THE REPUBLIC OF TURKEY
BAHÇEŞEHİR UNIVERSITY

DEVELOPING A DEEP LEARNING APPROACH FOR DETECTING BRAIN CANCER FROM MRI IMAGES

Master's Thesis

AHMAD TANASH

İSTANBUL, 2020

THE REPUBLIC OF TURKEY
BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
BIG DATA ANALYTICS AND MANAGEMENT

DEVELOPING A DEEP LEARNING APPROACH FOR DETECTING BRAIN CANCER FROM MRI IMAGES

Master's Thesis

AHMAD TANASH

Thesis Supervisor: ASSIST PROF. DR. SERKAN AYVAZ

İSTANBUL, 2020

THE REPUBLIC OF TURKEY
BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
BIG DATA ANALYTICS AND MANAGEMENT

Name of the thesis: Developing A Deep Learning Approach for Detecting Brain Cancer from MRI Images
Name/Last Name of the Student: Ahmad TANASH
Date of the Defense of Thesis: 08/01/2020

The thesis has been approved by the Graduate School of Natural and Applied Sciences.

Assist. Prof. Dr. Yücel Batu SALMAN
Graduate School Director

I certify that this thesis meets all the requirements as a thesis for the degree of Master of Science.

Assist Prof. Dr. Serkan AYVAZ
Program Coordinator

This is to certify that we have read this thesis and we find it fully adequate in scope, quality and content, as a thesis for the degree of Master of Science.

Examining Comitee Members	Signature
Thesis Supervisor Assist Prof. Dr. Serkan AYVAZ	----------------------------------
Member Assist Prof. Dr. Tarkan AYDIN	----------------------------------
Member Assist Prof. Dr Atınç YILMAZ	----------------------------------

DEDICATION

This thesis is dedicated to my father, for this endless love, support and encouragement may his soul rest in peace.

ACKNOWLGEMENT

First, I wish to thank Allah for the endless gifts for me and for surrounding me with all these wonderful and supportive people.

I wish to thank my wonderful family for their endless love and support, especially my great mother for her continuous encouragement and help.

I wish to thank my supervisor, Assist Prof. Serkan AYVAZ for his advises, help, guidance and patience with me and his encouragement during the preparation of this thesis.

I wish to thank the radiologist Ashraf Tanash for his advice and help in the medical part.

I would like to thank every person who helps facilitate the accomplishment of this thesis, my friends, my great college.

İstanbul, 2020 Ahmad TANASH

ABSTRACT

Developing A Deep Learning Approach for Detecting Brain Cancer from MRI Images

Ahmad TANASH

Big Data Analytics and Management

Thesis Supervisor: Assist Prof. Dr. Serkan AYVAZ

January 2020, 45 pages

Background: Brain tumor in general is considered as a fatal disease that attacks the nervous system of the body, while brain is considered as one of most important parts of our body because of its functionality, it is also considered as one of our most sensitive parts of the body specially in term of diseases that effect it functionality. Brain tumor detection is the process of classifying the brain to normal or abnormal, it is a challenging task to everyone who is related to the field due to our brain sensitive nature, complexity and the nature of the disease , also this process is considered as resources consumer for both of patient and doctors, but it is a primary step for the patient to decide the treatment plan.

Method: In this work, a Convolutional Neural Network (CNN) algorithm is proposed as a classification solution for brain MRI images, a network with a variety of convolutional, max-pooling and fully connected layer are used to achieve the desired goal.

Evaluation: To evaluate our proposed method a cross validation method used, accuracy, precision, recall and F1-score were used to evaluate cross validation results.

Results: The proposed method results are compered with each other, for each fold evaluation method are used to evaluate it, maximum results for three folds our five were 0.96 for accuracy, 0.90 for precision, 1.00 for recall and 0.91 for F1-score, while average results are 0.93 for accuracy, 0.85 for precision, 1.00 for recall and 0.92 for F1-socre.

Conclusion: An efficient brain tumor classification method for MRI images based on CNN has been built, the high results of the model makes it effective to aid doctors in the process of detection brain tumor.

Keywords: Deep Learning, Convolutional Neural Network, Brain Tumor, MRI Images.

ÖZET

Mrg Görüntülerinden Beyin Kanserini Saptamada
Derin Öğrenme Yaklaşimi Geliştirmek

Ahmad TANASH

Büyük Veri Analitiği ve Yönetimi

Tez Danışmanı: Dr. Öğr. Üyesi Serkan AYVAZ

Ocak 2020, 45 sayfa

Amaç: Beyin tümörü genel olarak vücudun sinir sistemine saldıran ölümcül bir hastalık olarak kabul edilirken, beyin işlevselliği nedeniyle vücudumuzun en önemli kısımlarından biri olarak kabul edilirken, aynı zamanda en hassas parçalarımızdan biri olarak kabul edilir. vücudun özellikle işlevselliğini etkileyen hastalıklar açısından. Beyin tümör tespiti, beyni normal veya anormal olarak sınıflandırma sürecidir, beyne duyarlı doğamız, karmaşıklığımız ve hastalığın doğası nedeniyle alanla ilgili herkes için zorlu bir görevdir, ayrıca bu süreç kaynaklar olarak kabul edilir. hem hasta hem de doktorlar için tüketicidir, ancak hastanın tedavi planına karar vermesi için birincil adımdır**.**

Yöntem: Bu çalışmada, beyin MRG görüntüleri için bir sınıflandırma çözümü olarak bir Konvolüsyon Sinir Ağı (CNN) algoritması önerilmiştir, istenen hedefe ulaşmak için çeşitli evrişimsel, maksimum havuzlama ve tamamen bağlı katmana sahip bir ağ kullanılmaktadır**.**

Değerlendirme: Önerilen yöntemimizi değerlendirmek için çapraz geçerlilik yöntemi kullanılmış, çapraz geçerlilik sonuçlarını değerlendirmek için doğruluk, kesinlik, hatırlama ve F1 skoru kullanılmıştır**.**

Sonuçlar: Önerilen yöntem sonuçları birbiriyle karşılaştırılır, her bir katlama değerlendirme yöntemi için bunu değerlendirmek için kullanılır, üç kat için maksimum sonuçlar beşimiz doğruluk için 0.96, hassasiyet için 0.90, hatırlama için 1.00 ve F1 skoru için 0.91'dir. ortalama sonuçlar doğruluk için 0.93, hassasiyet için 0.85, hatırlama için 1.00 ve F1-socre için 0.92'dir.

Sonuç: CNN'ye dayalı MRG görüntüleri için etkili bir beyin tümörü sınıflandırma yöntemi oluşturulmuştur, modelin yüksek sonuçları beyin tümörünün saptanması sürecinde doktorlara yardımcı olmayı etkili kılmaktadır.

Anahtar Kelimeler: Derin Öğrenme, Konvolüsyonel Sinir Ağı, Beyin Tümörü, MRG Görüntüleri.

CONTENTS

TABLES

FIGURES

ABBREVIATIONS

AI	:	Artificial Intelligence
ANN	:	Artificial Neural Network
CNN	:	Conventional Neural Network
CSF	:	Cerebrospinal fluid
CT	:	Computerized Tomography
DICOM	:	Digital Imaging and Communications in Medicine
DL	:	Deep Learning
DWT:	:	Wavelet Transformation
FLAIR	:	Fluid Attenuated Inversion Recovery
FN	:	False Negative
FP	:	False Positive
GLCM	:	Level Co-occurrence Matrix
JPEG	:	Joint Photographic Experts Group
KNN	:	K-Nearest Neighbor
ML	:	Machine learning
MRI	:	Magnetic Resonance Imaging
NN	:	Neural Network
PCA	:	Principle Component Analysis
ReLU	:	Rectified Linear Unit
RMSprop	:	Root Mean Square Propagation
SOM	:	Self-Organizing Maps
TE	:	Time of Echo
TF	:	Repetition Time
TN	:	True Negative
TP	:	True Positive
WM	:	White Matter

1. INTRODUCTION

1.1 OVERVIEW

Cancer is known as one of the most life-threatening and fatal diseases that humanity faced. Melanoma starts in one cell after that other cell gets infected, which builds up a block, blocks leads to tissues, which can be found in any part of the body, including the brain. Typically, cells are generated when the old cells die, but, in some cases, the body produces a new cell, and the old cells did not shrink or die; in this situation, a cancerous cell appears, when cancerous cells appear in the brain, they are called brain tumors.

A brain tumor is a collection of irregular cells in the brain that surround a brain very rigidly. Symptoms of this disease appear in the form of frequent headaches, unexplained vomiting, difficulty in speech and hearing, and gradual loss of body control. According to numbers, a brain tumor is one of the deadliest diseases that humans face in terms of patient survival. While 51 % of leukemia patients and 86% of breast cancer live and survives more than five years, less than two hundredth of tumor patients survive on the far side 5 years of their diagnosing (Stark Facts, 2018).

1.2 PROBLEM DEFINITION

Brain tumor detection is defined as a process of finding abnormal brain cells (tumor) using computer techniques or doctor judgment. Accurately classifying the brain is healthy or not is a challenging task for almost everyone related to the filed. It is the base and essential stage to decide the next step for the patient, necessary treatment, a preliminary idea about the tumor stage if it exists. Whether if we need to do a medical biopsy or not, the right examination for the brain can save time, cost, and other kinds of resources for doctors, hospitals, and patients.

Mainly tumor detection is considered as a core role in the treatment process, an early detection of a brain tumor increases the chances for a patient to be cured, while late detection is a big death causer in a considerable number of cases.

1.3 MOTIVATION

The classification of medical images of the brain, if it contains a tumor or not, is a fundamental and preliminary step for the patient. It determines the treatment planning and pathological analysis for the patient if the cancer exists.

Neurologist, the process of manual detection of the tumor is defined as a time-consuming process; it may take several hours with high concentration for a Neurologist to determine whether the brain contains a cancer or not only for a patient which makes the process open for human error.

While tumor detection is a challenging task for everyone related to the field, the need of an automated process has been increased with the increase of the brain tumor cases in the world according to statistics : for United States of America in 2019 approximately 87,000 new cases of malignant, non-malignant and other brain nervous system tumor are expected to be diagnosed (Ostrom, et al., 2018).

Since the manual process of tumor detection is a time killer and resources consumer process an automated process has become the suitable solution to detect brain tumor from Magnetic resonance imaging (MRI) Images to determine the next step for the patient and whether we need to do a biopsy or not with a less time span and higher accuracy rate than the manual process.

1.4 CONTRIBUTION

The contribution of this thesis is summarized in proposing a deep learning (DL) model for brain tumor detection from a collection of normal and abnormal brain MRI images, by developing a Conventional Neural Network (CNN) model with images processing techniques to recognize brain tumor and classify them.

1.5 OBJECTIVES

We have developed a deep learning approach using a convolutional neural network model for the detection of brain tumors and cancer. Our work can be used for:

a. Develop a fully automated model that can detect normal brains and abnormal brain from a dataset of MRI images.

b. Develop a deep learning algorithm to detect brain tumor images while focusing on the accuracy rate.

c. Develop a deep learning algorithm that aids doctors in detecting and classifies brain tumors.

d. Shrink the time and resources that is used to detect brain tumors for both of radiologist and patients.

1.6 THESIS STRUCTURE

This section describes the thesis rest of thesis format:

a. Chapter 2 (Literature review): This chapter defines the necessary concepts that we need in our project and gives a survey of other works which is related to the filed.

b. Chapter 3 (Methodology): This chapter clarifies and describes what we contribute to brain tumor detection.

c. Chapter 4 (Results and Discussion): This chapter shows and describes the outcomes and results that we acquired from the model that we developed.

d. Chapter 5 (Conclusion and Future Work): This chapter summarizes the work and the results of the project and what we will do in the future.

e. Chapter 6 (Discussion): This chapter describes our solution limitations and future work.

2.LITERATURE REVIEW:

2.1 INTRODUCTION

This chapter describes the following concepts for a further understanding of the problem and solution in this study:

a. The concept of brain tumors and their types.
b. The concept of brain tumor detection by Radiologist.
c. The concept of Magnetic Resonance Imaging (MRI).
d. The concept of machine learning and Artificial Neural Networks (ANN).
e. The concept of deep learning techniques and methods
f. The concept of Convolution Neural Network (CNN).

2.2 BRAIN TUMOR

2.2.1 Brain Tumors Definition

A brain tumor is considered as abnormal expansion of cells in the brain and central nervous system (King Hussein Cancer Foundation). Not all types of brain tumors are cancer, as those tumors cane be benign or malignant. When a tumor does not contain cancer cells it is called benign, and once they are removed the chance of them to regrow is rare. However, benign brain tumors may lead to serious health problems and issues and can they can grow up and become malignant. Malignant brain tumors are considered as cancerous, overgrowing, and attacking surrounding tissues rigidly and are usually life-threatening (Wrensch, Bondy, Wiencke, & Yost, 1993).

2.2.2 Brain Tumors Types

Brain tumor types are categorized into two main types primary and secondary, primary brain tumors: tumors that begin in the brain are classified by region, which they start to grow within the brain or by the cells that infect them.

Secondary brain tumors: caused by cancers that arise elsewhere in the body, such as the lung, for example, after that it spreads to the brain. This type of brain tumor is also called as metastatic brain cancer. Metastatic brain cancer is treated differently from primary brain cancer depending on where cancer originated in the body (Griesinger, et al., 2013).

2.2.3 Brain Tumor Causes

Brain cancer and tumor cases still ambiguous and unknown. Still, brain tumor has been found more widely spread in people whom their age average is more over than 60, it is also considered as the second common cancer in children after leukemia. Brain tumor has been spreads and grows in males more than females only meningococcal tumors are the only species found to affect females more than males. Brain tumor development relies on specific risk factors that leads into a release of a brain tumor:

a. A high dose of a certain radiation exposes, such as X-rays that are used in brain direct treatment.
b. Some rare genetic conditions increase the chance of developing brain tumors such as neurofibromatosis, Von Hebel Lindau syndrome and Turkot syndrome.
c. Age, the risk of brain tumor and cancer increases with age, brain tumor cases appear to be more frequent in elder people than others.

2.2.4 Brain Tumor Symptoms

People with brain cancer have many symptoms and signs, and they may also not be exposed to any of them, and may result from diseases other than brain cancer. Brain cancer symptoms can be divided into general and specific symptoms, and general symptoms result from the pressure the tumor generates for the brain or on the spinal cord, the symptoms are caused by the disruption of the functioning of a specific part of the brain due to the tumor. For many people with brain tumors, they are diagnosed after reviewing a doctor with complaints such as head pain or other changes.

i) General Symptoms of Brain Tumor and Cancer

For general indications and symptoms for a brain tumor, they are summarized in the point below:

a. Frequent headache.
b. Seizures or so-called convulsions.

c. Change on the level of memory or personality.

d. Nausea and vomiting.

e. General fatigue and exhaustion.

ii) Specific Symptoms of Brain Tumor and Cancer

For specific indications and symptoms for a brain tumor and cancer it depends on the part that it exists in it; the common specific symptoms of all effected part are summarized in the following points:

a. Sensation of pain or pressure of the place or around the cancer.

b. A change in the issuance of decisions and judgments This includes loss of initiative in dealing, as well as laziness and muscle weakness or paralysis.

c. Complete or partial vision loss may indicate a tumor in the occipital or temporal lobe of the brain.

d. A tumor may cause inability to lookup in the pineal gland.

e. A weakness in the facial muscles and numbness followed with a difficulty of swallowing and a double vision, those symptoms and indications are associated with stem brain tumor.

f. Vision disorder, including loss or double vision loss, this situation maybe acquire from tumors in the temporal or occipital lobe of the brain or brain stem.

2.2.5 Brain Tumor Diagnosis

Doctors relies on a range of diagnostic and examinations techniques and procedures and techniques to decide the presence of the tumor type in the brain whether is it a benign or malignant tumor, and to estimate its size, location, and speed of growth using the methods below:

a. A complete physical and neurological examination such as vision, hearing, and muscle reactions.

b. Magnetic Resonance Image (MRI) for a complete picture of the whole body and brain to determine the presence and location of any abnormal areas.

c. Computerized Tomography (CT) scan of the brain to look for any abnormal areas.
d. Spinal fluid examination to obtain a sample of spinal fluid and examine the presence of the abnormal cells.
e. Take a sample of the tumor through needle or in some certain cases through a biopsy to get some tissue to be tested in a lab for a further understanding of the tumor nature.

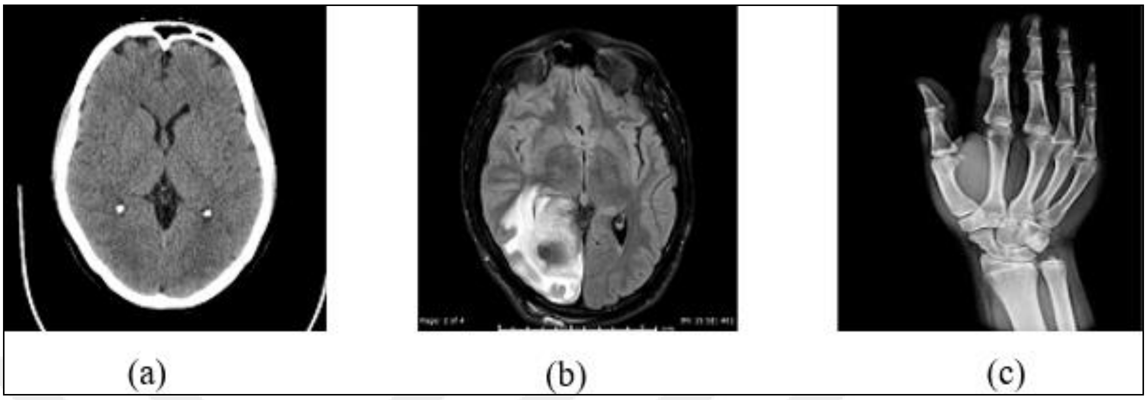
2.3 MAGNETIC RESONANCE IMAGING (MRI)

Magnetic Resonance Imaging (MRI) is the most common techniques in medical imaging filed; it a scanning technology creates a detailed image of the human body. While MRI is well known as a non-ionized and non-invasive imaging technique with a unique tissue variation, it seems an appropriate technique for brain imaging.

MRI can also define as a proton energy map of a tissue inside of the body, which makes it different than other kinds of medical images for example Computer Tomography (CT) and X-ray, where they can define as a map of density of the tissue in the body.

X-rays beneficial to show doctors a broken bone but not patient, soft tissue or organics also CT images are limited to one axial-plane of the patient organic, so it does not give a rich detail about the soft tissues insides, on the other hand, MRI provides images for the soft tissues and organics of patient with different axial-plane with a rich details to detect any of abnormality behavior for a brain. The figure 2.1 below shows the difference between those three types of images.

Figure 2.1: Example of a different medical image's types. (a) CT images for a healthy brain. (b) MR images of a typical flair for an abnormal brain. (c) X-ray image for a normal hand.



2.3.1 Magnetic Resonance Imaging Sequences

For brain tumor detection and diagnosis MRI are widely used in the preprocessing phase as a basic examiner (Vézina, 2008).it gives a variety of different images depending on the purpose of use and the which part are being examined, for our study T1, T2 and Fluid Attenuated Inversion Recovery (FLAIR) are used as input.

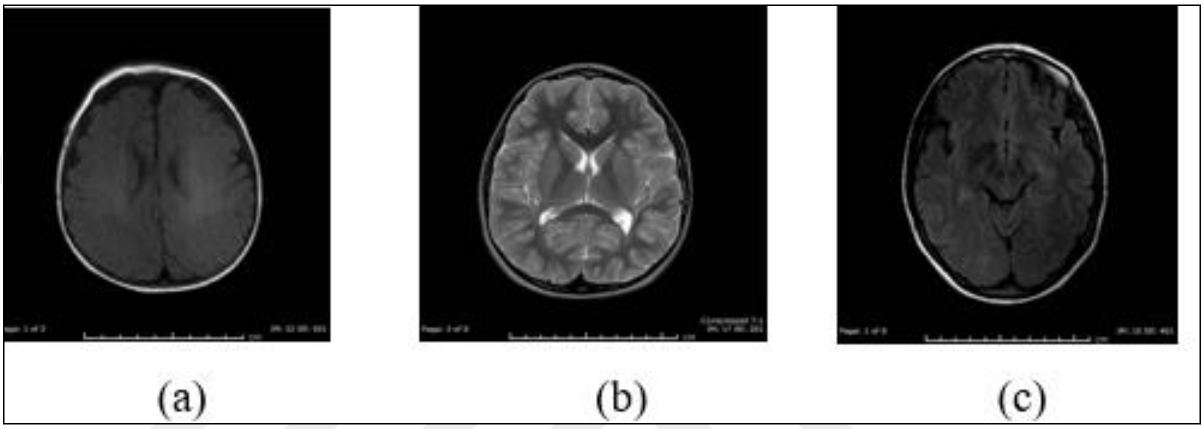
MRI technology relies on two essential concepts to generate different sequences which they are time of echo (TE) and Repetition Time (TR). TE is defined as a duration between the series of successive pulses applied to the same slide where TR is defined as the reception time between two continuous pulses used in the same sequence (Werring, et al., 2004).

T1-weighted images: contain the dark appearance of Cerebrospinal Fluid (CSF) and fluid. White matter (WM) is brighter than Gray matter (GM), for brain structure imaging T1 images is suitable, and fat seems more brilliant in this sort. TE and TR time (TR ->500msec, TE->14msec) is short of producing the images (Tahir, 2018).

T2-weighted images: T2 images contain stronger intensity of signal for CSF and fluid comparing to the tissue, and for that purpose, it appears bright. T2 riles on long time (TR->4000msec, TE->19msec) for TE and TR to produce images. T2 gives brighter color for water and fluid, which make it ideal for the edema tissue (Hajnal, et al., 1992).

FLAIR: it is similar to T2, but in FLAIR the CSF fluid is more attenuated, also abnormalities stays bright. FLAIR is sufficient for cerebral edema imaging. It relies on very long TE and TR time (TR->9000msec, TE->114msec) for producing images (Kato, Izumiyama, Izumiyama, Takahashi, & Itoyama, 2002). Figure 2.2 below represents the difference between these types of sequence in an MRI image.

Figure 2.2: Example of a different MRI image's types. (a) T1-weighted image. (b) T2-weighted image. (c) FLAIR image.



For a further understanding of we need to clarify the differences between the features of those types of images in terms of tissues, as shown in Table 2.1 below.

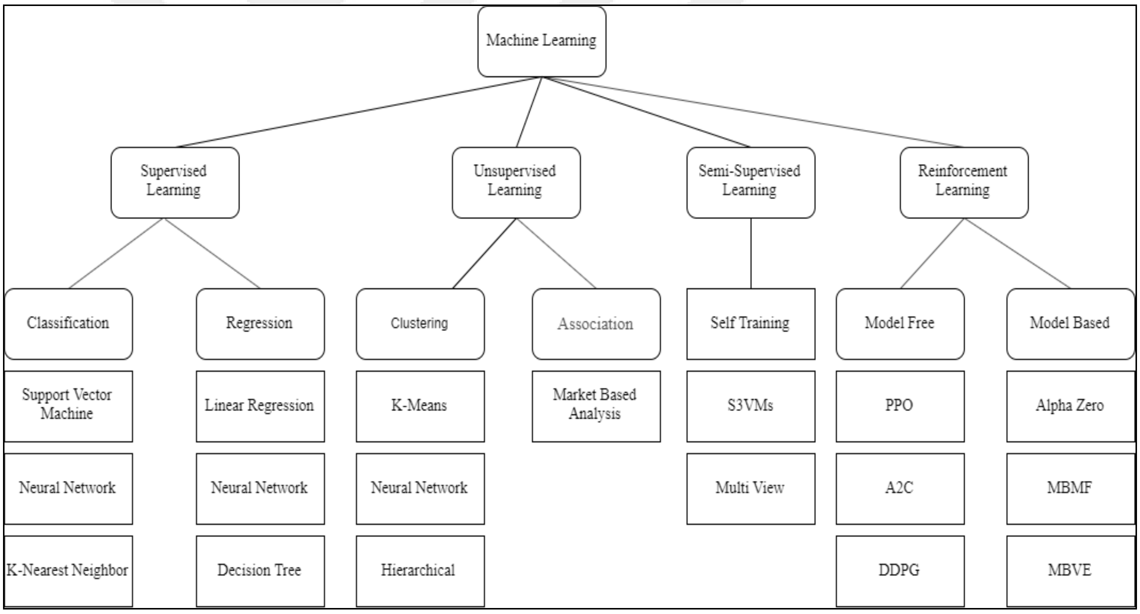
Table 2.1: Represent the variations based on various tissue types.

Tissues	T1-Weighted	T2-Weighted	Flair
Cerebrospinal Fluid (CSF)	Dark	Bright	Dark
White matter	Light	Dark grey	Dark grey
Cortex	Grey	Light Grey	Light Grey
Fat (within bone marrow)	Bright	Light	Light
Inflammation (impurity)	Dark	Light	Light

2.4 MACHINE LEARNING AND ARTIFICIAL NEURAL NETWORKS

Machine learning (ML) is known as a subfield in Artificial Intelligence (AI) which generally uses statistical techniques that allow the computer to be able to learn from data. Machine learning filed focuses on making predictions based on available data. Machine learning is divided into four main subsections, and under these subsections, there are a variety of algorithms to be used in terms of achieving machine learning goals. The figure below shows machine learning subsections and gives and a variety of examples for each subsection algorithms.

Figure 2.3: Machine learning subfields with algorithms as an example of each type of learning.



As the figure shows that the machine learning process is described into four main fields, for a further understanding of these four fields, they are described in detail.

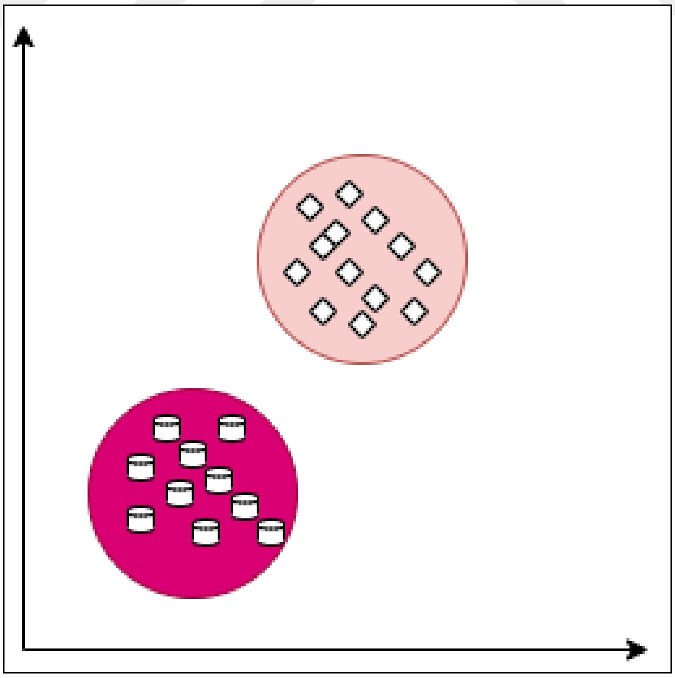
2.4.1 Unsupervised Learning

Unsupervised Learning is defined as a type of learning where the data is not labeled or classified, where the algorithms or the learner acts without guidance on the data. Unsupervised learning, known as a descriptive learning goal, is to find or devise new models or relationships in the data without any prior training.

Unsupervised learning problems and or algorithms are defined into two main fields:

a. Clustering: it is the task or the process of grouping or dividing your data into subgroups where the data in the new subgroups are like each other and different to the data in other groups.

Figure 2.4: A basic example of a clustering problem



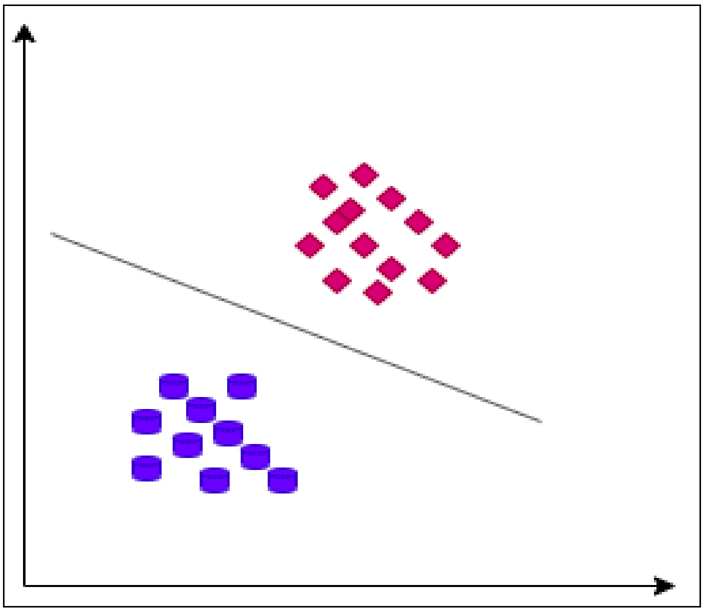
b. Association: it is a task or the problem that is used to find a rule that describes the most rule in the dataset.

2.4.2 Supervised Learning:

Supervised learning is defined as a type of learning where the data is labeled or classified into categories where the data already is already tagged with the correct answers. Supervised learning is also known as predictive learning goal is to understand the relationship between the input and output so it can predict or categorize any new income data. Supervised learning problems or algorithms are categorized into two main types.

1. Regression: the process of predicting the values of continuous variables within a linear relationship between the data and its labels.
2. Classification: the process of predicting or categorize the data into two or more specific labels. Classification is considered as the most used machine learning technique.

Figure 2.5: A basic example of classification learning.



2.4.3 Semi-Supervised Learning

Semi-Supervised learning is defined as a type of learning where some of the data are categorized and labeled, and the rest of it is unlabeled, it is a conjunction between supervised learning and unsupervised learning, where the algorithms are being able to learn from the labeled data and apply its knowledge on the unlabeled data.

2.4.4 Reinforcement Leaning

Reinforcement learning is defined as a deep learning method that describes how the software agent should take actions in the environment. This method relies on putting the agent in a sequence of operations, observe it, reward it, and trying to predict the correct path of the agent until we have a proper way or prediction, which also known as policy.

Integrating Machine Learning (ML) methods into brain tumor detection and recognition has always played a core role in terms of improving performance and accuracy in the detection process. The manual brain tumor detection process takes more time and consumes many resources than machine learning techniques. Machine Learning methods appear appropriate in this area as they can improve the accuracy and performance in the process of brain tumor detection while consuming fewer resources and time.

Recently, there has been increasing interests in using machine learning techniques with brain tumor detection. Several studies in the filed have been applied to brain tumor detection in side of enhancing the performance and the accuracy of the process. Some of the related work to the field are mentioned below

Sharma, Komal, et al (2014), In this paper, the authors focus on developing an automated method to detect brain tumors from MRI images to save radiologist time, by using machine learning techniques, the work is divided into three different parts, preprocessing which applied on brain MRI images includes noise reduction, edge detection, after that a texture features are extracted using Gray Level Co-occurrence Matrix (GLCM), and the last step is using Self-Organizing Maps (SOM) to classify brain MRI images.

Kumar et al., (2017), in this work the authors proposed a modified method of K-means and morphological operations for brain tumor segmentation and detection, it relies on two different algorithms, they achieved accuracy in brain tumor segmentation like manual segmentation in term of reproduce of the tumor tissue.

Chethan et al, (2016), this work proposes a K-Nearest Neighbor (KNN) algorithm with multistep techniques, they started with obtaining scanned brain MRI images, after that a preprocessing applied using size-changing and noise removal, last step is using KNN and fuzzy C-mean to sperate normal cells from tumor cells, that method contributes in determine tumor stage for patients.

2.5 DEEP LEARNING AND CONVOLUTION NEURAL NETWORK

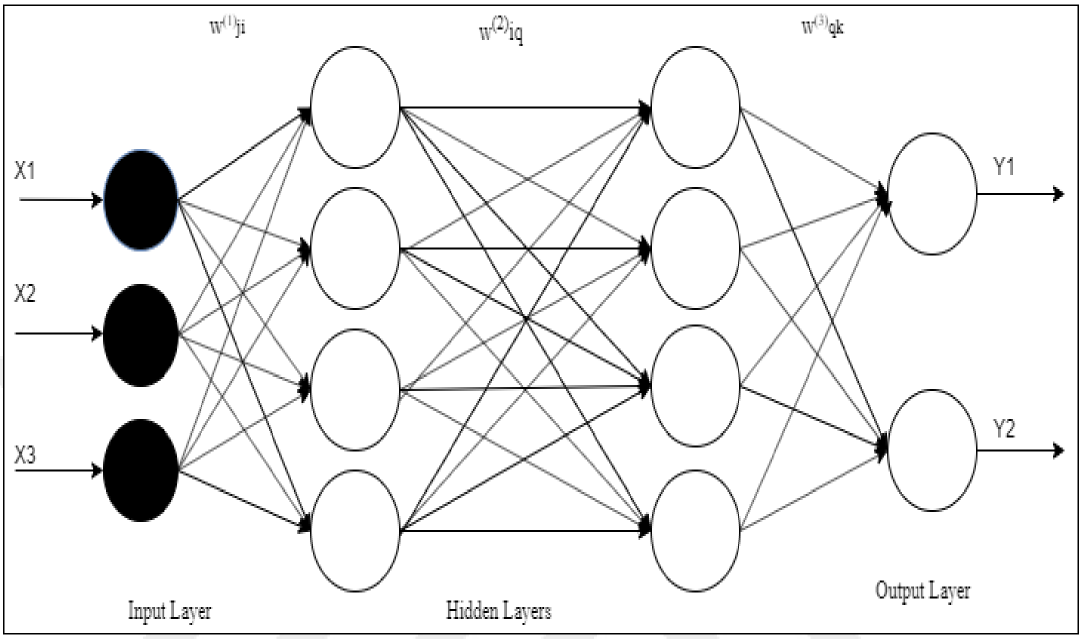
Deep Learning (DL) is considered as a subfield of machine learning that is inspired by Artificial Neural Network (ANN) to learn, “deep learning” was first presented in 2006 (Hinton, Osindero, & Teh, 2006).Deep learning known also as deep structured learning, structured learning is defined as a class of machine learning algorithms which focus on various layers to slowly obtain higher-level features through raw input. For example, for image processing, lower layers can identify boundaries, while high layers can identify human concepts, such as numbers or characters or expressions (Deng, 2014).

2.5.1 Artificial Neural Network (ANN)

Neural networks have gained widespread acceptance and acknowledgement as an effective machine learning algorithm by outperforming many algorithms, such as least squares support vector machines, in different applications such as pattern recognition and image classification (Schölkopf, Burges, & Smola, 1999) (Vapnik, 2010).

Artificial Neural Network (ANN) is an algorithm which emulates the way of the brain in processing data. ANN intent to build a map between input and output. Basically, ANN are built form neurons. A neuron is an essential component in the neural network that receives the data from another node (neuron) or from input source and compute an output, each output relates to a weight, weight is assigned according to the importance of it according to the other inputs. After that, the node applies a particular function to the averaged sum of its data. Figure 2.6 below Gives an example about the structure of an Artificial Neural Network.

Figure 2.6: A basic example of a Feed Forward Neural Network structure



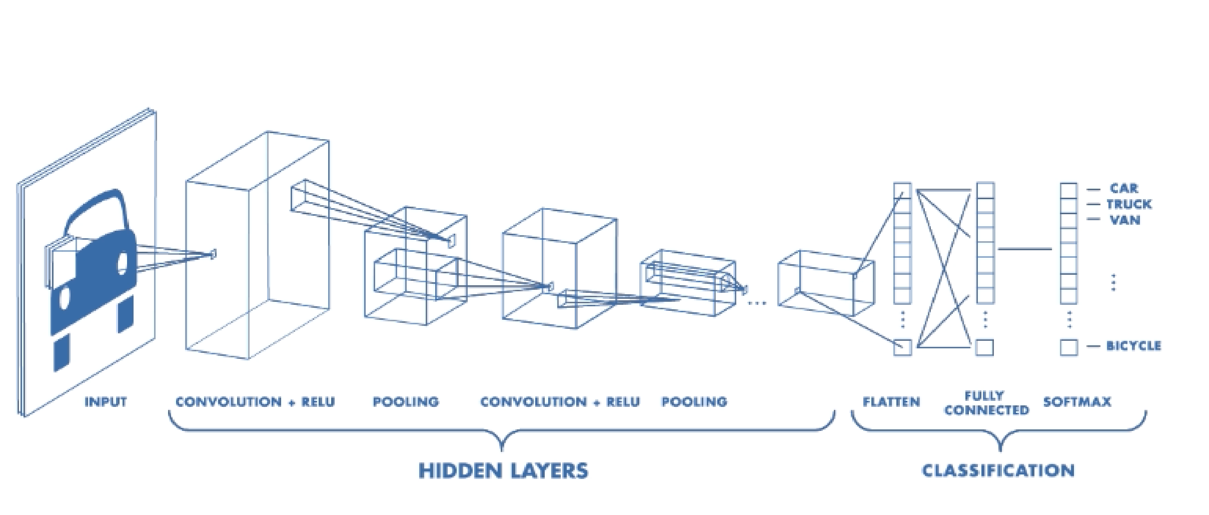
Artificial Neural Network contains three essential layers basically the first layer is called an input layer which is a set of node that provides the data from an external source to the next layer, in the input layer no calculation is accomplished in the data, after that the input from the next layer which called Hidden Layer where all of the calculations happens, hidden layer can be one layer of neurons or multiple layers of neurons ,in the end when all of the calculations the results goes to output layer this layer is responsible for the calculations and transmission of information from the network to external source.

2.5.2 Convolutional Neural Network

A Convolutional Neural Network (CNN) is considered as a deep learning technique that has been improved in the last ten years; it simulates the structure of the brain. CNN algorithm is an outstanding algorithm in the deep learning world, while traditional machine learning algorithms rely on manual feature extraction from input images, CNN and deep learning algorithms take images in its row format as input to select features.

A Convolutional Neural Network contains three layers input layer, output layer, and a variety of hidden layers between first and last layers, number of layers in between input and output is changeable according to the task that is performed on the CNN algorithm. Figure 2.6 shows an example of a CNN structure.

Figure: 2.7: Example of a CNN structure



Source: Chatterjee, C. C. (2019, July 31). Basics of the Classic CNN

CNN’s structure contains different hidden layers. The number of layers is changeable; it depends on the task performed on the algorithm. Basically, the components of the hidden layers are:

a. Convolutional layer: In this layer, a dot product is used between a particular filter and a specific area to create a new value for the local area.
b. Pooling layer: This pooling layer is responsible for shrinking the dimensions of an input image, also it reduces the computational power using dimensionality reduction.
c. Fully connected layer: this layer is reasonable about the calculations of the probability of each image which belongs to each class; it got its name because this layer relates to all the previous layer.

Recently, there has been an increasing interest in the use of deep learning techniques with brain tumor detection. Several studies in the filed have been applied to brain tumor detection in order to improve the performance and the accuracy of the process. Some of the related work are mentioned below.

Sapra et al. (2013), In this paper, the authors propose an image segmentation technique to detect brain tumor from MRI images using Probabilistic Neural Network (PNN), The PNN is used in automated brain tumor classification from MRI images, the system is proposed to classify more brain tumor cases more accurately.

Kharat et al. (2012), In the research the authors propose two different neural network (NN) to deal with brain tumor MRI images classification, they feature extraction had been handled using Wavelet Transformation (DWT), next step is using Principle Component Analysis (PCA) to reduce the feature number, last step is using two classifiers to classify brain tumor images.

V.P. Gladis Pushpa Rathi et al,(2014), this study focuses on brain tumor MRI images detection and classification using multiple kernel-based probabilistic clustering with deep learning classifier, three modules were used in for images classification, named as segmentation module, feature extraction module, and classification module, images are preprocessed to fit for segmentation while de-noising is done using median filter, segmentation is done using Multiple Kernel-based Probabilistic Clustering (MKPC), next step were feature extraction, a Linear Discriminant Analysis (LDA) used to select important features, at the end a deep learning classifier is used to classify normal and abnormal brain MRI images.

3. METHODOLOGY

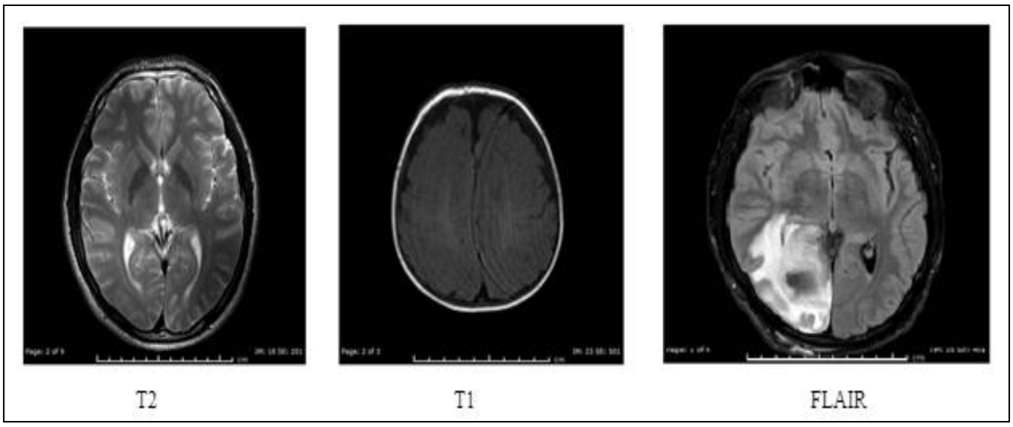
This chapter aims to clarify our contribution to solving the problem of the detection of brain tumors. This chapter defines the following concepts and techniques for our brain tumor detection methodology:

a. Dataset set gathering and splitting
b. Medical images converting from Digital Imaging and Communications in Medicine (DICOM) to Joint Photographic Experts Group (JPEG)
c. Preprocessing
d. CNN model for brain tumor detection
e. Evaluation metrics

3.1 DATASET

For this study, we acquired our dataset from two different resources for brain tumor MRI images we acquired our images from Cancer Archive web site; we used a data set called LGG-1p19qDeletion (Erickson, Bradley; Akkus, Zeynettin; Sedlar, Jiri; Korfiatis, Panagiotis., 2017), The data set contains MRI images of type T1, T2 and flair for brain tumor patients. For a healthy brain MRI images, we gathered a T1, T2, and flair images from a local hospital for different patients. Figure 3.1 below gives an example of MRI images that we collected.

Figure 3.1: An example of MRI images that we collected.



In total, our data set contains 710 MRI images for normal and abnormal brains; we split it into three different datasets, Training, testing, and validation. Training dataset contains 587 MRI images those images are spilt to 242 MRI images labeled as normal and another 345 images labeled as abnormal (Contains tumor), for validation a 73 MRI images used ,we split the we took 41 images labeled as normal and another 32 images labeled as abnormal, at the end we took 19 images as normal and another 31 abnormal brain images for our training data set. Figures 3.2, 3.3, and 3.4 below clarifies the distribution of the data for our data sets.

Figure 3.2: The distribution of training set, where NO represent healthy brain MRI images, and YES represent MRI brain tumor images.

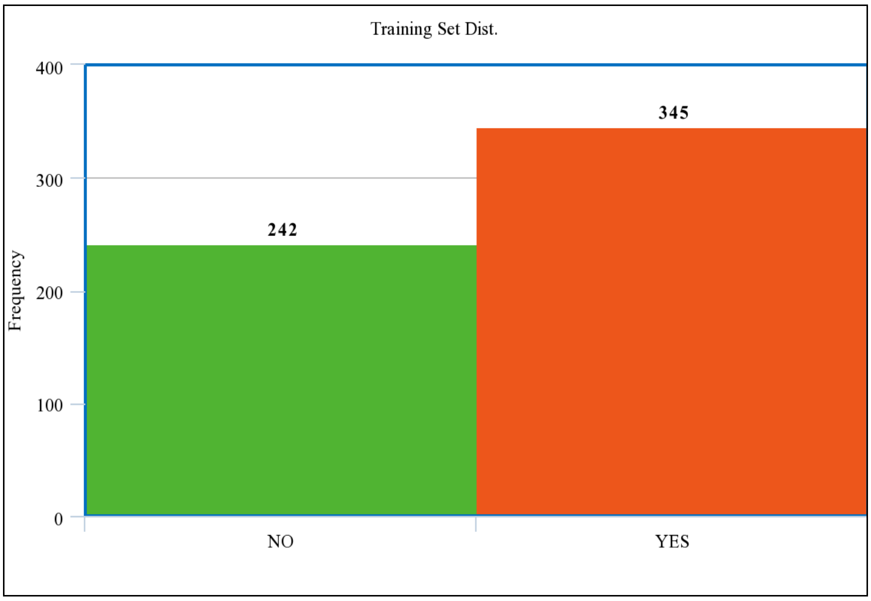


Figure 3.3: The distribution of validation set, where NO represent healthy brain MRI images, and YES represent MRI brain tumor images.

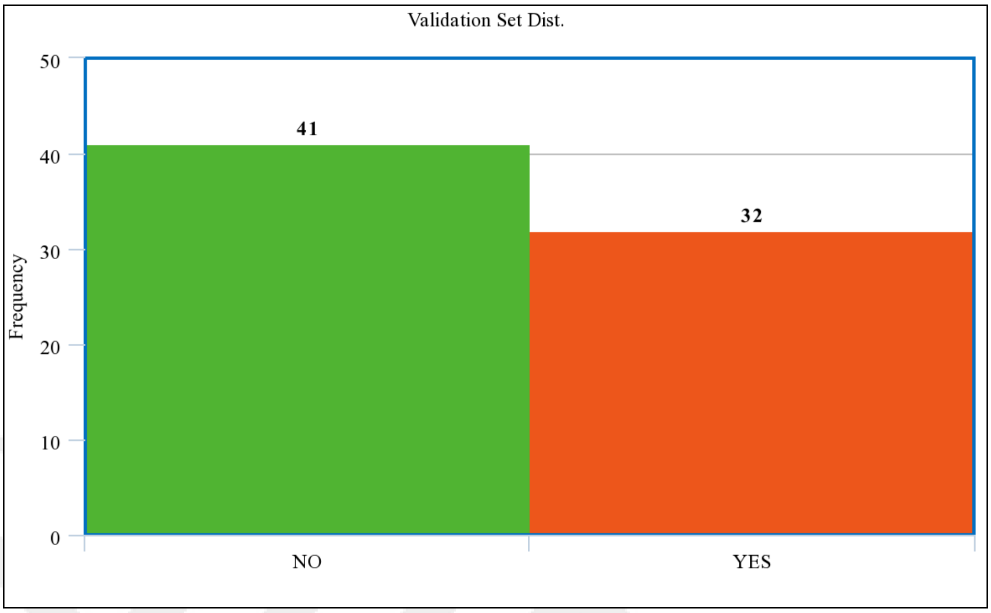
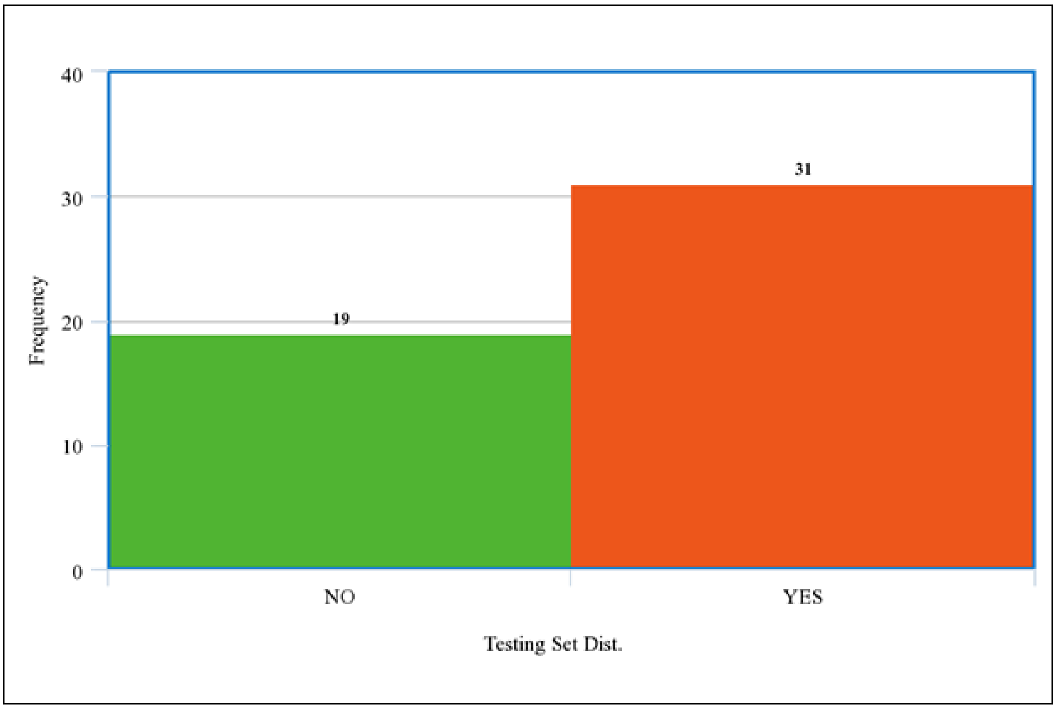


Figure 3.4: The distribution of the testing set, where NO represents healthy brain MRI images, and YES represent MRI brain tumor images.



3.2 FROM DICOM TO JPEG

As we mentioned before, our dataset has been gathered from two different resources; for healthy brain MRI images, the images came in JPEG format, while brain tumor MRI images came in DICOM format for images processing techniques we need to have a universal format of images.

For this problem, we had two different solutions to choose between them, the first solution is converting JPEG images to DICOM, and the other solution is converting the DICOM to JEPG format. Each solution gets its advantages and disadvantages. For the first solution, the images are less lossless in details than JPEG, but with a bigger size than JPEG, while on the other hand, JPEG format is more lossless than DICOM, but the size of images are suitable for our resources than DICOM.

Due to our hardware limitations in terms of resources and for the nature of our performed task we applied the second solution in our dataset, in the end, we had a 700 MRI images in JPEG format as a final solution for our format confliction problem.

3.3 PREPROCESSING

For image processing techniques preprocessing plays a core role in the field, but while we are dealing with a CNN algorithm which is a feature extraction algorithm, we performed one simple preprocessing step which is image resizing, the reason behind this step is that CNN algorithm deals learns better when all of the images got the same size, all of the images resized to size equals 428X 428 pixels.

3.4 DATA AUGMENTATION

Data augmentation is defined as a method to make the deep learning algorithm more generalize in the learning term, and it expands data size by applying a certain number of filters to a target dataset. Due to the limitations in our dataset size, we used a data augmentation technique on our training and validation, and we used the following filters and method on our data sets in data augmentation:

a. Rotation Rate: assigned to a value equal to 15.

b. Width Shift Range: assigned to 0.1.
c. Height Shift Range: assigned to 0.1.
d. Shear range: assigned to 0.1.
e. Brightness Range: assigned to a range between 0.5 to 1.5.
f. Horizontal Flip: this parameter takes a Boolean value; we assign it to TRUE.
g. Vertical Flip: as a horizontal flip, it takes Boolean values, we assign it to TRUE.
h. Rescale: we assign the re-scale value to (1/255).

We used the previous parameters on both of our validation set and training set, for testing, images kept as same without using data augmentation.

3.5 CONVOLUTIONAL NEURAL NETWORK (CNN) LAYERS

Convolutional Neural Network are considered as a subtype of artificial neural network that have provided significant breakthroughs in many areas regarding pattern recognition and image classification (Ojala, Pietikäinen, & Mäenpää, 2002). CNN is an algorithm that simulates the brain architecture in the behavior of finding and discovering patterns. To achieve that goal, CNN, contains a number of different layers and components, the various amount of layers that has been used in our solution is explained below.

3.5.1 Convolution Layer

The convolution layer represents as the main block of the convolution network structure. It's a feed-forward neural network; this layer is responsible for the main calculations in-network; it relies on learnable kernels as its core component. The example below explains convolution layer working criteria.

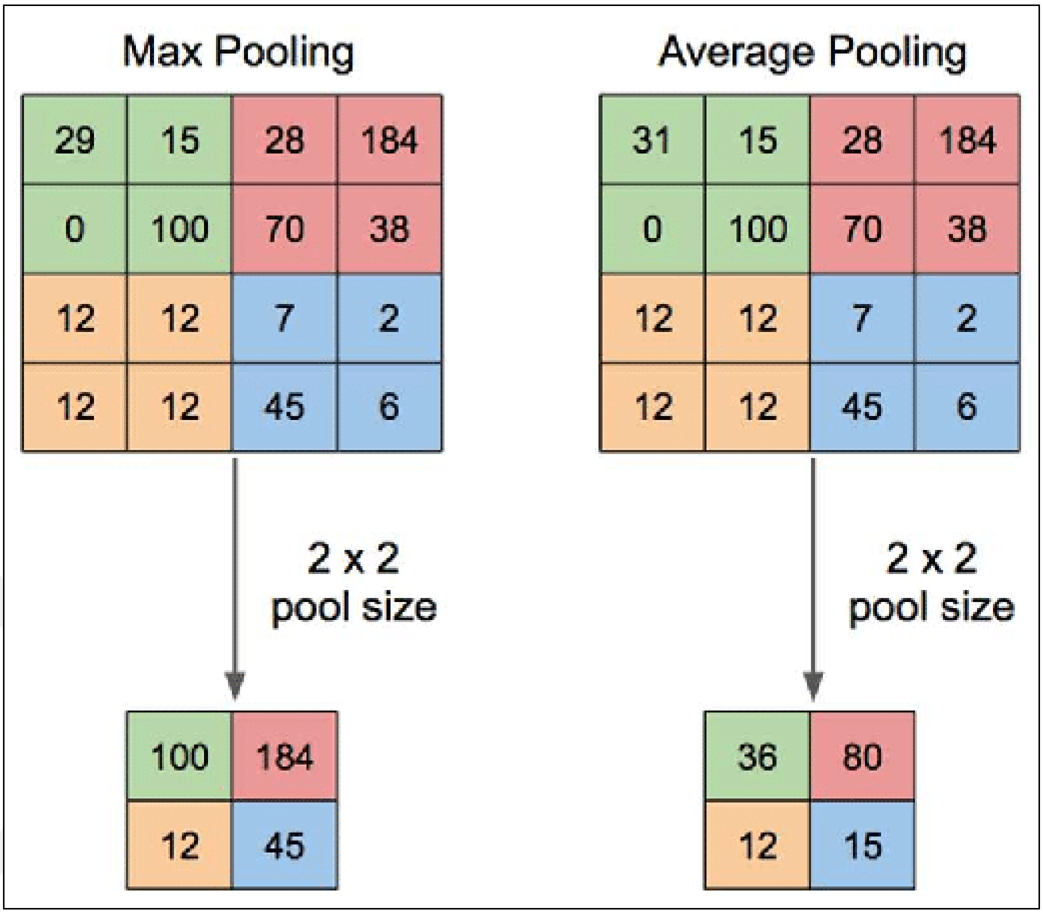
We start with defining a particular filter, this filter is slid over all the image to create a new one by using dot product to between filter and input image chunks, the output is a new scalar image, after applying the filter to the whole image, the output is image known as activation map.

Activation map size computed according to the following simple formula (M-I)/stride +1, where M equals the length of the input (image length), I equals the extent of filter and stride represents the sliding length. For example, if M =64, I= 3 and stride = 2 the output length equals to = (64-3)/ (2+1), at the end if we have more convolution layers the depth of the image continue to expand more and more as this layer is known as depth expanding layer.

3.5.2 Pooling Layer

The next step in CNN general architecture called pooling; this layer is responsible in terms of dimensionality reduction, computational complexity, and minimizing parameters total number in model. In general, two basic types of pooling layer are considered as common and basic in term of popularity and use, average pooling, and max-pooling layer. Average pooling is done by down-sampling the number of features into a rectangle shape and average the values of the original form for a specific amount of pixels. On the other hand, max-pooling is reducing the number and size of features by down-sampling the volume using the maximum value of the rectangle-shaped. Figure 3.5 below gives an example of both of max and average pooling.

Figure 3.5: An example of both of max and average pooling layer.



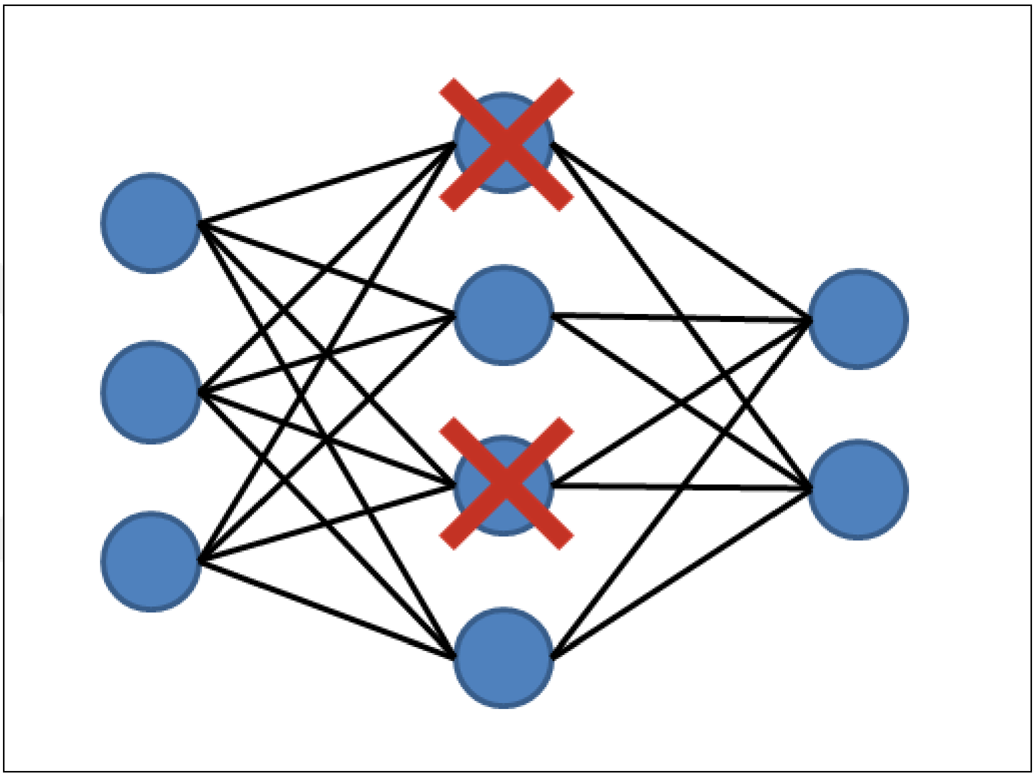
Source: Yani, Muhamad & Irawan, S, & ST, MT.. (2019). Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail. Journal of Physics: Conference Series. 1201

For the previous example, the pool size is assigned to (2x2), which offers an output that contains 4 pixels that represent values of 16 pixels. Max pooling is more common in deep learning than average pooling because of its nature in extracting extreme features more than average pooling more smoothly.

3.5.3 Dropout layer

While the network goes deeper and deeper, the amount of parameters is getting higher also, this tremendous number of parameters gives the model a chance to be overfitted, overfitting in general means that model is remembering not learning from data. Dropout is considered as one of the regularizations techniques that helps in overfitting reduction, as its name dropout layer core idea is to cut a specific number of layers during the training phase, this method prevents the network from taking the same paths every time in the learning phase. Dropout layer randomly drops the node according to dropout rate, a dropout rate value is a percentage value between zero and 1 used to cut a certain number of nodes in the training phase, for example, if dropout rate is 0.25 than quarter of the nodes randomly drops, Figure 3.6 below gives an example of a dropout layer with a dropout rate equals to 0.5.

Figure 3.6: An example of a dropout layer with dropout rate of 0.5.



Source: Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014, Jun 15).

Dropout is considered as a fundamental and core technique in the overfitting problem, and it improved the results of the learning process in supervised learning filed such as computer vision and speech recognition (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014).

3.5.4 Flatten Layer

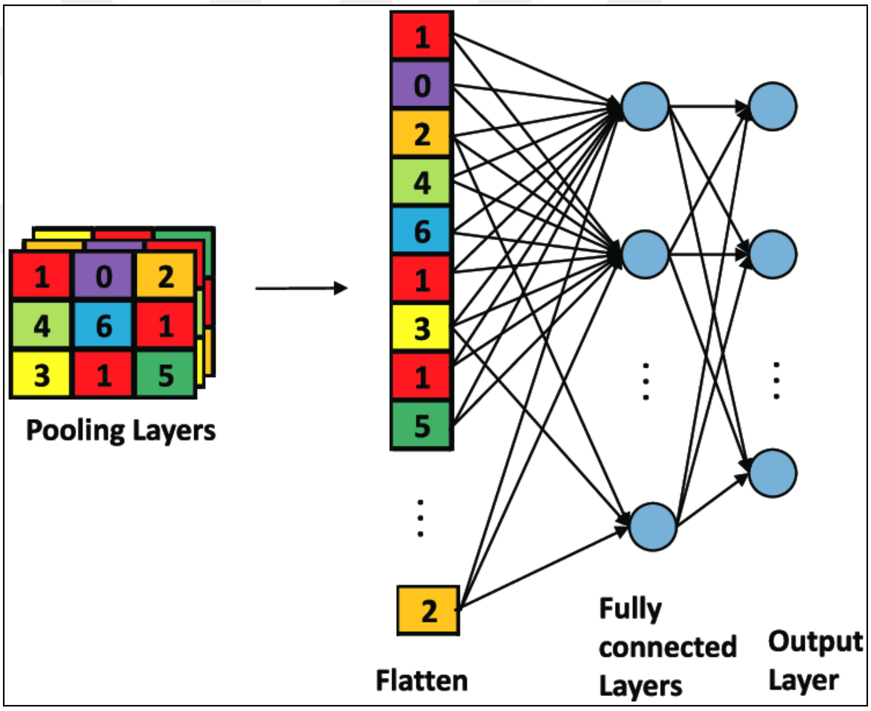
the final layer of the convolution network is called a fully connected layer (Dense), flatten layer deals with sequential data (1D), data from previous layers such as

Convolution layer comes in the form of 2D shape, dense layer can't perform with targeted input.

For this problem a layer should be in the middle between dense layer and 2D shape input from previous layers to solve the dimensionality problem, this layer called dense, the main goal of dense layer is to convert 2D input into sequential data in a shape of vector (1D).

Flatten layer break the structure of the multi-dimensional data and transform it into shape of 1D vector of data by applying the following formula((Z-(T-1), H - (M-1), N)) , where Z is the width of image, T is the height , M is size of the filter and N is number of filters. Figure 3.8 below gives an example of flatten layer.

Figure 3.7: An example of a flatten layer with an input comes from pooling layers

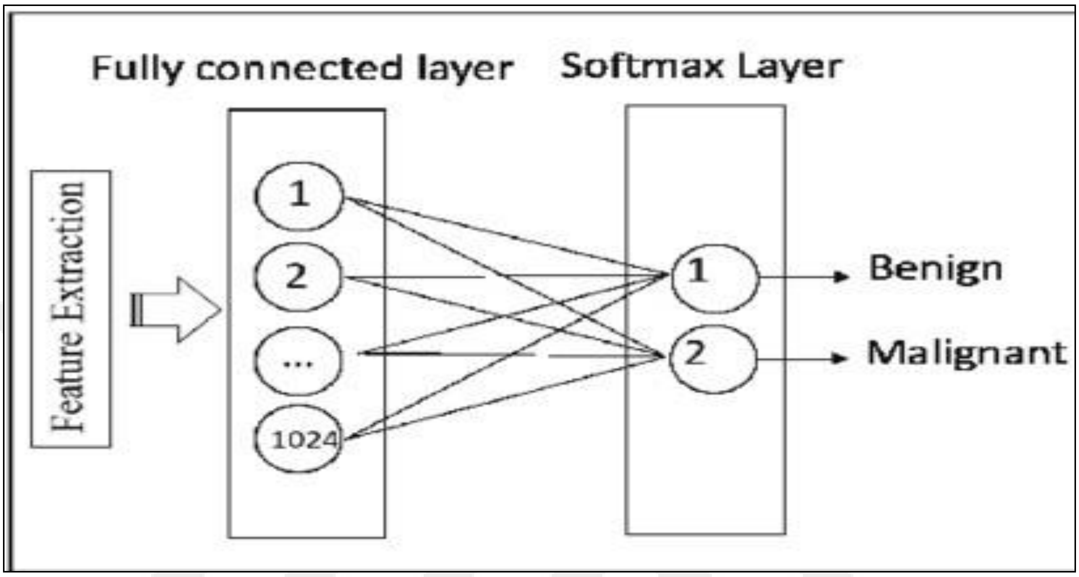


Source: Ng, W., Minasny, B., Montazerolghaem, M., Padarian, J., Ferguson, R., Bailey, S., & McBratney, A. B. (2019)

3.5.5 Fully Connected Layer (Dense)

The last layer in the convolutional neural network is called a fully connected layer also known as dense. A fully connected layer is responsible for classifying the input into different categories by extracting high-level features from them has been learned from the model. Fully connected layer deals with sequential data that comes from the flatten layer its labels them into their classes by using an activation function specified according to the performed task from the network. Figure 3.7 below gives an example of a fully connected layer.

Figure 3.8: Gives an example of a dense layer for an ANN.



Source: Soriano, D., Aguilar, C., Ramirez-Morales, I., Tusa, E., Rivas, W., & Pinta, M. (2017, November 8). Mammogram Classification Schemes by Using Convolutional Neural Networks.

3.5.6 Activation Functions

Activation function or transfer functions are defined as a function used to calculate the sum and bias of input to decide whether to drop the node or not, and Activation function provides the output for neural network layers by modifying the provided data and parameters in the layers using gradient batch processing.

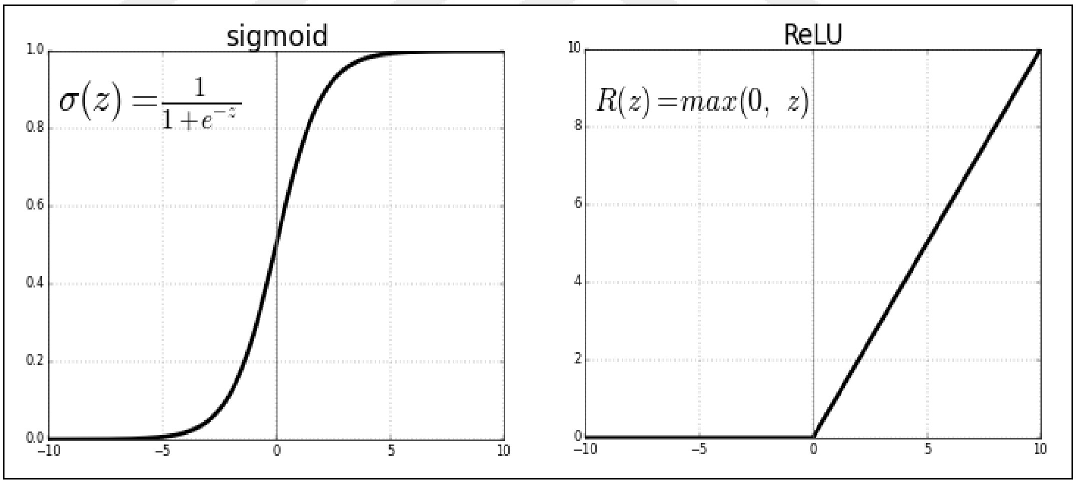
Activation functions are classified into linear and non-linear according to the tasks that it represents. Activation functions are widely used in deep learning techniques such as cancer detections system, fingerprint detection system.

Activation functions fill the gap between the data type and learning problem. A variety of activation functions exists with a different use gives a various output, for our problem, the activation functions that we used are mentioned below:

a. Rectified Linear Unit (ReLU): it is the most frequently used activation function in the structural learning application with state-of-the-art results, it had been introduced by Hinton and Nair in 2010. ReLU is a linear activation function that deletes all negative values by converting them into zero's within the following formula: *Y(x)= max (0,x)*, where Y represent our output and x is the input value, if x is a positive number it ReLU function returns values its self, otherwise if x is a negative number it return zero value instead of negative value.

b. Sigmoid: sigmoid function also known as logistic function is a type of a non-linear activation function that is used in neural network, sigmoid function is mainly used for predicting the output in deep learning models in binary classification problems by the following formula: $T(x)=1/(1+e^{-x})$, where T represents output and x represents input value, sigmoid function always provide values between -1 and 1 as shown in figure 3.10 below while ReLU provides values between 0 and maximum values in our input.

Figure 3.19: A visual representation of sigmoid and ReLU activation functions.



Source: Sharma, S. (2019, February 14). Activation Functions in Neural Networks

3.6 LAYERS AND NETWORK ARCHITECTURE

VGG16 model inspired our convolutional neural network. The model contains five convolutional layers. Each single convolutional layer is followed by a single max-pooling layer to separate each convolutional layer from each other. Before moving the data into the dense layers, we used a Flatten layer to convert the output of the layers to sequential, next layers two dense layers with one dropout layer to determine the class of the images.

Table 3.1: Shows our model architecture

	TYPE	*Filter Size*	*Stride*	*# of filters*	*FC Units*	*Activation*
Layer 1	Conv.	3x3	1x1	128	-	ReLU
Layer 2	Max-Pooling	2x2	2x2	-	-	-
Layer 3	Conv.	2x2	1x1	128	-	ReLU
Layer 4	Max-Pooling	2x2	2x2	-	-	-
Layer 5	Conv.	2x2	1x1	128	-	ReLU
Layer 6	Max-Pooling	2x2	2x2	-	-	-
Layer 7	Conv.	2x2	1x1	256	-	ReLU
Layer 8	Max-Pooling	2x2	2x2	-	-	-
Layer 9	Conv.	2x2	1x1	256	-	ReLU
Layer 10	Max-Pooling	2x2	2x2	-	-	-
Layer 11	Flatten	-	-	-	-	-
Layer 12	FC (Dense)	-	-	200	-	ReLU
Layer 13	Dropout	-	-	-	-	-
Layer 14	FC (Dense)	-	-	1	-	Sigmoid

a. Convolutional Layer

Our solution contains 5 convolutional layers the first three-layer have a 128 filter each with a filter size equals to 3x3 for the first one of them while the rest comes with filter size equals to 2x2, while the last two convolutional layers have 256 filter size with filter size equals to 2x2, Large filter size is used in the model to capture the non-linearity of the images using large numbers of filters for each layer.

b. Max-Pooling Layer

Our solution contains five Max-Pooling layers; each single Max-Pooling comes after each single convolutional layer. For each Max-Pooling layer, the stride is assigned to 2x2, and filter size is also attached to 2x2.

c. Sigmoid Layer:

Last layer in our solution architecture is Sigmoid layer, and it is the final layer where the output is produced; it gives the probability of each image class.

d. Activation Functions:

We used two different kinds of activation functions in our model; first activation function is ReLU, it is used to bring the non-linearity output from the layers, where second activation function is sigmoid which is used in the last layer, which used to give the probability of images classes.

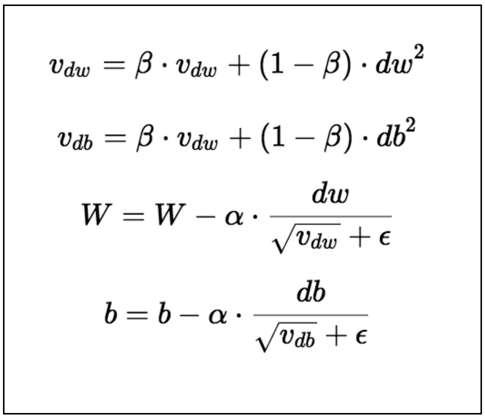
e. Loss Function:

A Categorical Cross Entropy (CSE) function has been used in model to represent the loss of Sigmoid function, a CSE function is defined in the following formula: $C = \sum T \log P$, for all predicted classes. Where T is the output and corresponding output of the Sigmoid layer.

f. Optimization Algorithm and learning rate:

We used a Root Mean Square Propagation (RMSprop) as an optimization algorithm, RMSprop optimizer guide the oscillations in a vertical direction so it can converge in somewhere in the training phase. Figure 3.11 shows RMSprop optimizer equation.

Figure 3.10: Describe RMSprop equation.



$$v_{dw} = \beta \cdot v_{dw} + (1 - \beta) \cdot dw^2$$

$$v_{db} = \beta \cdot v_{dw} + (1 - \beta) \cdot db^2$$

$$W = W - \alpha \cdot \frac{dw}{\sqrt{v_{dw}} + \epsilon}$$

$$b = b - \alpha \cdot \frac{db}{\sqrt{v_{db}} + \epsilon}$$

Source: Kathuria, A. (2018, August 20). Intro to optimization in deep learning: Momentum, RMSProp and Adam

Learning rate with a value equals to 0.0001 is used in our model to deliver the required output. Learning rate affects the overall performance of the model by changing the weight of the connected weights, which leads to overall performance changes in the proposed model.

3.7 CNN TRAINING PHASE

In the training phase, we used a google colab platform with the following parameters for our model to achieve the desired output:

a. Optimizer =RMSporp
b. Loss Type = Binary cross-entropy.
c. Learning rate = 1e-4.
d. Number of epochs = 30.
e. Steps per epoch= 5.
f. Validation Steps = 3.

All the previous parameters are used to achieve the desired output from our model, by tuning the parameters of the model in each step of the training process, using a cross-validation techniques with number of fold equals to five to counter overfitting.

3.8 EVALUATION METRIC

For assessing the CNN results, there are some well-known measurements utilized as benchmarks for contrasting and dissecting how efficient the model is, for our model recall, precision, accuracy, and F1-score used to measure our proposed model performance.

a. Accuracy: The accuracy of the classification of the method depends on the number of correctly categorized objects (Ojala, Pietikäinen, & Mäenpää, 2002). In object detection accuracy is a technique to measure to predicted how much is the predicted values according to its original values, accuracy is calculated by taking the number of correct prediction and divide it on the number of labeled data class that belongs to it, the formula of accuracy is: Accuracy = (TP+TN)/N, Where TP means true positive, and TN means true negative.

b. Recall: this is a technique used in the process of determining how the right values were predicted in our proposed model. Recall calculated using the following formula: Recall= TP/(TP+FN), where FN is false negative.

c. Precision: It is a technique used to the ratio of correct values that were predicted in the model. Precision is calculated using the following formula: Precision = TP/(TP+FP), where FP is false positive.

d. F1-score: This is a technique to find the balance between the precision and recall in the term of having an imbalanced classification dataset. F1-score is calculated using the following formula: 2*((Precision * Recall) / (Precision + Recall)).

All the previous techniques and benchmarks are driven from an output map called confusion matrix, and it is a map that contains four different components used in classification problem to measure the values of the mentioned above metrics. Confusion matrix component are:

a. True Positive (TP): TP is the case when the positive prediction is classified as it originally labeled.

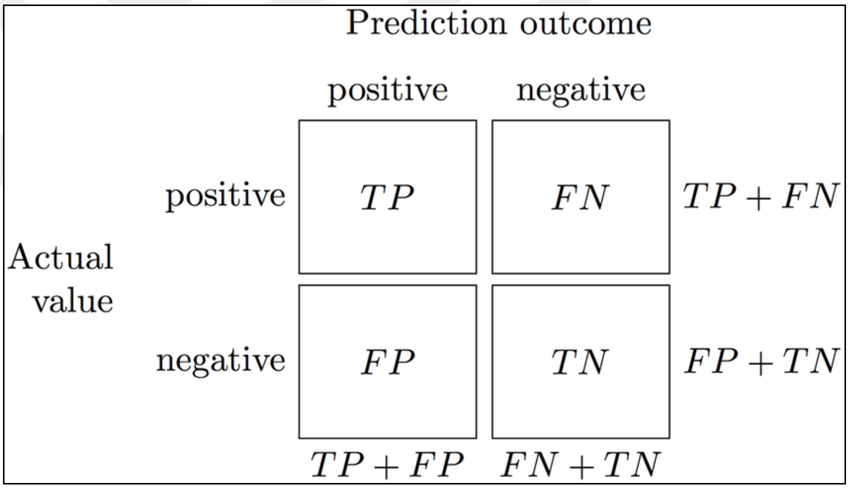
b. True Negative (TN): FP is the case when the negative predication is classified as it is labeled originally.

c. False Positive (FP): FP is the situation when the negative element is classified into a positive label.

d. False Negative (FN): it is the situation when the positive element is classified into a positive label.

Usually a confusion matrix is shown as in figure 3.12 below where TP is standing to the case where Yes meets to Yes cases in both sides of predicated and actual labels, TN stands to the case where NO meets NO cases in both sides of predicated and actual labels, FP stands to the cases where NO is the actual label but it meets to Yes labels in the predicted labels, and finally an FN stand for the cases where YES is the actual labels but it meets NO in the predicated labels.

Figure 3.11: Describes a general confusion matrix for a two classes classification problem.



		Prediction outcome		
		positive	negative	
Actual value	positive	TP	FN	$TP + FN$
	negative	FP	TN	$FP + TN$
		$TP + FP$	$FN + TN$	

Source: Masías, V. H., Valle, M., Morselli, C., Crespo, F., Vargas, A., & Laengle, S. (2016). Modeling Verdict Outcomes Using Social Network Measures: The Watergate and Caviar Network Cases

4. RESULTS

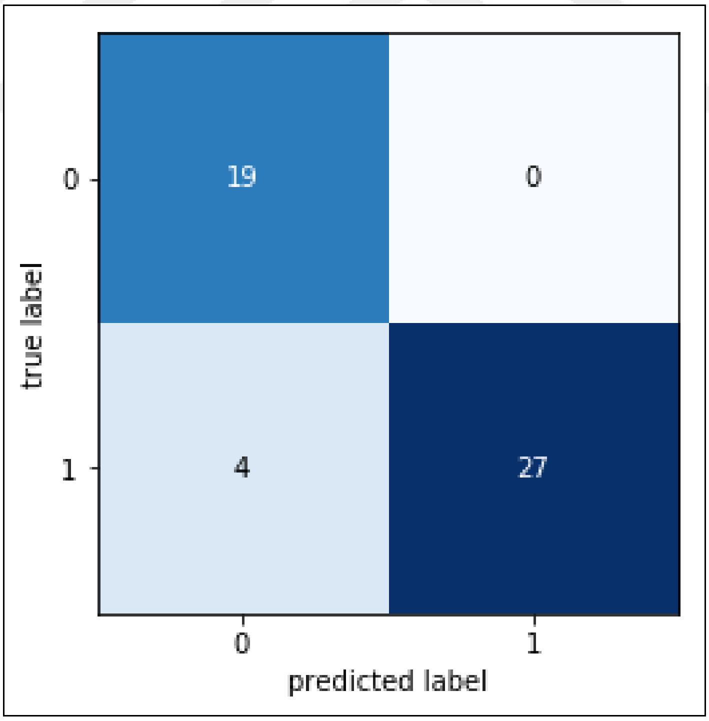
This chapter presents the results of our solution, all practical results of our convolutional neural network are presented and discussed in detail in this chapter, all the results are tabulated, discussed and evaluated in this chapter.

4.1 CROSS-VALIDATION EVALUATION

A cross-validation method applied to our datasets with the number of folds equals five; for each fold, a confusion matrix is used to calculate accuracy, recall, precision, and F1-score. At last, an average of the evaluation metrics is used to evaluate overall model output in general.

4.1.1 First Fold Results

Figure: 4.1: Fold One Confusion Matrix



By applying the mentioned evaluation metrics mentioned in section 3.7 on our obtained results from figure 4.1 which represent confusion matrix for the first fold where zero represent brain MRI without tumor images and 1 represents images with brain tumors, we obtain the following results:

a. Accuracy: (TP+TN)/N = (19+27)/50 = 0.92

b. Precision: TP/(TP+FP) = 19/ (19+ 4) = 0.83

c. Recall: TP/(TP+FN) = (19/ (19+0)) = 1.00

d. F1-score: 2*((Precision * Recall) / (Precision + Recall)) =0.91

The above obtained results represent the results of fold one applied on our testing set, which delivers are average results comparing to other folds. For the training process figure 4.2 represent the relationship between model accuracy in both training set and test set with the number of epochs, while figure 4.3 represents the relationship between model loss for training and validation sets in with the number of epochs.

Figure 4.2: Relationship between model accuracy with epochs in the training phase.

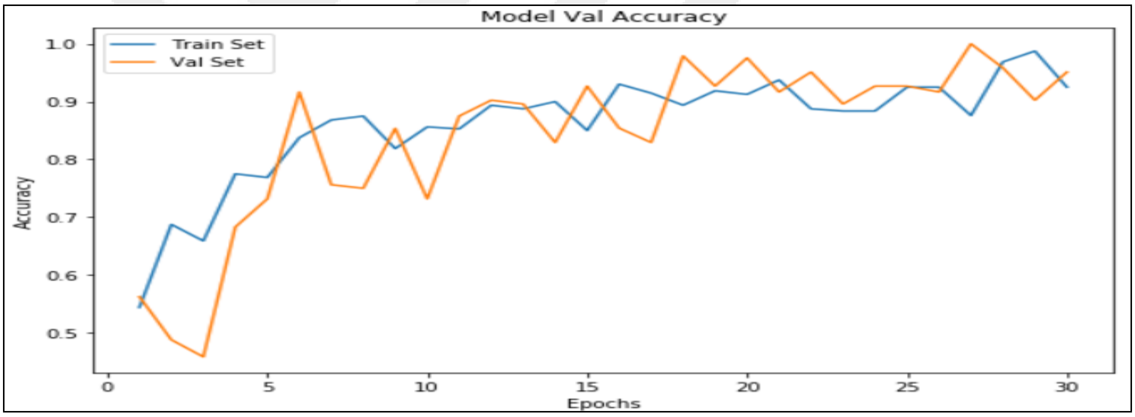
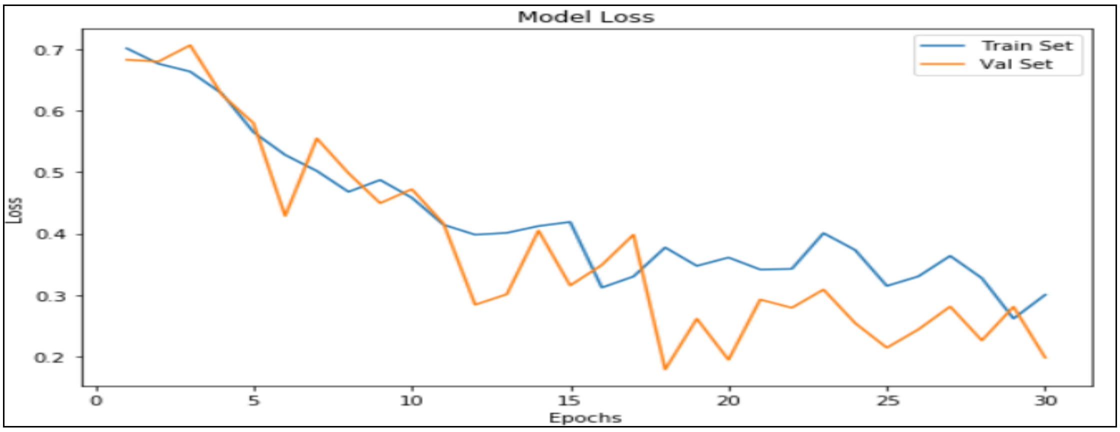
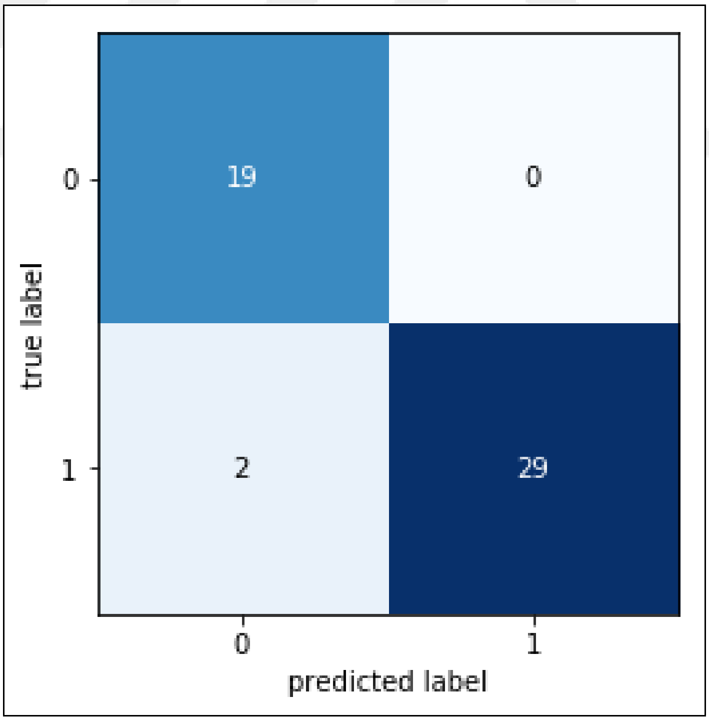


Figure 4.3: The relationship between model loss and epochs in the training phase.



4.1.2 Second Fold Results

Figure 4.4 Second Fold Confusion Matrix



The results of fold two on our testing set is obtained by applying the evaluation metrics on the confusion matrix of fold two is mentioned below:

a. Accuracy: (TP+TN)/N = (19+29)/50 = 0.96
b. Precision: TP/(TP+FP) = 19/ (19+2) = 0.90
c. Recall: TP/(TP+FN) = (19/ (19+0)) = 1.00
d. F1-score: 2*((Precision * Recall) / (Precision + Recall)) =0.95

Fold two gave the highest scores in terms of evaluation metrics; it is considered as the best case in terms of evaluation for the testing set. For training and validation sets accuracy and loss figures, 4.5 is representing the relationship between epochs and training accuracy, while figure 4.6 is representing the relationship between epochs and the loss for training and validation sets.

Figure 4.5: **Model Accuracy for training and testing sets in the training phase for the second fold.**

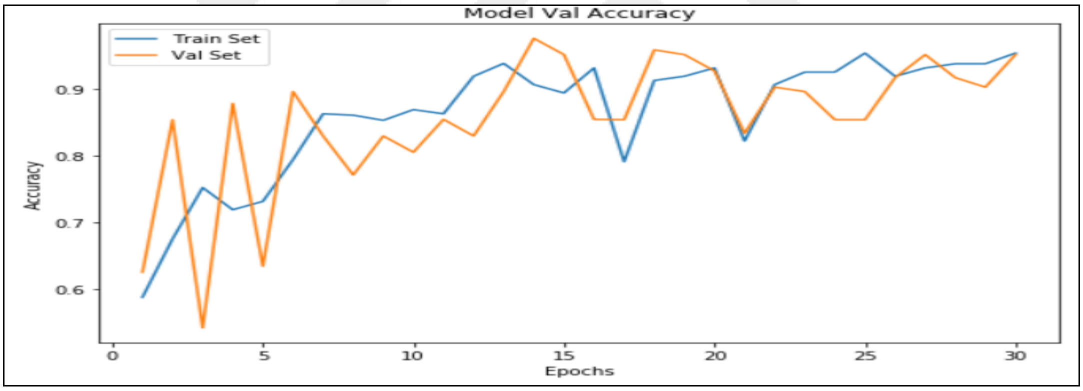
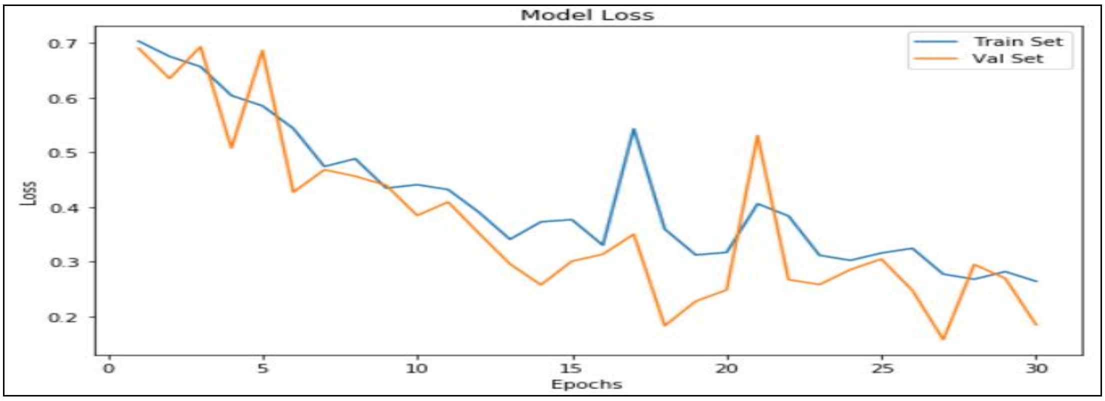
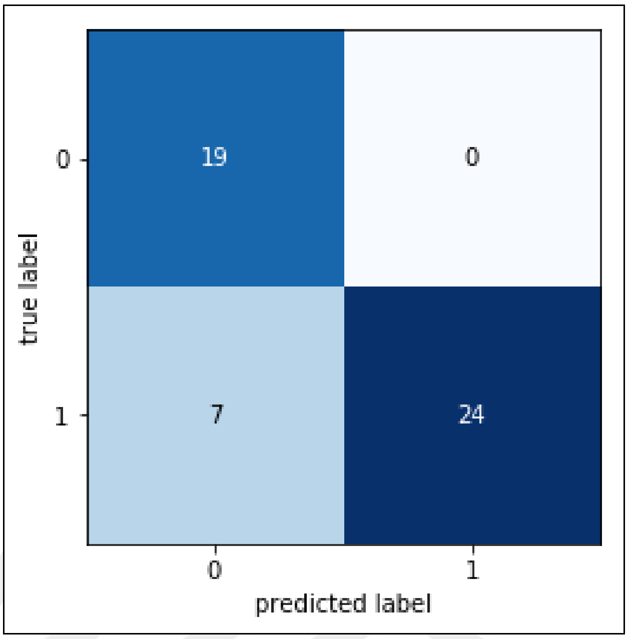


Figure 4.6: **Model loss for training and testing sets in the training phase for the second fold.**



4.1.3 Third Fold Results:

Figure 4.7 Third fold confusion matrix



By applying the evaluation metrics on confusion matrix of fold three, the following results obtained:

a. Accuracy: (TP+TN)/N = (19+24)/50 = 0.86
b. Precision: TP/(TP+FP) = 19/ (19+7) = 0.73
c. Recall: TP/(TP+FN) = (19/ (19+0)) = 1.00
d. F1-score: 2*((Precision * Recall) / (Precision + Recall)) =0.84

From the results above, the third fold is considered as worst fold in term of performance, for training and validation sets the figures below describe the relationship in terms of accuracy and loss with epochs.

Figure 4.8: Model Accuracy for training and validation sets for the third fold.

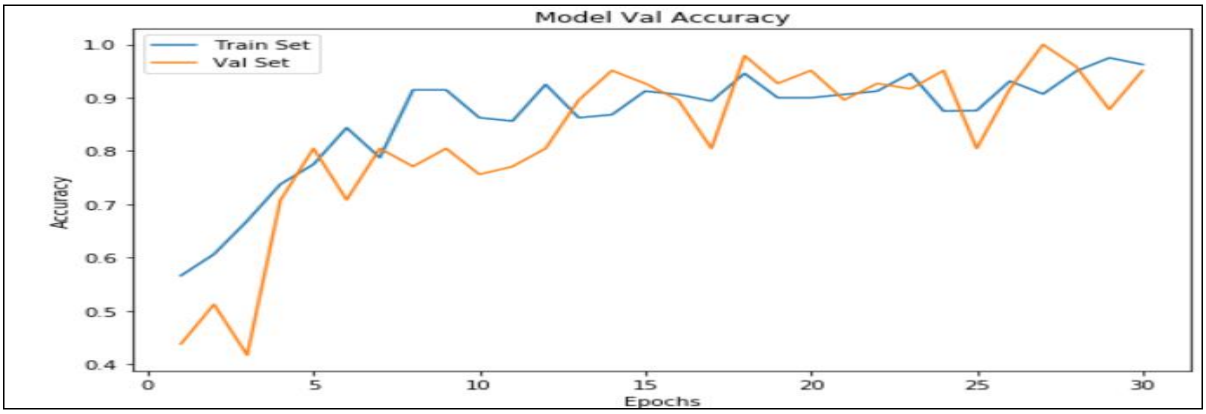
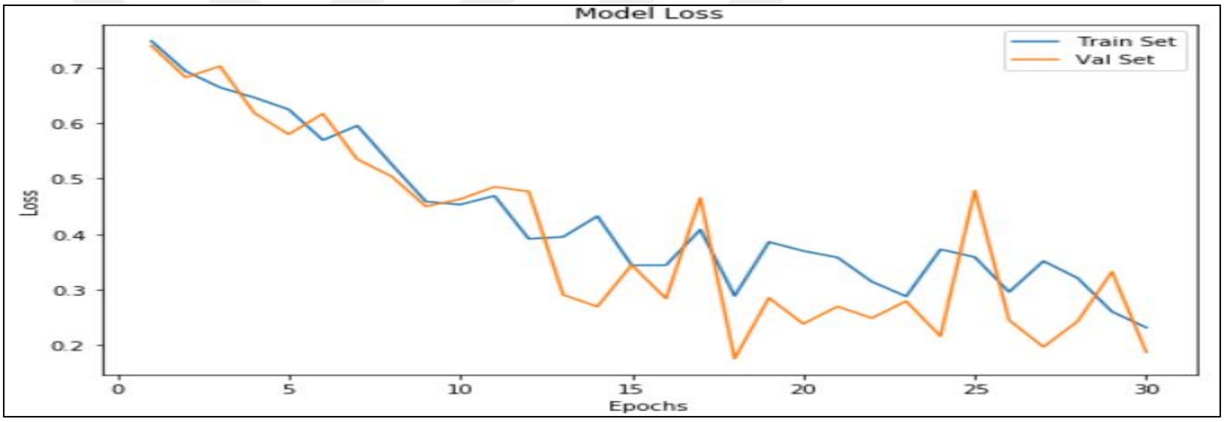
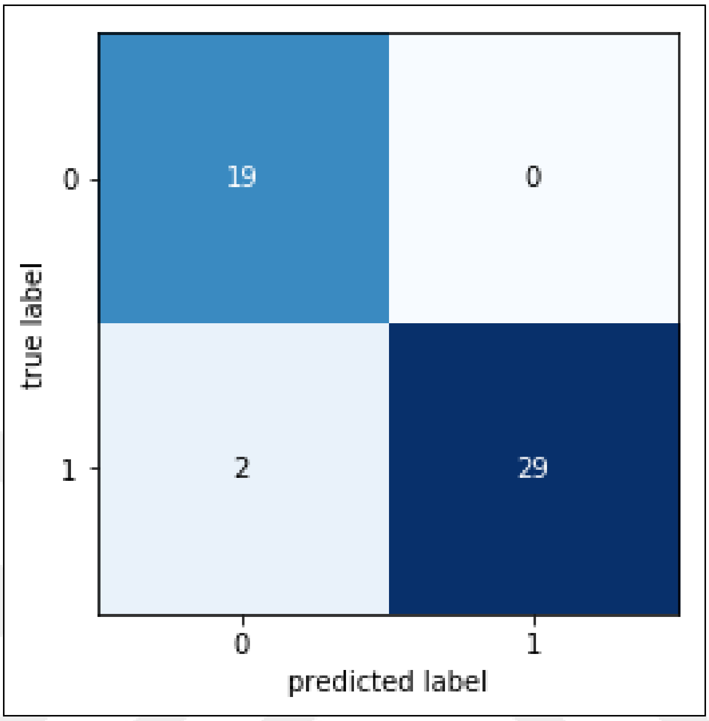


Figure 4.9: Model loss for training and validation sets for the third fold.



4.1.4 Fourth Fold Results:

Figure 4.10: Confusion matrix for fourth fold



From figure 4.10 above the following results are acquired:

a. Accuracy: (TP+TN)/N = (19+29)/50 = 0.96
b. Precision: TP/(TP+FP) = 19/ (19+2) = 0.90
c. Recall: TP/(TP+FN) = (19/ (19+0)) = 1.00
d. F1-score: 2*((Precision * Recall) / (Precision + Recall)) =0.95

This fold results similar to fold two results, it gave highest evaluation scores for training set, for training and validation sets figures below describes accuracy and loss for the model in the training approaches.

Figure 4.11: Relationship between accuracy and epochs for the fourth fold.

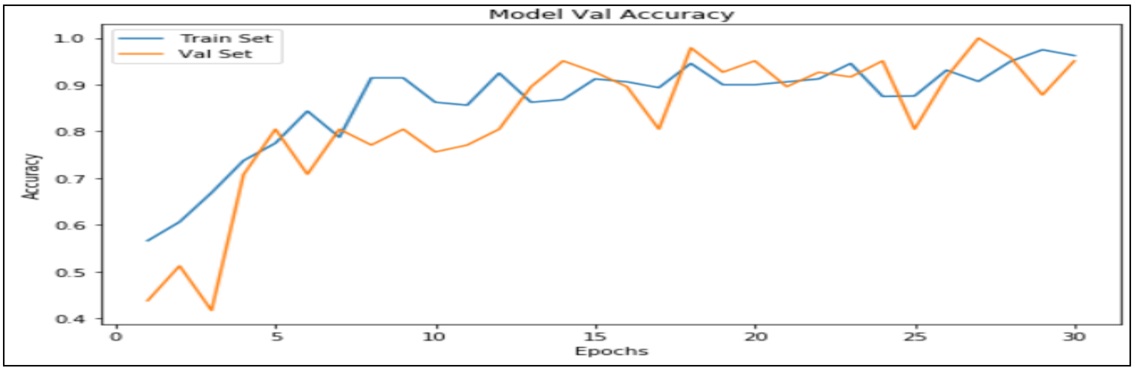
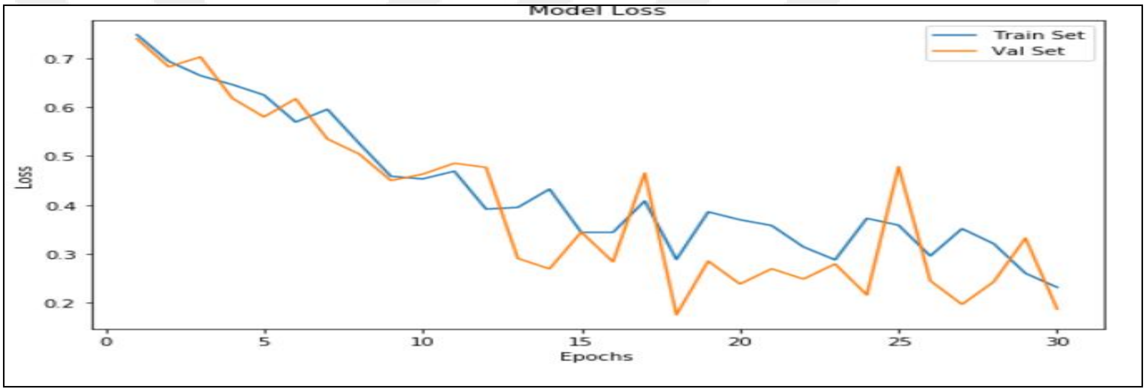
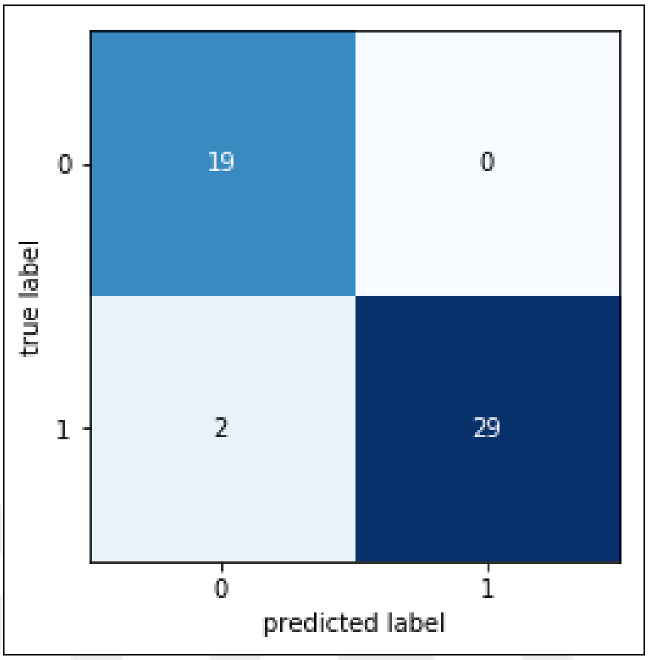


Figure 4.12: Relationship between model loss and epochs for the fourth fold.



4.1.5 Fifith Fold Results

Figure 4.13 Fifth Fold Confusion Matrix



As fold four, this fold results are similar to the second fold, and it delivers the same results in terms of our evaluation metrics.

a. Accuracy: (TP+TN)/N = (19+29)/50 = 0.96

b. Precision: TP/(TP+FP) = 19/ (19+2) = 0.90

c. Recall: TP/(TP+FN) = (19/ (19+0)) = 1.00

d. F1-score: 2*((Precision * Recall) / (Precision + Recall)) =0.95

For training set and validation set accuracy and loss in training approach each fold is different than each other.

4.1.6 Overall Results

In the last of the model evaluation, we take an average of all of the evaluation metrics to make a general result to judge our model performance; Table 4.1 shows the results of all folds and their averages.

Table 4.1: Folds results with averages of the results.

Fold	Accuracy	Precision	Recall	F1-score
Fold One	0.92	0.83	1.00	0.91
Fold Two	0.96	0.90	1.00	0.95
Fold Three	0.86	0.73	1.00	0.84
Fold Four	0.96	0.90	1.00	0.95
Fold Five	0.96	0.90	1.00	0.95
Average	0.93	0.85	1.00	0.92

From the previous results our model classified all normal brains MRI images totally accurate, but for tumor MRI images our error rate is equals to 0.15, in average of classifying brain MRI images our solution achieved an average result equals to 0.93, while our dataset is slightly unbalanced we also relied on F1-score to evaluate our final model performance from table 4.1 average result of our solution for F1-score is 0.92.

5. CONCLUSION

The purpose of this study is to develop a deep learning algorithm that detects brain cancer and tumor from images from Magnetic Resonance Images (MRI) with high accuracy, recall, precision and F1-score, for a further understanding of the problem we had, an MRI image were studied, how are MRI images collected to show different brain tissues, how doctors diagnose brain tumor manually, then we present a literature review about brain tumor detection with machine learning and deep learning, after that we presented our solution of brain tumor detection using CNN and present it results using cross-validation technique, after comparing our folds results to each other in section 4.1.6

for the aim of brain tumor detection, we built a deep learning architecture that relies on CNN algorithm to detect brain tumors from MRI images, data gathering, and model details are in chapter three, while the results are described in chapter four.

Our deep learning solution for brain tumor detection provided a maximum result for accuracy equals 0.96, precision 0.90, recall 1.00, and for F1-score, the maximum result is 0.95. Those results came out from 3 different folds, while we had an average result for accuracy equals 0.93, precision 0.85, recall 1.00, and for F1-score 0.92. From our maximum and average results of folds, we say that our main aim of the solution is achieved.

6. DISSUCION

6.1 LIMITATIONS

a. Our model is a deep learning model which needs a large dataset to learn model, comparing to the datasets in the filed this dataset is classified a small dataset, we could not succeed to obtain a larger dataset of brain tumor MRI images from a local hospital, so we used a dataset from the internet that contains a limited number of images that meets our criteria.

b. Limited computational resources prevented us from creating a deeper model to extract more features from our dataset.

c. Limited time to finish the project, some parts did not explore deeper and implemented better if more time were given.

6.2 FUTURE WORK

Since we achieved the main goal of this solution, there is some parts of the solution needs further work and development:

a. Gathering more data: gathering more data was one of the major limitations of the project to build a deeper network.

b. Applying image segmentation technique: The next step for this model should be segmentation after detection instead of general detection, we should perform image segmentation to localize brain tumors.

c. Increasing the precision value: by gathering and adding more training data to our model, it reduces the false positive cases, which increase the precision value.

REFERENCES

Books

Deng, L.; Yu, D. (2014). “Deep Learning: Methods and Applications” (PDF). Foundations and Trends in Signal Processing. 7 (3–4): 1–199

Periodicals

Ostrom, Q. T., Gittleman, H., Truitt, G., Boscia, A., Kruchko, C., & Barnholtz-Sloan, J. S. (2018). CBTRUS Statistical Report: Primary Brain and Other Central Nervous System Tumors Diagnosed in the United States in 2011–2015. Neuro-Oncology, 20(suppl_4), iv1–iv86. doi: 10.1093/neuonc/noy131

Wrensch, M., Bondy, M. L., Wiencke, J., & Yost, M. (n.d.). Environmental risk factors for primary malignant brain tumors: A review. Retrieved from https://link.springer.com/article/10.1007/BF01054274.

Griesinger, A. M., Birks, D. K., Donson, A. M., Amani, V., Hoffman, L. M., Waziri, A., … Foreman, N. K. (2013, November 1). Characterization of distinct immunophenotypes across pediatric brain tumor types. Retrieved from https://www.ncbi.nlm.nih.gov/pubmed/24078694.

Vézina, L.-G. (2008, October). Imaging of central nervous system tumors in children: advances and limitations. Retrieved from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2841016/.

David J. Werring, Duncan W. Frazer, Lucy J. Coward, Nick A. Losseff, Hilary Watt, Lisa Cipolotti, Martin M. Brown, H. Rolf Jäger, Cognitive dysfunction in patients with cerebral microbleeds on T2*-weighted gradient-echo MRI, *Brain*, Volume 127, Issue 10, October 2004, Pages 2265–2275, https://doi.org/10.1093/brain/awh253.

Hajnal, J. V., Bryant, D. J., Kasuboski, L., Pattany, P. M., Coene, B. D., Lewis, P. D., … Bydder, G. M. (1992). Use of Fluid Attenuated Inversion Recovery (FLAIR) Pulse Sequences in MRI of the Brain. Journal of Computer Assisted Tomography, 16(6), 841–844. doi: 10.1097/00004728-199211000-00001

Kato, H., Izumiyama, M., Izumiyama, K., Takahashi, A., & Itoyama, Y. (2002). Silent Cerebral Microbleeds on T2*-Weighted MRI. Stroke, 33(6), 1536–1540. doi: 10.1161/01.str.0000018012.65108.86

Schölkopf Bernhard, Burges, C. J. C., & Smola, A. J. (1999). Advances in kernel methods: support vector learning. Cambridge, MA: MIT Press.

Vapnik, V. N. (2010). The nature of statistical learning theory. New York: Springer.

Yani, Muhamad & Irawan, S, & ST, MT.. (2019). Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail. Journal of Physics: Conference Series. 1201. 012052. 10.1088/1742-6596/1201/1/012052.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014, Jun 15). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Retrieved from http://jmlr.org/papers/v15/srivastava14a.html.

Ng, W., Minasny, B., Montazerolghaem, M., Padarian, J., Ferguson, R., Bailey, S., & McBratney, A. B. (2019). Convolutional neural network for simultaneous prediction of several soil properties using visible/near-infrared, mid-infrared, and their combined spectra. Geoderma, 352, 251-267. doi: 10.1016/j.geoderma.2019.06.016

Soriano, D., Aguilar, C., Ramirez-Morales, I., Tusa, E., Rivas, W., & Pinta, M. (2017, November 8). Mammogram Classification Schemes by Using Convolutional Neural Networks. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-72727-1_6.

Masías, V. H., Valle, M., Morselli, C., Crespo, F., Vargas, A., & Laengle, S. (2016). Modeling Verdict Outcomes Using Social Network Measures: The Watergate and Caviar Network Cases. Retrieved from https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0147248.

Sharma, K., Kaur, A., & Gujral, S. (2014, January 1). [PDF] Brain Tumor Detection based on Machine Learning Algorithms: Semantic Scholar. Retrieved from https://www.semanticscholar.org/paper/Brain-Tumor-Detection-based-on-Machine-Learning-Sharma-Kaur/3b3b9a3dccb582e1455d0264630cfc1e18798455.

Kumar, S., Dabas, C., & Godara, S. (2017, December 12). Classification of Brain MRI Tumor Images: A Hybrid Approach. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050917326455.

Qasem, S. N., Nazar, A., Qamar, A., Shamshirband, S., & Karim�, A. (2019). A Learning Based Brain Tumor Detection System. Computers, Materials & Continua, 59(3), 713–727. doi: 10.32604/cmc.2019.05617

Varuna Shree, N., & Kumar, T. N. R. (2018, March). Identification and classification of brain tumor MRI images with feature extraction using DWT and probabilistic neural network. Retrieved from https://www.ncbi.nlm.nih.gov/pubmed/29313301.

Other Publications

Stark Facts. (2018). Retrieved from https://www.braintumourresearch.org/campaigning/stark- facts.

King Hussein Cancer Foundation. (n.d.). Retrieved from http://www.khcc.jo/en.

Tahir, M.N. (2018). Classification and characterization of brain tumor MRI by using gray scaled segmentationand DNN.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. Neural Computation, 18(7), 1527–1554. doi: 10.1162/neco.2006.18.7.1527

Chatterjee, C. C. (2019, July 31). Basics of the Classic CNN. Retrieved from https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add

Erickson, Bradley; Akkus, Zeynettin; Sedlar, Jiri; Korfiatis, Panagiotis. (2017). Data From LGG- 1p19qDeletion. The Cancer Imaging Archive. DOI: 10.7937/K9/TCIA.2017.dwehtz9v

Kathuria, A. (2018, August 20). Intro to optimization in deep learning: Momentum, RMSProp and Adam. Retrieved from https://blog.paperspace.com/intro-to-optimization- momentum-rmsprop-adam/.

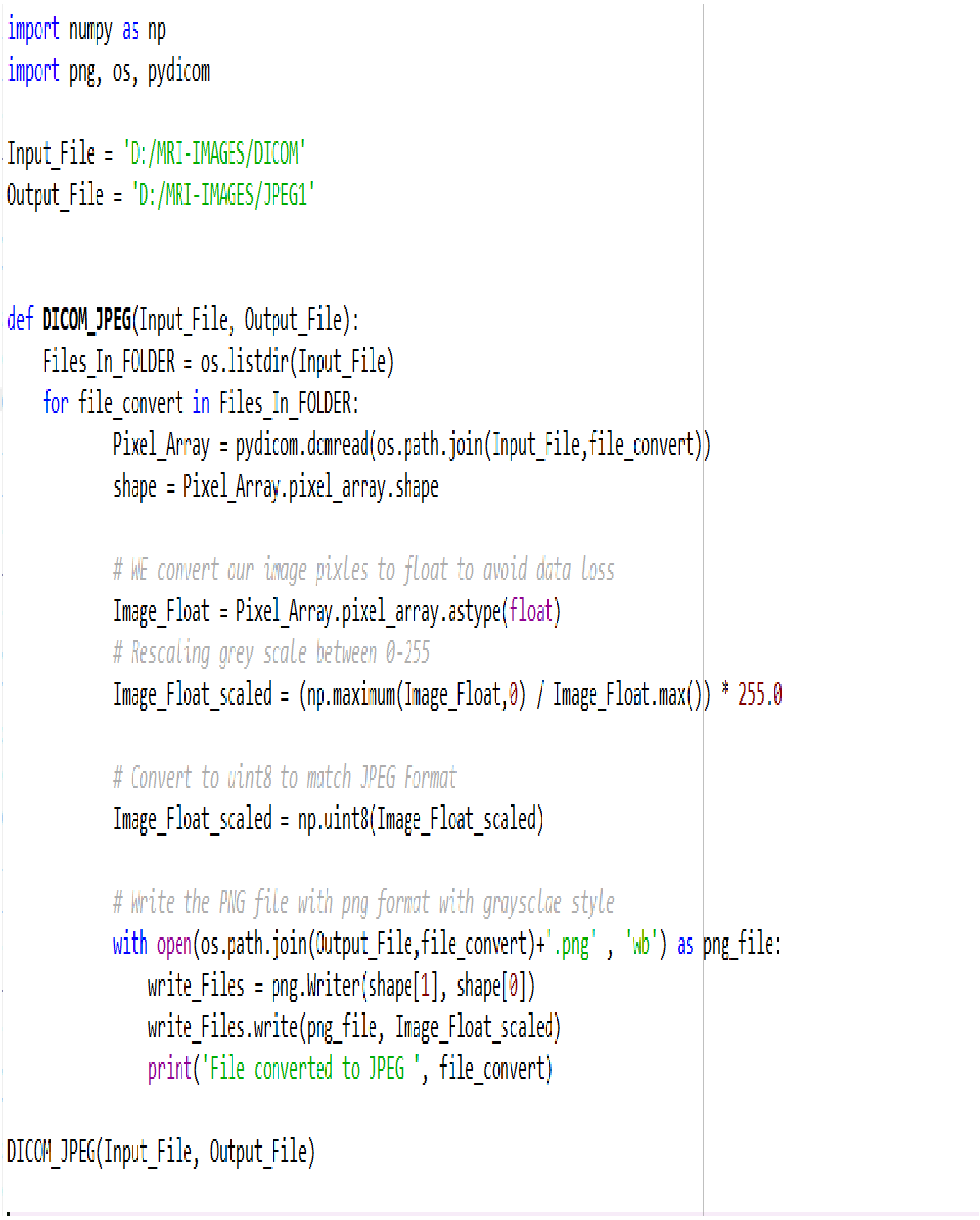
Subramanyam, Muthukumar. (2014). An Automatic Brain Tumor Detection and Segmentation Scheme for Clinical Brain Images.

Deshmukh, R.J., & Khule, R.S. (2014). Brain Tumor Detection Using Artificial Neural Network Fuzzy Inference System (ANFIS).

Sharma, S. (2019, February 14). Activation Functions in Neural Networks. Retrieved from https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

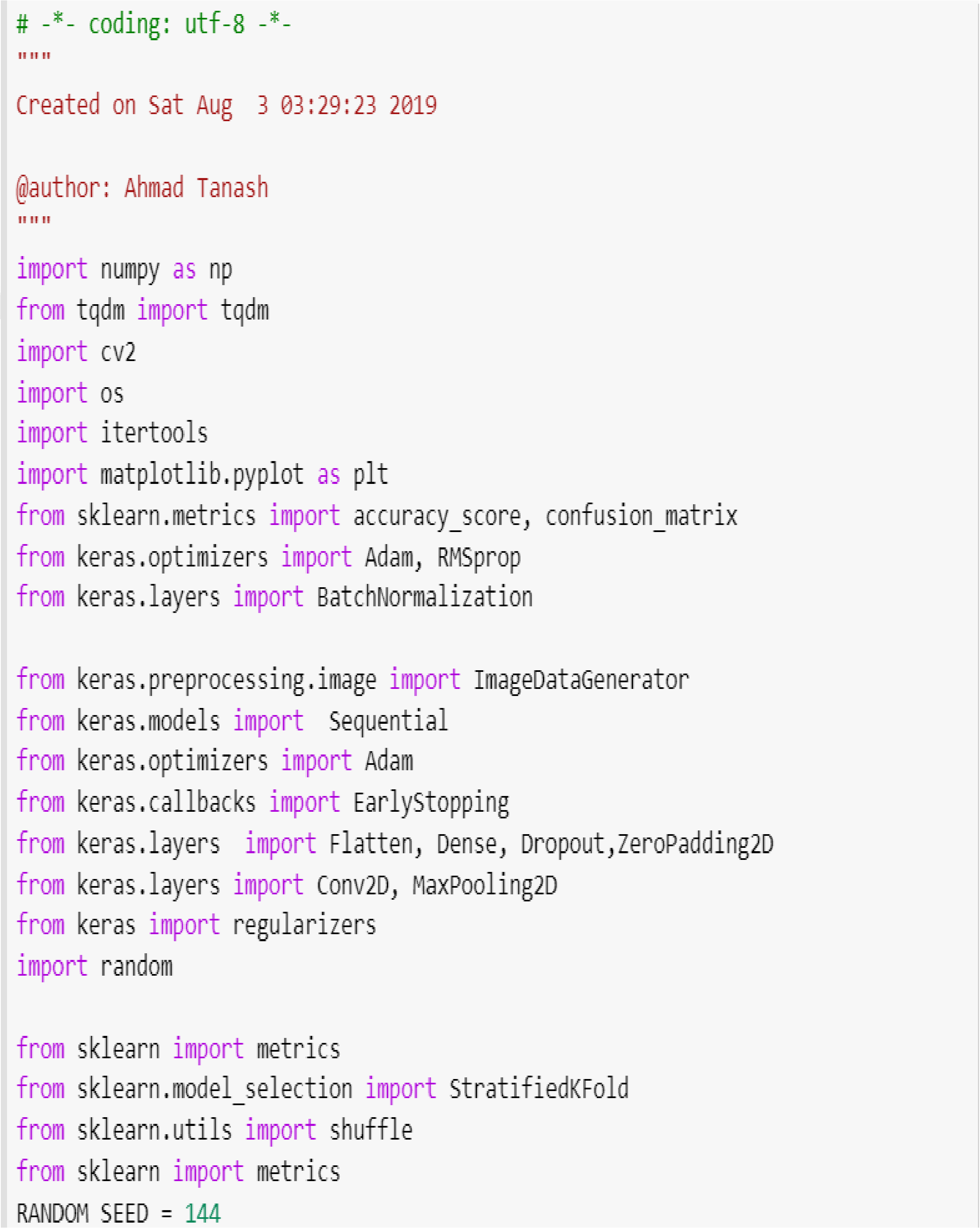
APPENDICES

Appendix A.1 DICOM to JPEG Code

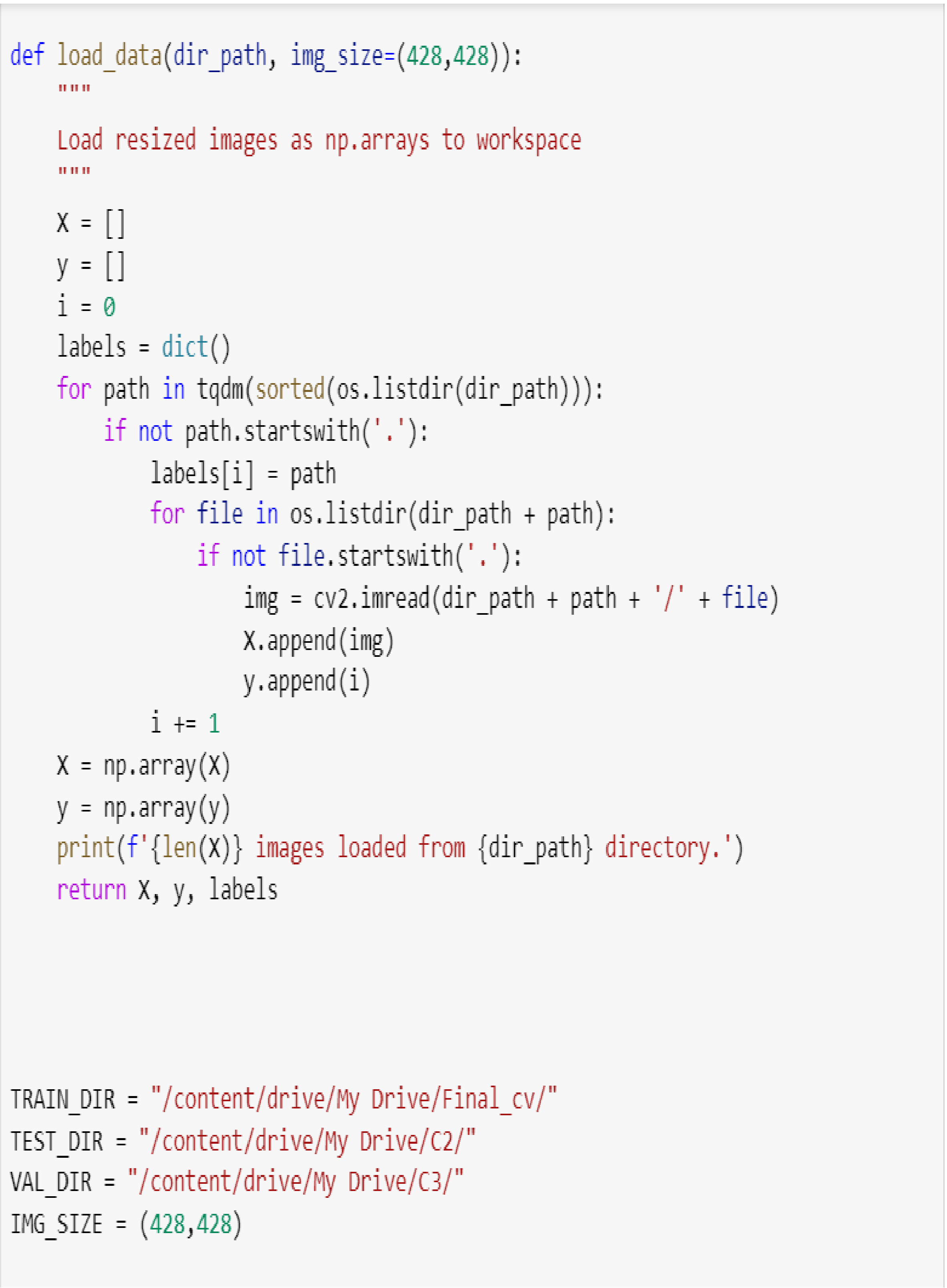


```
import numpy as np
import png, os, pydicom

Input_File = 'D:/MRI-IMAGES/DICOM'
Output_File = 'D:/MRI-IMAGES/JPEG1'


def DICOM_JPEG(Input_File, Output_File):
    Files_In_FOLDER = os.listdir(Input_File)
    for file_convert in Files_In_FOLDER:
            Pixel_Array = pydicom.dcmread(os.path.join(Input_File,file_convert))
            shape = Pixel_Array.pixel_array.shape

            # WE convert our image pixles to float to avoid data loss
            Image_Float = Pixel_Array.pixel_array.astype(float)
            # Rescaling grey scale between 0-255
            Image_Float_scaled = (np.maximum(Image_Float,0) / Image_Float.max()) * 255.0

            # Convert to uint8 to match JPEG Format
            Image_Float_scaled = np.uint8(Image_Float_scaled)

            # Write the PNG file with png format with graysclae style
            with open(os.path.join(Output_File,file_convert)+'.png' , 'wb') as png_file:
                write_Files = png.Writer(shape[1], shape[0])
                write_Files.write(png_file, Image_Float_scaled)
                print('File converted to JPEG ', file_convert)

DICOM_JPEG(Input_File, Output_File)
```

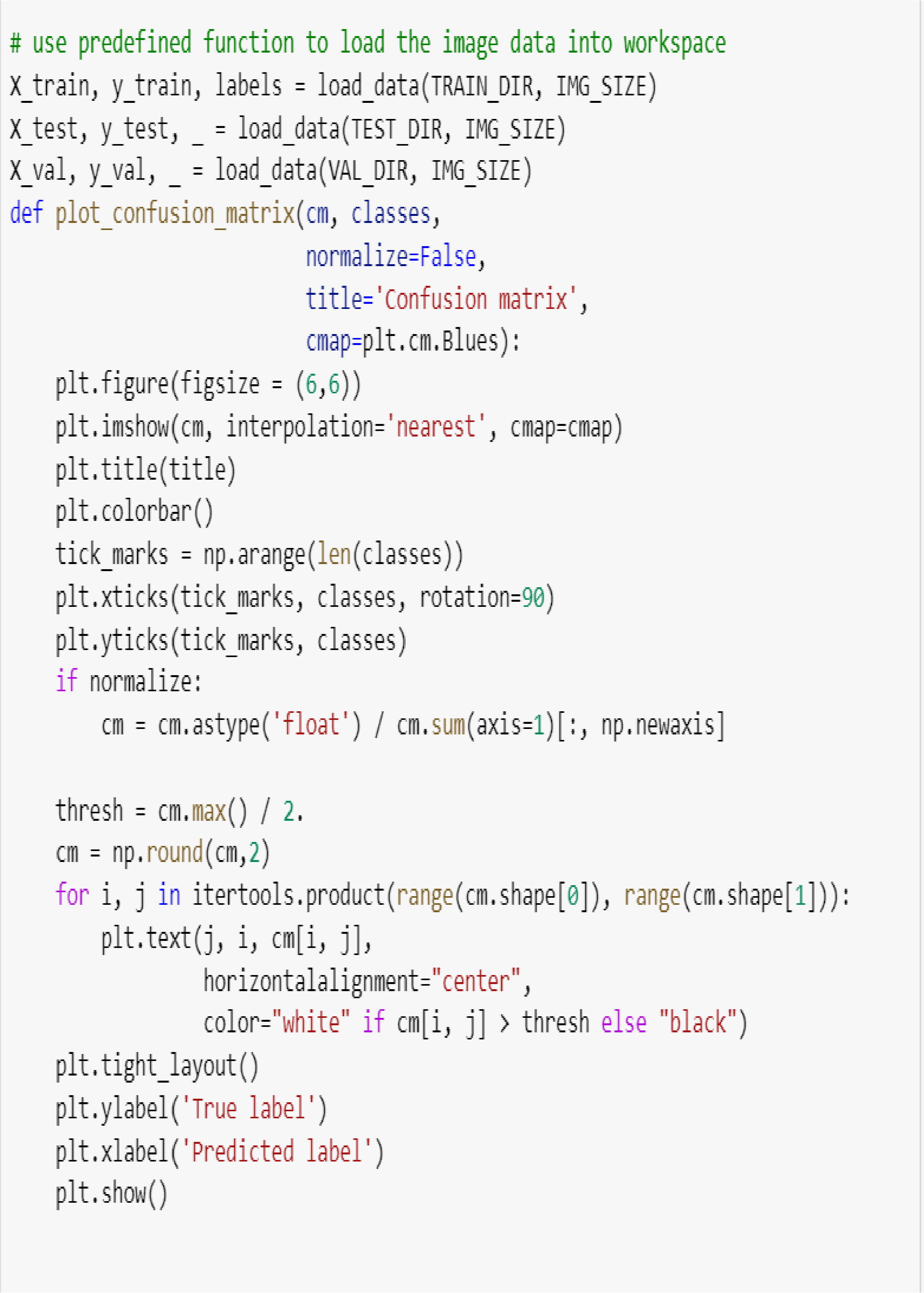
Appendix A.2 Source Code



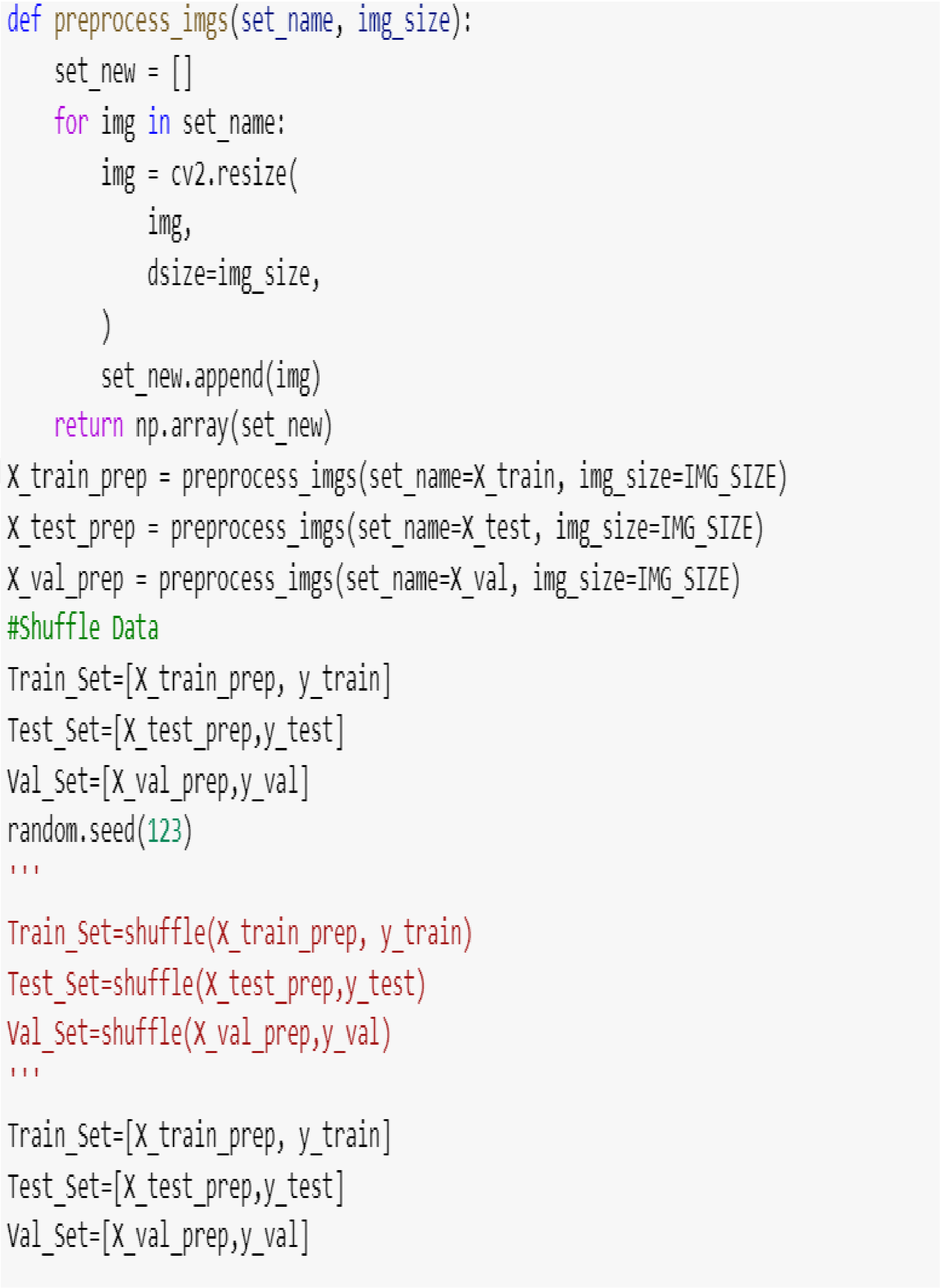
```
# -*- coding: utf-8 -*-
"""
Created on Sat Aug  3 03:29:23 2019

@author: Ahmad Tanash
"""
import numpy as np
from tqdm import tqdm
import cv2
import os
import itertools
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score, confusion_matrix
from keras.optimizers import Adam, RMSprop
from keras.layers import BatchNormalization

from keras.preprocessing.image import ImageDataGenerator
from keras.models import  Sequential
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping
from keras.layers  import Flatten, Dense, Dropout,ZeroPadding2D
from keras.layers import Conv2D, MaxPooling2D
from keras import regularizers
import random

from sklearn import metrics
from sklearn.model_selection import StratifiedKFold
from sklearn.utils import shuffle
from sklearn import metrics
RANDOM SEED = 144
```



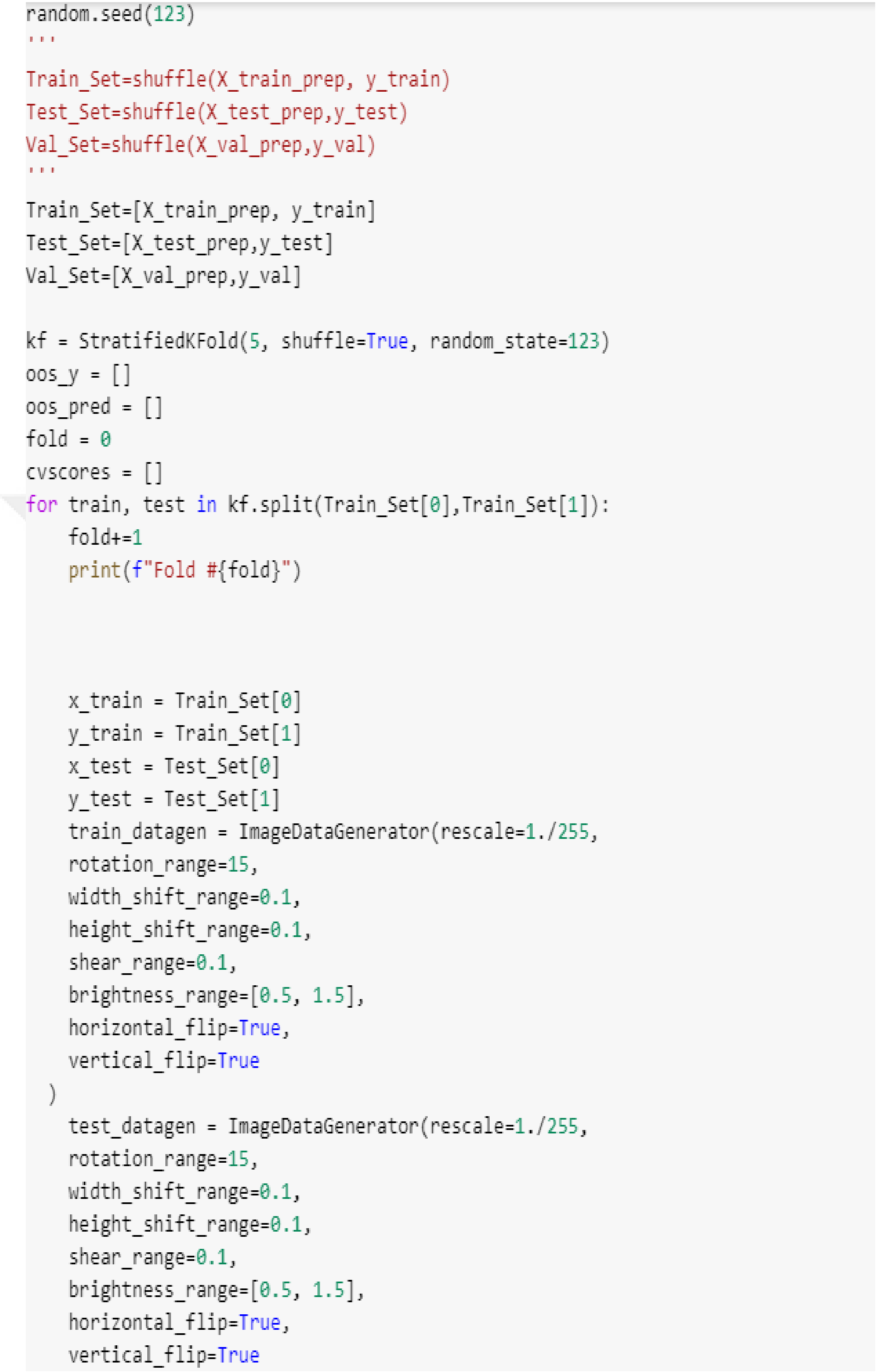
```python
def load_data(dir_path, img_size=(428,428)):
    """

    Load resized images as np.arrays to workspace
    """

    X = []
    y = []
    i = 0
    labels = dict()
    for path in tqdm(sorted(os.listdir(dir_path))):
        if not path.startswith('.'):
            labels[i] = path
            for file in os.listdir(dir_path + path):
                if not file.startswith('.'):
                    img = cv2.imread(dir_path + path + '/' + file)
                    X.append(img)
                    y.append(i)
            i += 1
    X = np.array(X)
    y = np.array(y)
    print(f'{len(X)} images loaded from {dir_path} directory.')
    return X, y, labels




TRAIN_DIR = "/content/drive/My Drive/Final_cv/"
TEST_DIR = "/content/drive/My Drive/C2/"
VAL_DIR = "/content/drive/My Drive/C3/"
IMG_SIZE = (428,428)
```



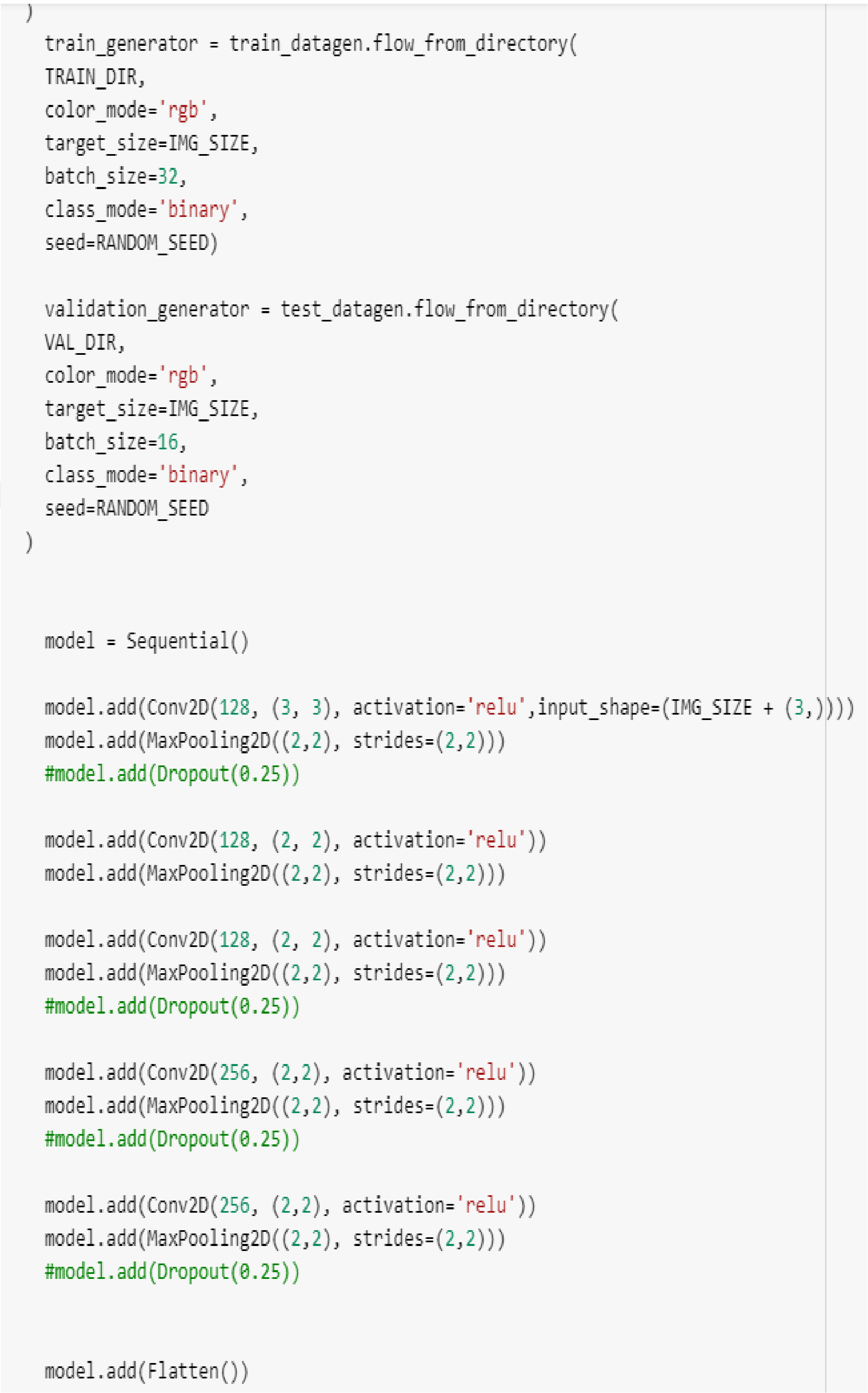
```python
# use predefined function to load the image data into workspace
X_train, y_train, labels = load_data(TRAIN_DIR, IMG_SIZE)
X_test, y_test, _ = load_data(TEST_DIR, IMG_SIZE)
X_val, y_val, _ = load_data(VAL_DIR, IMG_SIZE)
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    plt.figure(figsize = (6,6))
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=90)
    plt.yticks(tick_marks, classes)
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

    thresh = cm.max() / 2.
    cm = np.round(cm,2)
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.show()
```



```
def preprocess_imgs(set_name, img_size):
    set_new = []
    for img in set_name:
        img = cv2.resize(
            img,
            dsize=img_size,
        )
        set_new.append(img)
    return np.array(set_new)
X_train_prep = preprocess_imgs(set_name=X_train, img_size=IMG_SIZE)
X_test_prep = preprocess_imgs(set_name=X_test, img_size=IMG_SIZE)
X_val_prep = preprocess_imgs(set_name=X_val, img_size=IMG_SIZE)
#Shuffle Data
Train_Set=[X_train_prep, y_train]
Test_Set=[X_test_prep,y_test]
Val_Set=[X_val_prep,y_val]
random.seed(123)
'''
Train_Set=shuffle(X_train_prep, y_train)
Test_Set=shuffle(X_test_prep,y_test)
Val_Set=shuffle(X_val_prep,y_val)
'''
Train_Set=[X_train_prep, y_train]
Test_Set=[X_test_prep,y_test]
Val_Set=[X_val_prep,y_val]
```

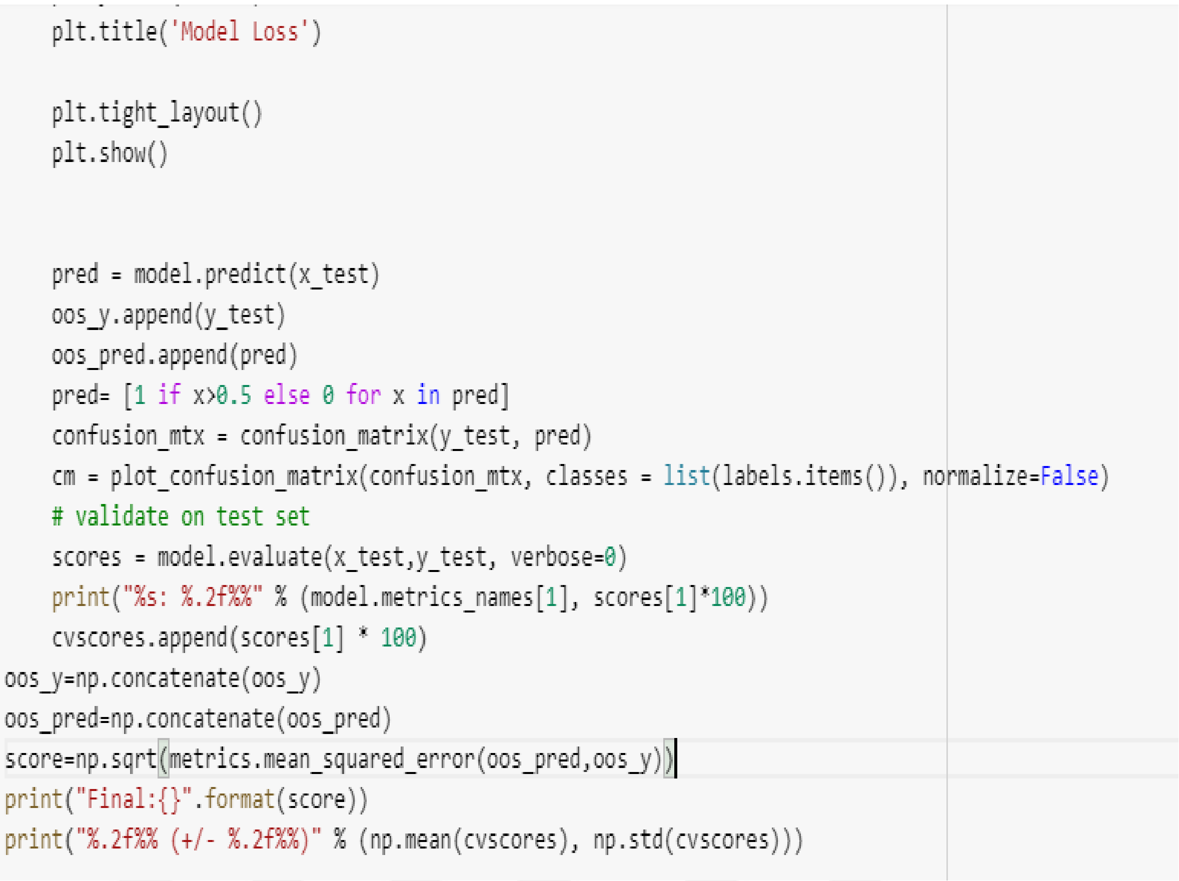


```
random.seed(123)
'''
Train_Set=shuffle(X_train_prep, y_train)
Test_Set=shuffle(X_test_prep,y_test)
Val_Set=shuffle(X_val_prep,y_val)
'''
Train_Set=[X_train_prep, y_train]
Test_Set=[X_test_prep,y_test]
Val_Set=[X_val_prep,y_val]

kf = StratifiedKFold(5, shuffle=True, random_state=123)
oos_y = []
oos_pred = []
fold = 0
cvscores = []
for train, test in kf.split(Train_Set[0],Train_Set[1]):
    fold+=1
    print(f"Fold #{fold}")



    x_train = Train_Set[0]
    y_train = Train_Set[1]
    x_test = Test_Set[0]
    y_test = Test_Set[1]
    train_datagen = ImageDataGenerator(rescale=1./255,
    rotation_range=15,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    brightness_range=[0.5, 1.5],
    horizontal_flip=True,
    vertical_flip=True
  )
    test_datagen = ImageDataGenerator(rescale=1./255,
    rotation_range=15,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    brightness_range=[0.5, 1.5],
    horizontal_flip=True,
    vertical_flip=True
```



```
)
  train_generator = train_datagen.flow_from_directory(
  TRAIN_DIR,
  color_mode='rgb',
  target_size=IMG_SIZE,
  batch_size=32,
  class_mode='binary',
  seed=RANDOM_SEED)

  validation_generator = test_datagen.flow_from_directory(
  VAL_DIR,
  color_mode='rgb',
  target_size=IMG_SIZE,
  batch_size=16,
  class_mode='binary',
  seed=RANDOM_SEED
)



  model = Sequential()

  model.add(Conv2D(128, (3, 3), activation='relu',input_shape=(IMG_SIZE + (3,))))
  model.add(MaxPooling2D((2,2), strides=(2,2)))
  #model.add(Dropout(0.25))

  model.add(Conv2D(128, (2, 2), activation='relu'))
  model.add(MaxPooling2D((2,2), strides=(2,2)))

  model.add(Conv2D(128, (2, 2), activation='relu'))
  model.add(MaxPooling2D((2,2), strides=(2,2)))
  #model.add(Dropout(0.25))

  model.add(Conv2D(256, (2,2), activation='relu'))
  model.add(MaxPooling2D((2,2), strides=(2,2)))
  #model.add(Dropout(0.25))

  model.add(Conv2D(256, (2,2), activation='relu'))
  model.add(MaxPooling2D((2,2), strides=(2,2)))
  #model.add(Dropout(0.25))



  model.add(Flatten())
```

```
model.add(Dense(200, activation='relu',activity_regularizer=regularizers.l1(1e-4)))
model.add(Dropout(0.5))

model.add(Dense(1, activation='sigmoid'))

model.compile(
loss='binary_crossentropy',
optimizer=RMSprop(lr=1e-4),
metrics=['accuracy']


#model.fit(Train_Set[0],Train_Set[1],batch_size=10,epochs=20,validation_split=0.20)
history=model.fit_generator(
train_generator,
steps_per_epoch=5,
epochs=30,
validation_data=validation_generator,
validation_steps=3,

# plot model performance
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs_range = range(1, len(history.epoch) + 1)

plt.figure(figsize=(15,5))

plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Train Set')
plt.plot(epochs_range, val_acc, label='Val Set')
plt.legend(loc="best")
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('Model Val Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Train Set')
plt.plot(epochs_range, val_loss, label='Val Set')
plt.legend(loc="best")
plt.xlabel('Epochs')
```



```
    plt.title('Model Loss')

    plt.tight_layout()
    plt.show()



    pred = model.predict(x_test)
    oos_y.append(y_test)
    oos_pred.append(pred)
    pred= [1 if x>0.5 else 0 for x in pred]
    confusion_mtx = confusion_matrix(y_test, pred)
    cm = plot_confusion_matrix(confusion_mtx, classes = list(labels.items()), normalize=False)
    # validate on test set
    scores = model.evaluate(x_test,y_test, verbose=0)
    print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
    cvscores.append(scores[1] * 100)
oos_y=np.concatenate(oos_y)
oos_pred=np.concatenate(oos_pred)
score=np.sqrt(metrics.mean_squared_error(oos_pred,oos_y))
print("Final:{}".format(score))
print("%.2f%% (+/- %.2f%%)" % (np.mean(cvscores), np.std(cvscores)))
```