
**Arab American University**

**Faculty of Graduate Studies**

# Classification of Colon Tumors based on Machine Learning Techniques and Deep Learning Techniques: A Comparative Study

# تصنيف أورام القولون على أساس تقنيات التعلم الآلي وتقنيات التعلم العميق: دراسة مقارنة

By

**Jumana Hisham Mustafa Abed**

Supervisor

**Prof. Dr. Mohammed Awad**

**This Thesis Was Submitted in Partial Fulfillment of the Requirements for the Master's Degree in Computer Science**

**September, 2022**

# Classification of Colon Tumors based on Machine Learning Techniques and Deep Learning Techniques: A Comparative Study

By

**Jumana Hisham Mustafa Abed**

This Thesis was Defended Successfully on **22/09/2022** and Approved By:

| Committee Members | Signature |
| --- | --- |
| 1. Supervisor: **Prof. Mohammed Awad** |  |
| 2. Internal Examiner: **Dr. Ahamd Ewais** | |
| 3. External Examiner: **Dr. Eman Daraghmi** | |

# Declaration

I declare that the content of this thesis is my own research work unless otherwise referenced. I certify that this thesis does not contain any material published before by another person or has been submitted elsewhere for any degree or qualification.

Student name: Jumana Hisham Mustafa Abed

Signature: 

Date: 24/8/2023

# Dedication

I dedicate this thesis to my family, friends, and work colleagues who have given me a lot of support. Special thanks to my husband for his support and encouragement for me all the time. I also dedicate this thesis to my supervisor, Prof. Dr. Mohammed Awad, thanking him for all his support and assistance.

## Acknowledgments

I would like to seize this opportunity to express my deep regards to Prof. Mohammed Awad for his advice, support, and time he has spent reviewing my work. Prof. Mohammed provided valuable suggestions which have had a significant impact and have helped in overcoming many obstacles in writing this thesis in the best way.

# Abstract

Classification of Colon Tumors based on Machine Learning Techniques and Deep Learning Techniques: A Comparative Study

**By: Jumana Abed**

**Supervisor: Prof. Dr. Mohammed Awad**

The rapid evolution in technology nowadays especially in the medical sector has encouraged researchers to build Artificial Intelligence (AI) models to help in the diagnosis of critical diseases that threaten human lives such as cancers. Colon cancer is considered the third cause of death worldwide. The symptoms of colon cancer are not noticeable at the early stages but it could spread fast damaging human organs. The diagnosis of Colon cancer can be automated using Artificial Intelligence powerful models at early stages, with less time, more accuracy, and fewer costs. In this thesis, Machine Learning (ML) algorithms and Deep Learning (DL) algorithms were used to classify colon tumors using a local dataset from the Ministry of Health of symptoms and risk factors and another global colon cancer tissue images dataset. Eight different machine learning algorithms were used; Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Radial Basis Function Neural Network (RBFNN), Recurrent Neural Network (RNN), and Multi-Layer Perceptron Neural Network (MLPNN). Convolutional Neural Network (CNN), Visual Geometry Group -16 (VGG-16), and Visual Geometry Group 19 (VGG-19) were used for deep learning algorithms. A comparison of performance results for the eight machine learning algorithms was carried out to evaluate the model with the best accuracy for classifying colon cancer symptoms and risk factors. Another comparison

between deep learning models is also done to evaluate the model with the best accuracy for classifying colon cancer tissue images. The results for the first experiment on a local dataset containing features selected by domain experts with machine learning algorithms showed that Ensemble and Naïve Bayes algorithms were able to achieve the best accuracy of 96.2%. Other algorithms; SVM, DT, KNN, RBF, RNN, and MLPNN achieved 87.80%, 92.4%, 79.2%, 66%, 92.5%, and 90.6% respectively. The same algorithms were applied o the local dataset after using mutual information feature selection. SVM, DC, Ensemble, and Recurrence Neural Networks achieved the best accuracy of 94.3%. For the second experiment using Deep learning techniques against the global dataset LC25000, VGG-19 was able to achieve the best classification accuracy of 98.94% while VGG-16 achieved 97.54% and CNN 83.94%. The ensemble algorithm is recommended for colon tissue tumor classification using the local dataset while VGG-19 is recommended with superior accuracy to classify colon tissue tumor images between cancerous and benign tumors.

# Table of Contents

# List of Figures

# List of Tables

# List of Abbreviation

| | |
|---|---|
| AI | Artificial Inelegance |
| ANNs | Artificial Neural Networks |
| ML | Machine Learning |
| DL | Deep Learning |
| DT | Decision Tree |
| NB | Naïve Bayes |
| SVM | Support Vector Machine |
| KNN | K-Nearest Neighbor |
| RBF | Radial Basis Function |
| MLPNNs | Multi-Layer Perceptron Neural Networks |
| VGG-16 | Visual Geometry Group-16 |
| VGG-19 | Visual Geometry Group-19 |
| CNN | Convolutional Neural Network |
| ROC | Receiver Operating Characteristic Curve |
| AUC | Area Under Curve |
| TP | True Positive |
| TN | True Negative |
| FP | False-Positive |
| FN | False-Negative |
| TPR | True Positive Rate |
| TNR | True Negative Rate |

# CHAPTER 1

# INTRODUCTION

## 1.1 Introduction

Cancer is the second leading cause of death worldwide; Colorectal cancer is the third cause of women’s death and the second cause of men’s death globally among cancers [1]. The abnormal growth of body cells inside the body will lead to the form of tumors. Those abnormal cells will affect the organs' functionality and threat the patient’s life as body cells are grown with special capabilities and functionality, other cells could lose their ability and resist growing abnormally. Those cells will form a tissue that’s called a tumor. Colon cancer is one of the most common cancers that affect elder people and those who have bad lifestyle habits such as smoking, lack of physical activity, and drinking alcohol. Genetic disorders can also be a cause of Colon cancer but in a small percentage. The incidence of Colon cancer can be reduced or prevented if it’s diagnosed at early stages and the precancerous polyps can be removed safely if detected at early stages [2]. Doctors avoid asking young people or people with no family history of Colon cancer to do screening as it can be harmful to expose to such radiation. But patients older than 50 years old are asked to do regular screening every year, especially for those who have a family history of colon cancer by doing a colonoscopy (going inside the colon) and blood tests to check the overall health of the patient, as detecting such tumors at early stages increases surviving potentials. The diagnosis of Colon cancer depends on the clinical examination and observation of metastases, their location, and size. Also, it depends on symptoms such as changes in bowel habits, blood

with stool, nausea, weight loss, and abdominal distension. Anemia is considered a secondary sign of Colon cancer.

Treating Colon cancer can differ according to its stage, it can be treated by surgery, chemotherapy, or radiation therapy.

To overcome the exhaustive screening for patients, surgeries, and false/late diagnoses, computer-aided systems can be used to help doctors in making a medical diagnosis of body tumors. Machine learning and deep learning techniques can be used to make feature extraction and tumor image classification into benign or cancerous tumors. Using machine learning in medical fields has proven its superior performance and efficiency in making a good diagnosis of Colon cancer. Also, deep learning showed significant capabilities in the prediction of visual features for classifying medical images. An aided system using Machine Learning and Deep Learning techniques can be built to detect and classify the Colon tumor, this system can help doctors improve decision-making and get accurate results. In this thesis, Machine Learning and deep learning techniques were used to classify colon tumors. A comparison between different machine learning algorithms is carried out to find the best algorithm with the local dataset. Another comparison is done using three different deep learning algorithms using the global tissue images dataset.

Artificial Intelligence refers to the implication of computers to model human intelligence in different life aspects [3]. A medical management system is an example of using AI in the health sector since Machine Learning achieves high classification accuracy. Many challenges are facing modern Medicine in terms of a huge amount of knowledge that needs acquiring and analysis to solve complex medical cases.

Developing medical artificial intelligence models has always been a part of the development of Artificial Intelligence to help doctors in the diagnosis [4]. Machine Learning is a part of Artificial Intelligence that can be used to build a predictive model that can improve spontaneously from data [5] as it has the power to automate the predictions from data patterns [6]. Building a predictive model that can make the right decisions and find solutions for problems requires Classification Machine Learning algorithms. Supervised Machine Learning is a group of algorithms that make predictions and classifications according to the supplied data into discrete values [7] such as Support Vector Machine (SVM), Naïve Bayes, Logistic Regression, Perceptron, Decision Tree, Random Forest, Neural Network and so on. The following algorithms were discussed to classify Colon Cancer: Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Radial Basis Function, Recurrent Neural Network, and Multi-Layer Perceptron Neural Network.

The first model, the powerful algorithm Support Vector Machine (SVM), this algorithm has tough regularization properties i.e. generalization of the model against new data [8]. SVM can fit linear and non-linear data and it can be used for classification and regression [9].

The second model is Naïve Bayes; this algorithm works with an assumption called "class conditional independence" which states that the effect of a value for a given attribute on a class is the independence of the values of the other attributes [10]. It works by finding the finest mapping between data and classification within the problem. The third model is the Decision Tree (DT), the tree is organized and sorted according to feature values. Each feature is represented by a node in that tree, and the value of that

feature is represented by the branch. The classification process starts from the root node and the following tree will be sorted according to feature values [11]. K-nearest neighbor is the fourth algorithm, this algorithm is known for its simplicity and good output result. The idea is to assign a class for all instances shown at distance k in the training data [12]. KNN classifier is mostly used in pattern recognition regardless of its calculation complexity [12]. The ensemble learning algorithm is the fifth on our list, this algorithm has significant and powerful features in achieving great results. The ensemble works by combining data modeling, data mining, and data mixture into one framework. This framework will extract data features and apply several learning algorithms to make predictions. The framework will use those predictions to continue obtaining better prediction results through voting schemes [13]. The sixth algorithm is the Radial Basis Function (RBFNN) which is a feedforward NN that uses radial basis as an activation function for its neurons. It works by finding the Euclidean distance between each neuron and the center of the RBF neuron and multiplying by connection weight to find the output of each neuron [14] [15]. The seventh algorithm Recurrent Neural Network (RNN) works by adding a feedback connection for each hidden neuron, so the last state of each hidden neuron will be used to find the neuron output value. RNN is performing very well in many ML applications. The eighth algorithm is Multi-Layer Perceptron Artificial Neural Networks (MLPNNs), this algorithm works by mimicking the biological neuron to solve different problems related to classification, prediction, pattern recognition, and so on. ANNs are the interconnection of nodes, each neural network consists of input, output, weights, and the activation function [16]. The above-mentioned eight machine learning algorithms were applied to the local dataset for Colon cancer classification. To classify Colon cancer tissue images into benign or

cancerous, CNN, VGG-16, and VGG-19 are based on the Conventional Neural Network which uses convolutional layers for feature extraction and fully connected layers for classification. VGG-19 and CNN are used to compare their accuracy with VGG-16. A colon cancer tissue images global dataset (LC25000) were applied to VGG-16, VGG-19, and CNN models to classify tumor tissue images into benign or cancerous tissues.

## 1.2 Objectives

The main goal of this study was to improve the accuracy of classification of colon tumors using Machine Learning and Deep Learning applied to local and international datasets, and to study, analyze and differentiate between those two approaches in terms of classification accuracy, where the efficiency of classification between Colon Cancer features dataset and medical images dataset are presented too. In general, ML and DL are used to find the best accurate methodology to diagnose colon tumors as benign or cancerous. An accurate diagnosis is very important at the early stages as it threatens life expectancy.

### 1.2.1 Specific Objectives of the Study

Other specific objectives of the study are the following:

- Several machine learning techniques were applied to classify colon tumors using a local dataset to find the best algorithm with the highest classification accuracy for colon tumors.
- Applying Deep Learning (DL) models by using the VGG-16, VGG-19, and CNN models to classify colon tumors using the global tissue images dataset.

- Recommending the best deep learning model to classify colon tumor tissue images with the best accuracy.
- Identifying the improvements that can be included in future research and determining the limitations encountered during the realization of the study.

## 1.3 Contribution

In Palestine, there were no studies of Machine Learning and Deep Learning applications in the field of classifying colon tumors into benign or cancerous tumors. A machine learning model is built using a Palestinian local dataset collected from the Ministry of Health to classify colon tumors at their early stages which will increase the quality and expectancy of people's life. Also, a rich global dataset of colon tumor tissue images is used to classify the colon tissue into benign or cancerous tissues using powerful deep learning techniques such as; CNN, VGG-16, and VGG-19.

## 1.4 Overview

In chapter one, the problem statement "colon tumors classification" is explained along with the objectives and the contribution. The remaining thesis is organized as follows: Chapter two will provide a background for the two datasets that were used; the local dataset, and the colon tissue images global dataset, then literature is discussed, also, as other classification techniques of Colon cancer disease. Chapter three discussed the methodologies used in this study in general, the data collection step will be discussed, and then data preprocessing steps will be shown, including feature selection and solving missing values issues. Then, the building model phase and Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN),

Ensemble, Radial Basis Function, Recurrent Neural Network, and Multi-Layer Perceptron algorithms were explained. VGG-16, VGG-19, and CNN models for Colon tissue image classification are explained. Finally, several performance measurements were presented to evaluate the machine learning techniques. In chapter four, the experiments and the results of each dataset with each model were discussed, also, an evaluation of the accuracy performance of each model for each dataset is provided to show the best one for the global and local datasets. Chapter five presents the conclusion and future work.

# CHAPTER 2

# BACKGROUND

## 2.1 Background

The evolution of technology to improve the process of diagnosis and treatment of cancer patients is bringing too much attention to the world. The ability of Machine Learning and Deep Learning algorithms to achieve a great level of accuracy is a huge step forward in the medical field. Many hospitals are moving toward using predictive models to detect and diagnose cancer tumors in their early stages. Colon cancer is affecting the mortality rate and it's considered the third cause of death worldwide [1]. Cancer is dangerous as it grows irrepressibly and irregularly. The spread of this disease is very fast, the cells start to divide in an abnormal way causing damaged cells to move in the body without having clear symptoms at the early stages. Researchers had introduced many studies in the field of Artificial Intelligence to classify and predict Colon cancer [2] [3] [4] [5].

In this thesis, numerous models were conducted to classify colon tumors such as Ensemble, Naïve Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Radial Basis Function, Recurrent Neural Network and Multi-layer perceptron Neural Network (MLP). These models were applied to a local dataset and the results were compared to decide which model has the best classification accuracy.

Also, CNN the convolutional neural network, VGG-16, and VGG-19 models are used in the Colon cancer tissues dataset to classify benign and cancerous tumor tissues.

## 2.2 Dataset Description

In this research, two datasets were used:

### 2.2.1 Colon Cancer Histopathological Image Global Dataset (LC25000)

In this part of the thesis, a set of histopathological images of Colon benign and cancerous tissues was used [6]. From those tissue images, doctors can decide whether the patient has colon cancer or not. The images were obtained from pathology glass slides observed in the laboratory from the Global dataset of Lung and Colon cancer tissue images (LC25000) that was made available for researchers. The dataset contains 500 images of colon tissues (250 benign colon tissue and the other 250 colon cancerous), and images were validated by HIPAA (The Health Insurance Portability and Accountability Act of 1996). The authors used the Augmentor python framework to expand the dataset to 25,000 images of two classes; Colon and Lung tissue images. The colon folder contains two subfolders: colon_aca subfolder with 5,000 images of colon adenocarcinomas (cancerous tissues) and colon_n subfolder with 5,000 images of benign colonic tissue. In this thesis, we used 7593 images divided between benign and cancerous tissue images. Figure 2.2- 1 below shows the distribution of Colon tissue images.

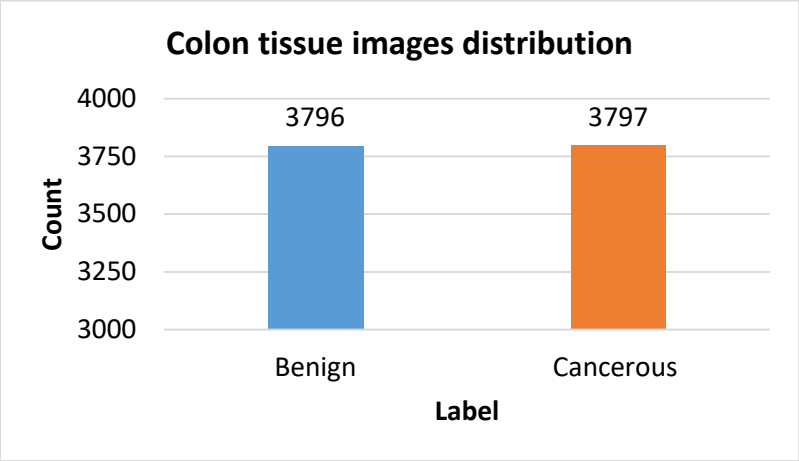


Figure 2.2-1: The distribution of colon tissue images

Figure 2.2-2 shows a sample from the training set of histopathological images of Colon benign and cancerous tissues used in the training process. The image labeled as ‘colonn’ is a benign tissue image and the image labeled as ‘colonca’ is a cancerous tissue image.

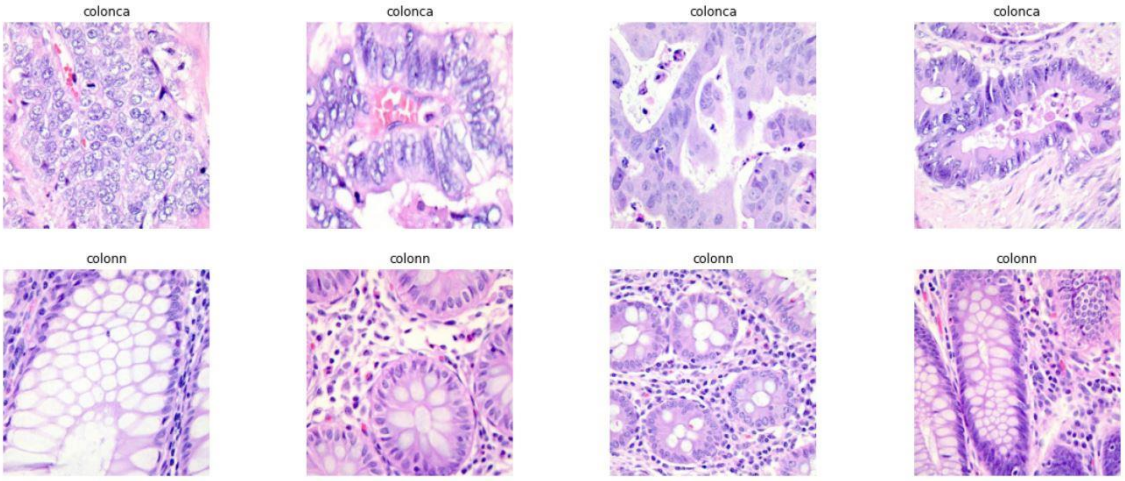


Figure 2.2-2: Benign and Cancerous tissue images used in the training process

### 2.2.2 Local Dataset

The local dataset was taken from the Ministry of Health, it contains data from 21 attributes from 350 patients, later the attributes were reduced to ten attributes after consulting a physician specialist. The patients are from different Palestinian cities:

Ramallah, Nablus, Jerusalem, Salfit, Jenin, Biet Lahim, Qalqiliah, Tulkarem, and Hebron. The patients' results were classified into two categories:

0 Benign

1 Colon Cancer

**Table 2.2-1: Valid ranges & description for each feature (Local Dataset)**

| # | Feature Name | Description | Range of Value | Type of value |
|---|---|---|---|---|
| 1. | Place of Residency | The city of patient | (0-9) | Discrete variable |
| 2. | Family History | Does the patient have a family member with a history of cancer? | 1-101-110-111-9-99 | Discrete variable |
| 3. | Age when diagnosed | Age of the patient | 0.01–0.97 | Continuous variable |
| 4. | Work | Does the patient work or not? | 0 (No)<br>1 (Yes) | Binary variable |
| 5. | Job | Type of patient job | (1 - 99) | Discrete variable |
| 6. | Sex | Gender of the patient | 0 (Male)<br>1 (Female) | Binary variable |
| 7. | Marital status | Marital status of the patient | (1 – 9) | Discrete variable |
| 8. | PMH | Past Medical History | (0 – 99) | Discrete variable |
| 9. | Smoking_Yes_No | Does the patient smokes or not? | 0 (Yes)<br>1 (No)<br>99 (Missing) | Discrete variable |
| 10. | PA_Rate | Physical activity of the patient | 0 (Low)<br>1 (Moderate)<br>2 (High) | Discrete variable |

The details about each feature for the local Dataset:

- Age: In the MoH dataset, the ages of patients are ranges from 1 and 91 years old. The youngest patient is 1 year old and the oldest is 91 years old. The most repeated value (Mode) is 48 years old and the average patient's age is 53. Figure 2.2- 3 shows how the data is distributed according to Age.

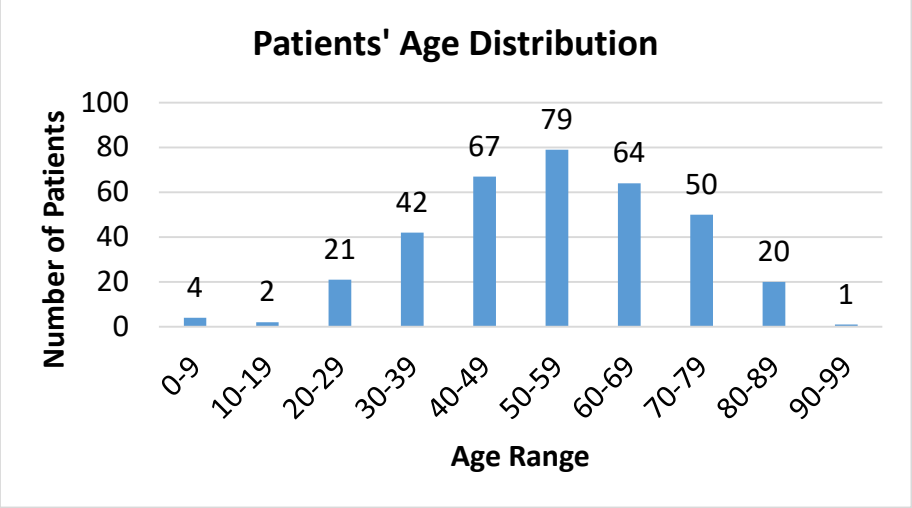


Figure 2.2-3: Patients' Age Distribution

- Sex: The gender of the patients in the MoH dataset is shown in Figure 2.2- 4.

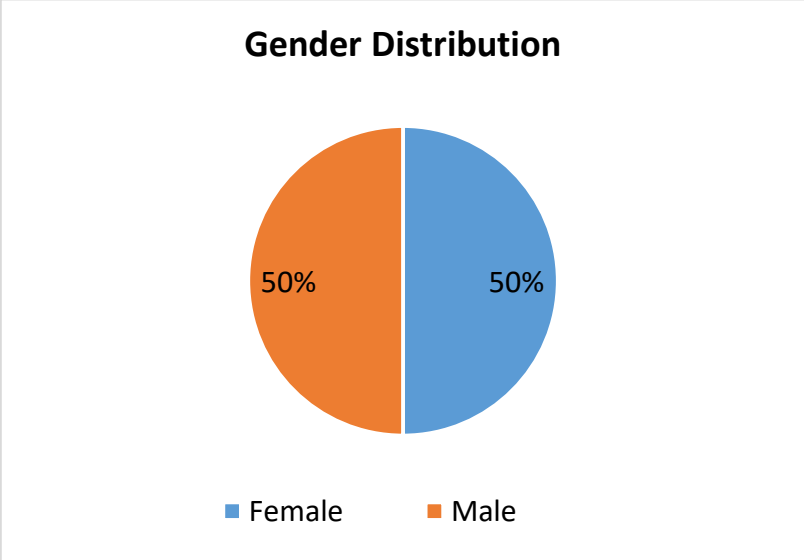


Figure 2.2-4: Gender Distribution

- Place of Residency: Figure 2.2- 5 shows that the most of patients in the MoH dataset are residents of Nablus city with a total of 20.57% of total patients. The lowest percentage is for Jerusalem city with a total of 2.57%.

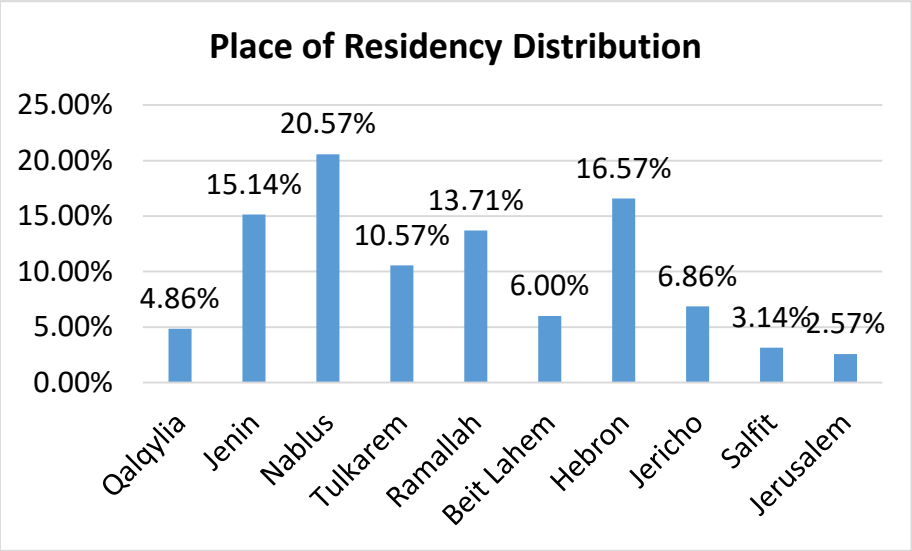


Figure 2.2-5: Place of Residency Distribution

- Marital Status: from the 243 patients diagnosed with Colon cancer, there's 74.49% of them are married while 0.82% are divorced as shown in Figure 2.2- 6.

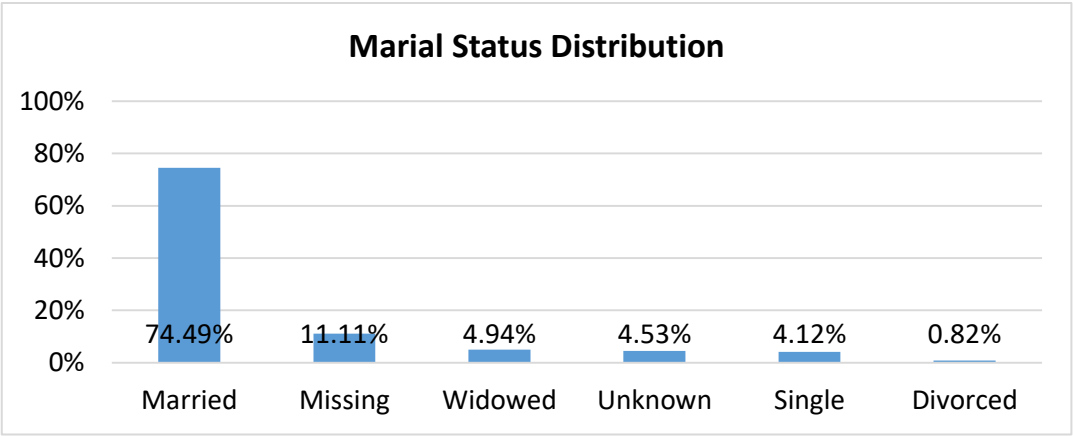


Figure 2.2-6: Marital Status Distribution

- Past Medical History (PMH): of 243 patients diagnosed with Colon cancer, 44.03% of them have a different past medical history such as Hypertension (HTN) and Diabetes-Mellitus (DM) together or others with HTN only. 37.08% are free i.e. do not have any history of medical issues. Figure 2.2-7 shows the distribution of the patient's past medical history in the MoH dataset.

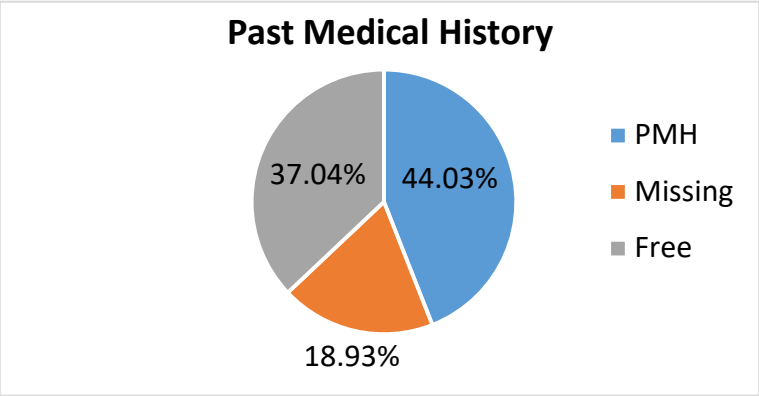


Figure 2.2-7: Past Medical history of Patients

- Physical Activity Rate: This variable indicates the rate of a patient's physical activity whether it's low, moderate, or high. Low physical activity is affecting the mortality rate. In the MoH dataset, among the patients diagnosed with Colon cancer, 4.53% are physically inactive or have a low-activity lifestyle while 33.74% are moderate as shown in Figure 2.2-8.

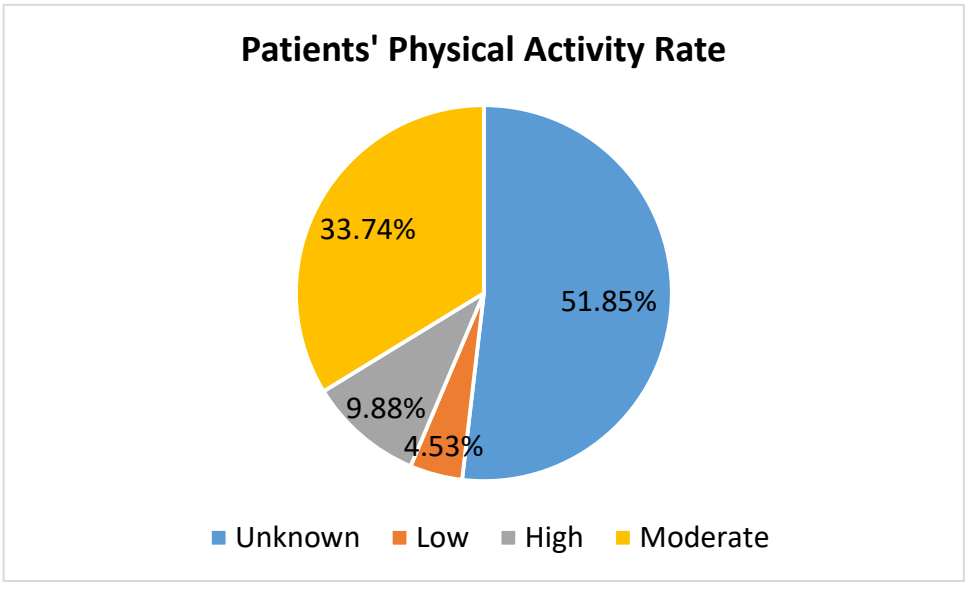


Figure 2.2-8: Patients' Physical Activity Rate

- Smoking: This variable indicates if the patient smokes or not. Smoking is considered a high-risk factor for colon cancer. In the MoH dataset, the majority of patients in the percentage of 62.55% are not smoking while 26.75% are smoking as shown in Figure 2.2-9

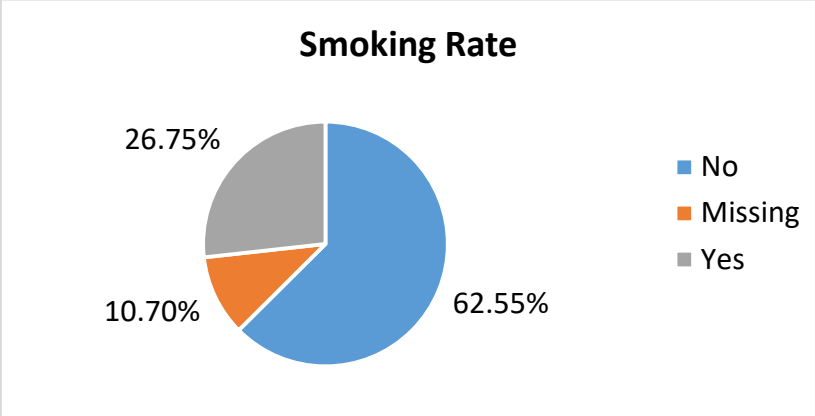


Figure 2.2-9: Patients' Smoking Rate

- Family history: Genetics is considered a risk factor that increases the probability of having Colon cancer. Among patients diagnosed with cancer in the MoH dataset, family history is unknown for most patients while only 19 of them had a past medical history of related diseases.
- Working: 80.29% of all MoH dataset patients are working in full-time jobs while 6.29% are not. 76.13% are diagnosed with Colon cancer.
- Job: Among the patients diagnosed with Colon cancer, the majority of them are 37% work as staff nurses. The minimum percentage is 0.41% for teachers and housewives.

## 2.3 Related Work

Previous studies have discussed colon cancer classification and detection from different perspectives and they used several input types. Recently, the methods related to gene data extraction which becomes an area of interest for study with another deep neural network existence of supervised and unsupervised machine learning techniques that makes it available for researchers to work with different input data types and get no good results but even superior results. In [7], the authors developed a shallow neural network in supervised learning for Colon cancer genes by narrowing its computational process to get better results and optimal goals. The results were excellent and stable when compared to other deep neural networks. In [8], the authors focused on using supervised learning in three different machine learning techniques for cancer classification. C4.5 decision tree bagged decision tree and boosted decision tree. Classification using bagged and boosted decision trees showed a better performance than C4.5. In [9], Fourier transforms infrared (FTIR) spectroscopy was used to differentiate between colon cancer patients and healthy people. They extracted 16 features from the user data and used compared the performance of the multilayer perceptron neural network and support vector machine with cross-validation. Fahmi and others in [1] did a hypothesis test using a t-test, Mann-Whitney-Wilcoxon, KNN, and Decision Tree to detect the real genes of colon cancer patients. They characterized it into 2 clusters with 20 genes for each, those genes would be very effective in the early diagnosis of colon cancer.

Between the 70’s and 80’s, the invention of DL and non-DL methods based on learning algorithms began to see the light the researchers by using radiographs in CAD

systems to diagnose different and various types of cancers. In this paper, the focused is on the work related to colon cancer. The early diagnosis of colon cancer has a high survival rate of 70% at stage 0 but this rate goes down to 13% if it was diagnosed at stage IV [3]. In [2], Wang et. al. presented a software system to extract the features during colonoscopy and assist the endoscopist and provide feedback. This software uses an edge-cross and rule-based classifier to detect the polyp edges. Their software was able to detect 97.7% of the edges correctly on random 53 shots from a video file of a single colonoscopy procedure. Pu et. al. [10] proposed a CAD system that uses the CNN model to classify NBI and BLI colon images. Their system has achieved very good results and is similar to experts' diagnoses. In [4], Chen and the authors developed a Computer-aided classification system with a deep neural network for colonic polyps. In this study, the authors used NBI from endoscopy giving real-time prediction of Colon polyps. They tested the system by training the DNN using 2157 images of neoplastic polyps and hyperplastic polyps. In their study, they were able to detect hyperplastic polyps with a sensitivity of 96.3% and in approximately 1.5 seconds.

Deep learning algorithms and techniques are taking attention these days for their superior performance and accuracy achieved. For example, Convolutional Neural Network was used by Park and Sargent [5] to extract features from polyp's images and it achieved a sensitivity of 86%. In the same context, Urban and other authors in [11] used a dataset of 8641 colonoscopy images for 200 patients in evaluating the CNN model. That model was able to detect polyps with a validation accuracy of 96.4%. Ozawa et. al. [12] used the CNN model with a Single Shot MultiBox detector in polyp classification and detection. In their work, they used a private dataset and they were able to achieve a good performance in colon cancer. Zeng et. al. [13] also used the CNN

model but with coherence tomography imaging to detect normal colon tissues and neoplastic colon tissues. CNN model was based on RetinaNet and it was the first study in the context of real-time detection of colon cancer, it showed a good potential for detection accuracy. Nadimi et. al. [14], used the CNN model for detecting polyps from images captured during capsule endoscopy. To detect the regions of the Colon polyp faster, they used Faster R-CNN. The study included about 11 thousand images and they were able to achieve high-performance metrics compared to previous studies. Hasan et. al. [15] came up with a working model with CNN with contour transformation for polyp detection from images and video data. They evaluated that model with the VGG19 model and found that VGG19 achieved the best accuracy. Blanes-Vidal et. al. [16] examined five different CNN algorithms for polyp localization. They used VGG-16, VGG-19, AlexNet, GoogleNet and ResNet50. Zhoa et. al. [17] proposed a CNN model with CRCNet for the optical detection of Colon cancer. CRCNet is a dense layer with 169 layers. The authors conducted three tests using three different datasets and their model, the results were good for two datasets and better than the performance of the endoscopist. Park et. al. [18] proposed a CNN model to classify normal, adenoma, and adenocarcinoma colon images extracted from colonoscopy. Their contribution was in terms of enhancing the performance of CNN, the best result achieved was by using 43 fully connected CNN. The experiments showed that their model is better than VGG-16 and DenseNet-121 and ResNet-152. Another study using CNN was done by Tamaki et. al. [19] by comparing the performance of three different CNN models using SVM to classify NBI colon images. Results showed that CNNs with fewer layers can achieve better performance when compared with CNN with too many layers. Several kinds of literature proposed different Deep Learning algorithms for classifying Colon cancer

using tissue images; Ponzio et. al. [20], proposed a CNN model to classify between normal tissue and affected tissues. The model was trained using the ImageNet dataset and achieved an accuracy of 90%. In another hand, they used also transfer learning approaches and applied it again based on CNN, the accuracy increased to 96% using the same test data. Alom et. al. in [21], proposed a model of a Densely Connected Recurrent Convolutional Network (DCRN) using different datasets, the results then were compared to previous DCRN study. The model F-1 score was better in ratio of 3.4% and 4.5% For nuclei classification. Yoon et. al. [22], used a Deep CNN for image data to locate and classify different tumor types. In their study, they used a dataset from National Cancer Center and applied it to DCNN with five different VGG configurations, the accuracy of the configurations ranges between 82.50% and 93.48%. Bychkov et. al. [23] also used CNN but combined it with recurrent neural network to train the model using e dataset of Colon cancer tissues. The model was able to achieve an accuracy better than human experts. Patel et. al. [24] demonstrated a comparative study using six different CNN models using a video dataset that contains two different types of Colon polyps. The results showed that CNN can achieve a very good performance in the classification of Colon cancer. Mangal S. et. al in [25] used LC25000 dataset to classify lung and colon cancer by building a CNN model. Their model achieved an accuracy of 96% for lung cancer classification and 97% accuracy for colon cancer classification. The same dataset was used also by S.Garg in [26], in their paper, they trained eight distinct models: CNN, VGG16, NASNetMobile, InceptionV3, InceptionResNetV2, ResNet50, Xception, MobileNet, and DenseNet169. Models achieved a superior accuracy ranges between 96% and 100%. Bukhari S. et. al in [27] applied LC25000 global dataset three different CNN models; ResNet-18, ResNet-34

and ResNet-50. The accuracy results are 93.04 % for ResNet-18 and 34 while ResNet-50 achieved 93.91%.

Many wrong lifestyle habits can lead to a risk of having colon cancer, smoking, drugs, family history and others are risk factors for colon cancer [28]. Understanding the relationship between risk factors can help in diagnosis studies. A classification tree model was proposed by Nicola J. and Martha L. in [29] to investigate and evaluate the interaction and relationship between risk factors for colon cancer early diagnosis.

Several deep learning and machine learning techniques have achieved several good results with different datasets to classify colon cancer and other medical diseases as mentioned above. So, in this thesis, CNN, VGG-16, and VGG-19 were used with the LC25000 dataset to achieve better classification accuracy. Also, eight different machine learning algorithms were used; Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Radial Basis Function, Recurrent Neural Network, and Multi-Layer Perceptron with a local dataset from the Palestinian Ministry of Health for colon cancer patients. Those algorithms were used by several researchers for different classification purposes and achieved good results. So, the aim was to use those algorithms with different datasets to get a better classification of colon tumors.

# CHAPTER 3

# THE PROPOSED METHOD

## 3.1 Proposed Method

This chapter of the thesis explains the methodologies and models in Machine Learning (ML) that were used and applied to datasets for the comparison of different classification algorithms. Then, the best suitable model/classifier that can improve classification accuracy in cases of colon tumors is considered. The work started from the local dataset of colon cancer from the Palestinian Ministry of Health (MoH). After that, the models are described: Decision Tree (DT), Naïve Bayes (NB), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Ensemble, Radial Basis Function, and Multi-layer perceptron Neural Network (MLPNN). Then, a comparison between the eight models is shown in terms of performance metrics when applied to the local dataset. To enrich the thesis, Deep Learning (DL) was used to classify benign and cancerous Colon tissues using a global dataset LC25000. The same global dataset was applied to three different deep learning models; CNN, VGG-16, and VGG-19. The results of the three models are then compared in terms of classification accuracy.

Finally, the performance metrics used to evaluate the models are explained. Machine Learning (ML) models diagram is shown in Figure (3.1-1), Figure (3.1-2) shows the Deep Learning (DL) model diagram of the Convolutional Neural Network (CNN), Figure (3.1-3) shows the VGG-16 model and Figure (3.1-4) shows the VGG-19 model.

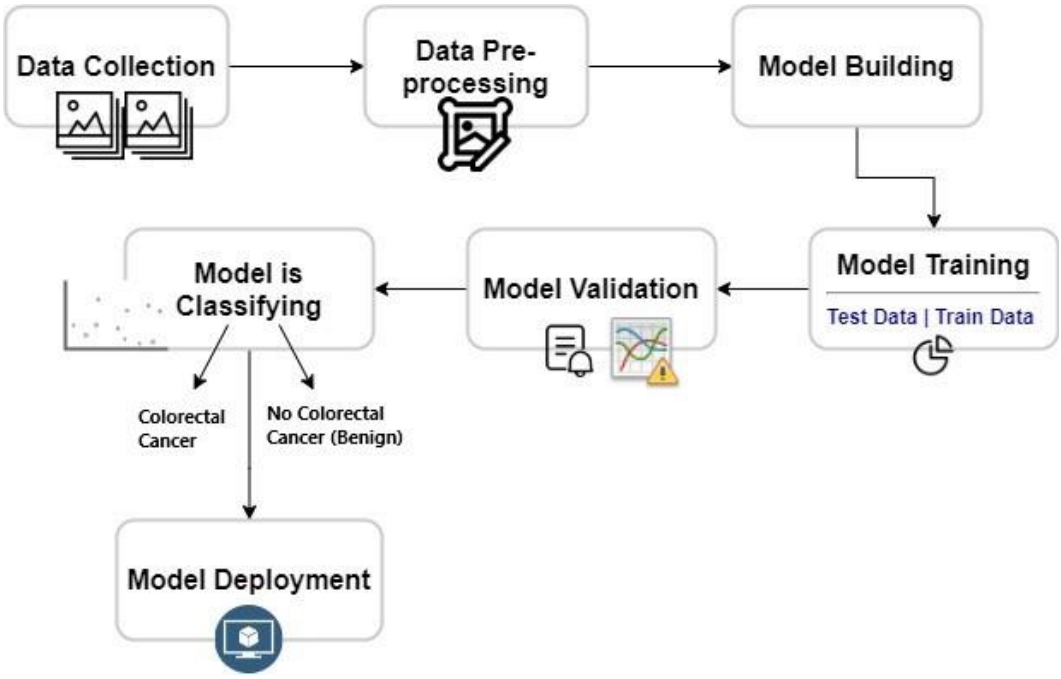


Figure 3.1-1: Block diagram of the Machine Learning (ML) models

For all machine learning and deep learning models, the process will start by taking the pre-processed data and then building the model. After that, each model will be trained using 80% of the dataset which is called training data. If the model performance is satisfying, the model will be tested using the other portion of the dataset which is 20% of the dataset called test data. Finally, the model will be deployed and can be used for further classification.

The input data is fed into the CNN network from the input layer, then it will pass by the convolutional layers for feature extraction. After that, the fully connected layers do the classification and pass the result to the output layer.

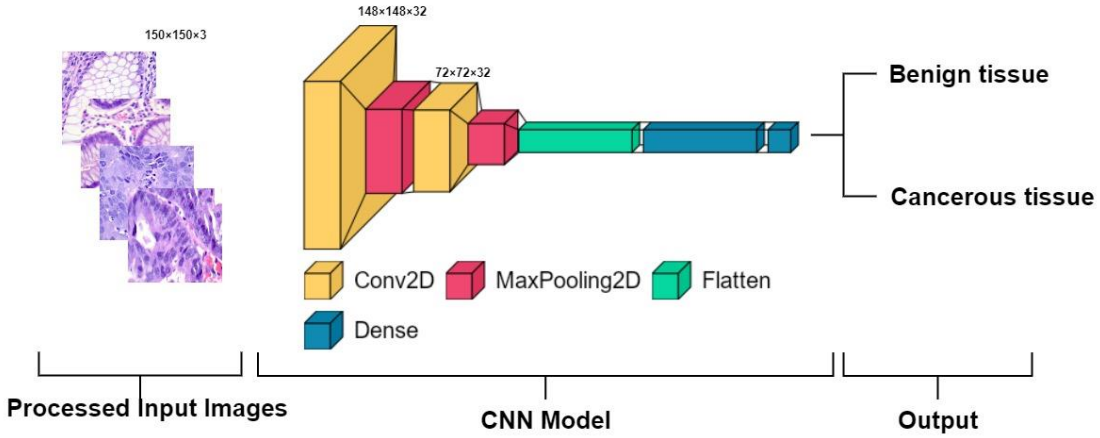


Figure 3.1-2: Block diagram of the CNN model

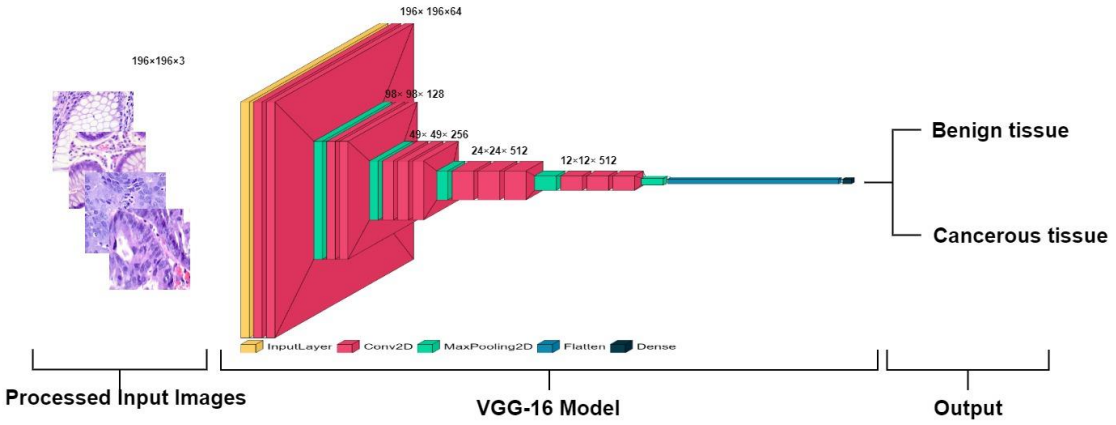


Figure 3.1-3: Block diagram of the VGG-16 model

The VGG-16 model was built by adding 13 convolutional layers and 3 dense layers (fully connected layers). The data will be passed through 16 convolutional layers to extract the features and the classification result will be directed by the multiple dense layers. A softmax activation function is used in the last dense layer for the final classification result.

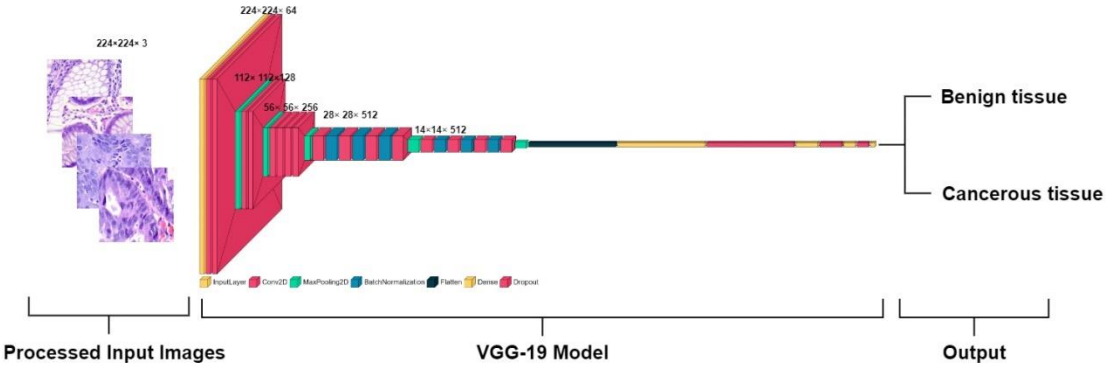


Figure 3.1-4: Block diagram of the VGG-19 model

The VGG-19 model was built by adding 16 convolutional layers and 3 dense layers (fully connected layers). For CNN, VGG-16, and VGG-19 models, the pooling layers are used to reduce the matrix size. Then, the softmax dense layer is added for the final classification output value.

## 3.2 Data Collection

Two datasets will be used; one is a local dataset from the Ministry of Health (MoH) and a global Colon tissue images dataset from Kaggle. The local dataset used in this study was collected by Abu Zuhri et. al. [30] from National Cancer Registry (MoH) which contains data for 243 cancer patients with 21 attributes. The other 107 records were for patients suspected to have Colon cancer. The data contains information about the patient's age, gender, family history, smoking, physical activity, marital status, place of residency, working status, job, and past medical history.

## 3.3 Data Preprocessing Phase

The data preprocessing phase contains many steps such as data aggregation, normalization, cleaning, dealing with missing values, and feature selection [31]. In this study, feature selection and dealing with missing value operations performed on the local dataset are explained in this section.

### 3.3.1 Feature Selection

Abu Zuhri et. al. [30] have done an intensive literature review and consulted experts in the field of Colon cancer to decide which features are the most important among the 21 collected attributes from each patient. Then, they decided to keep the ten attributes mentioned before in Table 2.2 1 and exclude other unrelated attributes.

### 3.3.2 Mutual Information Feature Selection

A Mutual Information technique is also used against the local tabular dataset. Mutual information works by evaluating the dependency between two variables to measure the amount of information every two variables have about each other. A zero mutual information score means that variable X doesn’t have information in it about the second variable Y [32]. Table 3.3-1 shows the scores of the 13 features after Mutual Information is done. The features with non-zero scores were selected, which are: Religion, Occupation, Marital Status, Place of Residency, PMH, Age at diagnosis, Family history, ICCC, Place of Treatment, Basis of Diagnosis, and Behavior. We used ML techniques on the local dataset with the features selected by the experts as Abu Zuhri et. al. [30] have done. Also, the same ML techniques are used against the local dataset with the features extracted using Mutual Information.

**Table 3.3-1: Features Scores using Mutual Information**

| Feature | Score |
|---|---|
| Gender | 0 |
| Smoke | 0 |
| Religion | 0.072705 |
| Occupation | 0.078695 |
| Marital status | 0.120763 |
| Place of Residency | 0.157876 |
| PMH | 0.164578 |
| AGE at DX | 0.254521 |
| Family History | 0.383547 |
| ICCC | 0.588188 |
| Place of Treatment | 0.598447 |
| Basis of Diagnosis | 0.599402 |
| Behavior | 0.599402 |

### 3.3.3 Dealing with Missing Values

Missing records can affect the output of the model, so it's important to do a data cleaning by several approaches such as: ignoring the record with missing values, filling missing values manually, filling missing values with the global variable that is usually 'Unknown' or use the mean value to fill the missing value [33]. Variables with time stamps were removed, as they do not reflect any category. In this dataset, Abu Zuhri et. al. [30] have done 3 steps against MoH local dataset:

1. Duplicate records were deleted.
2. The dataset contains 3 attributes for past medical history (PMH1, PMH2, and PMH3). Those values were combined into one attribute called PMH.
3. Inconsistent entries in each record were deleted or replaced by an 'Unknown' value.

## 3.4 Building Models Phase

This section is divided into two phases; the first phase is the application of various Machine Learning models applied to the local dataset is explained in detail. The local dataset collected from MoH was applied to Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Recurrent Neural Network, Radial Basis Function Neural Network, and Multi-Layer Perceptron. A 5-fold cross-validation is used with the local dataset to divide the local dataset. The results of those models were then compared according to classification accuracy.

The second phase is the application of VGG-16, VGG-19, and CNN deep learning methodologies on the Colon tissues images (LC25000) dataset from Kaggle, and the classification results are then compared. Two different datasets are used in this thesis because there's no local colon cancer image dataset to apply to deep learning algorithms. Also, there's no tabular data for colon cancer risk factors or symptom variables. So, a tabular local dataset was used for machine learning and a global colon cancer tissue images dataset was used with deep learning algorithms.

- **First Phase**

In this phase, the machine learning algorithms used to train and test several models with the local dataset for Colon cancer classification are shown in figure (3.4-1).

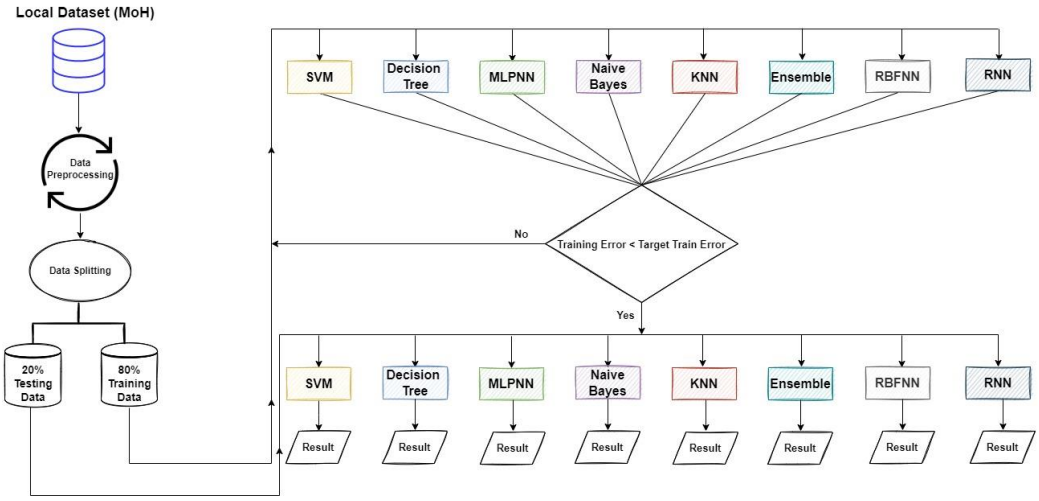


Figure 3.4-1: The general diagram for the first phase of classification using the local dataset by all models

The training dataset was applied to each of the eight algorithms to build the classification model. A target training error is set to $1e^{-5}$ the training process was repeated until the training error is less than the target training error. After the training process, the model was tested against the remaining 20% of the local dataset.

### 3.4.1 Support Vector Machine (SVM)

SVM is a supervised learning tool used to solve classification and regression problems for linear and non-linear data by assigning labels to objects [34]. SVM is considered a powerful tool for recognizing objects such as handwriting images and microarray gene expressions [35]. It consists of four concepts: 1. hyperplanes 2. maximum margin hyperplanes 3. soft margins 4. kernel function.

SVM uses hyperplanes to separate the local dataset of unseen data and a larger distance between hyperplanes gives a better generalization of the classifier. For the local data, it will be classified into two classes; 0 and 1 as shown in Figure 3.4-2. SVM will start making one hyperplane (line) to separate the data into 2 categories/classes which

are in our case; 0 and 1. Data points are defined as support vectors. Of course, there's an infinite number of possible lines to be set as a hyperplane, SVM will decide which line separates the data by calculating the margin between the hyperplane and the closest support vector to it. SVM will continue to set a new hyperplane until it reaches a hyperplane with the largest margin value so that any new data point would be classified to the correct class. Dealing with a dataset with only two variables or two classes is not always the case, most cases are for non-linear data.

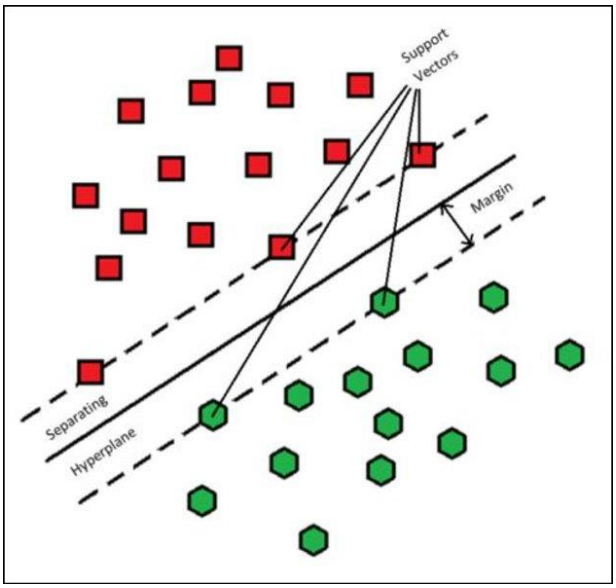


Figure 3.4-2: SVM Classifier [36]

Our training dataset $\mathrm{T} = \{(x_1 y_1 \ldots . x_l y_l)\} \in (\mathrm{X} \times \mathrm{Y})^l$, where: $x_i \in \mathrm{x} = R^n$ , $y_i \in (1, 0)$ $\mathrm{i} = 1, \ldots . , \mathrm{l}$ and a decision function $\mathrm{f(x)\ X} = \mathrm{R} \rightarrow \mathrm{Y}$

X is the features vector and Y is the expected label/class either 0 or 1 for the training input data. The hyperplane shown in (Figure 3.4.1- 1) separates the data points by a certain margin. Hyperplane equation can be written as:

$$(\omega \cdot \mathrm{x})^T + \mathrm{b} = 0$$

(Eq 3.4-1)

Where w is the weights of the features, X is the features vector and b is a bias factor. Many hyperplanes can separate the data into classes but the optimal hyperplane should meet (Eq 3.4-1) [37]. Then, another two parallel hyperplanes are set by following (Eq 3.4- 2) so that no other points are laying between the optimum hyperplane and those two parallel hyperplanes [34].

$(\omega \cdot x)^T + b = 1$ and $(\omega \cdot x)^T + b = -1$ (Eq. 3.4- 2)

The margin between hyperplanes can be increased gradually until the maximum margin is reached. The distance between hyperplanes is: $\frac{2}{|W|}$ so, to maximize the margin, we need to minimize |w|. All data points 'i' in the local dataset should ensure to meet (Eq. 3.4- 3) which leads to the separation of the data points into one of the two classes defined which are 0 or 1 as defined before.

$w.x_i - b \geq 1$ **or** $w.x_i - b \leq -1$ (Eq. 3.4- 3)

For non-linear data, SVM should handle assigning data points to multiple classes that couldn't be divided into just two classes as shown in Figure 3.4- 2. For such cases, Kernel functions are used to transform data into high-dimensional data for easier data classification by using a pattern recognition class of algorithms. The training vector of $x_i$ is assigned to a function $\emptyset$ to be transformed into high-dimension space. Then, a linear hyperplane with a maximum margin value is assigned. There are many kernel functions to use in SVM, the choice depends on the problem and the model we're trying to build.

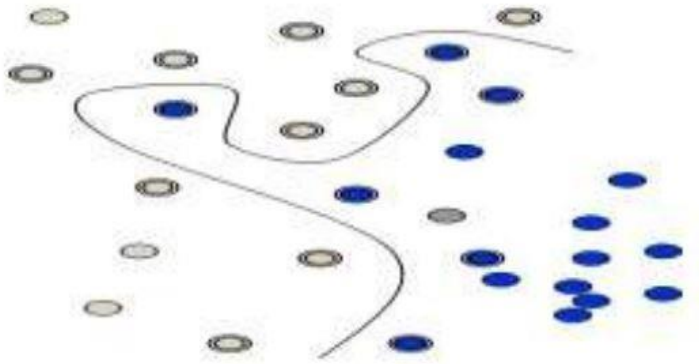

Figure 3.4-3: Multi-dimensional Classification [34]

### 3.4.2 Naïve Bays (NB)

Naïve Bayes is a probabilistic classifier that uses Bayes' mathematical theorem to calculate the conditional probability $P(A|B) = \frac{P(A)\ P(B|A)}{P(B)}$ which calculates the frequency of certain values within the dataset [38]. NB can be used as a replacement for logistic regression because of its capability of treating variables as if they're independent of each other. The algorithm works by assuming that one parent is having multiple children, and each child is considered an independent attribute (Figure 3.4- 4). So, for the local dataset having 10 attributes vectors and $x_1$ is one of the attribute vectors $\{x_1, x_2, \ldots\ldots, x_{10}\}$. The target classes are m $\{c_0, c_1\}$, $c_0$ is 0 and $c_1$ is 1. So, for each attribute in the local dataset, the probability value will be calculated $P(c_i|x)$ among all classes m. If the highest probability of the incoming data belongs to class $c_1$, then it will be assigned to class 1 [39].

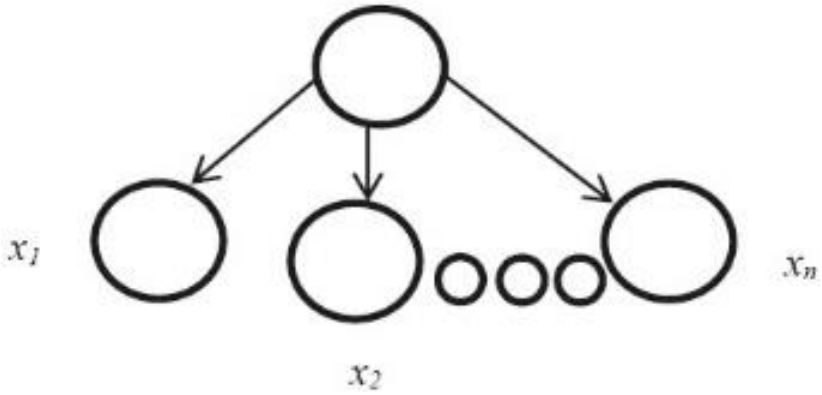


Figure 3.4-4: Naive Bayes Structure [39]

### 3.4.3 Decision Tree (DT)

Decision Trees are mostly used to solve complex space classification problems as it considered part of supervised learning algorithms and it's capable of solving problems for hug data with different input types and even missing data or data with errors [40]. The idea is to build a tree that can take any input data and classify it into its correct class 0 or 1 with good performance. The tree follows the heuristic function and the top-down approach. It starts from the root node and starts going down in the tree by taking a test every time the data passes into a test leave node, the test node will check if the input meets the threshold. In simple words, the algorithm will move down into the tree to predict according to test answers [41]. Decision trees have three important concepts:

1. Node labeling: building the tree starts to split the local dataset from a single root node and then split the node into test nodes, the algorithm will continue to split the nodes recursively into smaller and smaller subsets of test nodes using the Gini index splitting criterion. The maximum number of splits is set to 100. The terminal node will be given a class name [42].

2. Stop splitting: decision tree will continue to split down as needed but that could cause the algorithm to overfit. That can be solved by using stop-splitting criteria or by using a post-pruning algorithm [43]. In our study, the maximum number of splits is set to 100.
3. Post-pruning: to avoid the overfitting of the decision tree, different post-pruning methods can be used to shrink the growly decision tree and remove the unnecessary parts to maximize its performance [43] [41].

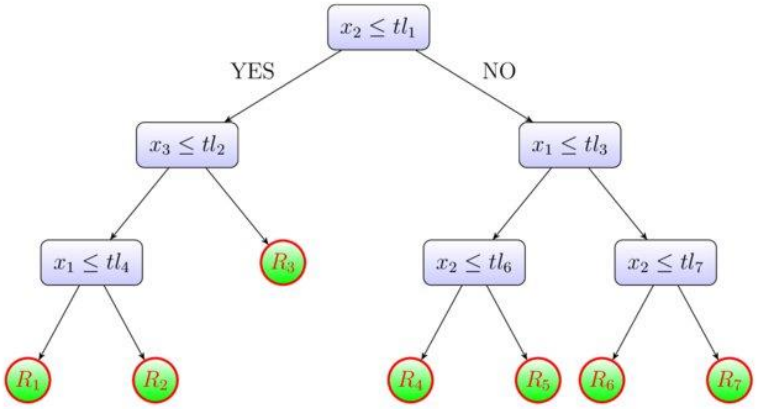


Figure 3.4-5: Decision Tree [44]

### 3.4.4 K-Nearest Neighbor (KNN)

KNN is a multiclass supervised learning classifier algorithm that works by setting up boundaries between data points [45]. The term ‘k’ refers to the number of neighbors around the new incoming data that should be taken into account when assigning it to a class. Having a small number of target classes ‘k’ will help avoid classifier noise while a large number of target classes ‘k’ will increase the computational power [46]. In this study, the K value was set to 3, then, the input data point was assigned to class 0 or 1 by

calculating the Euclidean distance for the incoming data point against all data points in the space (Eq. 3.4-4) [47].

$$d(q,p) = \sqrt{(q1-p1)2 + \cdots ... + (qn-pn)2} \quad \text{(Eq. 3.4-4)}$$

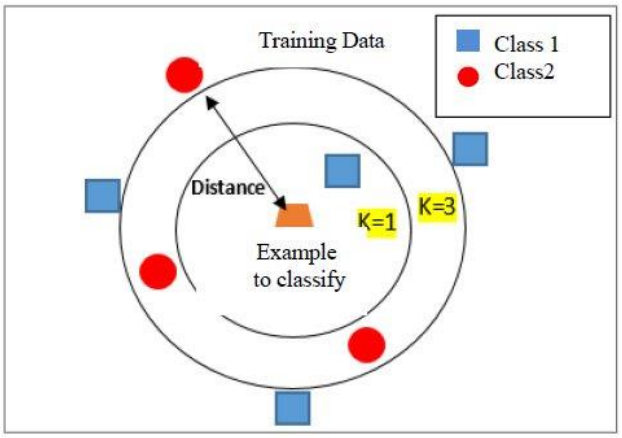


Figure 3.4-6: KNN Classifier [48]

### 3.4.5 Ensemble Algorithm

An ensemble algorithm is a combination of multiple learners together to solve a problem. All learner techniques are aggregated by a combination rule and the output is then generated by a single technique [49]. The ensemble is being widely used in Machine Learning for classification and regression problems; especially in bioinformatics [50]. The combination of different techniques improved the performance and the accuracy of classification and regression models. In general, Ensemble consists of two steps; one is generating the base learners, and two is combining the results of learners [49]. The homogeneous Ensemble technique combines three methods, 1) Bagging, 2) Boosting, and 3) Random Subspace. The combination of all methods can exceed the prediction accuracy when compared to the accuracy of a single method.

### 3.4.5.1 Bagging Ensemble Method

Bagging is an algorithm that works as an aggregator for ensemble methods. It works by creating multiple subsets of the same size taken from the dataset, then, it will create a single predictor for each subset as in Figure 3.4-7. Those subsets will be used to train the model in parallel R several times. After that, the result of all predictors is aggregated to make the final result. Bagging is widely used in decision trees and there're multiple bagging methods such as random forests and large-scale bagging [51].

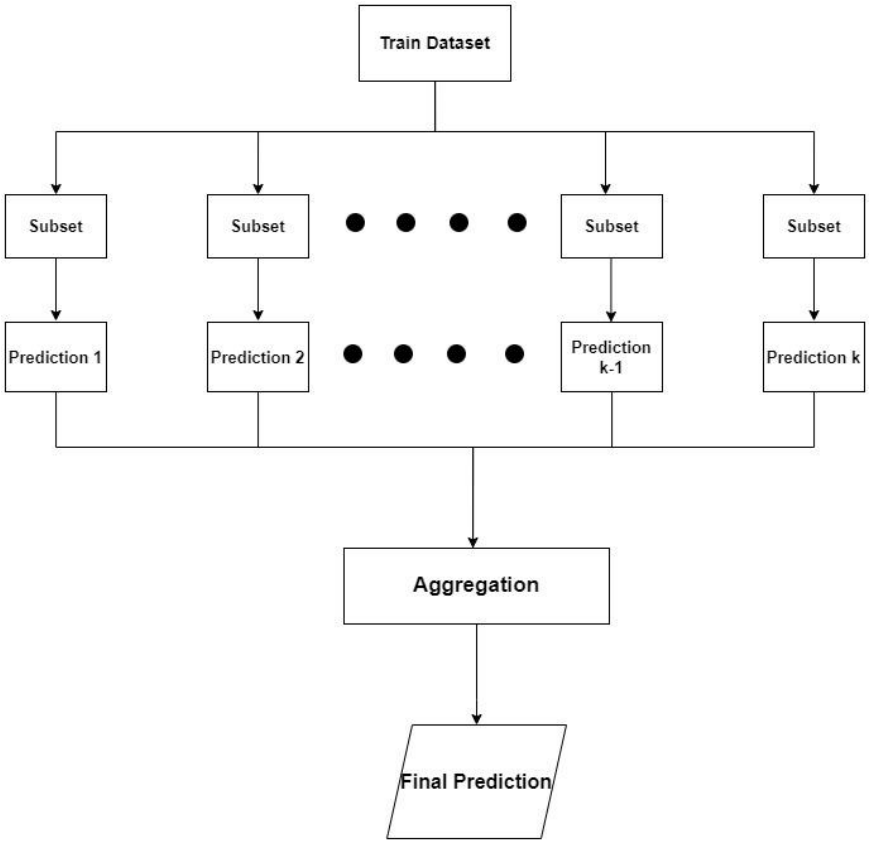


Figure 3.4-7: Bagging Ensemble Architecture

#### 3.4.5.2 Boosting Ensemble method

Boosting algorithm works like Bagging but it assigns a weight for each subset of training data to predict the classification performance on the previous iteration so that any data that were misclassified will be trained again in the next iteration [52]. The weight of each subset is modified on each iteration based on its performance. The weights of Misclassified results will be increased. The boosting goal is to improve the performance of the weak classification [53]. Different methods can be used in boosting such as AdaBoost, Multiboost, and Adaboost [50].

#### 3.4.5.3 Random Subspace Ensemble method

It works the same as bagging by creating a random set of training features, each set is trained individually, and then, all weak predictors/classifiers are aggregated together. The final predictor/classifier is chosen by a majority vote [54].

### 3.4.6 Radial Basis Function Neural Network (RBFNN)

RBF is a feedforward neural network with 3 layers; an input layer, one hidden layer, and an output layer. The activation function of hidden layers' neurons is the radial basis function [55]. RBF was introduced as a solution for the interpolation of data in multidimensional space. It's very effective to solve a problem that requires huge data. The main idea of RBFNN is about using the Euclidean distance equation to find the distance between each input to the center of the RBF neuron. In addition to the activation function of the hidden neurons, there's a Gaussian kernel function and a bias value. RBFNN calculates the Euclidean distance between input vector x and the center

of the Gaussian function. Then, the output of the hidden neuron will be multiplied by weight to form the output value as explained in (Eq. 3.4.6-1) [56].

$$y_k(\mathrm{x}) = \sum_{j=1}^{n_h} w_{kj} \, . \, h_j(\mathrm{x}) + b_k \qquad \text{(Eq. 3.4.6- 1)}$$

$h_j(\mathrm{x})$ can be calculated by (Eq. 3.4.6 -2) where $b_k$ is the bias, $\sigma_j$ is Gaussian function width, $c_j$ is the Gaussian function center and $w_{kj}$ is the weight for connection k and j.

$$h_j(\mathrm{x}) = \exp\left(-\|\mathrm{x} - c_j\|^2 \; / \; {\sigma_j}^2\right) \qquad \text{(Eq. 3.4.6- 2)}$$

### 3.4.7 Recurrent Neural Network (RNN)

RNN is a feedforward Neural Network with a feedback connection to each neuron with a time step. At time t, the neurons will receive input from themselves or from another neuron that contains a recurrent edge that could be a hidden neuron [57]. So, a feedback loop is added around each hidden neuron [58]. The output of each hidden neuron is the result of the activation function of the current state of the neuron with the hidden state of the neuron as shown in (Eq. 3.4.7- 1) where $B_u(t)$ is the current state of neuron, $A_z(t-1)$ is the hidden state of the neuron and $b_0$ is the bias value.

$$z(t) = f_n(A_z(t-1) + B_u(t) + b_0) \qquad \text{(Eq. 3.4.7- 1)}$$

There're many forms of RNN such as: "Long-Short term memory", Hopfield and Elman. RNN in all forms is very effective and can achieve superior accuracy in text detection, language identification, and other machine learning applications.

### 3.4.8 Multi-Layer Perceptron Neural Networks (MLPNNs)

Neural Networks are built to mimic the human brain by building a network of interconnected neurons over many layers together. Such networks can solve different hard regression, classification, and recognition problems efficiently [56]. Multi-Layer Perceptron is one of the Neural Networks that consists of the input layer, at least one hidden layer, and an output layer. All neurons are fully connected to the neuron of the previous layer and the next layer Figure 3.4-8. For each connection, a weight value is assigned, this value is multiplied by the neuron input to form the output of the next connected neuron.

MLPNN is like any feedforward neural network that must go through 2 steps in the learning process; step one is going forward by predicting the output and error value. Step two is going backward propagation by taking the error value and using it to modify the weight values associated with each connection. Those steps are repeated many times until the error is reduced [59]. Below is a full explanation with equations of one forward and backward iteration through the learning process of MLPNN.

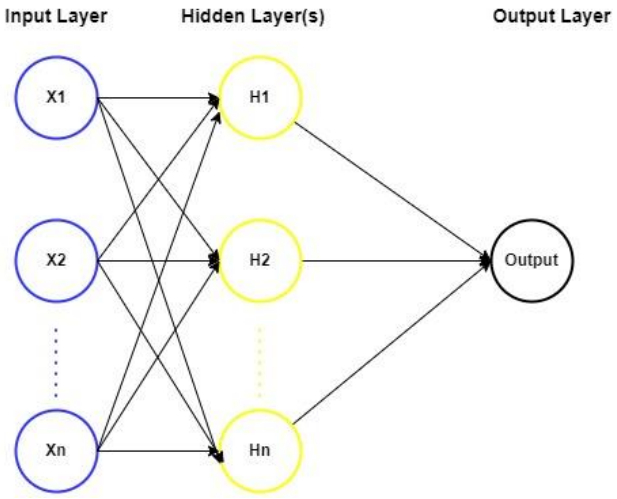


Figure 3.4-8: MLPNN Architecture

1. Data Entry: each neuron will receive an input from the training dataset, the corresponding target output values are also known to NN.
2. Process Data: input data is processed by multiplying the input by the connection weight value according to (Eq. 3.4.8-1).

$$Y_{ij} = f(\sum_{i=1}^{n} Xi * w_{ij}) \quad \text{(Eq. 3.4.8-1)}$$

$Xi$ is the input data, $w_{ij}$ is the weight of the connection between node $i$ and node $j$. $f$ is the activation function assigned for the neuron.

3. Calculate output: output value for the forward process is calculated according to the value of the last neurons connected to the output layer multiplied by the connection weight value.
4. Calculate error: error is calculated by subtracting the output value from the target value, if the error is above the threshold value, a backpropagation process is required to adjust connection weights. Mean Square Error (Eq. 3.4.8- 2):

$$MSE = \frac{1}{2}\sum_{i}^{n}(y_d - y_j)^2 \quad \text{(Eq. 3.4.8- 2)}$$

$y_d$ is the target output and $y_j$ is the actual output value.

5. Modify weights between the last hidden layer and output layer: connection weights are modified starting from the last layer in a backward direction until the first layer is reached. Weights are modified using (Eq. 3.4.8- 3).

$$\Delta w_{i+1} = \alpha . E . x_i \quad \text{(Eq. 3.4.8- 3)}$$

Where the weight between the last hidden layer and the output layer is modified by multiplying the error value $E$ by a constant value $\alpha$ and the input value $x_i$.

6. Calculate output difference: for the output layer, the difference between the actual output and target output is calculated by (Eq. 3.4.8- 4).

$$\Delta_n = t_n - Y_n \quad \text{(Eq. 3.4.8- 4)}$$

$Y_n$ is the actual output value and $t_n$ is the target output value.

7. Calculate the error signal for the output layer: error signal $\delta_n$ value for the output layer is calculated by (Eq. 3.4.8- 5).

$$\delta_n = \Delta_n Y_n (1 - Y_n) \quad \text{(Eq. 3.4.8- 5)}$$

8. Modify weights: weights of output layers n and j by calculating $\Delta w_{jn}$ using (Eq. 3.4.8- 6), then, adjust the weight using (Eq. 3.4.8- 7)

$$\Delta w_{jn} = l\, nX_n \quad \text{(Eq. 3.4.8- 6)}$$

$$w_{jn} = w_{jn} + \Delta w_{jn} \quad \text{(Eq. 3.4.8- 7)}$$

9. Calculate the error signal for the hidden layer: between hidden layers k and j, the error signal $\delta_j$ is calculated using (Eq. 3.4.8- 8)

$$\delta_j = (t_k - Y_k) Y_k \sum w_{jk}\, \delta_n \quad \text{(Eq. 3.4.8- 8)}$$

10. Modify weights: weights of hidden layer k and hidden layer j by calculating $\Delta w_{jk}$ using (Eq. 3.4.8- 9), then, adjust the weights using (Eq. 3.4.8- 10)

$$\Delta w_{jk} = l\ \delta_j X_j \quad \text{(Eq. 3.4.8- 9)}$$

$$w_{jk} = w_{jk} + \Delta w_{jk} \quad \text{(Eq. 3.4.8- 10)}$$

The backpropagation process is repeated until the error is reduced.

- **Second Part**

In this part, we will explain the Deep Learning model based on a convolutional neural network (CNN) implemented by the (VGG-16) and (VGG-19) approaches.

### 3.4.9 Convolutional Neural Network (CNN)

The wide variety of machine learning research has evolved valuable solutions in different aspects of life. Moreover, deep learning is known for its outstanding performance in image, audio, and natural language processing applications [60]. DL is preferred for its several features such as:

- Universality: DL is being used to solve different problems and find solutions for various applications and domains.
- Generalization: same DL method, algorithms, and techniques can work with different data types and applications.
- Scalability: DL networks can be improved to powerful networks such as ResNet.

Most DL network used now is Convolutional Neural Network (CNN) for their power in extracting the features without any previous intervention [61]. CNN is involved in the applications of speech processing, image classification, pattern recognition, and face recognition. CNN works with two-dimensional data, this data is then organized into 3 dimensions; width, height, and depth. The CNN consists of two different layer architectures; feature extraction layers and classification layers. Such networks are built with a very large number of interconnected layers to mimic human neurons. During the training of the CNN network against the training data, the detection of the target class is recognized while data is going through the network's hidden layers connected using activation functions. Training CNN is computationally exhaustive because of is rich in image input data points, weights, and features. As a result, the training is done in different and multiple steps or as it’s called ‘epochs’ and the data is divided according to the ‘batch’ value for system memory usage purposes. So, training data is divided into batches, each batch is trained in an epoch until all data is trained by the CNN model. Each epoch consists of several iterations that are calculated by the number of training input data points divided by batch size [62]. Figure 3.4-9 shows the structure of the CNN model in two parts; the first part is the feature extraction consisting of the input layer, convolutional layer, and pooling layer, and the second part is the classification part that consists of a network of connected layer and output layer. The input layer is an image with a fixed size and it can be modified to a size that gives the best accuracy. The image is then moved to a convolutional layer that has a kernel function. The image size is then reduced by the pooling layer. The feature extraction part will produce a matrix of features for the image data, this matrix will go into a network of fully connected layers to combine all features, and the category/class for

each feature matrix is represented by an output neuron [63]. Detailed steps of CNN are explained below.

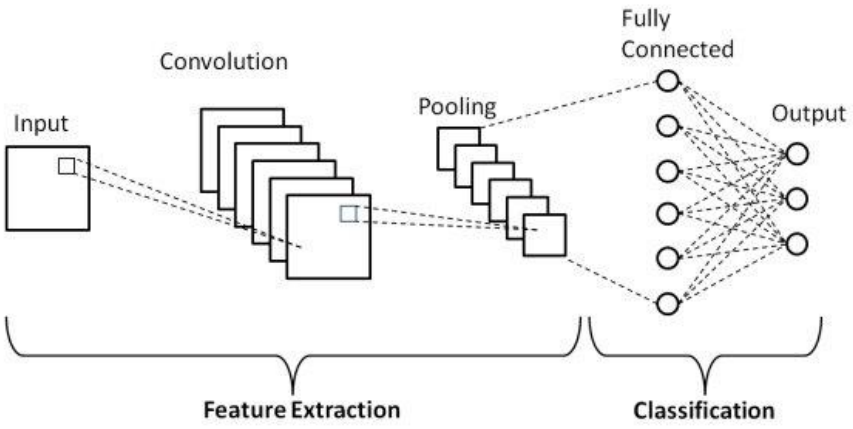


Figure 3.4-9: General Structure of CNN [69]

*The General Steps of the CNN model:*

1. **Convolution**

For each convolutional layer, a kernel function is defined to extract the features from input data images. Multiple convolutional layers with different kernel functions will extract different features from input data images which emphasize the diversity of features when the number of convolutional layers with kernel functions is increasing. This layer is also called the up-sampling layer [64]. In the convolutional process, a small sample of input data image will be multiplied by a matrix of the convolutional kernel function. That process will be repeated until the whole image is scanned. The result of multiplication will be inserted as a new matrix at the end of the process as shown in Figure 3.4-10.

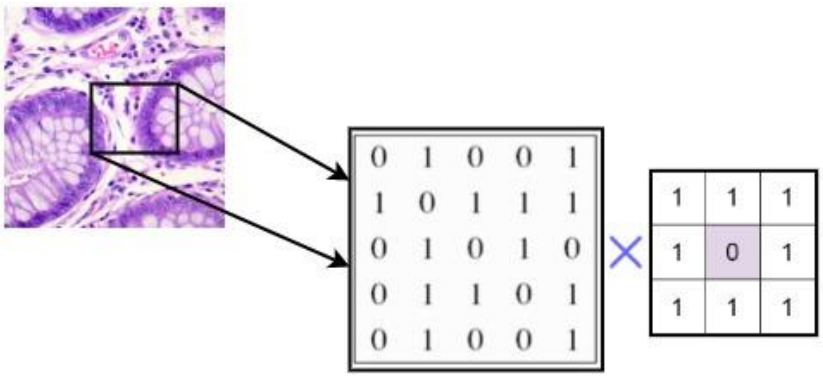


Figure 3.4-10: Convolution Process

A layer of zero values will be added at the convolutional step outside the image array. This layer is added in the image frame as shown in Figure 3.4-11 to extend the area where the kernel function is moving to cover the full image area which will help achieve better detection accuracy.

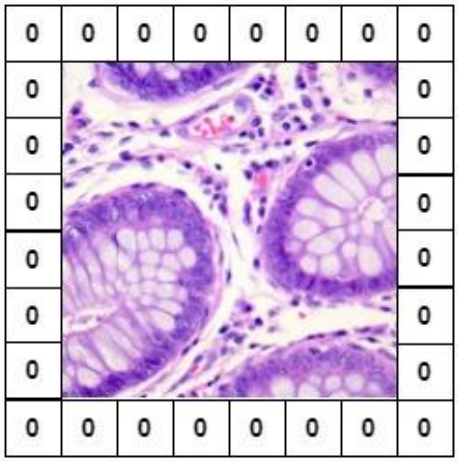

Figure 3.4-11: Padding Process

The activation function is another important element in the convolutional process as it transforms the input data image into a certain range of values, in other words, the values will be scaled. The input values of the network are scaled by normal, then, when stepping forward in the network, the input values will be multiplied by the weights which will increase the values into very larger values. In this condition, the values need to be scaled into an acceptable range using the activation function. As a result, the next layers will be affected by those changes which in the end will determine the output

value of the network. So, the activation function must be a non-linear function and differentiable such as the sigmoidal function and tan function [65].

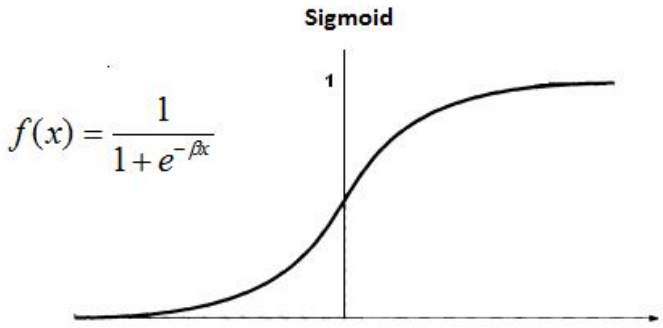


Figure 3.4-12: Sigmoidal function

**2. Pooling**

In this layer, a pooling function is used to replace the network output values with summary values to decrease the number of extracted and repeated features, input values, and weights that will eventually decrease the computational power. The pooling layer is moving in slices to cover the whole specific output value but the operation is done individually in each slice. Different pooling functions can be used in this layer, max and average functions are the most popular. The Max function will extract the maximum number in each 2*2 rectangle slice and move that value into a feature map to form the output of the pooling layer as shown in Figure 3.4-13 [66].

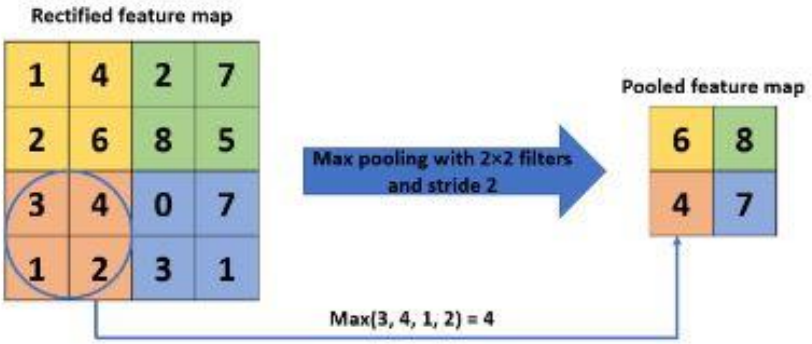


Figure 3.4-13: Max Pooling function in CNN [66]

### 3. Flattening

After having featured maps from the pooling layer, this data should be inserted into a connected neural network, so it should be converted to a column of data instead of a feature map as shown in Figure 3.4-14. The collection of feature maps will be converted into a collection of columns. Each column represents a feature map as shown in Figure 3.4- 15.

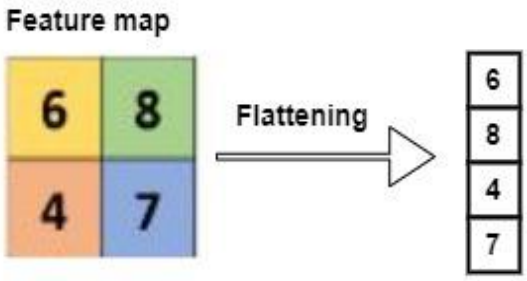


Figure 3.4-14: Flattening of a feature map

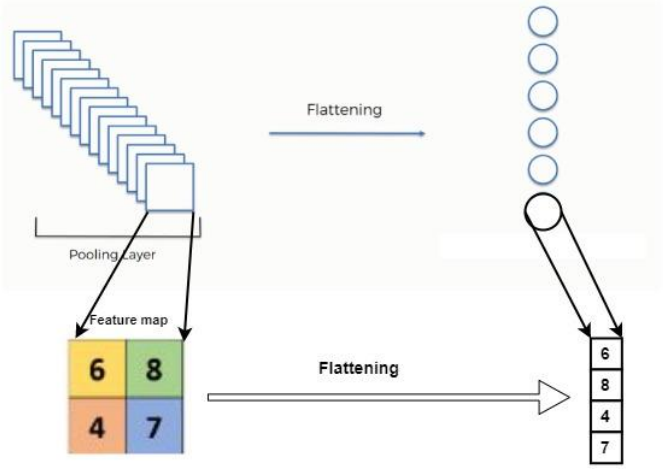


Figure 3.4-15: Flattening in CNN

### 4. Fully-Connected Layer

At the end of the CNN network, fully connected layers are added to take the data from the flattening layer and strengthen the network classifier. These interconnected layers are much like a Multi-Layer perceptron feedforward neural network. The input for the fully-connected layer is the vectors from the flattening layer and the output is the

final CNN output as shown in Figure 3.4- 16 [60]. This layer acts as a representative between the input layer and the output layer.

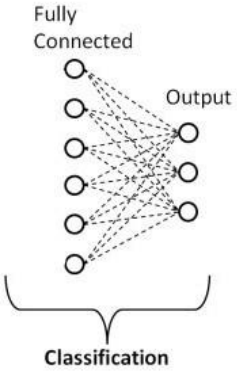


Figure 3.4-16: Fully-connected Layer in CNN

Convolutional Neural Network (CNN) is meant to process image data which is represented as binary data called pixels arranged in a grid. The architecture of CNN is arranged in a way to extract special features in image data starting from the convolutional layer, pooling layer, and fully connected layer as shown in Figure 3.4- 17. In this thesis, the CNN model was built using the Keras library in Python programming language, it consists of a convolutional layer of 32 and is followed by a Pooling layer of size = 2*2. Another Convolutional layer of 32 and Pooling layer of size = 2*2 is followed. The last layers are the flattening layer and two dense layers as shown in Figure 3.4- 18.

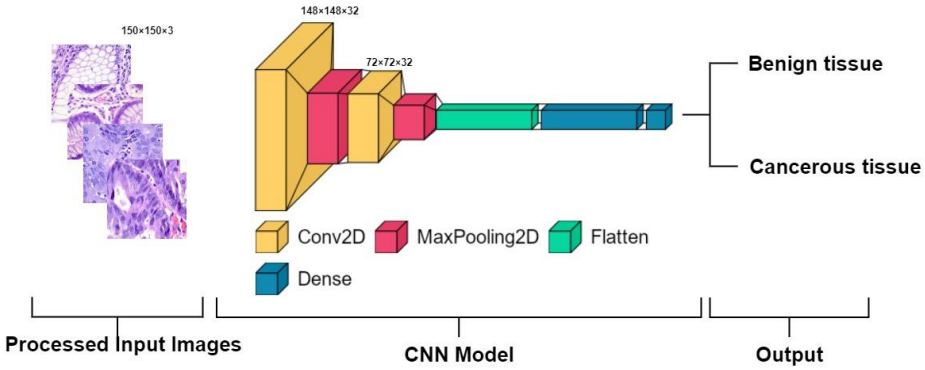


Figure 3.4-17: CNN model architecture

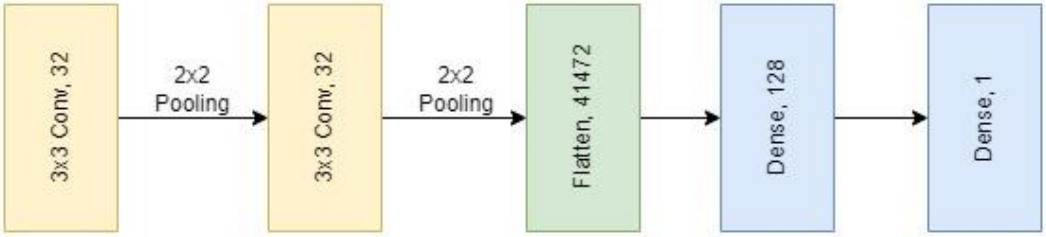


Figure 3.4-18: Stages of CNN model

The below Table 3.4- 1 shows the details of the CNN model.

**Table 3.4-1: Convolution Layers Implementation of CNN model**

| Layer | Convolution | Conv. Output Dimension | Pooling | Pooling Output Dimension |
|---|---|---|---|---|
| Layer 1 | Convolution Layer of 32 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 148*148*32 | Max Pool, Stride = 2 , Size = 2*2 | 74*74*32 |
| Layer 2 | Convolution Layer of 32 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 74*74*32 | Max Pool, Stride = 2 , Size = 2*2 | 36*36*32 |

The flattening Layer is then followed to transform the images from a two-dimensional array to a one-dimensional array of 41472 pixels. Two dense fully connected layers are then followed to take the flattened data, the first dense layer has 128 neurons and the second dense softmax layer has 1 neuron to return the classification output.

### 3.4.10 Visual Geometry Group-16 Model (VGG-16)

VGG-16 stands for Visual Geometric Group and it's another architecture from the CNN model that was introduced by K. Simonyan and A. Zisserman from the University of Oxford [67]. VGG-16 model contains more than 16 million parameters and 16

convolutional layers distributed in 3 groups. Each convolutional group contains three 3×3 convolutional layers followed by one 2×2 max pooling layer. Two fully connected layers are then added with 4096 hidden neurons with a “ReLU” activation function and one fully connected layer with 1000 hidden neurons [68][69]. A final dense layer is added with softmax activation function for predication purposes as shown in Figure 3.4.19. In the VGG-16 model, ImageNet pre-trained model was used to transform the weights into our model. Figure 3.4- 20 shows the stages of a VGG-16 described above.

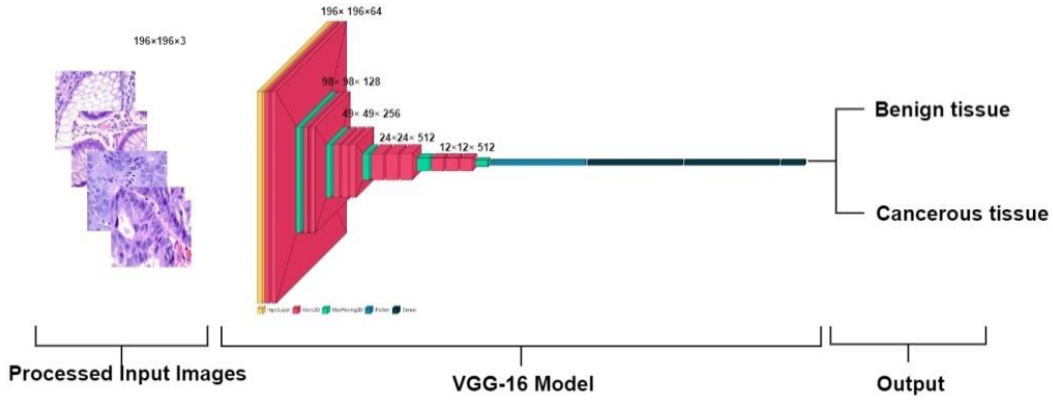


Figure 3.4-19: VGG-16 Architecture

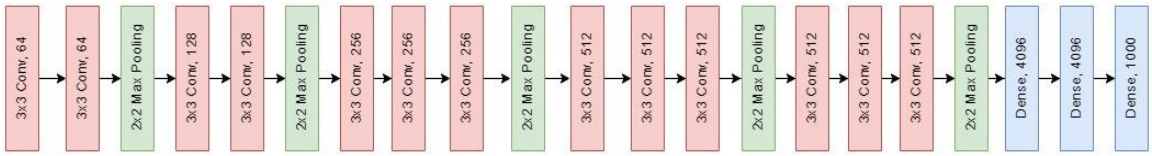


Figure 3.4-20: Stages of VGG-16 model

Table 3.4- 2 shows the details of VGG-16 stages including Input/output dimensions for each layer.

**Table 3.4-2: Convolution Layers Implementation of VGG-16 model**

| Stage | Layer | Convolution | Conv. Output Dimension | Pooling | Pooling Output Dimension |
|---|---|---|---|---|---|
| First | Layers 1&2 | Convolution Layer of 64 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 196*196*64 | Max Pool, Stride = 2 , Size = 2*2 | 98*98*64 |
| Second | Layers 3 &4 | Convolution Layer of 128 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 98*98*128 | Max Pool, Stride = 2 , Size = 2*2 | 49*49*128 |
| Third | Layers 5, 6, 7 | Convolution Layer of 256 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 49*49*256 | Max Pool, Stride = 2 , Size = 2*2 | 24*24*256 |
| Fourth | Layers 8, 9, 10 | Convolution Layer of 512 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 24*24*512 | Max Pool, Stride = 2 , Size = 2*2 | 12*12*512 |
| Fifth | Layers 11, 12, 13 | Convolution Layer of 512 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 12*12*512 | Max Pool, Stride = 2 , Size = 2*2 | 6*6*512 |

The fifth stage is then followed by 2 fully connected dense layers of 4096 neurons and then a dense softmax layer to produce the classification result.

### 3.4.11 Visual Geometry Group-19 Model (VGG-19)

Visual Geometric Group 19 is another CNN model that was introduced by K. Simonyan and A. Zisserman from the University of Oxford [67]. VGG-19 is a very efficient model for image classification and feature extraction. This model contains 19 layers including convolutional layers, Max pooling layers, fully connected layers, and dropout layers. 3×3 filters were used for convolutional layers with the “ReLU”

activation function, then, they were followed with max-pooling layers with size = 2×2. The dropout layer is used in the training process to overcome the overfitting problem by converting some activation functions to zeros. Figure 3.4- 21 shows the architecture of the VGG-19 model and Figure 3.4- 22 shows the stages of VGG-19 layers. ImageNet pre-trained model was used and transformed its weights to our model, so only fully connected layers need to be trained [70] [71].

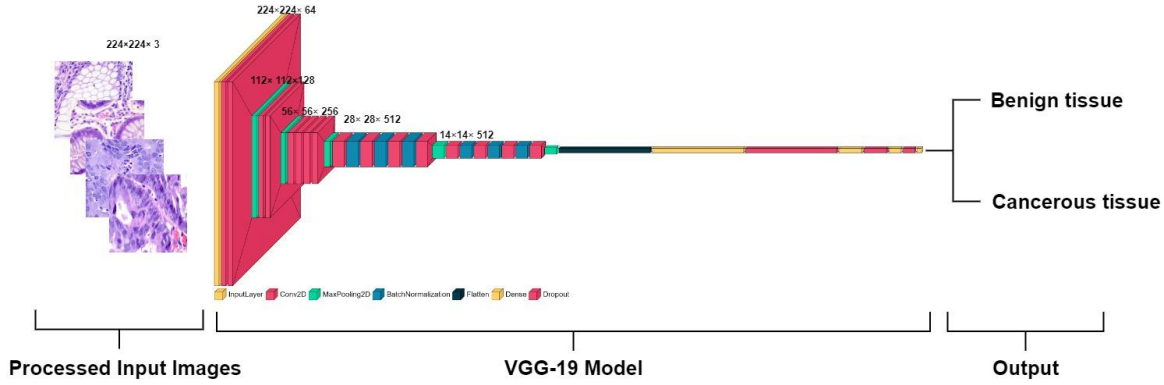


Figure 3.4-21: VGG-19 Architecture

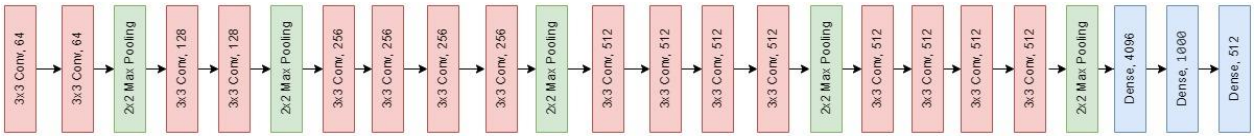


Figure 3.4- 1: Stages of the VGG-19 model

Table 3.4- 1 shows the details of VGG-19 stages including Input/output dimensions for each layer.

**Table 3.4-3: Convolution Layers Implementation of the VGG-19 model**

| Stage | Layer | Convolution | Conv. Output Dimension | Pooling | Pooling Output Dimension |
|---|---|---|---|---|---|
| First | Layers 1&2 | Convolution Layer of 64 Channel of (3*3) Kernel With padding = | 244*244*64 | Max Pool, Stride = 2 , Size = 2*2 | 112*112*64 |

| | | | | | |
|---|---|---|---|---|---|
| | | 1, Stride = 1 | | | |
| Second | Layers 3 &4 | Convolution Layer of 128 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 112*112*128 | Max Pool, Stride = 2 , Size = 2*2 | 56*56*128 |
| Third | Layers 5, 6, 7,8 | Convolution Layer of 256 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 56*56*256 | Max Pool, Stride = 2 , Size = 2*2 | 28*28*256 |
| Fourth | Layers 9, 10,11,12 | Convolution Layer of 512 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 28*28*512 | Max Pool, Stride = 2 , Size = 2*2 | 14*14*512 |
| Fifth | Layers 13,14,15,16 | Convolution Layer of 512 Channel of (3*3) Kernel With padding = 1, Stride = 1 | 14*14*512 | Max Pool, Stride = 2 , Size = 2*2 | 7*7*512 |

The fifth convolutional layer is followed by three fully connected layers with 4096, 1000, and 512 hidden neurons respectively. The last dense layer with 2 neurons and softmax activation function produce the classification result.

## 3.5 Performance Metrics Selection

Measuring the performance of Machine Learning techniques is an essential part of the ML pipeline. For each ML model, the performance is evaluated by measuring the difference between the actual model output with the target output values from the test dataset [81]. The most important performance metric to assess the model is precision, accuracy, sensitivity, specificity, f-score, confusion matrix, and receiver operator area under curve. Those metrics are all shown for all machine learning models and deep

learning models used in this thesis: Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Radial Basis Function, Recurrent Neural Network, Multi-Layer Perceptron Neural Network, CNN, VGG-16, and VGG-19. Accuracy, precision, sensitivity, and specificity metrics are calculated from the values extracted from the confusion matrix. The confusion matrix consists of four parameters True-positive, true-negative, false-positive, and false-negative which reflect the model prediction performance, they're used to calculate the other performance metrics. We started with an explanation of the confusion matrix and its parameters, and other metrics explanations followed.

- **Confusion Matrix:** A Confusion matrix is a two-dimensional array where rows show the actual prediction and columns show the predictions [72]. It's a matrix of n×n that shows the number of data predicted as one of four outcomes as shown in Figure 3.5- 1:
  - True-Positive (TP): indicates how many classes were predicted correctly by a model, which means the class is true and it was predicted as true by the model.
  - True-Negative (TN): indicates the number of false classes which were predicated as false by the model.
  - False-Positive (FP): indicates the number of false classes predicted incorrectly by the model, which means the class is false but was predicted as true by the model.
  - False-Negative (FN): indicates the number of true classes predicted incorrectly by the model, which means the class is true but was predicted as false by the model.

The summation of TP, TN, FP, and FN reflects the total number of the input dataset. To get the high accuracy of the model, the number of TP and TN should be large. Also, a lower FP and FN mean a lower error ratio.

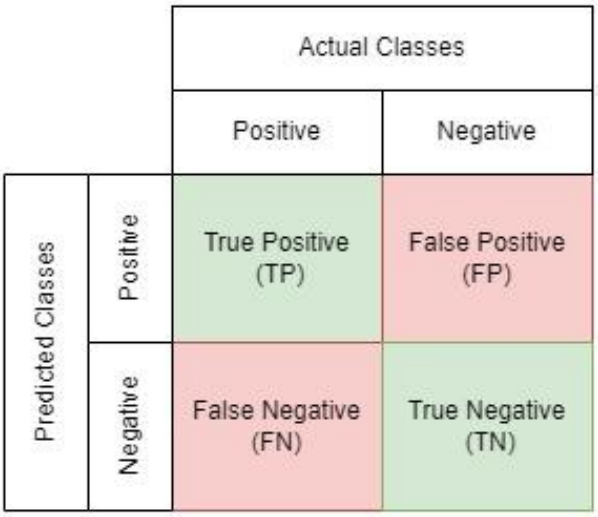


Figure 3.5-1: Confusion Matrix

- **Accuracy:** measures the model predication rate, suppose that we have a dataset of N classes, Np is the number of TP and TN correct predicted classes, and accuracy is the percentage of correct predicated classes over the total number of the dataset. It's calculated by Eq. (3.5- 1)

$$Accuracy = \frac{Np}{N}$$

- **Sensitivity/Recall:** measures the true positive rate predicted by the model during the test, it's calculated by Eq. (3.5- 2) [73].

$$Sensitivity = \frac{TP}{TP + FN}$$

- **Precision:** measure the rate of true positive predictions over the total positive predictions. Precision values are close to one means the model was able to correctly predict almost all true positive classes. It's calculated by Eq. (3.5- 3)

$$Precision = \frac{TP}{TP+FP}$$

- **Specificity:** measures the rate of true negative classes predicted correctly by the model. It's calculated by Eq. (3.5- 4) [73]

$$Specificity = \frac{TN}{TN+FP}$$

- **F-Score:** measures the accuracy of binary classifiers using sensitivity and precision values. The f-score is close to 1 means the classifier has the perfect precision and recall values. Zero f-score means the classifier has a recall value of zero or precision is zero.

$$F - Score = 2 \times \frac{Precision \times Recall}{Precision+Recall}$$

- **Receiver Operating Characteristic Curve (ROC):** it's also known as the AUC curve that ranges between 0 and 1. It's used to visualize the performance to plot the true-positive rate (TPR) and false-positive rate (FPR) across the data points. The Best classifier result is achieved when the curve points are in the upper left corner, i.e. in (0,1) coordinates of ROC space. ROC diagonal divides the curve into two areas, upper and lower. The upper area contains the good predictions and the lower area contains the bad predictions as shown in Figure 3.5- 2.

Classifiers with high AUC are considered better than other classifiers with lower AUC.

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

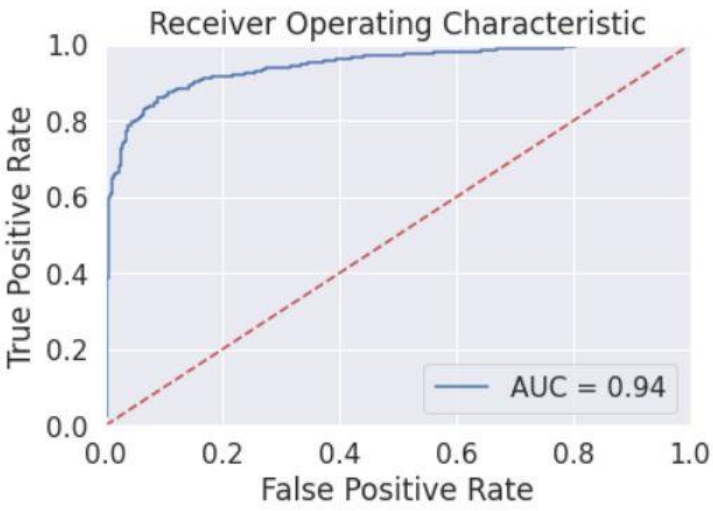


Figure 3.5-2: ROC

# CHAPTER 4

# EXPERIMENTS AND RESULTS

## 4.1 Experiments and Results

In this chapter, all machine learning and deep learning classification models that were explained in the previous chapters are applied against the datasets, reviewed, and discussed in detail. This chapter is organized into two parts; the first part is for the experimental of machine learning models (Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Radial Basis Function, Recurrent Neural Network, and Multi-Layer Perceptron Neural Network) on the local dataset one time for the data with features selected by experts from the domain as done by Abu Zuhri et. al. [30], and then with local dataset after feature selection using mutual information. The accuracy, sensitivity, precision, specificity, and F-Score for all models are calculated, Confusion matrix and Receiver Operation Curve (ROC) are illustrated. Based on those performance results, the best machine learning classification model on a Colon cancer local dataset was recommended. The second part is for the experimental of deep learning classification models (CNN, VGG-16, and VGG-19) against global colon cancer tissue images from Kaggle. For the global dataset, the accuracy, sensitivity, precision, specificity, and F-Score for all models are also calculated, Confusion matrix and Receiver Operation Curve (ROC) are illustrated. Based on those performance results, the best deep learning classification model was recommended, and the models are built over Python programming language for the global colon cancer tissue image dataset.

## 4.2 Computing Environment

The computing environment for the first part of the experiments on the local dataset was done by Lenovo ThinkPad Intel Core i5-8250U CPU 1.60GHz 1.80GHz, 16GB RAM, 240GB SSD with Windows 10 Pro Operating System. Experiments were applied to MATLAB R2018b. The second part of the experiments on the global colon cancer tissue images dataset was done using a virtual machine obtained from Google Cloud Platform to get a powerful computational power, the virtual machine is e2-highmem-8 (Efficient Instance 8 CPUs, 64 GB RAM), None GPU with Debian 10 Operating System. Experiments were applied on JupyterLab with Python 3 environment that includes scikit-learn, pandas, TensorFlow, Keras libraries, and more.

## 4.3 Deep Learning Practical Experiments for Global Tissue Images Dataset

In this section, three different deep learning models were used to classify colon tumor tissue images as benign tissue or cancerous tissue. The results are split into 3 subsections: the first subsection is for the CNN model, the second subsection is for the VGG-16 model and the last subsection is for the VGG-19 model. From LC25000 global dataset, 7593 images were taken and they were divided between benign tissues and cancerous tissues. Dataset was divided into the following: 5313 images for training, 1140 images for validation, and 1140 images for testing. All models were trained with the same epoch number = 20 and the same batch size = 64. For all models, the learning rate was set to 1e-5. In deep learning experiments, each model was trained four times,

each run with a different image size to find which size will achieve the best accuracy, training, validation, and testing images were all resized according to each experiment.

### 4.3.1 Deep Learning Practical Experiment for CNN model

The Convolutional Neural Network model was built using Python programming language by JupyterLab. The training and testing process was repeated four times with different image sizes to evaluate its performance and decide which image size is the best for CNN with the LC25000 dataset. The table below shows the test accuracy for different image sizes. Using image size = 224×224 with a CNN model to classify Colon cancer tissue images is the best. Figure 4.3-1 shows the CNN model training accuracy progress during the epochs for each image size. As seen, the CNN model is performing better in terms of training accuracy increasing when the image size is set to 224×224.

**Table 4.3-1: CNN model accuracy with different image sizes**

| Image Size | Accuracy |
|---|---|
| 224×224 | 83.94% |
| 150×150 | 69.82% |
| 148×148 | 82.89% |
| 196×196 | 77.45% |

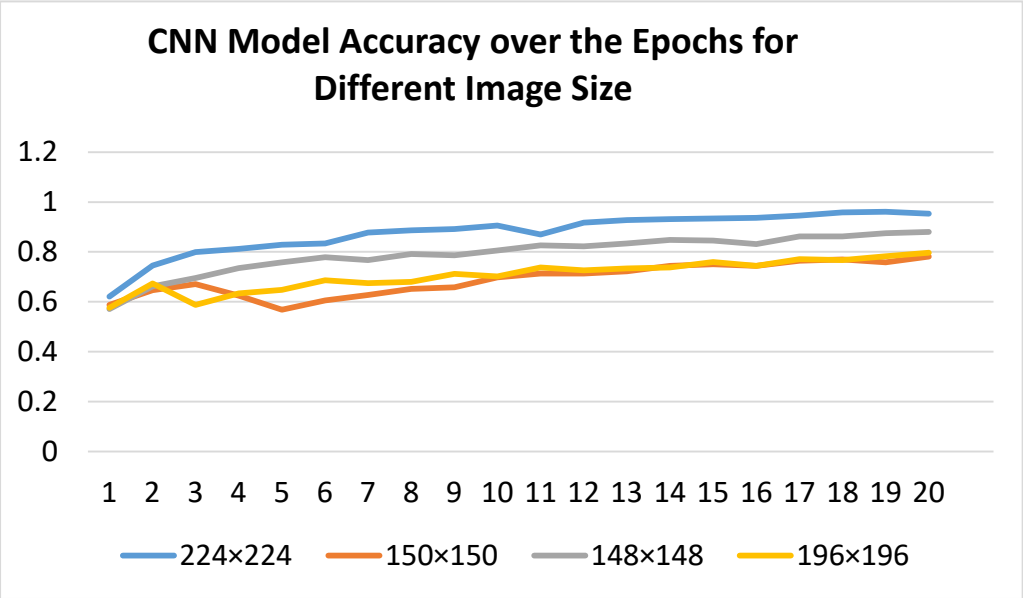


Figure 4.3-1: CNN model training accuracy over the epochs for different image size

During the training process of the CNN model when the image size is set to 224×224, the learning was fast as seen in Figure 4.3-2 the training loss is decreasing rapidly while the training and validation accuracy are increasing until it stopped learning between epochs 17 and 20 (numbers in the chart are not multiplied by 100 to show the percentage).

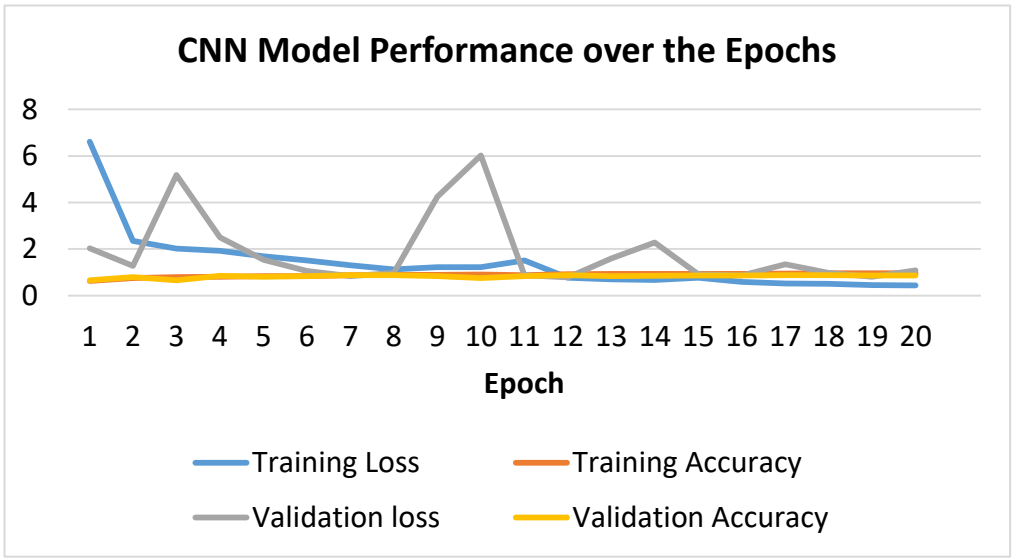


Figure 4.3-2: CNN model performance over the epochs when image size is 224×224

In summary, CNN achieved a testing accuracy of 83.94%. Figure 4.3-3 below shows the confusion matrix and ROC for the CNN model with a Precision of 77%, Sensitivity of 96%, Specificity of 61%, and f-score of 85% obtained from the confusion matrix after specifying the TP, TN, FN, and FP values.

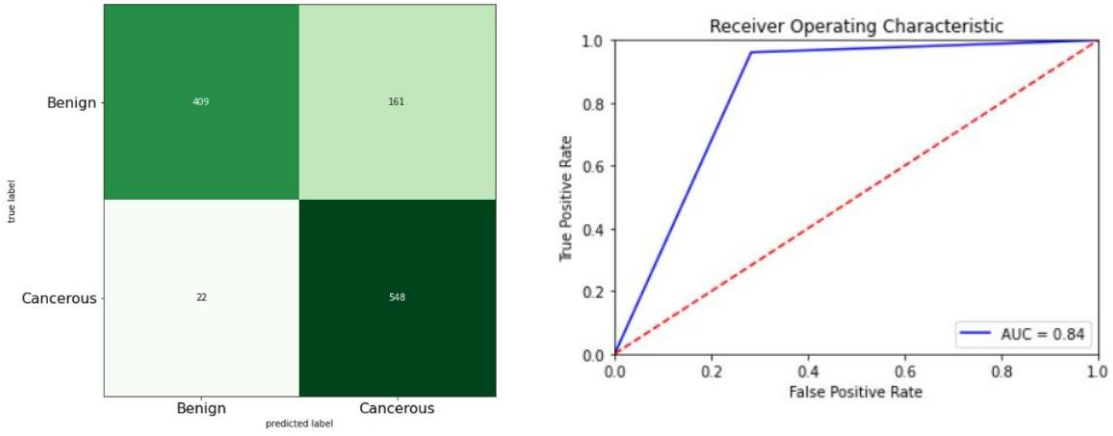


Figure 4.3-3: Confusion Matrix and ROC for CNN model when image size is 224×224

### 4.3.2 Deep Learning Practical Experiment for VGG-16 model

VGG-16 model was built using Python programming language using JupyterLab. VGG-16 model can take different image sizes, according to the studies there's no optimal image size, the performance of the mode depends on the size of the model and dataset. So, the model was trained and tested four times with image sizes 224×224, 196×196, 148×148, and 150×150 to compare model performance. As shown in Table 4.3-2, the VGG-16 model achieved the best accuracy of 97.54 % when using image size = 196×196 to classify Colon cancer tissue images.

**Table 4.3-2: VGG-16 model accuracy with different image sizes**

| Image Size | Accuracy |
|---|---|
| 224×224 | 60.78% |
| 150×150 | 97.28% |
| 148×148 | 96.14% |
| 196×196 | 97.54% |

During the training process, the training and validation loss for VGG-16 when the image size is 196×196 quickly decreased as the accuracy increased until it reached 99% at epoch = 20 as shown in Figure 4.3-4.

According to our results, the VGG-16 model is a powerful model to classify colon cancer tissue with an accuracy of 97.54%, Precision, Sensitivity, Specificity, and F-score of 98%. Those values are obtained from the confusion matrix and ROC is shown in Figure 4.3-5.

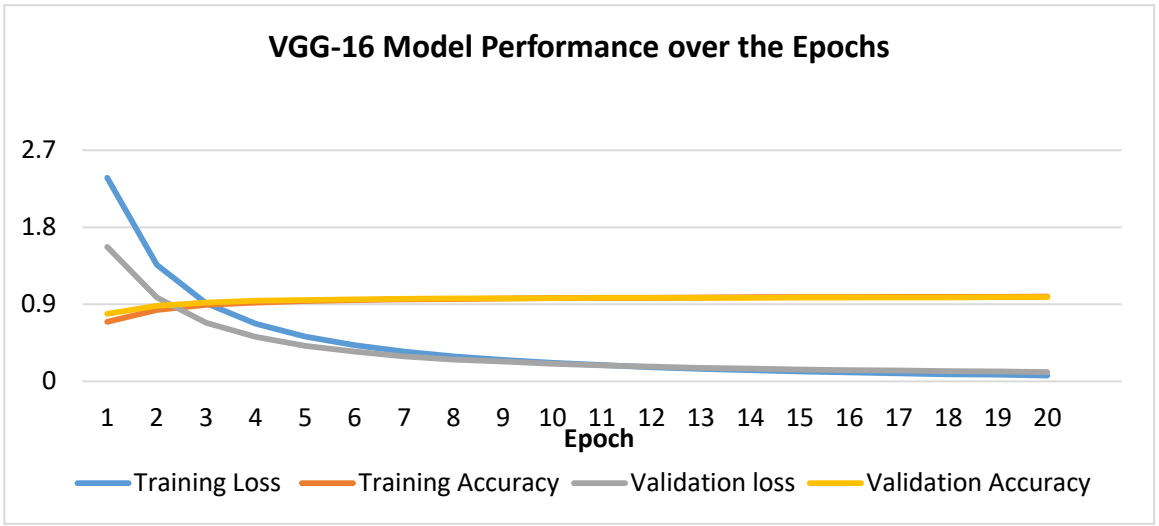


Figure 4.3-4: VGG-16 model performance over the epochs when image size is 196×196

.

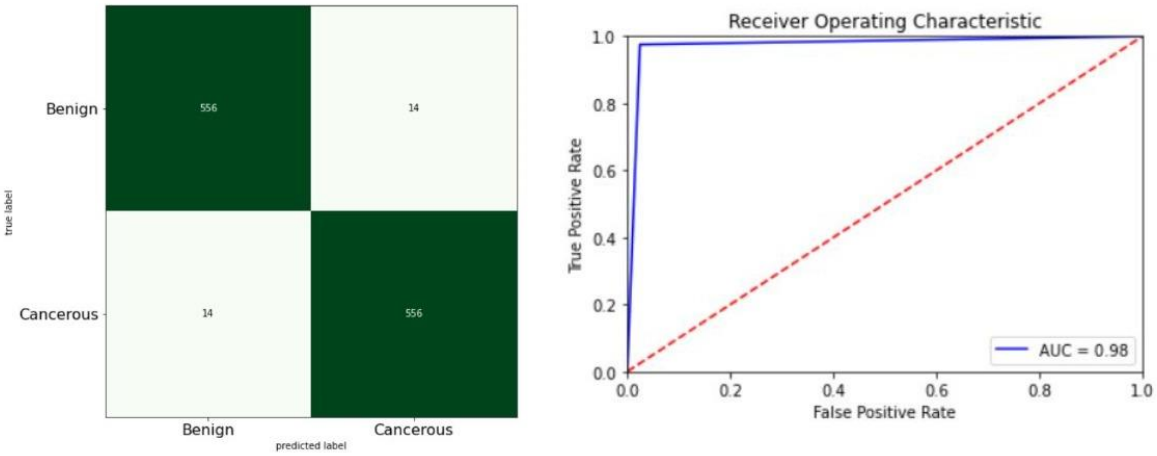


Figure 4.3-5: Confusion Matrix and ROC for VGG-16 model when image size is 196×196

### 4.3.3 Deep Learning Practical Experiment for VGG-19 model

Same as CNN and VGG-16, the VGG-19 model was built also using Python programming language and JupyterLab. The model was applied four times with different image sizes; 224×224, 196×196, 148×148, and 150×150 to compare model performance. As shown in Table 4.3-3, the VGG-19 model achieved the best accuracy of 98.94% when using image size = 224×224 to classify Colon cancer tissue images.

Table 4.3-3: VGG-19 model accuracy with different image sizes

| Image Size | Accuracy |
|---|---|
| 224×224 | 98.94% |
| 150×150 | 98.24% |
| 148×148 | 94.99% |
| 196×196 | 97.98% |

During the training of the model when the image size is 224×224, validation accuracy and loss were fluctuating between epochs 12 and 15 and also between epochs 17 and 19 causing overfitting and that's obvious as the loss increased on those epochs as shown in Figure 4.3-6.

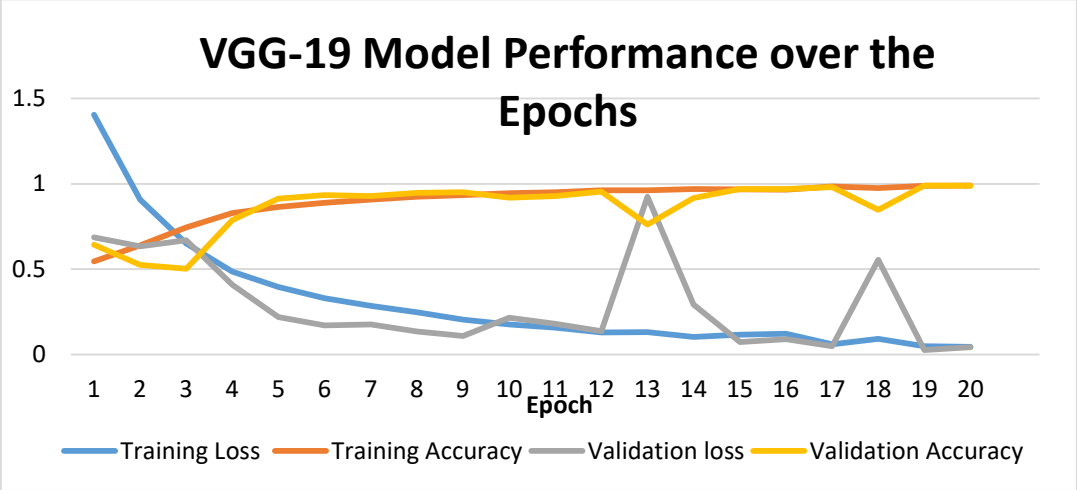


Figure 4.3-6: VGG-19 model performance over the epochs when image size is 224×224

According to our results, the VGG-19 model is also a powerful model to classify Colon cancer tissue with an accuracy of 98.94%, Precision of 100%, Sensitivity of 98%, Specificity of 100%, and F-score of 99%. Those values are obtained from the confusion matrix and ROC is shown in Figure 4.3-7.

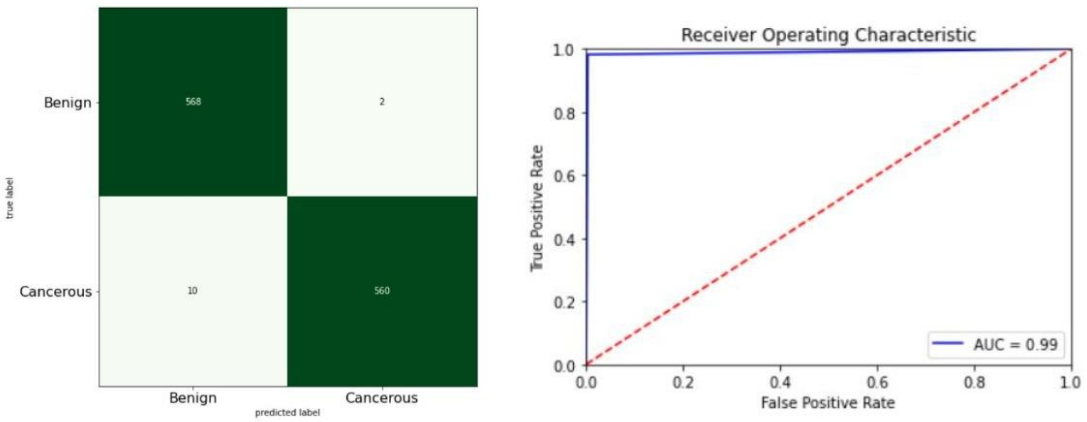


Figure 4.3-7: Confusion Matrix and ROC for VGG-19 model when image size is 224×224

In summary, many types of researchers have proposed different models to classify Colon cancer images using different deep learning algorithms and different medical datasets. As shown in Table 4.3-4 compares the performance of the proposed models that were used in this thesis. Figure 4.3- 8 illustrates the accuracy of the three models and based on that, the VGG-19 model is recommended to classify colon cancer tissue images (LC25000) dataset into a benign or cancerous image with the best accuracy achieved = 98.94% and minimum loss. VGG-19 is the best for our study with the LC25000 dataset of tissue images because it contains a huge number of images. VGG-19 model performs well with huge datasets for a large number of layers it has.

**Table 4.3-4: Best performance results for all models**

| Model | Accuracy | Sensitivity/Recall | Precision | Specificity | F-Score |
|---|---|---|---|---|---|
| CNN | 83.94% | 96% | 77% | 61% | 85% |
| VGG-16 | 97.54% | 98% | 98% | 98% | 98% |
| VGG-19 | 98.94% | 99% | 100% | 100% | 99% |

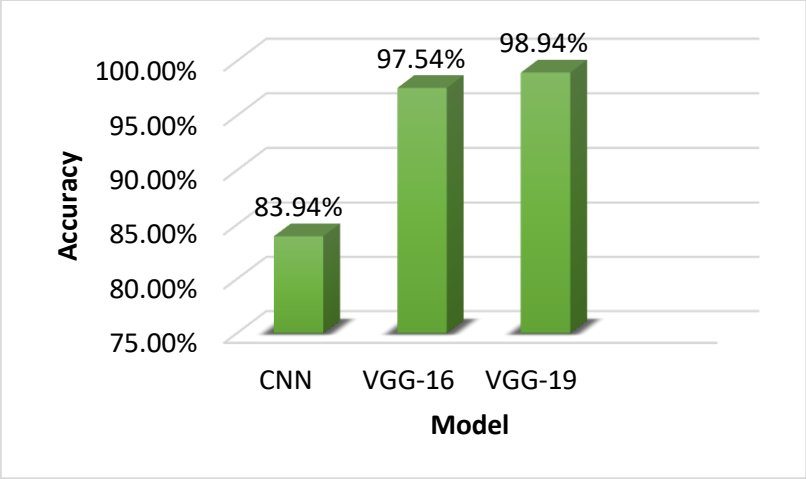


Figure 4.3-8: Accuracy results for all models

## 4.4 Machine Learning Practical Experiments for the Local Dataset

In this experiment, a local dataset was applied against eight machine learning algorithms; Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), K-nearest neighbor (KNN), Ensemble, Radial Basis Function, Recurrent Neural Network, and Multi-Layer Perceptron Neural Network. The local dataset from MoH has 350 records, it was divided into training data a testing data. For all classification algorithms, the models were built several times starting from 10 neurons and up to 50 neurons. Then, for each model, and each number of neurons, the testing accuracy, testing precision, testing sensitivity/recall, testing specificity, and testing f-score were calculated. The best performance result is then considered. The results are divided into subsections starting from classifier results, then, Recurrence Neural Network results, Multi-Layer Perceptron results, and the final subsection for Radial Basis Function results.

### 4.4.1 Classifiers Experiment Results on a Local Dataset

Starting from the results of the classifier, which are: Support Vector Machine, Naïve Bayes, Decision Tree, Ensemble, and KNN. Cross-validation of 5 folds is used to assess the accuracy of the classifier by dividing the local dataset that contains 244 records into 5 folds, each fold contains an equal size of records, and in our case, each fold contains 49 records. The training process will be repeated five times starting by training using four folds and validating the model using the remaining fold with unseen data. Table 4.4-1 shows the performance metric for all classification algorithms using the local dataset.

**Table 4.4-1: Classification training results for local dataset**

| Classifier | Training Accuracy | Training Recall | Training Precision | Training Specificity | Training F-Score |
|---|---|---|---|---|---|
| SVM | 88.5% | 94.6% | 92.5% | 79.9% | 93.5% |
| **NV** | **100%** | **100%** | **100%** | **100%** | **100%** |
| DC | 94.3% | 96.4% | 95.3% | 89.0% | 95.8% |
| **Ensemble** | **98.8%** | **100%** | **98.2%** | **95.7%** | **99.1%** |
| KNN | 88.5% | 89.2% | 93.9% | 83.0% | 91.5% |

The naïve Bayes algorithm achieved a superior training accuracy of 100% then, the Ensemble algorithm achieved the second-best training accuracy of 98.8% close to the Decision Tree achieved 94.3%. The training sensitivity/recall values for NB and Ensemble are the same at 100% and the DC algorithm achieved a good training recall value of 96.4%. SVM and KNN got the same training accuracy of 88.5%. In Figure, the training performance of the above classification algorithms is illustrated.

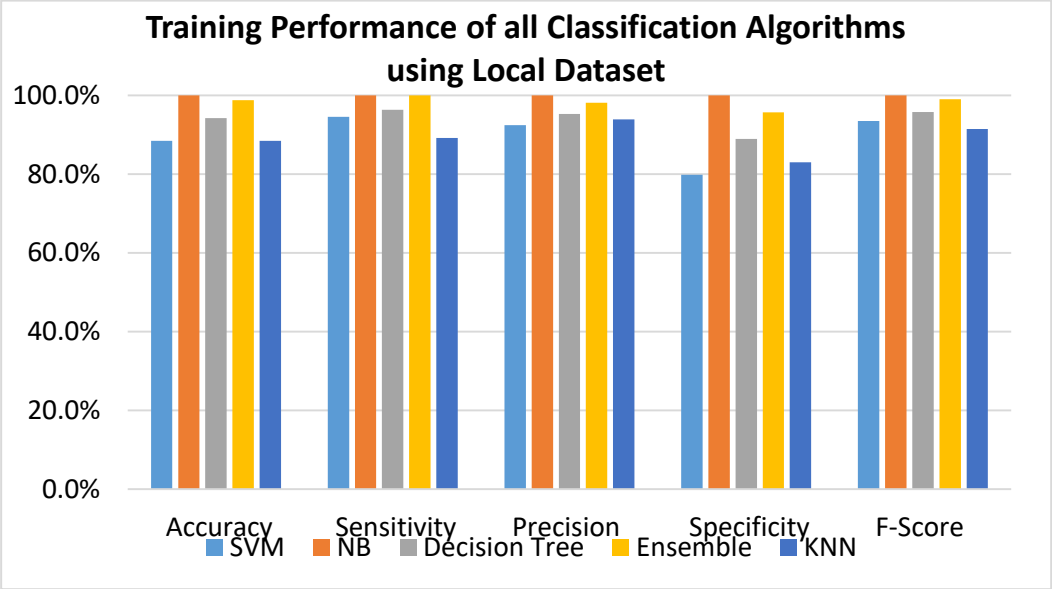


Figure 4.4-1: Training Performance of all Classification Algorithms using Local Dataset

Then, each model was tested using unseen data from 53 records to assess its performance. The model was exported in MATLAB and the function prediction

function was used against the test input data. For each model, a confusion matrix and receiver operating curve are plotted. Table 4.4-2 shows the testing performance metrics for all classification algorithms.

**Table 4.4-2: Classification testing results for local dataset**

| Classifier | Testing Accuracy | Testing Recall | Testing Precision | Testing Specificity | Testing F-Score |
|---|---|---|---|---|---|
| SVM | 87.8% | 94.4% | 87.1% | 70.5% | 90.6% |
| **NV** | **96.2%** | **100%** | **94.7%** | **88.2%** | **97.3%** |
| DC | 92.4% | 92.1% | 97.2% | 93.3% | 94.6% |
| **Ensemble** | **96.2%** | **100%** | **94.7%** | **88.2%** | **97.3%** |
| KNN | 79.2% | 83.7% | 86.1% | 68.7% | 84.9% |

The experiment summary is, for the local dataset, among SVM, NB, DC, KNN, and Ensemble algorithms, the best testing accuracy was achieved using Naïve Bayes and Ensemble of 96.2%. both models achieved the same accuracy, recall, precision, specificity, and f-score values. Other models achieved testing accuracy of 87.8%, 92.4%, and 79.2% for SVM, DC, and KNN respectively as illustrated in Figure 4.4-2. Testing confusion matrices for the above five classification algorithms are shown in Figure 4.4-3. After investigation, the KNN algorithm achieved the lowest accuracy because it's sensitive to data with outliers, the local dataset contains 178 outlier values.

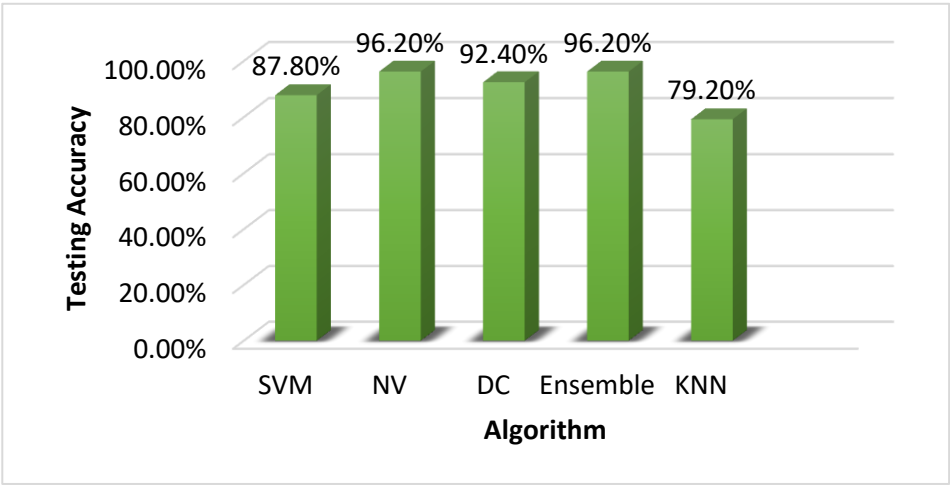


Figure 4.4-2: Testing accuracy for all machine learning algorithms

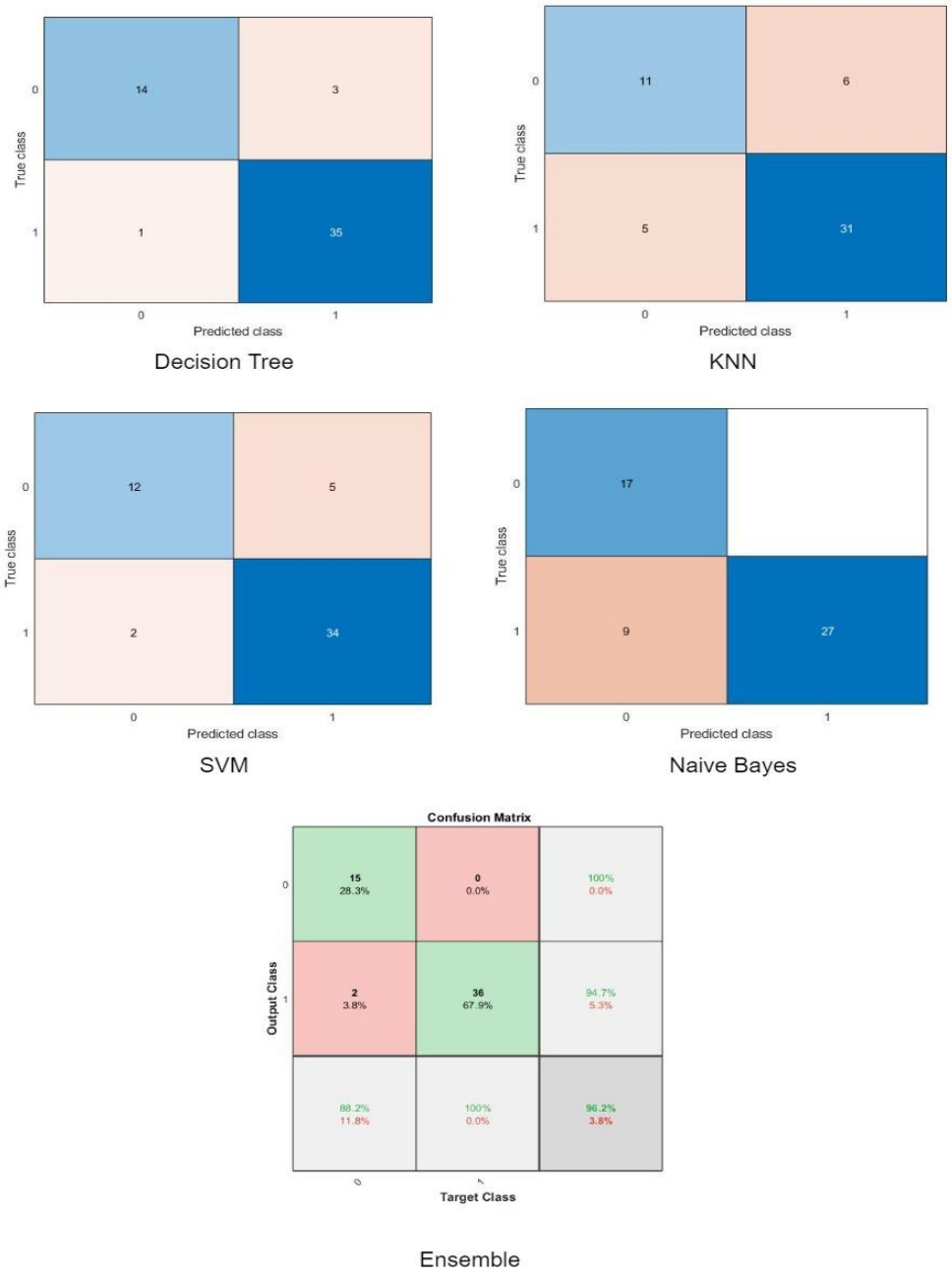


Figure 4.4-3: Testing Confusion Matrices for the Five Classification Algorithms

Figure 4.4-3 shows the ROC for all classification algorithms, according to the accuracy values achieved by Naïve Bayes and Ensemble algorithms, both algorithms' curve is close to the top left corner of the curve which indicates the ability of algorithms to correctly predict the data into target classes.

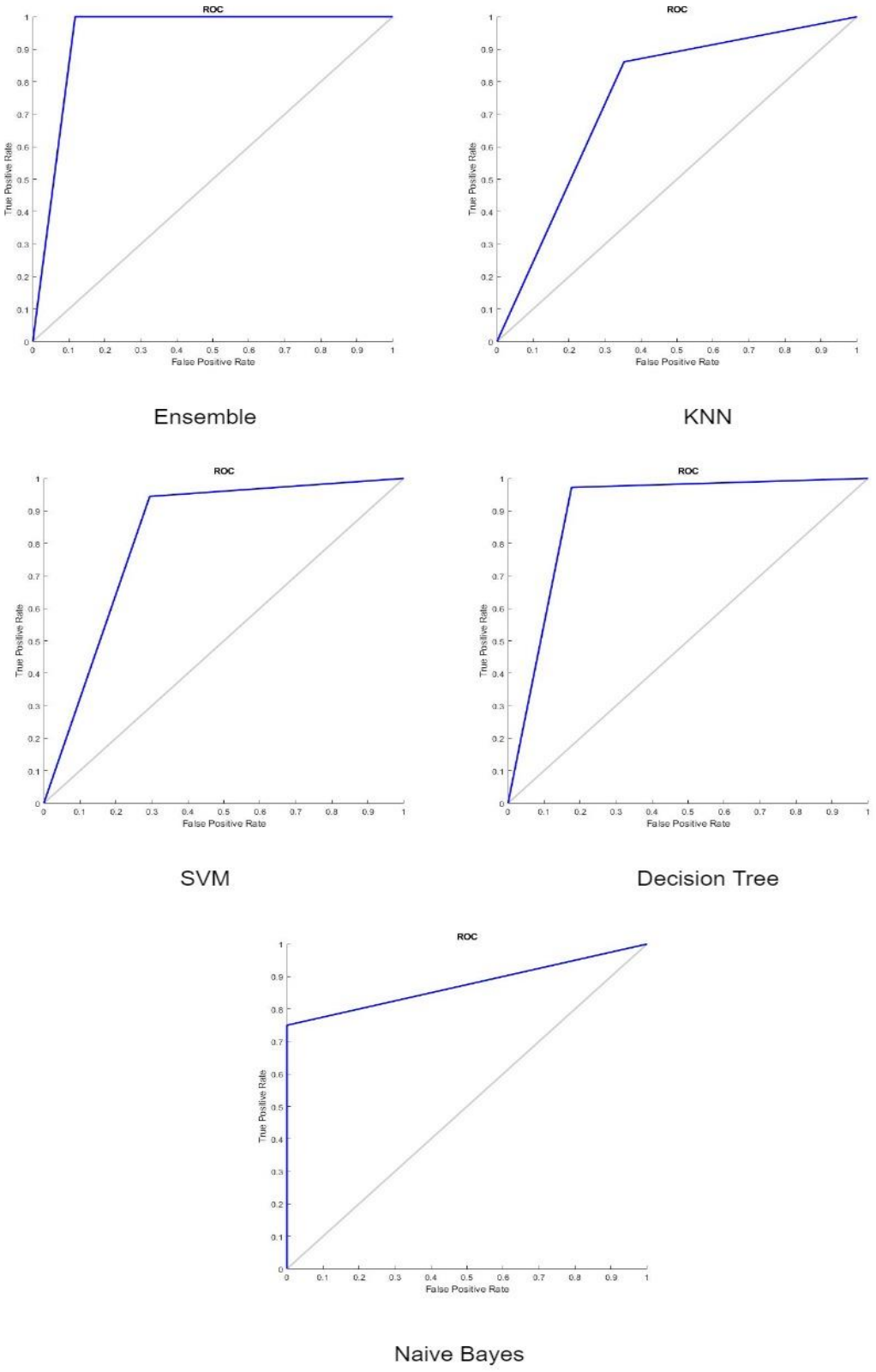


Figure 4.4-4: Testing ROC for the Five Classification Algorithms

### 4.4.2 Recurrence Neural Network Experiment Results on a Local Dataset

For Recurrence Neural Network, the model was trained and tested several times starting from 5 neurons and increasing them gradually until 50 neurons. Table 4.4-3 below shows how training and testing performance is increasing until it reaches 20 neurons then, the training accuracy dropped to 94.4% and the testing accuracy dropped to 81.1%. Then, it's fluctuating between 25 neurons and 50 neurons. In general, the best performance metrics are achieved when using 15 neurons, as seen, the training and testing accuracy is the best at 98.80% and 92.5% respectively. Also, at 20, 40, and 45 neurons, the training accuracy is high but the testing results were not that good when compared to other results. The summary is, Recurrence Neural Network can be used as a classifier with the local dataset to achieve perfect accuracy and 15 neurons are enough. Figure 4.4-4 shows the confusion matrix and ROC for the RNN testing model when 15 neurons are used.

**Table 4.4-3: Recurrence of Neural Network Training and Testing Classification Results**

| N# of Neurons | Training Accuracy | Training Recall | Training Precision | Training Specificity | Training F-Score | Testing Accuracy | Testing Recall | Testing Precision | Testing Specificity | Testing F-Score |
|---|---|---|---|---|---|---|---|---|---|---|
| **5** | 98.2% | 95.7% | 98.2% | 98.2% | 97.5% | 79.2% | 91.6% | 80.4% | 52.0% | 85.6% |
| **10** | 98.2% | 97.1% | 98.8% | 97.6% | 97.5% | 84.9% | 94.4% | 85.0% | 64.7% | 89.5% |
| **15** | **98.80%** | **100%** | **100%** | **97.6%** | **98.4%** | **92.5%** | **100%** | **90.0%** | **76.4%** | **94.7%** |
| **20** | 97.6% | 95.7% | 98.2% | 97.1% | 96.7% | 92.5% | 100% | 90.0% | 76.4% | 94.7% |
| **25** | 94.40% | 87.30% | 94.70% | 94.20% | 92.2% | 81.1% | 91.50% | 82.50% | 58.80% | 86.80% |
| **30** | 94.40% | 85.90% | 94.20% | 94.70% | 92.2% | 88.7% | 92.20% | 87.50% | 70.60% | 89.80% |
| **35** | 98.80% | 95.70% | 98.20% | 99.40% | 98.4% | 84.9% | 97.20% | 83.30% | 58.80% | 89.70% |
| **40** | 98.50% | 95.70% | 98.20% | 98.80% | 98.0% | 88.7% | 100% | 85.70% | 64.70% | 92.30% |
| **45** | 97.30% | 97.10% | 97.10% | 97.60% | 96.3% | 86.8% | 97.20% | 85.30% | 64.70% | 90.90% |
| **50** | 97.30% | 94.30% | 97.60% | 97.10% | 96.3 | 88.7% | 97.20% | 87.50% | 70.50% | 92.10% |

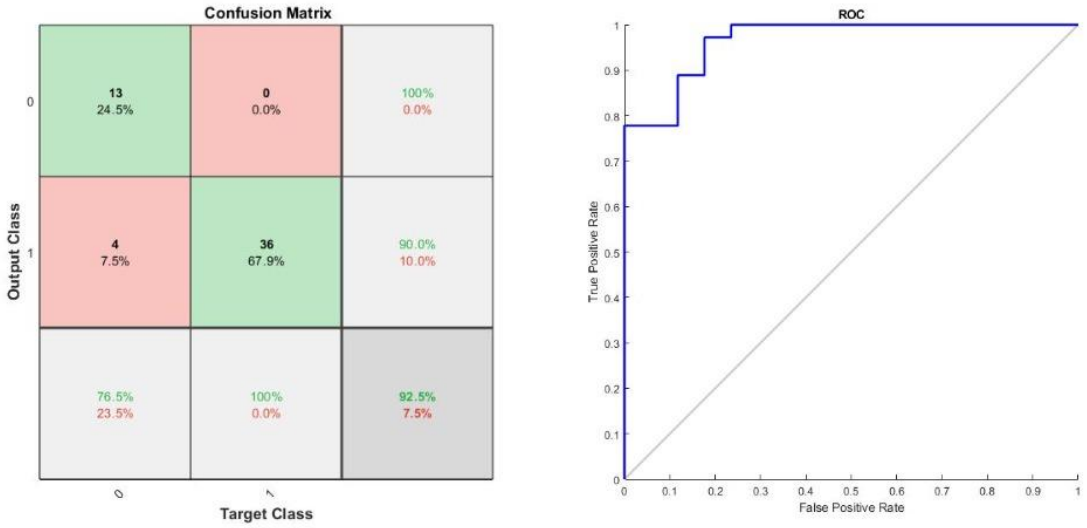


Figure 4.4-5: Confusion Matrix and ROC for RNN testing model using 15 neurons

### 4.4.3 Multi-Layer Perceptron Neural Network Experiment Results on a Local Dataset

For Multi-Layer Perceptron Neural Network, the model also was trained 10 times starting from 5 neurons to 50 neurons. Dataset was divided into 70% for training, 15% for validation, and 15% for testing. Table 4.4-4 shows the training and testing results for MLPNN.

**Table 4.4-4: MLP Neural Network Training and Testing Classification Results**

| N# of Neurons | Training Accuracy | Training Recall | Training Precision | Training Specificity | Training F-Score | Testing Accuracy | Testing Recall | Testing Precision | Testing Specificity | Testing F-Score |
|---|---|---|---|---|---|---|---|---|---|---|
| **5** | 86.8% | 92.8% | 69.9% | 89.9% | 91.3% | 79.2% | 92.8% | 74.2% | 64% | 82.5% |
| **10** | 89.7% | 94.14% | 80.1% | 91.5% | 92.8% | 88.7% | 94.5% | 89.7% | 75% | 92% |
| **15** | 91.3% | 92.8% | 89.1% | 94.5% | 93.6% | 88.7% | 90.4% | 95% | 81.1% | 92.6% |
| **20** | 91.0% | 93.5% | 85.1% | 93.5% | 93.5% | 83.0% | 89.1% | 86.8% | 68.7% | 87.9% |
| **25** | 87.3% | 88.6% | 92.5% | 84.4% | 90.5% | 83.0% | 81.5% | 94.0% | 86.6% | 87.3% |
| **30** | 89.9% | 92.2% | 84.4% | 92.7% | 92.4% | 88.7% | 94.7% | 90% | 73.3% | 92.3% |
| **35** | 97.5% | 98.2% | 96.0% | 98.2% | 98.2% | 86.8% | 92.3% | 90% | 71.4% | 91.1% |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **40** | 90.6% | 92.2% | 87.8% | 94.0% | 93.1% | 90.6% | 94.7% | 92.3% | 80% | 93.5% |
| **45** | **91.0%** | **93.0%** | **87.0%** | **94.0%** | **93.0%** | **90.6%** | **94.7%** | **92.3%** | **80%** | **93.5%** |
| **50** | 89.3% | 94.1% | 79.0% | 91.0% | 92.5% | 81.1% | 85.2% | 85.2% | 73.6% | 85.2% |

Increasing the number of neurons has increased the model training accuracy as shown in Table 4.4-4 but it hasn't reflected that much in the testing accuracy. For example, when using 35 neurons, the training accuracy was 97.5% but the testing accuracy was 86.6% and that result is not the best of all. When using 45 neurons the testing accuracy was the best achieving 90.6% which is the highest. The same accuracy result was achieved when using 40 neurons with a small difference in the training accuracy between 40 and 45 neurons. In summary, MLPNN is a good classifier for the local dataset with a large number of neurons. Figure 4.4-5 shows the confusion matrix and ROC for model testing with the best results using 45 neurons. The overall ROC is laying close to the top left corner of the curve indicating good results.

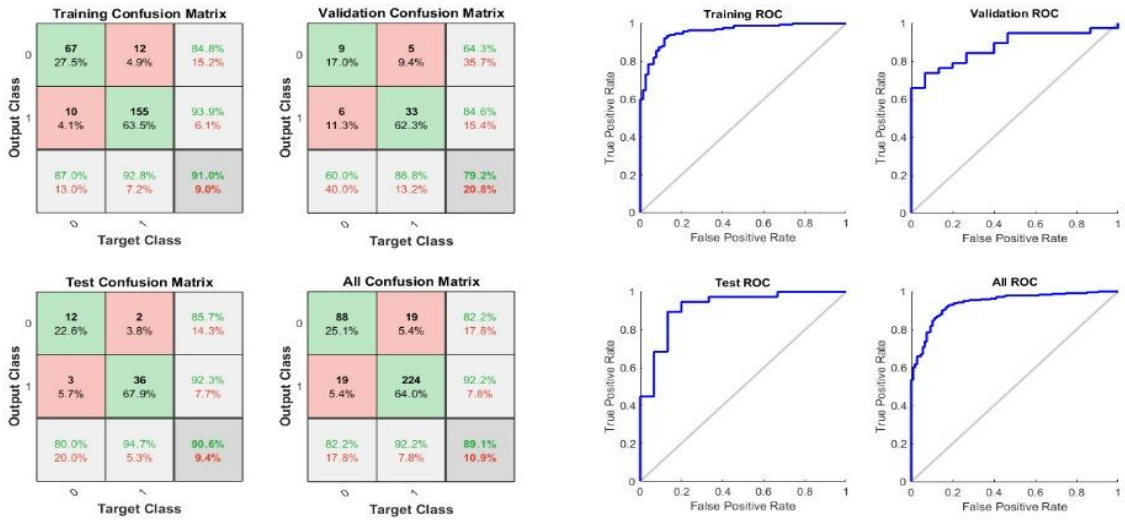


Figure 4.4-6: Confusion Matrix and ROC for MLPNN testing model when using 45 neurons

### 4.4.4 Radial Basis Function Neural Network Experiment Results on a Local Dataset

For RBFNN, the model was trained 10 times starting from 5 neurons to 50 neurons. In this experiment, the dataset was divided into 80% for training and 20% for testing. We found that the training performance did not change or improve during the 10 runs. Table 4.4-5 shows that the accuracy is 70.9% and did not improve, as a result, the performance results of testing are constant, the achieved accuracy is 66% while recall is 100% which means the model was able to predict the true positive classes. Specificity is 0% which means the model was not able to predict the false positive values. Eventually, those values affect the precision and f-score value. In summary, RBFNN is not recommended as a classifier for the local dataset if the number of neurons is small. The model could achieve better results if the number of neurons increased.

**Table 4.4-5: RBF Neural Network Training and Testing Classification Results**

| N# of Neurons | Training Accuracy | Training Recall | Training Precision | Training Specificity | Training F-Score | Testing Accuracy | Testing Recall | Testing Precision | Testing Specificity | Testing F-Score |
|---|---|---|---|---|---|---|---|---|---|---|
| **5** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **10** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **15** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **20** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **25** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **30** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **35** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **40** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **45** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |
| **50** | 71.4% | 100% | 71.4% | 0% | 83.3% | 61.4% | 100% | 61.4% | 0% | 76.1% |

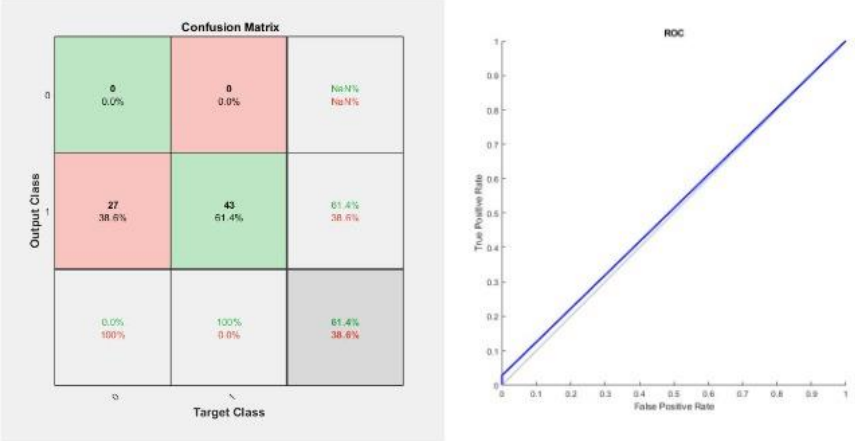


Figure 4.4-7: Confusion Matrix and ROC for RBFNN testing model from 5 to 50 neurons

Table 4.4- 6 below shows the best testing accuracy, precision, sensitivity, and specificity for all machine learning models. As we can see in the table, Ensemble and Naïve Bayes achieved the best accuracy compared to the other six models. RNN and Decision Tree achieved almost close accuracy which is considered very good if compared to other models. Of course, higher accuracy will result from a higher sensitivity rate which means the model was able to predict most of the true classes. The ensemble algorithm achieved a sensitivity value of 100% which means it was able to predict all true classes correctly. Also, higher precision and specificity values will increase the accuracy; which is obvious in the results for the Ensemble algorithm. On the other hand, Radial Basis Function got the lowest accuracy of 66% with sensitivity values of 100% which is considered good but the accuracy was affected by specificity and precision values as they're lower than others. This means the model was not able to predict the true negative and true positive values over all positives. RBFNN bad performance happened because the number of neurons should be consistent with the number of data in the local dataset we have. As said before, it needs up to 300 neurons

to achieve better performance. As for KNN, the data contains outliers that affected the performance of the algorithm. In summary, Ensemble and Naïve Bayes are recommended to classify colon tumors with a high accuracy rate. Also, DC, RNN, and MLPNN achieved fair accuracy. The discrepancy values between the eight models are shown in Figure 4.4- 8 and Table 4.4-6.

**Table 4.4-6: Discrepancy between testing values of the eight models applied on the local dataset with feature extraction by experts**

| Classifier | Accuracy | Recall | Precision | Specificity | F-Score |
|---|---|---|---|---|---|
| SVM | 87.80% | 94.4% | 87.1% | 70.50% | 90.6% |
| **NV** | **96.2%** | **100.0%** | **94.7%** | **88.2%** | **97.3%** |
| DC | 92.4% | 92.1% | 97.2% | 93.3% | 94.6% |
| **Ensemble** | **96.2%** | **100%** | **94.7%** | **88.2%** | **97.3%** |
| KNN | 79.2% | 83.7% | 86.1% | 68.7% | 84.9% |
| RBFNN | 66% | 71.90% | 100% | 0% | 76.1% |
| RNN (N = 15) | 92.50% | 90% | 100% | 76.40% | 94.7% |
| MLPNN (N = 45) | 90.6% | 94.7% | 92.3% | 80% | 93.5% |

**Table 4.4-7: Discrepancy between testing values of the eight models applied on the local dataset with feature extraction using mutual information**

| Classifier | Accuracy | Recall | Precision | Specificity | F-Score |
|---|---|---|---|---|---|
| **SVM** | **94.3%** | **100%** | **92.5%** | **81.2%** | **96.1%** |
| NV | 75.5% | 75.4% | 100% | 0% | 86% |
| **DC** | **94.3%** | **100%** | **92.5%** | **81.2%** | **96.1%** |
| **Ensemble** | **94.3%** | **100%** | **92.5%** | **81.2%** | **96.1%** |
| KNN | 92.5% | 100% | 90% | 76.4% | 95% |
| RBFNN | 24.5% | 0% | 0% | 100% | 0% |
| **RNN (N = 15)** | **94.3%** | **100%** | **92.5%** | **81.2%** | **96.1%** |
| MLPNN (N = 45) | 92.5% | 91% | 100% | 100% | 95% |

To show the discrepancy between the performances of ML techniques and the local dataset, the experiments were run on the same dataset but each time with different features. At first, the experiments took place using the same features selected by the experts as done by Abu Zuhri. Then, a feature selection using Mutual information was used against the local dataset. Table 4.4-7 shows the results of ML experiments against

the local dataset after performing a feature selection using Mutual Information. SVM, DC, Ensemble, and RNN achieved the same testing accuracy values of 94.3% while the worst value was for the Radial basis function. Naïve Bayes algorithm achieved better accuracy using the features extracted by experts compared to mutual information feature extraction, unlike SVM whose testing accuracy is 94.3% with mutual information features which is much better than its accuracy when using experts features. The same behavior is observed for the Ensemble algorithm.

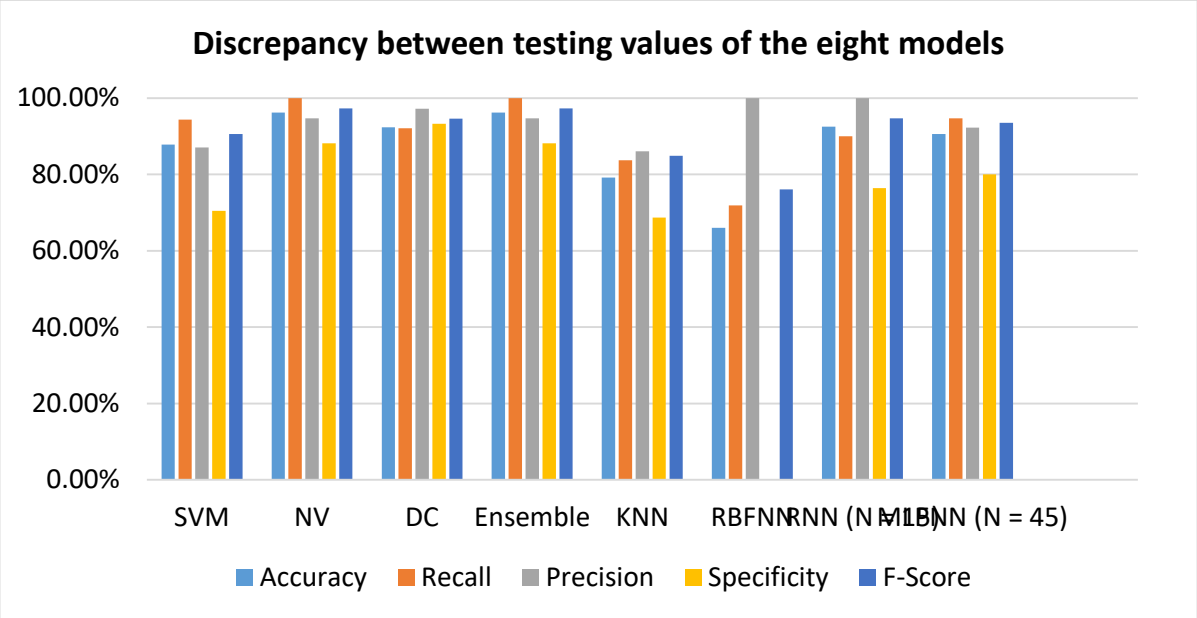


Figure 4.4-8: Discrepancy between testing values of the eight models of local dataset with experts feature selection

## 4.5 Challenges and Limitation

Several challenges and obstacles were encountered during data enquiring, the collected data from the Ministry of Health is not rich enough in terms of records, and variables and it did not cover a wide variety of populations. Also, it contains missing data and unknown values. Another limitation is the lack of a Palestinian colon cancer

images dataset. It would be good to have a local colon cancer images dataset to compare the model's results against the global colon cancer images dataset. Also, training CNN, VGG-16, and VGG-19 models did not work on a personal computer due to computational power limitations. So, I had to search for substitution by using Google Cloud Platform services to create a virtual sensor with a powerful CPU and high memory. Training and testing VGG-16 and VGG-19 models took a lot of time. VGG-16 took around 6 hours to finish and VGG-19 took around 9 hours.

# CHAPTER 5

# CONCLUSION AND FUTURE WORK

## 5.1 Conclusion

The evolution of medical-aided systems has pushed researchers to build models using machine learning and deep learning techniques to help in classifying and predicting life threaten diseases such as Cancer. Colon cancer is considered the third cause of death worldwide. Many studies were conducted using several AI algorithms to build computer-aided systems that can speed up the diagnosis process and increase human life expectancy by early diagnosis. In this thesis, we used eight different machine learning algorithms against a local dataset from the Ministry of Health to classify if the patient has a Colon tumor or not depending on ten variables. The algorithms were used on the local dataset with features selected by experts from the domain and one more time on the local dataset after performing feature selection using mutual information. Machine learning algorithms results for a local dataset with experts selected features are as follows: SVM accuracy is 87.80%, NV accuracy is 96.2%, DC accuracy is 92.4%, Ensemble accuracy is 96.2%, KNN accuracy is 79.2%, RBFNN accuracy is 66%, RNN with 15 neurons accuracy is 92.50% and MLPNN with 45 neurons accuracy is 90.6%. We recommended Ensemble and Naïve Bayes as they achieved the highest accuracy. As for the local dataset with mutual information feature selection, the results are as follows: SVM, Ensemble, Decision Tree, and Recurrence Neural Network achieved a testing accuracy of 94.3%. KNN and Multi-layer Perceptron neural network achieved 92.5%. The lowest accuracy was for Naïve Bayes with 75.5% and Radial basis function neural network with 24.5%. Another experiment was

conducted to classify Colon cancer tissue images using LC25000 global dataset. We applied the dataset against CNN, VGG-16, and VGG-19 models and did a performance comparison between them. The experiment showed that VGG-19 achieved the best accuracy of 98.94% while VGG-16 achieved 97.54% and CNN with the lowest accuracy of 83.94%.

## 5.2 Future Work & Recommendations

In future work, we're planning to train our model using a recent and large local dataset that contains more important variables in a collaboration with Palestinian hospitals, laboratories, clinics, and medical universities. Other machine learning algorithms can be used in a hybrid system to build a more powerful classification model. For deep learning, CNN, VGG-16, and VGG-19 are very powerful models to do image classification but different other models can be used against a global dataset and compare the performance results with our results of CNN, VGG-16, and VGG-19. The models can be developed by updating the tissue images dataset regularly and adding recent results from laboratories and hospitals, an updated and recent dataset can make the model more accurate. To make this model accessible for medical sector staff, a web-based interface can be developed so the staff can use the model to diagnose the samples taken from patients and get a fast and accurate diagnosis.

## Bibliography

[1] M. A. Fahami, M. Roshanzamir, N. H. Izadi, V. Keyvani, and R. Alizadehsani, "Detection of effective genes in colon cancer: A machine learning approach," *Informatics Med. Unlocked*, vol. 24, 2021, doi: 10.1016/j.imu.2021.100605.

[2] Y. Wang, W. Tavanapong, J. Wong, J. H. Oh, and P. C. de Groen, "Polyp-Alert: Near real-time feedback during colonoscopy," *Comput. Methods Programs Biomed.*, vol. 120, no. 3, 2015, doi: 10.1016/j.cmpb.2015.04.002.

[3] L. F. Sánchez-Peralta, L. Bote-Curiel, A. Picón, F. M. Sánchez-Margallo, and J. B. Pagador, "Deep learning to find colorectal polyps in colonoscopy: A systematic literature review," *Artificial Intelligence in Medicine*, vol. 108. Elsevier, p. 101923, Aug. 01, 2020, doi: 10.1016/j.artmed.2020.101923.

[4] P. J. Chen, M. C. Lin, M. J. Lai, J. C. Lin, H. H. S. Lu, and V. S. Tseng, "Accurate Classification of Diminutive Colorectal Polyps Using Computer-Aided Analysis," *Gastroenterology*, vol. 154, no. 3, 2018, doi: 10.1053/j.gastro.2017.10.010.

[5] S. Y. Park and D. Sargent, "Colonoscopic polyp detection using convolutional neural networks," *Med. Imaging 2016 Comput. Diagnosis*, vol. 9785, p. 978528, 2016, doi: 10.1117/12.2217148.

[6] A. A. Borkowski, M. M. Bui, L. Brannon Thomas, C. P. Wilson, L. A. Deland, and S. M. Mastorides, "Lung and Colon Cancer Histopathological Image Dataset (LC25000)," Accessed: Jul. 01, 2022. [Online]. Available:

https://github.com/beamandrew/medical-data.

[7] H. Chen, H. Zhao, J. Shen, R. Zhou, and Q. Zhou, “Supervised Machine Learning Model for High Dimensional Gene Data in Colon Cancer Detection,” 2015, doi: 10.1109/BigDataCongress.2015.28.

[8] A. C. Tan and D. Gilbert, “Ensemble machine learning on gene expression data for cancer classification.,” *Appl. Bioinformatics*, vol. 2, no. 3 Suppl, 2003.

[9] S. Toraman, M. Girgin, B. Üstündağ, and İ. Türkoğlu, “Classification of the likelihood of colon cancer with machine learning techniques using FTIR signals obtained from plasma,” *Turkish J. Electr. Eng. Comput. Sci.*, vol. 27, no. 3, 2019, doi: 10.3906/elk-1801-259.

[10] L. Zorron Cheng Tao Pu *et al.*, “Computer-aided diagnosis for characterization of colorectal lesions: comprehensive software that includes differentiation of serrated lesions,” *Gastrointest. Endosc.*, vol. 92, no. 4, 2020, doi: 10.1016/j.gie.2020.02.042.

[11] G. Urban *et al.*, “Deep Learning Localizes and Identifies Polyps in Real Time With 96% Accuracy in Screening Colonoscopy,” *Gastroenterology*, vol. 155, no. 4, 2018, doi: 10.1053/j.gastro.2018.06.037.

[12] T. Ozawa, S. Ishihara, M. Fujishiro, Y. Kumagai, S. Shichijo, and T. Tada, “Automated endoscopic detection and classification of colorectal polyps using convolutional neural networks,” *Therap. Adv. Gastroenterol.*, vol. 13, 2020, doi: 10.1177/1756284820910659.

[13] Y. Zeng *et al.*, “Real-time colorectal cancer diagnosis using PR-OCT with deep learning,” *Theranostics*, vol. 10, no. 6, 2020, doi: 10.7150/thno.40099.

[14] E. S. Nadimi *et al.*, “Application of deep learning for autonomous detection and localization of colorectal polyps in wireless colon capsule endoscopy,” *Comput. Electr. Eng.*, vol. 81, 2020, doi: 10.1016/j.compeleceng.2019.106531.

[15] M. M. Hasan, N. Islam, and M. M. Rahman, “Gastrointestinal polyp detection through a fusion of contourlet transform and Neural features,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 3, 2022, doi: 10.1016/j.jksuci.2019.12.013.

[16] V. Blanes-Vidal, G. Baatrup, and E. S. Nadimi, “Addressing priority challenges in the detection and assessment of colorectal polyps from capsule endoscopy and colonoscopy in colorectal cancer screening using machine learning,” *Acta Oncol. (Madr).*, vol. 58, no. sup1, 2019, doi: 10.1080/0284186X.2019.1584404.

[17] D. Zhou *et al.*, “Diagnostic evaluation of a deep learning model for optical diagnosis of colorectal cancer,” *Nat. Commun.*, vol. 11, no. 1, 2020, doi: 10.1038/s41467-020-16777-6.

[18] H. C. Park, Y. J. Kim, and S. W. Lee, “Adenocarcinoma recognition in endoscopy images using optimized convolutional neural networks,” *Appl. Sci.*, vol. 10, no. 5, 2020, doi: 10.3390/app10051650.

[19] T. Tamaki *et al.*, “Computer-aided colorectal tumor classification in NBI endoscopy: Using local features,” *Med. Image Anal.*, vol. 17, no. 1, 2013, doi: 10.1016/j.media.2012.08.003.

[20] F. Ponzio, E. Macii, E. Ficarra, and S. Di Cataldo, “Colorectal cancer classification using deep convolutional networks an experimental study,” in *BIOIMAGING 2018 - 5th International Conference on Bioimaging, Proceedings; Part of 11th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2018*, 2018, vol. 2, doi: 10.5220/0006643100580066.

[21] Z. Alom, C. Yakopcic, T. M. Taha, and V. K. Asari, “Microscopic Nuclei Classification, Segmentation and Detection with improved Deep Convolutional Neural Network (DCNN) Approaches.”

[22] H. Yoon *et al.*, “Tumor Identification in Colorectal Histology Images Using a Convolutional Neural Network,” *J. Digit. Imaging*, vol. 32, no. 1, 2019, doi: 10.1007/s10278-018-0112-9.

[23] D. Bychkov *et al.*, “Deep learning based tissue analysis predicts outcome in colorectal cancer OPEN,” *Sci. REPoRTS* |, vol. 8, p. 3395, 2018, doi: 10.1038/s41598-018-21758-3.

[24] K. Patel *et al.*, “A comparative study on polyp classification using convolutional neural networks,” *PLoS One*, vol. 15, no. 7 July, 2020, doi: 10.1371/journal.pone.0236452.

[25] S. Mangal Engineerbabu, A. Chaurasia Engineerbabu, and A. Khajanchi, “Convolution Neural Networks for diagnosing colon and lung cancer histopathological images.”

[26] S. Garg and S. Garg, “Prediction of lung and colon cancer through analysis of histopathological images by utilizing Pre-trained CNN models with visualization of class activation and saliency maps,” 2020, doi: 10.1145/3442536.3442543.

[27] S. U. K. Bukhari, A. Syed, S. K. A. Bokhari, S. S. Hussain, S. U. Armaghan, and S. S. H. Shah, “The Histological Diagnosis of Colonic Adenocarcinoma by Applying Partial Self Supervised Learning,” *medRxiv*, 2020.

[28] R. Labianca *et al.*, “Colon cancer,” *Crit. Rev. Oncol. Hematol.*, vol. 74, no. 2, pp. 106–133, May 2010, doi: 10.1016/J.CRITREVONC.2010.01.010.

[29] N. J. Camp and M. L. Slattery, “Classification tree analysis: A statistical tool to investigate risk factor interactions with an example for colon cancer (United States),” *Cancer Causes Control*, vol. 13, no. 9, 2002, doi: 10.1023/A:1020611416907.

[30] I. Ghrouz, N. El Sharif, S. Najjar, M. Awad, and M. A. Z. Abu Zuhri, “Colorectal cancer risk factor assessment in Palestine using machine learning models,” *Int. J. Med. Eng. Inform.*, vol. 1, no. 1, 2022, doi: 10.1504/ijmei.2022.10045260.

[31] S. B. Kotsiantis and D. Kanellopoulos, “Data preprocessing for supervised leaning,” *Int. J. …*, vol. 1, no. 2, 2006, doi: 10.1080/02331931003692557.

[32] F. Amiri, M. Rezaei Yousefi, C. Lucas, A. Shakery, and N. Yazdani, “Mutual information-based feature selection for intrusion detection systems,” *J. Netw. Comput. Appl.*, vol. 34, no. 4, 2011, doi: 10.1016/j.jnca.2011.01.002.

[33] S. A. Alasadi and W. S. Bhaya, “Review of data preprocessing techniques in data

mining," *J. Eng. Appl. Sci.*, vol. 12, no. 16, 2017, doi: 10.3923/jeasci.2017.4102.4107.

[34] H. Bhavsar and M. H. Panchal, "A Review on Support Vector Machine for Data Classification," *Int. J. Adv. Res. Comput. Eng. Technol.*, vol. 1, no. 10, 2012.

[35] W. S. Noble, "What is a support vector machine?," *Nature Biotechnology*, vol. 24, no. 12. 2006, doi: 10.1038/nbt1206-1565.

[36] M. W. Ashour, F. Khalid, A. A. Halin, and L. N. Abdullah, "Machining process classification using PCA reduced histogram features and the Support Vector Machine," 2016, doi: 10.1109/ICSIPA.2015.7412226.

[37] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, 2020, doi: 10.1016/j.neucom.2019.10.118.

[38] A. Yasar and M. M. Saritas, "Performance Analysis of ANN and Naive Bayes Classification Algorithm for Data Classification," *Int. J. Intell. Syst. Appl. Eng.*, vol. 7, no. 2, 2019, doi: 10.18201/ijisae.2019252786.

[39] M. Langarizadeh and F. Moghbeli, "Applying naive bayesian networks to disease prediction: A systematic review," *Acta Inform. Medica*, vol. 24, no. 5, 2016, doi: 10.5455/aim.2016.24.364-369.

[40] L. Rokach and O. Maimon, "Top-down induction of decision trees classifiers - A survey," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 35, no. 4, 2005,

doi: 10.1109/TSMCC.2004.843247.

[41] P. Geurts, A. Irrthum, and L. Wehenkel, “Supervised learning with decision tree-based methods in computational and systems biology,” *Molecular BioSystems*, vol. 5, no. 12. 2009, doi: 10.1039/b907946g.

[42] F. Provost and P. Domingos, “Tree induction for probability-based ranking,” *Mach. Learn.*, vol. 52, no. 3, 2003, doi: 10.1023/A:1024099825458.

[43] L. A. Wehenkel, *Automatic Learning Techniques in Power Systems*. 1998.

[44] A. Akbari, L. Ng, and B. Solnik, “Drivers of economic and financial integration: A machine learning approach,” *J. Empir. Financ.*, vol. 61, 2021, doi: 10.1016/j.jempfin.2020.12.005.

[45] T. M. Cover and P. E. Hart, “Nearest Neighbor Pattern Classification,” *IEEE Trans. Inf. Theory*, vol. 13, no. 1, 1967, doi: 10.1109/TIT.1967.1053964.

[46] P. Cunningham and S. J. Delany, “K-Nearest Neighbour Classifiers-A Tutorial,” *ACM Computing Surveys*, vol. 54, no. 6. 2021, doi: 10.1145/3459665.

[47] S. Zhang, M. Zong, X. Zhu, D. Cheng, and X. Li, “Learning k for kNN classification,” *ACM Trans. Intell. Syst. Technol*, vol. 8, no. 43, 2017, doi: 10.1145/2990508.

[48] I. Boulnemour, B. Boucheham, and L. Abdelmadjid, “On Enhancing the Accuracy of Nearest Neighbour Time Series Classifier Using Improved Shape Exchange Algorithm On Enhancing the Accuracy of Nearest Neighbour Time

Series Classifier Using Improved Shape Exchange Algorithm," no. September, 2020.

[49] Z. H. Zhou, *Ensemble methods: Foundations and algorithms*. 2012.

[50] M. Hosni, I. Abnane, A. Idri, J. M. Carrillo de Gea, and J. L. Fernández Alemán, "Reviewing ensemble classification methods in breast cancer," *Computer Methods and Programs in Biomedicine*, vol. 177. 2019, doi: 10.1016/j.cmpb.2019.05.019.

[51] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, no. 2. 2020, doi: 10.1007/s11704-019-8208-z.

[52] A. Rahman and S. Tasnim, "Ensemble Classifiers and Their Applications: A Review," *Int. J. Comput. Trends Technol.*, vol. 10, no. 1, 2014, [Online]. Available: www.internationaljournalssrg.org.

[53] T. M. Khoshgoftaar, D. J. Dittman, R. Wald, and W. Awada, "A review of ensemble classification for DNA microarrays data," 2013, doi: 10.1109/ICTAI.2013.64.

[54] L. I. Kuncheva, J. J. Rodríguez, C. O. Plumpton, D. E. J. Linden, and S. J. Johnston, "Random subspace ensembles for fMRI classification," *IEEE Trans. Med. Imaging*, vol. 29, no. 2, 2010, doi: 10.1109/TMI.2009.2037756.

[55] J. Ghosh and A. Nag, “An Overview of Radial Basis Function Networks,” 2001.

[56] Montazer, Gholam Ali, D. Giveki, M. Karami, and H. Rastegar, “Radial Basis Function Neural Networks: A Review,” *Comput. Rev.*, vol. 1, no. 1, pp. 52–74, 2018.

[57] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” 2015.

[58] A. C. Tsoi, “Recurrent neural network architectures: An overview,” 1998.

[59] F. Murtagh, “Multilayer perceptrons for classification and regression,” *Neurocomputing*, vol. 2, no. 5–6, 1991, doi: 10.1016/0925-2312(91)90023-5.

[60] L. Alzubaidi *et al.*, “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” *J Big Data*, vol. 8, p. 53, 2021, doi: 10.1186/s40537-021-00444-8.

[61] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, 2017, doi: 10.1145/3065386.

[62] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, “Review on Convolutional Neural Networks (CNN) in vegetation remote sensing,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 173. 2021, doi: 10.1016/j.isprsjprs.2020.12.010.

[63] V. H. Phung and E. J. Rhee, “A High-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets,” *Appl. Sci.*, vol. 9, no. 21, 2019, doi: 10.3390/app9214500.

[64] H. Gu, Y. Wang, S. Hong, and G. Gui, “Blind channel identification aided generalized automatic modulation recognition based on deep learning,” *IEEE Access*, vol. 7, 2019, doi: 10.1109/ACCESS.2019.2934354.

[65] W. You, C. Shen, D. Wang, L. Chen, X. Jiang, and Z. Zhu, “An Intelligent Deep Feature Learning Method with Improved Activation Functions for Machine Fault Diagnosis,” *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2019.2962734.

[66] H. Gholamalinezhad and H. Khosravi, “Pooling Methods in Deep Neural Networks, a Review.”

[67] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2015.

[68] A. Ajit, K. Acharya, and A. Samanta, “A Review of Convolutional Neural Networks,” 2020, doi: 10.1109/ic-ETITE47903.2020.049.

[69] S. Tammina, “Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images,” *Int. J. Sci. Res. Publ.*, vol. 9, no. 10, 2019, doi: 10.29322/ijsrp.9.10.2019.p9420.

[70] J. Jaworek-Korjakowska, P. Kleczek, and M. Gorgon, “Melanoma thickness prediction based on convolutional neural network with VGG-19 model transfer learning,” in *IEEE Computer Society Conference on Computer Vision and*

*Pattern Recognition Workshops*, 2019, vol. 2019-June, doi: 10.1109/CVPRW.2019.00333.

[71] M. Mateen, J. Wen, S. Song, and Z. Huang, “Fundus Image Classification Using VGG-19 Architecture with PCA and SVD,” 2018, doi: 10.3390/sym11010001.

[72] M. Heydarian, T. E. Doyle, and R. Samavi, “MLCM: Multi-Label Confusion Matrix,” doi: 10.1109/ACCESS.2022.3151048.

[73] T. Reddy Gadekallu *et al.*, “electronics Article,” doi: 10.3390/electronics9020274.

## الملخص

شجع التطور السريع في التكنولوجيا في الوقت الحاضر وخاصة في القطاع الطبي الباحثين على بناء نماذج الذكاء الاصطناعي للمساعدة في تشخيص الأمراض الحرجة التي تهدد حياة الإنسان مثل السرطان. يعتبر سرطان القولون السبب الثالث للوفاة في جميع أنحاء العالم. لا تظهر أعراض سرطان القولون في مراحله المبكرة إلا أنه قد ينتشر في الأعضاء البشرية بشكل سريع ويسبب الضرر. يمكن أتمتة تشخيص سرطان القولون باستخدام نماذج قوية للذكاء الاصطناعي في المراحل المبكرة، مع وقت أقل ودقة أكبر وتكاليف أقل. في هذا البحث، استخدمنا خوارزميات التعلم الآلي وخوارزميات التعلم العميق لتصنيف سرطان القولون باستخدام مجموعة بيانات محلية من وزارة الصحة للأعراض وعوامل الخطر ومجموعة بيانات عالمية أخرى لصور أنسجة سرطان القولون. تم استخدام ثمانية خوارزميات مختلفة للتعلم الآلي؛ آلة المتجهات الداعمة (SVM) ، نموذج بايز الساذج (Naïve Bayes) ، شجرة القرار (DC)، الجار الأقرب لـ K (KNN)، نموذج Ensemble ، وظيفة الأساس الشعاعي (RBFNN)، الشبكة العصبية المتكررة(RNN) ، والشبكة العصبية متعددة الطبقات . (MLPNN)بالنسبة لخوارزميات التعلم العميق، استخدمنا CNN و VGG-16 و VGG-19 . تم إجراء مقارنة لنتائج الأداء لخوارزميات التعلم الآلي الثمانية لتقييم النموذج بأفضل دقة. يتم أيضًا إجراء مقارنة بين نماذج التعلم العميق لتقييم النموذج بأفضل دقة لتصنيف صور أنسجة سرطان القولون. أظهرت نتائج التجربة الأولى على مجموعة بيانات محلية ميزات اختارها خبراء المجال مع خوارزميات التعلم الآلي أن خوارزميات Ensemble و Naïve Bayes كانت قادرة على تحقيق أفضل دقة بنسبة 96.2٪. أتت نتائج الخوارزميات الأخرى كما يلي SVM :و DT و KNNو RBF و RNN و MLPNN حققت 87.80% و 92.4٪ و 79.2٪ و 66٪ و 92.5٪ و 90.6٪ على التوالي. تم تطبيق نفس الخوارزميات على مجموعة البيانات المحلية بعد استخدام اختيار ميزة المعلومات المتبادلة. حققت الشبكة العصبية SVM و DC و Ensemble و Recurrence أفضل دقة بنسبة 94.3٪. بالنسبة للتجربة الثانية باستخدام تقنيات التعلم العميق مقابل مجموعة البيانات العالميةLC25000 ، تمكنت VGG-19 من تحقيق أفضل دقة تصنيف بنسبة 98.94٪، بينما حققت 97.54% VGG-16وحققت CNN دقة تصل الى 83.94%.

يوصى باستخدام خوارزمية المجموعة (Ensemble) لتصنيف أورام أنسجة القولون باستخدام مجموعة البيانات المحلية بينما يوصى باستخدام VGG-19 بدقة فائقة لتصنيف صور ورم أنسجة القولون بين الأورام السرطانية والحميدة.