REPUBLIC OF TURKEY

ALTINBAS UNIVERSITY

Institute of Graduate Studies

Information Technologies

AN INTELLIGENT SYSTEM FOR LUNG CANCER DIAGNOSIS USING A MACHINE LEARNING ALGORITHM BASED FEATURE SELECTION METHOD

Ruaa Azzah SUHAIL

Master's Thesis

Supervisor

Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2022

AN INTELLIGENT SYSTEM FOR LUNG CANCER DIAGNOSIS USING A MACHINE LEARNING ALGORITHM BASED FEATURE SELECTION METHOD

Ruaa Azzah SUHAIL

Information Technologies

Master's Thesis

ALTINBAŞ UNIVERSITY

2022

The thesis titled AN INTELLIGENT SYSTEM FOR LUNG CANCER DIAGNOSIS USINGA MACHINE LEARNING ALGORITHM BASED FEATURE SELECTION METHOD prepared by RUAA AZZAH SUHAIL and submitted on 17/08/2022 has been **accepted unanimously** for the degree of Master of Science in Information Technologies.

Asst. Prof. Dr. Sefer KURNAZ
supervisor

Thesis Defense Committee Members:

Asst. Prof. Dr. Sefer KURNAZ	Faculty Of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Oguz KARAN	Faculty Of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Serdar KARGIN	Faculty Of Biomedical Engineering, Beykent University	_____

I hereby declare that this thesis meets all format and submission requirements of a master's thesis.

Submission date of the thesis to Institute of Graduate Studies: ___/___/___

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Ruaa Azzah SUHAIL

Signature

DEDICATION

I devote and pledge this research work to my supervisor who is salient for guiding me through whole research work as well as my family for always assisting me in my hard time

PREFACE

First and foremost, I would like to thank my supervisor Asst. Prof. Dr. Sefer Kurnez for guiding and helping me along the way in writing this dissertation. Thanks to my parents, whose support and encouragement are always there for me, no matter what path I choose to follow.

Finally, I want to thank my supporting and loving husband and our three amazing daughters Rawan, Rola, and Sarah, who never cease to inspire me. They are the most important people in my life.

ABSTRACT

AN INTELLIGENT SYSTEM FOR LUNG CANCER DIAGNOSIS USING A MACHINE LEARNING ALGORITHM BASED FEATURE SELECTION METHOD

SUHAIL, Ruaa

M.Sc., Information Technology, Altınbaş University,

Supervisor: Asst. Prof. Dr. Sefer Kurnez

Date: 08/2022

Pages:102

Lung cancer is a lethal illness that affects both men and women equally, making early detection challenging. To reduce the death rate and raise the survival rate, it is vital to identify and classify lung cancer tumors at an early stage. Proper prognosis and treatment will save the lives of a greater number of patients. On the other hand, by the beginning of 2020, the world would have been infected with the coronavirus, a deadly disease (COVID-19). Among the several COVID-19 screening techniques available, CT imaging is an excellent way to identify sickness. Whereas other disease detection techniques are time consuming. Machine learning opens the door to predict, detect, and classify these diseases, but deep learning under machine learning provides a wide range of options for

model categorize COVID and NOCOVID photos, as well as lung cancer images (Benign, Malignant, Normal), and a test dataset is used to evaluate their performance, which produces expected accuracy for all models. The model's performance is evaluated using metrics such as

(accuracy, recall, and the F1-score). With a score of (100 %) for the lung cancer dataset, (CNN and VGG-16) earned the best accuracy of any model.

Keywords: Lung cancer, COVID-19, CT, SVM, LSVM, CNN, DNN, VGG.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

LDCT : low-dose computed tomography

CT : computed tomography imaging

ARDS : Acute respiratory distress syndrome

CAD : Computer-aided detection

ROI : Region of interest

MRI : Magnetic resonance imaging

ML : Machine learning

DL : Deep learning

SVM : Support vector machine

LSVM : Linear support vector machine

BPNN : propagation neural network

m-MKL : Multi-class learning

ANN : Artificial neural networks

RT-PCR : Real-time reverse transcription polymerase chain reaction

DNN : Deep Neural Networks

LSTM : long short-term memory

CNN : convolutional Neural Network

DNN : Deep neural networks

VGG : Visual Geometry Group

EDA : Exploratory data analysis

SL : Supervised Learning

UL : Unsupervised Learning

1VR : One-versus-rest

1V1 : one-versus-one

PET : positron emission tomography

MRI : magnetic resonance imaging

NLP : Natural Language Processing

RELU : rectified linear units

SGDA : Stochastic Gradient Descent with Adamax

1. INTRODUCTION

Lung cancer is a common disease that is causing an increase in deaths. For both men and women, the main cause of mortality from cancer is lung cancer. Cancer in the lungs is the direct result of long-term use of tobacco products such as cigarettes and chewing tobacco. Excessive cell proliferation causes malignant tumors to develop in the lungs. Despite advances in detection, diagnosis, and treatment, lung cancer still claims many lives. There are two major categories of lung cancer based on cell size. The cells are either small or large in size. According to the size and location of the lymph nodes, cancer is divided into four stages [1].

Lung cancer is one of the worst forms of cancer, with a one-third post-diagnosis survival rate. Females are seventy-five percent more likely than men to get lung cancer as a result of smoking. People who have never smoked account for around 10-15% of all instances [2]. Disease survival is the second lowest in all malignancies, with just over 30% surviving one year after diagnosis and only 5% surviving ten years after diagnosis. Recent developments in computed tomography imaging (CT) have allowed for earlier detection of illness [3]. Both a chest X-ray and a low dose computed tomography (LDCT) scan are commonly used to screen for lung cancer in patients [4].

A lung disease is defined as any problem in the lungs that prevents the lungs from functioning normally.

Infections of the Air Sacs in the Lungs (Alveoli)

There is a network of air sacs called alveoli that are connected by a system of tubes called bronchioles. These air sacs make up the bulk of your lungs' structure. Among the lung diseases that affect your alveoli are:

Pneumonia. Coronavirus (COVID-19) infection of the alveoli caused by bacteria or viruses.

A. Tuberculosis Mycobacterium tuberculosis causes progressive pneumonia.
B. Emphysema. Toxins that weaken the fragile connections between alveoli may lead to this problem. In the majority of cases, smoking is to blame. Emphysema also reduces airflow, affecting your airways.

C. Pulmonary edema. Air sacs and surrounding tissues are flooded with fluid from small blood veins in your lung. There are two types of pulmonary hypertension: heart failure and back pressure in the pulmonary blood vessels. Fluid leaks may also be caused by lung damage.

D. Lung cancer. It may start in any section of your lungs and present itself in many ways. Near or in the air sacs, it is most common in the middle of your lung.

E. Acute respiratory distress syndrome (ARDS). This is a life-threatening lung damage induced by a major sickness that was completely unexpected. One such case is COVID-19. The majority of patients with ARDS need breathing help from a machine known as a ventilator until their lungs recover.

F. Pneumoconiosis. a collection of disorders that result from breathing something that damages the lungs.

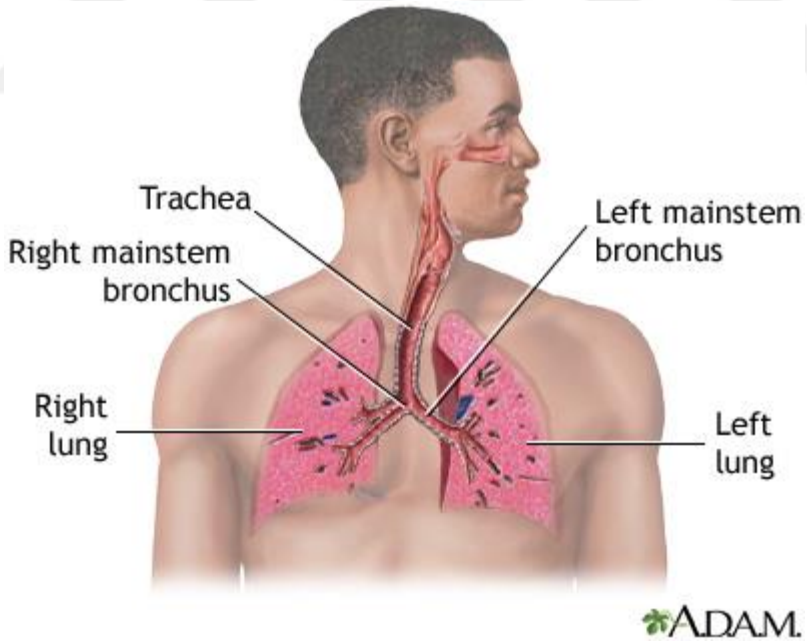


Figure 1.1: The most common lung diseases include: Asthma Chronic bronchitis COPD (chronic obstructive pulmonary disease) Emphysema Pulmonary fibrosis.

CAD systems increase radiologists' workflow, both of which reduce error rates Radiologists can use computer-aided detection (CAD) to locate suspicious areas (masses, nodules, and polyps) on a patient's body. Computer-aided design (CAD) has been utilized in CT, MRI,

and ultrasound imaging. CAD systems include three key steps for detecting and diagnosing cancer Segmentation based on ROI.

Feature extraction (the kind of feature), and classification [5]. To produce a large number of candidate locations that are very likely to contain the pulmonary nodules, a region-generating technique is applied [3]. In automated and semi-automated pulmonary lung nodule detection, methods and algorithms are used. The structure is made up of five parts: Image capture, lung segmentation, nodule identification and segmentation, and feature extraction [6]. ML and DL algorithms [7] may increase data interpretation's quality, consistency, and diagnostic power. ML uses mathematical and statistical tools to teach a computer a job by finding hidden patterns. patterns in inputs. Deep Learning, a subfield of ML, was born as a result of this (DL).

Many significant efforts have been made to automate the classification of lung cancer using ANN, naive Bayes, and other methods that used CT images and their model gave around 94 percent accuracy on average [8], used to train the support vector machine (SVM), back-propagation neural network (BPNN), and K-nearest neighbor classifiers (KNN) [9]. When our data has exactly two classes, we can use support vector machines (SVM) [1]. Multi-class learning classifiers are possible with many kernels (m-MKL) [10].

CNN may also be utilized for image classification, which can predict binary findings, such as 0 for benign nodules and 1 for malignant nodules, which are significant for this analysis [11]. Artificial neural networks (ANNs) are a mathematical model that tackles the problem of categorization and prediction. Most of the time, hidden layers transform the input into a format that the output layer can utilize. Neural networks have input and output layers, as well as hidden layers that transform the input into something that the output layer can utilize.

The novel coronavirus sickness (COVID-19) was caused by the SARS-CoV-2 infection that emerged in late 2019 [1]. COVID-19 transmission is reliant on close inter-individual contact and the transmission of breath droplets.

COVID-19 is spread via micron-sized droplets released from a COVID-19 infected person's mouth and/or nose while sneezing, coughing, or even speaking. The older population has been found to be substantially more susceptible to infection than younger age groups.

Because vaccines and therapeutic treatments aren't available, isolation and early patient detection are the most effective ways to keep the disease from spreading.

Real-time reverse transcription polymerase chain reaction (RT-PCR) was formerly the sole method for detecting COVID-19 in respiratory samples, but its time-consuming nature and erroneous findings have made it difficult to utilize in preventing the virus's spread in the population [2].

Furthermore, due to a shortage of RT-PCR test-kits, infected cases cannot be diagnosed in a timely manner, allowing the virus to propagate among the healthy population without being detected. Researchers are constantly exploring alternate testing methods, such as radiological images like x-rays and chest CT scans, to ease the inefficiency and scarcity of testing kits for COVID-19.

Medical Image Analysis has supported RT-PCR diagnosis by recognizing COVID-induced pneumonia in chest X-rays and CT images. These approaches can be used to accurately determine the radiological aspects of this disease. CT scan images are more effective at detecting soft tissues and producing findings [3]. In order to automate the whole process of evaluating CT scan pictures and detecting whether or not a person is COVID positive, researchers have developed DL approaches [4]. It is now possible to identify COVID-19 and differentiate it from other kinds of pneumonia as well as normal cases using an artificial intelligence system [5]. DNNs (Deep Neural Networks) have been suggested as an approximation approach. This approach [6] is a useful option for estimating the solution to a Partial Differential Equation.

DNN was used to identify COVID-19 in CT images and chest X-rays [7]. Deep learning and machine learning approaches can be used to provide alerts for maintaining social distance, diagnosing and treating COVID-19, tracking COVID-19 events, creating dashboards, forecasting, and another possible control mechanism [12].

A common deep learning algorithm, LSTM (long short-term memory) predicts COVID-19 data quickly [10]. LSTM uses specific features to capture the training set's sequence pattern information. CNN is a prominent deep learning approach for extracting meaningful

knowledge from noisy data for model predictions. Adding enormous layers to a CNN improves its prediction accuracy.

1.1 PROBLEM STATEMENT

Many issues have recently surfaced, and no effective solution for lung ailments like as cancer and pandemic Corona viruses has been discovered. The complexity of these diseases makes it difficult to obtain accurate results. An effective technique that will give an accurate diagnosis that's what doctors need to support their work and give more information. Previous research on this problem used a variety of strategies, but there were flaws in the efficiency of the algorithms and the time it took to develop them.

1.2 RESEARCH GOALS

The objective of this thesis is to resolve the problems described above by applying Machine learning algorithms can forecast lung cancer and coronavirus pandemics using off-the-shelf CNN, DNN, and VGG-16 algorithms.

1.3 THESIS ORGANIZATION

1. Thesis has 5 chapters. Chapter 1 presents an overview, problem statement, objective, thesis scope, research importance, and thesis organization.
2. Chapter 2 includes design and implementation algorithms for the proposed system.
3. According to the study methods outlined in Chapter 3, a full description of the tools and procedures utilized in classification and diagnosis is provided in the suggested system for diagnosing and detecting lung cancer and coronavirus.
4. Chapters 4 contain the modeling and simulations of the model, including the results derived from the implementation of the suggested system, their testing, and comparisons with earlier studies.

Finally, Chapter 5 presents the conclusion and future work.

2. LITERATURE REVIEW

2.1 INTRODUCTION

Lung cancer and COVID-19 detection and diagnosis are the subject of numerous studies and research projects. This chapter discusses the scientific history and classification algorithms and focuses on developing machine learning algorithms using data processing to detect lung cancer and COVID-19. It appears that lung cancer kills the most people in the world. Early detection of lung cancer can improve people's chances of survival; however, some of these lesions may go undetected because they are small could be camouflaged (hidden away). Data processing has evolved into an active field recent years' research topic [12]. Various preprocessing steps have been proposed. But high accuracy, sensitivity, and specificity are important. To achieve this, preprocessing of the acquired original data is needed.

As 2019 ended, COVID-19, also known as the Zika virus, began to spread throughout the world, causing an alarming situation. Its infections are frequently accompanied by respiratory symptoms. Even competent doctors have difficulty diagnosing COVID-19 [13,14].

The following algorithms are used (SVM ,LSVM) to check text data (csv), in another hand we have convolutional Neural Network (CNN), (DNN), Visual Geometry Group (VGG) to check pictures data because of each classifier's performance in terms of accuracy, sensitivity, and specificity evaluated Lung disease classification comprises data acquisition, data preprocessing, exploratory data analysis (EDA), and finally classification, these features used to train the classifiers of our interest of comparison (SVM, LSVM) for text data, (CNN, DNN and VGG) for CT scan medical images.

2.2 PRE-PROCESSING:

Data pre-processing can be considered as the main stage conducted on raw data to prepare them for the analytic tasks. If data have impurities such as duplicate or missing values, then analytic tools might be misled and provide a wrong result. As a result, preprocessing data prior to data analysis is a requirement. In this step, the missing data issue, which is produced during data transformation or collection, is resolved by removing the noise records with incorrect readings. Also, the missing data are filled with a global constant [13].

2.3 DATA CLEANING

Real-world data is notorious for being insufficient, noisy, and inconsistent. The methods of data cleaning have the task of "cleaning" data by removing or detecting outliers, for the purpose of filling missing values, resolving inconsistencies, and smoothing noisy data. Users are less likely to believe in the data mining results if they suspect the data is tainted with some sort of error. The mining process can also be thrown off by dirty data, resulting in an unreliable result [14].

2.3.1 Missing Values

It is possible to fill in the blanks with many different approaches, such as using a global constant, computing the measure of central tendency on the feature, or utilizing all instances of the feature in a similar class as a feature median or mean [14].

2.3.2 Data Transformation

Data transformation entails converting data to a specific range, such as [0.0, 1.0], in order to generalize its values. The goal of normalizing data is to assign equal weight to all features. For algorithms based on distance, such as nearest-neighbor classification and clustering, normalization is particularly beneficial. Normalization will aid in the acceleration of the learning process [15].

A. Min-Max Normalization

This method denotes the application of linear transformations to primary data. Assume that maxA and minA represent the feature's maximum and lowest values, as well as the value allocated to it (A) within the range [new minA, new maxA] according to equation (2.1) [15].

$$v_i^{`} = \frac{v_i - min_A}{max_A - min_A}\left(new_{max_A} - new_{min_A}\right) + new\_min_A \qquad (2.1)$$

Where $v_i^{`}$ represent features that normalized.

B. Score Normalization

In this approach, the attributes' means and standard deviations are computed, and the data is normalized using equations. (2.2) [15].

$$v^{`} = \frac{v - \mu}{\sigma} \qquad (2.2)$$

An experiment's value is v, whereas V' is its normalized value. The study's mean is, and the standard deviation is.

2.4 EXPLORATORY DATA ANALYSIS (EDA)

Analyzing huge datasets, exploratory data analysis (EDA) is used to identify the most essential properties of the data. For (EDA), in addition to the formal modeling or hypothesis testing process, the main objective is to examine what the data can tell us [16]. (EDA) is a quantitative data analysis tradition that may be traced back to John Tukey's pioneering work. EDA provides a framework for a wide range of data analytic activities, as well as resolving the many data and design difficulties that applied researchers encounter.

Graphics and data visualization that is interactive (Revelation), a focus on model construction, diagnosis, and assessment (Residuals), resolving the basic measuring difficulties that come with different distributions (Re-expression), Development of procedures that are resistant to incorrect or misleading results due to real-world data is one of the main terms and computational tools in EDA (Resistance) [16].

2.5 CLASSIFICATION

Classification is a supervised learning approach for predicting the target class of each sample in data using predefined classes. Classification algorithms requires classes that defined and being based on data attribute values [17]. Using the model, we may predict the class label of the presented data during the classification phase, which is separated into two stages: training and classification. The training phase involves building a classifier that describes a previous defined set of data classes, where the classification algorithm forms the classifier by learning from training data and related class labels, this phase can be described as a learning phase where the function y= f(x) can predict the label of the related class for the given data.

Where the accuracy of the prediction is first computed. If used the training data for the calculation of accuracy, it will be optimistic (classifier leads to over fitting the data), so in this state, in this case, a separate training set is employed for the test set.

Using the testing set, the classifier may be used in the future to classify data that has no known class label. The majority of the work in supervised classification is completed before the actual categorization. Although supervised classification is more accurate than unsupervised

classification, it is very dependent on the training, and misclassifications will be discovered. The growth of training data must be closely monitored in supervised categorization. If the training data is weak or non-representative, the classification results will be poor.

As a result, as compared to unsupervised classification, supervised classification needs more money and effort [18].

2.6 MACHINE LEARNING (ML)

Machine learning (ML) is a method of computer training to learn from experience." Machine learning algorithms are primarily concerned with discovering and extracting natural patterns in datasets, as well as analyzing the feature extraction process [19]. Machine learning makes use of two primary strategies. These are the "Supervised Learning (SL)" and "Unsupervised Learning (UL)" strategies [20].

The main distinction between the various types of machinery is that the supervised learning (SL) algorithm learns about a function by using both input and output. Unsupervised learning (UL), on the other hand, learns about data relationships solely from the input vector [20]. There is no definitive answer to the question of which learning algorithm to use, because each technique has advantages and disadvantages, and none can provide the best recognition performance in all situations [19]. All of the machine learning techniques are explained in Figure (2.1).

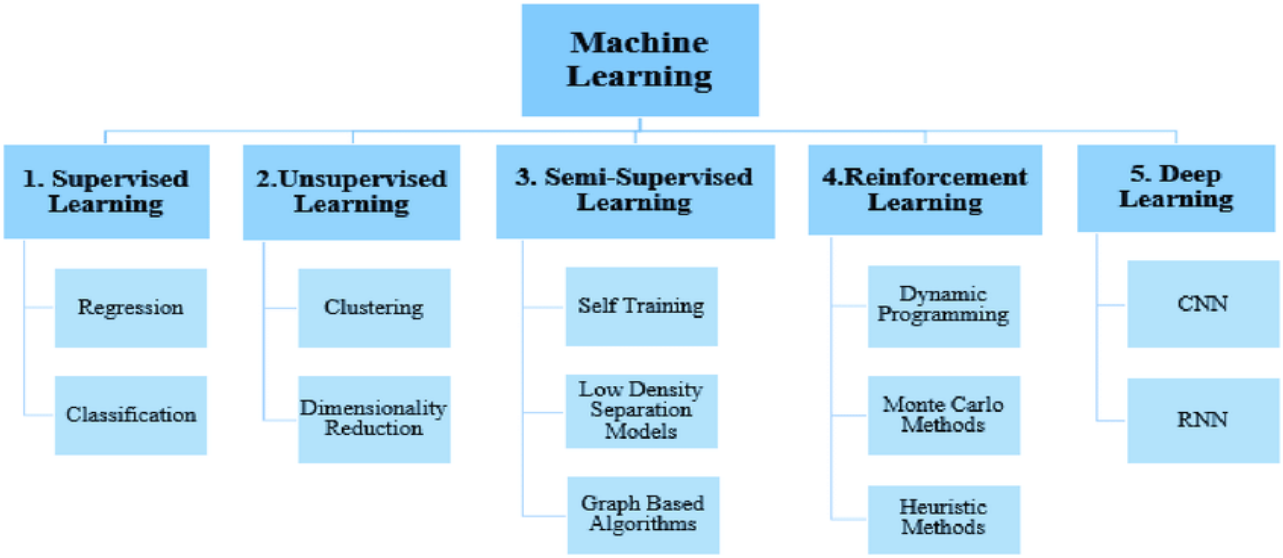


Figure 2.1: Machine Learning types [20].

A. Supervised Learning:
   Machine learning algorithms that need outside help are supervised. Input dataset is used to build train and test datasets. In the training dataset, you must predict or group an output variable. [21] Every algorithm learns patterns from the training dataset and applies them to the test dataset to create a prediction or classification. In this type, the algorithm makes a function that translates inputs to desired outputs. Unsupervised learning teaches a learner a function to modify a vector map's type by looking at many input/output samples. [22].
B. Unsupervised Learning
   It is used to conclude datasets comprising of input information without labeled reactions. An unsupervised learning approach known as cluster analysis is the most frequent method for discovering patterns or groups in data [22][23].

2.6.1 Support Vector Machine (SVM)

The algorithm was invented in (1995), introduced this algorithm to resolve the kind of specific regression and classification problems [24]. This method relies on the statistical learning method , it determines the ideal hyperplane that maximizes the class's boundary. SVM is a classification approach that identifies a flat classification boundary between two data classes [25] [26]. This algorithm is a supervised classification method in which a set of inputs is given along with their labels. The attributes vector represents the types

of these inputs. As previously stated, this algorithm works by creating a hyperplane to distinguish the two forms in order to achieve the greatest possible separation between them. Bounding planes are two planes parallel to the classifier that move through multiple points (see Figure (2.2)). Distance between these two surfaces is known as the "margin." In the bounding planes, the support vectors are just the points [28].

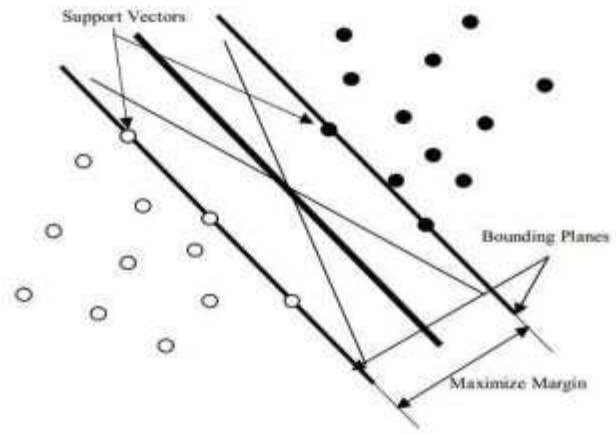


Figure 2.2: SVM Hyperplanes between two classes **[29].**

Because of the generalization principle based on the theory of structural risk reduction (SRM), the method is based on statistical learning theory's assured risk limitations, Support Vector Machines have the prospect of capturing extremely large feature areas [26]. At MLP classifiers, during the training process, weights are changed to reduce total errors between the network outputs and the desired performance. Because the decision boundaries between classes gained by training are difficult to resolve and the generalization ability is dependent on the training technique, the network's performance suffers significantly when data amounts are minimal [31] [32]. As indicated in the next section, the SVM algorithm may be divided into two types: linear and non-linear.

2.6.1.1 Linear SVM

Linear SVM is a fast machine learning (data mining) technique for classifying huge data sets. It creates a linear SVM by cutting planes. If the linear SVM hyper-plane is provided, the super vector machine is linear, When i=1,2...z, and z represents paired training (xi, yi), the direct equation defining the hyper-plane is y (1, -1) [33].

$$W.\ x+b=0 \qquad (2.3)$$

W is a weight vector, W = w1, w2, wn, b is a bias, and x are features of the weight vector. An equation is used to classify data [33].

$$(w.x.\ y)=sgn(w.x.b) \qquad (2.4)$$

A linear classifier using the hyperplane (f(x)) to predict the class y for each point (x) is called a "hyperplane" (f(x)). This hyperplane (f(x)) is produced as a sequence of points (x Rm) fulfilling the equation (f(x)=0) [32].

$$W^T.x + b \geq 1 = +1$$
$$W^T.x + b < 0 \text{ for } y = -1$$

The Lagrange approach can handle the problem of maximizing the margin. Every training point xi is explained by using a Lagrange multiplier (αi – i).

$\boldsymbol{\alpha_i = 0} \longrightarrow \boldsymbol{x_i}$ Has no influence on the hyper-plane

$\boldsymbol{\alpha_i > 0} \longrightarrow \boldsymbol{x_i}$ These points provide as support for vectors that are closest to the hyperplane.

When determining the αi value, you may also compute weight and bias using the formula below [34].

$$W = \Sigma \alpha i x i \quad (2.7)$$

The SVM average that is not zero (a_i) must be taken into consideration for points with (a_i = 0). Decisions are not affected when the support vector has (a_i = 0) [26].

A. Hard-Margin SVM

A hyperplane or function may be identified using the SVM method.

$$g(x) = w^T x + b \quad (2.8)$$

With a maximum margin, this successfully divides two groups. Figure (2.6) shows a separating hyper-plane corresponding to a hard-margin SVM (also called a linear SVM [26]. The training data must be entirely segregated by at least one hyperplane in the hard margin SVM. This is usually accomplished by mapping the characteristics into a high-dimensional space where the two classes are separable using a Kernel [35].

.

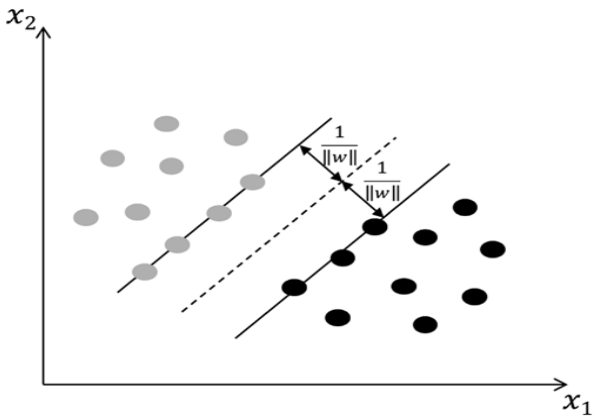


Figure 2.3: Hard-maximum-margin separating hyperplane [35].

As you may expect, there exist an endless number of separating hyper planes for two separable classes. This is seen in Figure (2.3) on the right side. Where the hyperplane results from a basic logistic regression model (GLM), an LDA (another common classification tool), and an HMC, as shown in Figure (2.4), if you want to design a decision boundary with good generalization performance, draw a border that gives the maximum separation between the two classes. That is exactly what the HMC is doing [36].

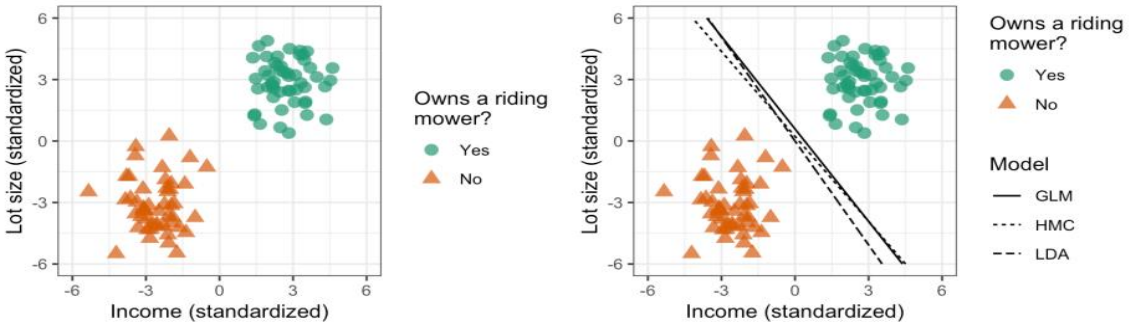


Figure 2.4: Simulated binary classification data with two separable classes.

The dataset is on the left. Raw data containing examples of decision boundaries (in this case, dividing hyperplanes) from several machine learning algorithms. [37].

B. Soft-Margin SVM

Nonlinear data mapping in the following part will make it more likely that the data will not be fully separated by a straight line, even if this is possible in a real-world situation. Noise and outliers in the data make a smooth decision border preferable than one that loops around them [38]. Different SVM variants approach this problem in various ways. It's preferable to use the easiest technique rather than necessitating it.

$$y_k(w'x + b) \geq 1 \qquad (2.9)$$

introduce "slack variables" sk ≥ 0 and allow

$$y_k(w'x + b) \geq 1 - sk \qquad (2.10)$$

This permits the (a) point to be a minor distance (sk) on the incorrect side of the hyperplane without breaking the specified limitation. Add a new component to the Lagrangian penalizing excessive slacks to prevent a simplistic solution when any line may "split" the data [39].

$$min_{w,b}L = \frac{1}{2}w'w - \sum \lambda_k(y_k(w'x_k + b) + s_k - 1) + \alpha \sum s_k \qquad (2.11)$$

When is decreased, more data are permitted to lie on the incorrect side of the hyperplane These data would then be categorized as outliers, resulting in a smoother decision boundary. When the data are not completely separable, slack variables xi are added into the

SVM objective function. An example of this would be the locations indicated by an X in Figure (2.5), to account for error in the misclassification. In this scenario, SVM is not looking for a hard margin that will perfectly categorize all data. Instead, SVM has evolved into a soft-margin classifier, which means it properly classifies the majority of the data while permitting a few points near the dividing border to be misclassified [40][41].

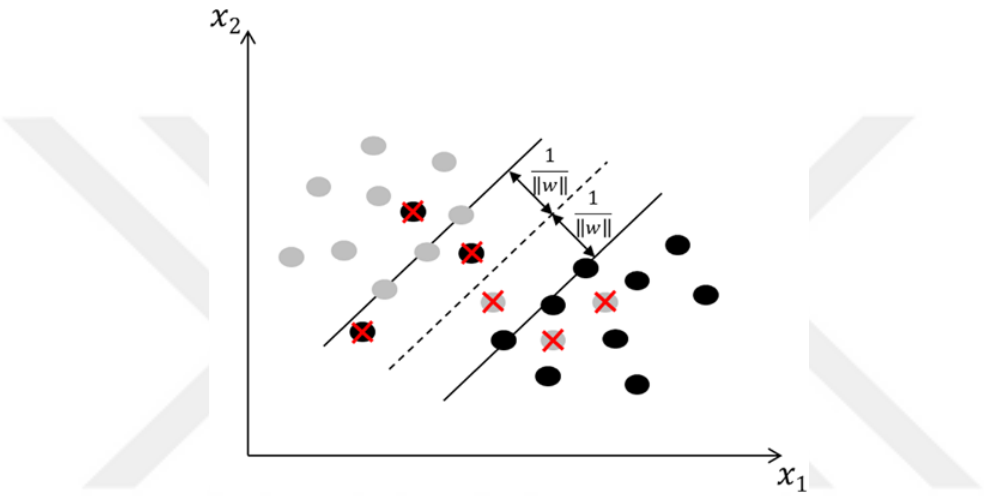


Figure 2.5: A few misclassifications, as part of soft-margin SVM [1].

2.6.1.2 Non-linear svm

In the case of non-linear classification, the use of non-linear kernel functions does not consider the suitable classification approach for this sort of categorization [42]. It is easy to train and deploy linear support vector machines (SVMs), however when given a high number of training cases and few attributes, these models seem to have poor performance on difficult datasets.

Non-linear SVM, on the other hand, may provide better quality across a wider variety of challenges and is the favored solution in many cases despite its lack of critical power [43].

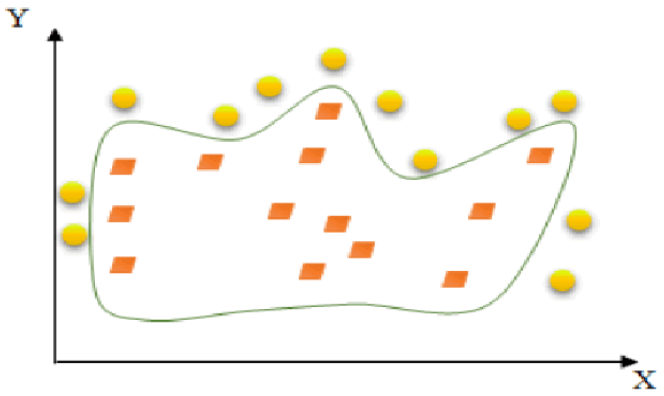


Figure 2.6: SVM concept for non-linear case[2].

A- Kernel svm

Kernel**:** A linear hyperplane can divide linear data. Non-linear data is widespread in datasets. Kernels convert low-dimensional to high-dimensional data. Later, new mapping can be linearly separated. [44].

The testing and training data are moved to a high-dimensional feature space using the functions kernel. Using kernels instead of mapping functions may save a lot of time in computation since kernels are more efficient [45]. SVM transforms raw data into a higher-dimensional space using the kernel function K (x, x'). (x) Equation's dot product transformation function (2.12). The goal is to convert information to a higher dimension. Calculating hyperplanes (2.13) [46].

$$K(x, x') = \{\Phi(x), \Phi(x')\} \quad (2.12)$$

$$f(x') = \sum_{n=1}^{N} \alpha_n y_n K(x, x') + b \quad (2.13)$$

Where α_n is a Lagrange multiplier and y_n is a membership class label (+1, 1) with n = 1, 2, 3, ..., N [46].

Here is a list of the different kernel functions: In the book, you can read more about how kernel functions work. The ones below are taken from that list and are listed here just to make note of them [45][47].

A. Polynomial: For non-linear modeling, a polynomial mapping is a popular approach. The second kernel is frequently preferred since it avoids the hessian becoming Zero.

$$K(x, x') = \langle x, x \rangle^d \quad (2.14)$$

B. Radial Basis Function: Radial basis functions are often Gaussian.

$$K(x, x') = exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) \quad (2.15)$$

C. Exponential Radial Basis Function: When discontinuities are tolerable, a radial basis function yields a piecewise linear solution.

$$k(x, x') = exp\left(-\frac{\|x-x'\|}{2\sigma^2}\right) \quad (2.16)$$

D. Linear kernel: If the data is linear, you can split it with a separating hyperplane.

$$k(x, x') = (x, x') \quad (2.17)$$

Many others exist, such as Fourier, splines, B-splines, additive kernels, and tensor products. You might read the book if you want to learn more about kernel functions [48].

B - Multiclass svm

Weston and Watkins (1999) and Platt (1999) were the first to extend the SVM binary classification to the multiclass scenario (2000). Researchers proposed a number of ways to deal with the multi classification problem, including one-versus-the-rest, pair-wise classification, and the multi classification formulation, which are all described below in turn [49]. There are some circumstances in life that require classification into more than two categories. Multiclass classification issues (k > 2) are often split down into binary problems that a generic SVM can solve. One-versus-rest (1VR) and one-versus-one (1V1) [50].

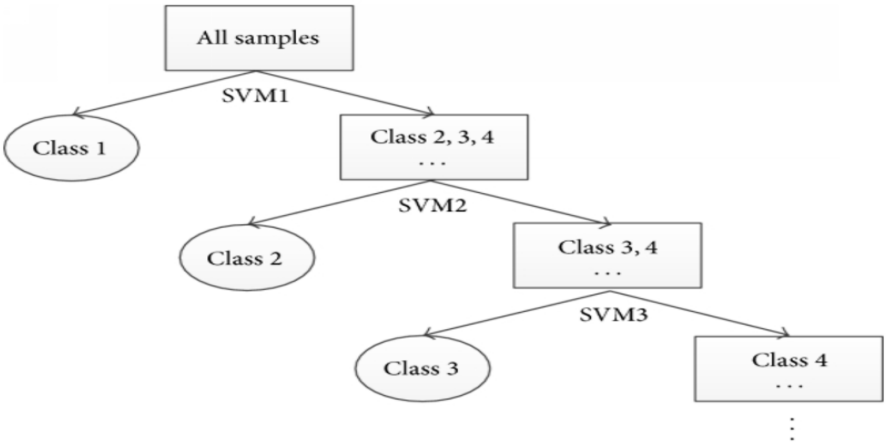


Figure 2.7: The multiclass SVM classifier [3]

1-One-Vs-Rest for Multi-Class Classification

This method includes M binary classes, one of which is trained with all data and is positive, while the other classes are negative. To develop this binary classifier model, the two classes are separated into two groups: one class and the rest. The classifier has been trained to determine whether a class belongs to the first or second group [51].

2. One-Vs-One for Multi-Class Classification

This approach creates M(M-1)/2 binary classifiers, each trained on M classes of data. [52].

2.6.1.3 C (penalty) and gamma

Nonlinear SVMs with a Gaussian RBF kernel have C and Gamma parameters. Conventional SVMs highlight positive and negative examples. Mislabeled or rare cases may lead to faulty models. Cortes and Vapnik created "soft margin" SVM in 1995 to solve this problem. Since certain samples can be "ignored" or moved, the overall fit is better. The C parameter in the soft margin cost function shows little bias and much variation. High error risk reduces prejudice.

A small C gives you higher bias and lower variance. Which controls the influence of each individual support vector; this process involves trading error penalty for stability.

SVMs use dot product classification.

A. However, in 1992, Boser, Guyan and Vapnik advocated replacing each dot product with a nonlinear kernel function to describe more intricate interactions (such as a Gaussian radial basis function or Polynomial kernel). The Gaussian radial basis function's free parameter is called gamma.

$$K(x_{i,}x_{j)} = \exp\left(-y\left|\left|x_i - x_j\right|\right|^2\right), y > 0 \qquad (2.18)$$

B. A Gaussian with a large variance has a small gamma, hence (x j) has more impact. If (x j) is a support vector, a small gamma suggests its class will affect (x I even if their distance is large. High gamma means minimal variance; hence the support vector is ineffective. Large gamma models are biased and have limited variance.

2.7 MEDICAL IMAGE ANALYSIS

Medical imaging refers to the mechanisms that offer visual information for the human body. Medical imaging helps to assist radiologists and physicians in improving the diagnosis and treatment procedure quality. Medical imaging encompasses a variety of imaging modalities and is an important part of disease diagnosis and care. To name a few, ultrasound, positron emission tomography (PET), magnetic resonance imaging (MRI), X-ray and computed tomography (CT) as well as hybrid modalities, are among them [53].

These techniques are important for detecting anatomical and functional details regarding various body organs for diagnosis and research [54]. Medical imaging is essential in modern healthcare. With applications in lung cancer detection, medical image annotation, image-guided treatment, recognition, classification, and segmentation, machine learning plays a critical role [55].

2.8 DEEP LEARNING

Artificial Neural Networks (ANNs) with several hidden layers are known as Deep Learning. It has been recognized as one of the most effective tools in recent decades, and its capacity to handle massive volumes of data has made it quite prominent in the literature [56]. In certain applications, like as pattern and image recognition, deep hidden layers have begun to outperform standard techniques [57].

DNN and CNN are the most popular types of deep neural networks. The CNN inherits its name from the use of convolutional filters that collect multi-dimensional inputs to search for local

features. A CNN can have multiple types of layers, including fully connected layers, pooling layer, non- linear layer, and convolutional layers. Parameters are available for the layers of convolutional and fully connected, but not for the non-linear and pooling layers [58]. In machine learning problems, CNN performs exceptionally well, especially in environments that interact with image data. The findings achieved by CNN are very important in areas such as the largest image classification dataset (ImageNet), Natural Language Processing (NLP), and computer vision [59].

2.9 DATASET AUGMENTATION

Combining color modification and affine image transformations is the most common modern technique for data augmentation. Shearing, scaling (zoom in/out), rotation, and reflection can all be described as affine transformations. After various types of affine transformations, the same picture Geometric distortions or deformations are typically used to maximize the number of samples for deep neural model training, to balance the scale of datasets, and to boost their reliability [60]. Blurring, sharpening, white balancing, enhancing brightness or contrast, and histogram equalization are the most popular methods.

2.10 CONVOLUTIONAL NEURAL NETWORK

Over the last decade, CNN has achieved ground-breaking results in a range of pattern recognition areas, image processing to speech detection are only a few examples. The greatest advantageous feature of Convolution Neural Networks is that they reduce the total of parameters available in Artificial Neural Networks [61].

This accomplishment has inspired developers and researchers to consider bigger models to resolve complicated jobs that were previously impossible to solve using traditional ANNs. The main principle of CNN solving problems should not be extracted by featured algorithms like others, instead, it used all pixels of input images, that are spatially dependent [62].

In other means, for instance, in the case of a face recognition app, attention should not be paid to the position of the face in the images. It is only essential, regardless of their location in the given images, to detect their relative features face to face. The abstract characteristics of input propagation to deeper layers would also be another significant aspect of CNN. eg, the edge can

be identified in initial layers and then in the second layer the simpler shapes, and higher-level characteristics like the faces in next layers [63].

The CNN layers can be described by two types of terminology, as shown in figure (2.8). The first terminology is called simple layer terminology (on the right-hand side of the figure), where the CNN is observed as a cascaded number of plain layers. Every layer after completing its data processing pushes its output results to the subsequent layer [64]. The second terminology is called complex layer terminology (on the left-hand side). In that way, a limited number of relatively complex layers are regarded as the convolution network, with each layer being several stages [65].

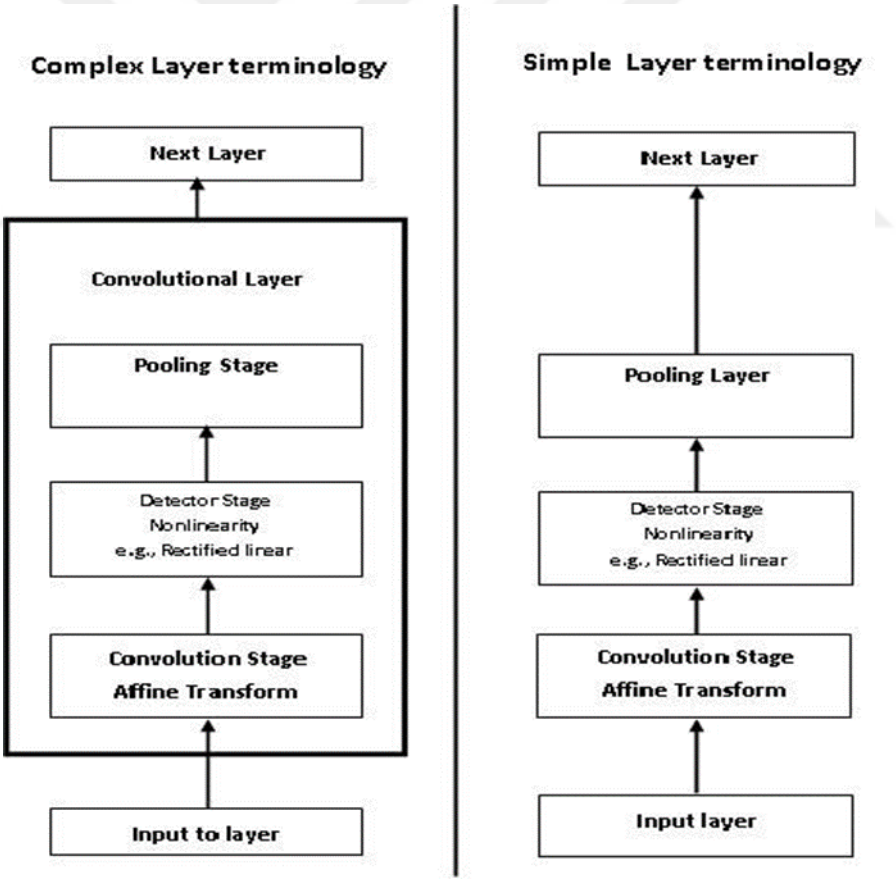


Figure 2.8: Components of a typical CNN Layers [65].

2.10.1 Convolution Operation

Convolution is per-pixel operations, the same mathematical operation applied to all pixels in the image or frame. Therefore, the neural network complexity and time required to perform convolution operations increase when the size of the image or frame increases. A filter (sometimes called the kernel) is a two-dimensional matrix of real numbers or a smaller-sized matrix in comparison with the input dimensions of the image or frame. The filter coefficients vary from one application to another [66]. Convolution is achieved by dragging a filter from the top left corner of the picture to the lower right corner of the image and then passing over each pixel there. The image region that is under the filter is called the receptive field; this has the same dimensions as the filter.

Convolution is performed by multiplication operation between each number of the kernel and the corresponding value of image or frame and sum the result of the multiplication. Then, the output is normalized by the summation of the filter kernel. The result value becomes the new intensity of the pixel under the center of the kernel as shown in figure (2.9) [67].

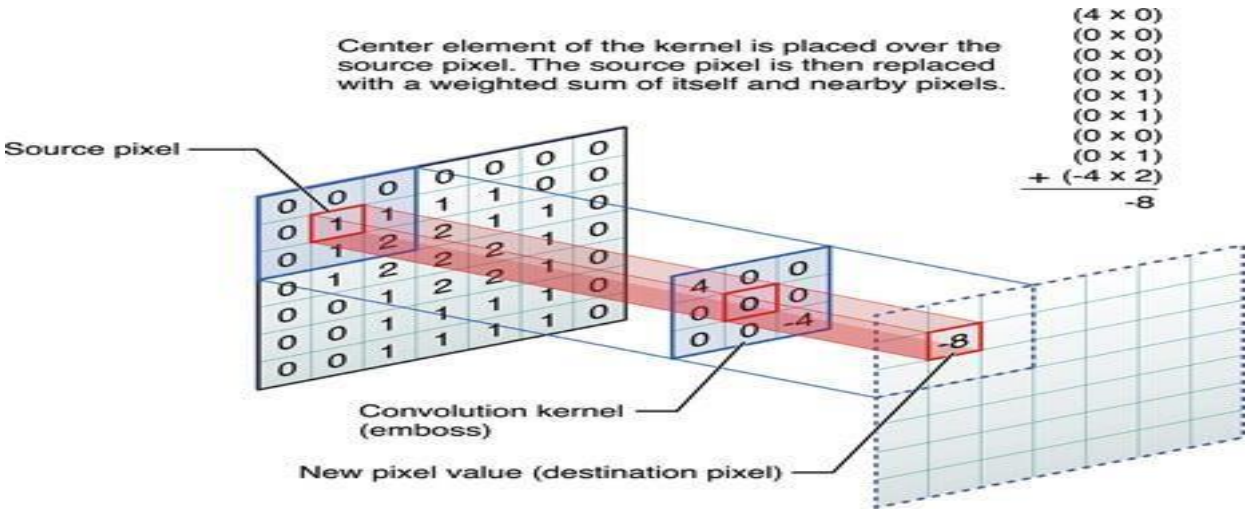


Figure 2.9: Convolution layer operation, sliding filter over the given image data of input[67].

RGB images (three channels) require three convolution filters. This 3*3*3 filter builds one feature map per output channel. [68] Figure 2.10 Size, number of filters, stride, and zero paddings effect convolution performance.

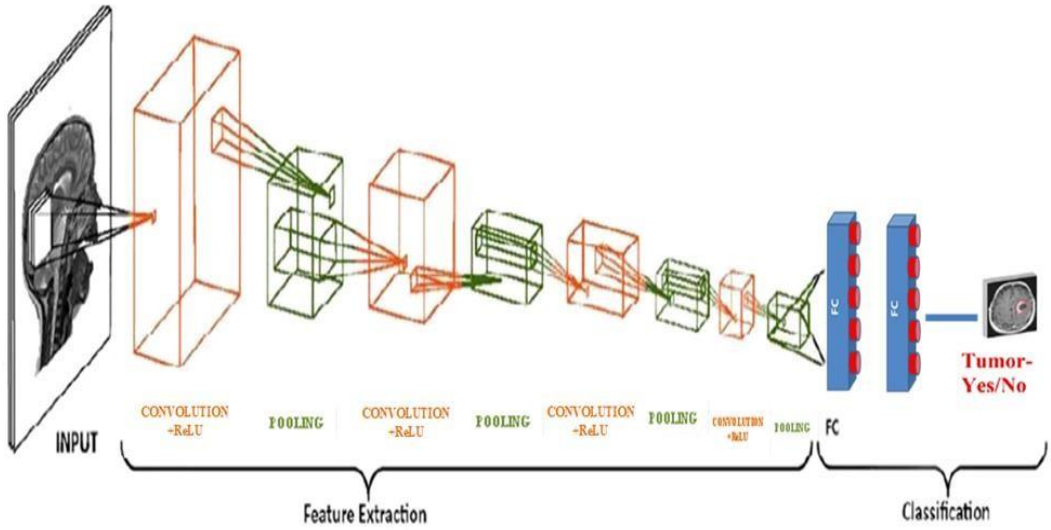


Figure 2.10: The depth corresponding to the number of filters used for the network convolution [53].

A. The Size and Number of Filters

The size and number of filters have a direct effect on the size and number of output feature maps. The feature map number used in a convolution process is often the same as the number of filters used. Additionally, the depth of filters used must equal the number of input channels**.** The size of filters used will be effective on the size of output features maps. In the case of convolution without empty space, the filter size increases, which reduces the size of the output feature map, in contrast, if the size of the filter decreases, the map features will increase, for example when the filter is the size of the image, the output will be only one pixel [69].

B. Stride

The CNN has many additional solutions that provide many possibilities to even reduce the parameters, Stride is one of these choices. The number of pixels the filter slides over the input image or frame matrix can be expressed as the stride. Therefore, when the stride is equal to 1, this means filter one pixel per step is moved over the input image, while when if the stride is equal to 2, this means the filter is jumped two pixels at each move. The overlap can be

manipulated by controlling the stride. Equation (2.24) formalizes the size of the output with a certain stride, N×N (image size), and F×F (filter size), as shown in figure (2.19) [70].

$$O = 1 + \frac{N - F}{S} \tag{2.19}$$

Where N is the input size, F is the filter size, and S is the stride size

C. Padding

One of the disadvantages of the convolution stage is the potential for information loss at the image's border. When the filter slides, they are never captured. The use of zero paddings is a very effective method to solve the problem. No paddings are added to the input matrix's borders so that convolution output can be controlled while still allowing the input picture or frame to be included in its output. In this case, slide the filter overall input matrix. Another advantage of zero paddings is that it allows you to control the generated size. The zero-padding formula has been updated as shown in Eq. (2.20) [71].

$$O = 1 + \frac{N + 2P - F}{S} \tag{2.20}$$

This padding approach prevents network output size from diminishing with depth (P=1 in figure (2.12)). [72].

D. The formula of the Convolutional
Eq. (2.21) determines convolutions for the following layer's pixel. [73].

$$net\,(\mathrm{i},\mathrm{j}) = (x * w)\,[\mathrm{i},\mathrm{j}] = \sum_{m}\sum_{n} x\,[m,n] \times w\,[\mathrm{i}-m,\mathrm{j}-n] \tag{2.21}$$

X is the input picture, W is the kernel or filter matrix, and * is the convolution operation.

2.10.2 Nonlinearity

To produce a non-linearity layer in a convolutional neural network, a convolutional layer generates a feature map, which is fed into an activation function.

The activation function performs an element-by-element action across the input volume, thus the dimensions of the input and output are equal [74]. A logistic (sigmoid) or hyperbolic-tangent activation function is the most common type of activation function used in neural networks. However, newer investigations indicate, in particular in CNN, that rectified linear units (ReLU), are beneficial over conventional activation functions.

In spite of the fact that the non-linearity layer and the convolutionary layer are supposed to be independent, it is fairly uncommon for the literature to combine the two. These kinds of works employ various notes to describe its layers and architecture. The common forms of nonlinear features as shown in figure (2.11). For the following purposes listed, Rectified Linear Unit (ReLU) was mostly used [75]:

As a function and gradient, the ReLU formulations are simpler, as shown in the two equations below (2.22) (2.23) [76]:

$$ReLU(x) = max(0, x) \tag{2.22}$$

$$\frac{d}{dx} ReLU(x) = \{1 \text{ if } x > 0;\ 0\ otherwise\} \tag{2.23}$$

Full sigmoid and tanh causes back-propagation problems. Gradient signal fades as neural network complexity increases. "Vanishing gradient" Except for the center, these functions have essentially little gradient. The steady input gradient is constant. The function isn't unique, but its execution may be. [77]. Since the zero in the gradient leads to achieving a complete zero, the ReLU produces a sparser representation.

Sigmoid and tanh, on the other hand, yield non-zero gradient effects, which may be counterproductive to training. The x axis represents the function's input, while the y axis represents the function's output.

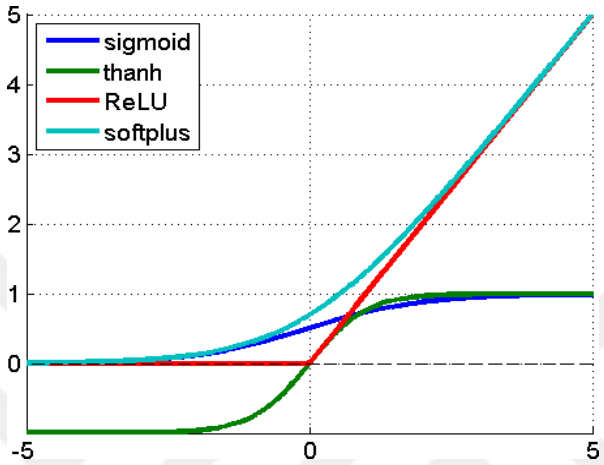


Figure 2.11: Common types of nonlinearity functions [62].

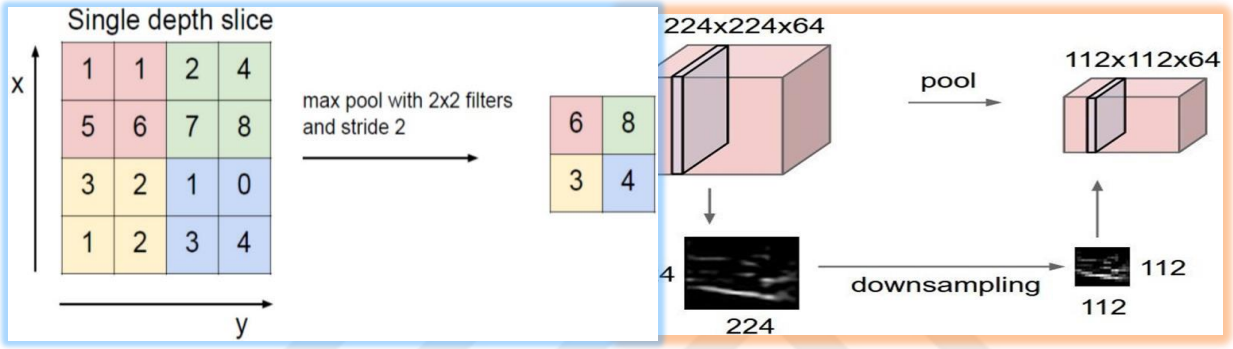
2.10.3 Pooling Layer

CNN pools for two reasons. Pooling creates a predetermined-size feature map for categorization. Applying maximum pooling to all 256 filters will yield a 256-dimensional output. Second, down-sampling lessens the complexity of following layers. Image processing applications are similar. Pooling doesn't influence filter count. Max-pooling uses pooling strategies. The maximum value of each sub-region is returned. Max-pooling size 2*2 is still most popular.

As shown in figure (2.12) (a and b). Stride 2 can be used to prevent down sampling, although it is not widely used. It's important to note that down sampling doesn't maintain the information's position. The only time this technique is appropriate is when the existence of data is more important than the location of that data [78].

Additionally, for the abovementioned reasons, the pooling layer controls the network overfitting by eliminating the computation and parameters number used in the network, and it scales the staple representation of the input image.

Therefore, pooling motivates the network to be very invariant to the small change in the input image such as transformation, distortions, translation. Further, there are other methods of pooling such as summation and averaging. However, since it produces substantial results when down sampling the input size by 75%, taking the maximum-pooling is the most broadly used and promising technique in the literature [79].



(a) (b)

Figure 2.12: Demonstration of Max-pooling. a: The 2x2 max-pooling filter and stride 2. b: applied max pooling on the single feature map [62].

2.10.4 Fully-Connected Layer

Neural networks that are fed forward are exactly the same as neural networks that are completely linked. The last layers of the network are all linked. A flattened version of the last pooling or convolutional layer's output is used as input for the fully connected layer. It resembles how neurons are organized in a typical neural network. Consequently, every node in the preceding and next levels is linked to a node in the totally connected layer, as illustrated in figure (2.13). In this figure, it is noticeable that each of the nodes in the last frames of the (convolution or ReLU or pooling layer preceding it) are connected as a vector to the first layer from the Fully-connected layer [80].

The fundamental disadvantage of a fully linked layer is that it has multiple parameters that must be complexly computed in training instances. Therefore, the number of nodes and connections are tried to be eliminated [81].

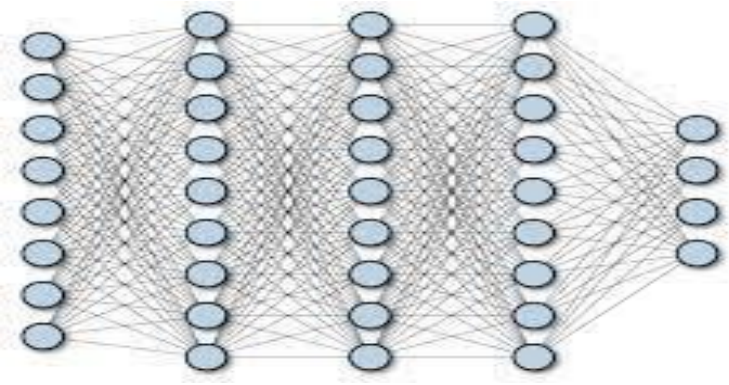


Figure 2.13: Fully-Connected Layer [82].

2.10.5 Dropout Network

The dropout method sometimes is used when the neural network is suffering from overfitting. Sometimes it is used to increase the training speed. Usually, it is used with a large neural network. Dropout is considered as the significant way to reduce the ratio of error for the network as it is applied to training examples. The dropout way is used at hidden layers, usually between the last layers in deep learning, as shown in figure (2.14). Because a completely linked layer has a lot of characteristics and connections, it's a big deal.

The nodes with probability (p), usually used (p=0.5) are left in the network while nodes with probability (1-p) are dropout out. So all nodes have a probability of fewer than 0.5 dropouts from the network. Besides all edges coming from and go to this dropout node are also removed. The network of nodes and edges eliminating are used for the training stage on the dataset [83].

The dropout technique considers as the best tool to reduce the complexity of network. Interactions between nodes that are tightly matched, causing them to learn more robust characteristics that are more generalizable to new data. One of the widespread techniques that drop out a node from the network is called Dropout Network [84].

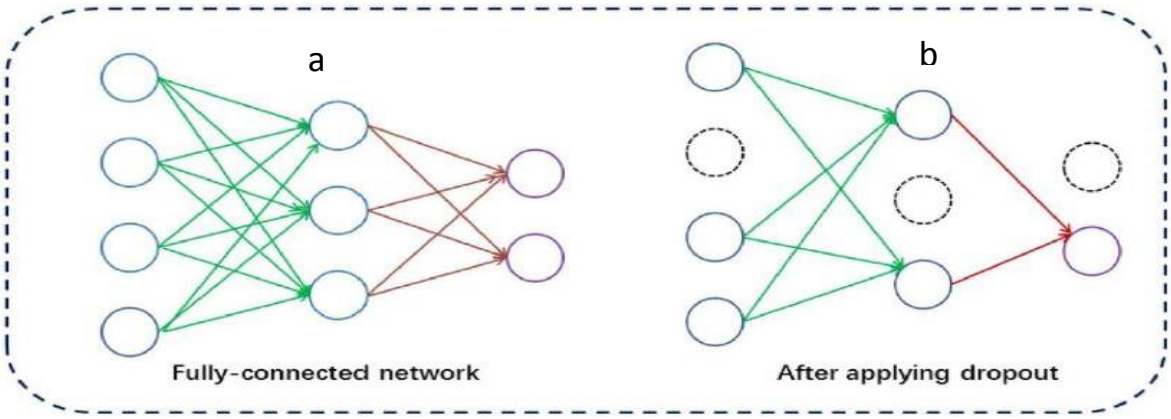


Figure 2.14: (a) Network before Dropout, (b) Network after Dropout network [84].

2.10.6 Normalization Layer In Neural Networks

The input layer is normalized by modifying and scaling the activations, e.g., if it has features from 0 to 1 and also from 1 to 1000, also must normalize those features to accelerate learning. Why not do the same for the values in the hidden layers that change all the time and increase the training pace by 10 times or more if it supports the input layer. There are various types of normalization, the most famous of which is Batch Normalization, as shown in algorithm (2.1) [85].

Input: Values of X over a mini-batch:β ={x1…..m}; Parameters to be learned :ϒ, β
Output: Scale and Shift
Step1: **Begin**
Step2: Calculate the mini-batch mean ($\mu\beta$) of the input layers.
Step3: Compute the mini-batch variance of the ($\sigma^2\beta$) layers input .
Step4: Normalize the layer inputs utilizing the prior calculated batch Statistics $\bar{\bar{X}}_1$.
Step5: Scale and shift to get output from the layer {yi $=\bar{X}+\beta \equiv BN, (x_i)$}
Step6: Return Batch Normalization
END

Figure 2.15: Algorithm Batch Normalization

Batch normalization is a method for accelerating neural network learning and stability. A batch standardization is accomplished in a neural network by normalization, which fixes the means and variances of the inputs of each layer, to compute the mini-batch mean by the equation (2.24), *m* represents the mini-activation batch's values. [86]:

$$\mu\beta = \frac{1}{m}\sum_{i=1}^{m} \text{Xi} \qquad (2.24)$$

Where μ: refers to the mean, Xi: is an instance of the input data, β: is a subscript that refers to the current batch. While computing the mini-batch variance σ2 β by the equation (2.25):

$$\sigma^{2\,\beta} = \frac{1}{m}\sum_{i=1}^{m}(xi - \mu\beta)^2 \qquad (2.25)$$

At the last, normalize the layer inputs by using the prior calculated batchstatistics as in the equation (2.26) [87]:

$$Xi = \frac{\text{Xi} - \mu\beta}{\sqrt{\alpha^2\beta + \varepsilon}} \qquad (2.26)$$

Where (ε)The smoothing term assures numerical stability within the operation by stopping division by a zero value.

2.10.7 Softmax Layer

Most agree that the SoftMax function, sometimes termed the normalized exponential function, best shows categorical distributions. SoftMax takes an N-dimensional vector of real units as input. The outcome is an N-by-M vector of 0-1 values (0 to 1). Large value is around one. Small value is near zero. One output value is possible. Even if production changes, output probably won't. [88].

The probability distribution of an N-dimensional vector is computed using the SoftMax function. The multiclass classification strategy in machine learning algorithms, deep learning, and data science drives the usage of SoftMax at the output layer. The target class for the input data set function may be determined by calculating the output likelihood appropriately [89], A

normalized exponent of the output values may be found commonly in the output layer. The SoftMax is a differentiable function that represents the output likelihood. Furthermore, the exponential factor raises the likelihood of maximum values. The SoftMax is equation (2.27).

$$Oi = \frac{e^{zi}}{\sum_{i=1}^{m} e^{zi}} \qquad (2.27)$$

SoftMax output Oi M is the number of output nodes and e is the exponential function. [90].

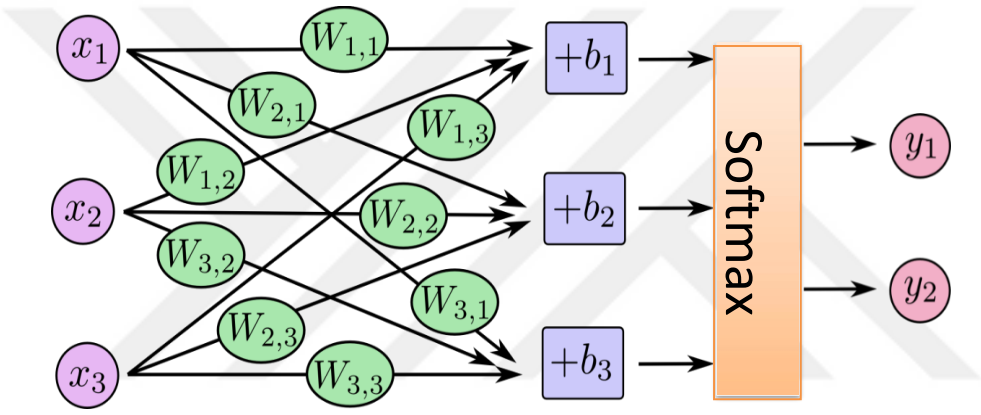


Figure 2.16: The layer of softmax position in the network [62].

2.11 LOSS FUNCTION

A loss function is used in machine learning to compare algorithmically predicted values to actual label values. Then, this error is minimized by implementing any optimization approach. Used presents the Cross Entropy Loss Function [91]. As a result of these two sets of data, the loss is computed, and its significance is defined by how well the model works. In each training or validation set, the total number of errors is recorded. To know how well or badly a model does after each optimization iteration, we look at its loss value (Loss). The loss value is a measure of how well or badly a model performs after each optimization cycle. For binary classification, binary cross entropy was used. The binary cross entropy error rate is expressed as a value between 0 and 1. It is mathematically represented by the equation (2.28):

$$J(z)=[y\log P(y)+(1-y)\log(1-P(y))] \qquad (2.28)$$

The actual labels are y, and the anticipated labels are P(y). Because y is multiplying with log, the first term will also be zero when the actual labels are 0. When y equals 1, the second term (1-y) equals 0 and is multiplied by log. And if y = P(y) then J(y)(Loss) will be zero.

2.12 STOCHASTIC GRADIENT DESCENT WITH ADAMAX (SGDA)

Optimizing a real function with suitable smoothness qualities may be accomplished by iterative means of stochastic gradient descent (commonly known as SGD) (e.g. differentiable or sub differentiable). It can be considered as a stochastic estimate of gradient descent optimization as it substitutes the true gradient (obtained from the whole dataset) with an average of it (computed from a randomly picked data sub-set). Stochastic gradient descent in machine learning has become a significant method of optimization. The trouble of reducing [92] a function objective in the form of average is known by both statistical analysis and machine learning (2.11) [93].

2.13 SHUFFLE

Shuffling methods are simply aimed at mixing data and can keep logical connections between features optional. It arbitrarily sets the data in an attribute or a number of attributes from a dataset. The critical data could be changed to be with other values from a separate record for the same attribute [94]. There are many options for shuffle dataset in python, in CNN Shuffle = (True or False) where Shuffle is set to True only instructs the neural network to collect training batches randomly from the training set. When Shuffle is set to False, the batches are collected one by one from the training dataset, using the same order that the inputs appear in the set.

2.14 NUMBER OF EPOCH

Hyper settings like the number of epochs dictate how many times the training dataset is run by the learning algorithm Using hyper parameters like epochs, the learning algorithm may control how often the training dataset is used. There are one or more batches in an epoch. The batch gradient descent learning epoch, for example, has only one batch.

The "batch size" option specifies the number of samples in each batch, and a nested for-loop iterates over those batches in this for-loop [95]. Epochs may range from hundreds to thousands, allowing the learning process to continue until the model's inaccuracy is acceptable. There are instances in the literature and tutorials of epoch counts of 10, 100, 500, 1000, and many more.

2.15 CREATING THE NETWORK

Deeper neural networks can be advantageous depending on the application. However, it introduces new criteria that must be learned. The back- propagation method is used to train the convolutional filters in the CNN. Structure of filter depends on what you are trying to accomplish. However, do not have complete control over these filters in CNNs, and their values are calculated by learning operations [96]. Figure (2.17) shows elements of CNN when all layers are connected.

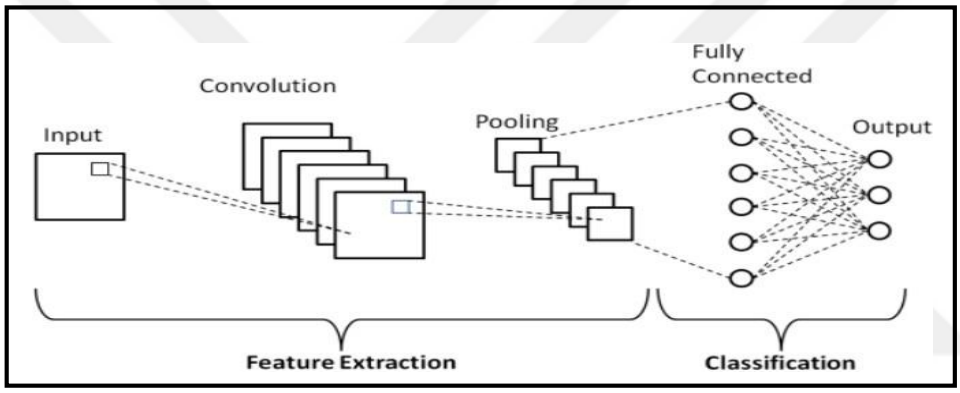


Figure 2.17: Elements of the CNN [97].

2.16 DEEP NEURAL NETWORKS (DNN)

A deep neural network (DNN) is a neural network that has more hidden nodes than a regular one. To perform more sophisticated input computations and to outperform "shallow" networks, a deep neural network uses a nonlinear transformation for each hidden layer.

A nonlinear f(x) should be used for each of the hidden layers of a single hidden layer neural network because its hidden layer depth does not boost its ability to express itself.

By dividing the procedure into DNN layers, distinct pulmonary nodule sizes may be determined. DNN shaves more than one hidden layer (l) between input and output (Good fellow et al., 2016). Each layer contains neurons that modify the input.

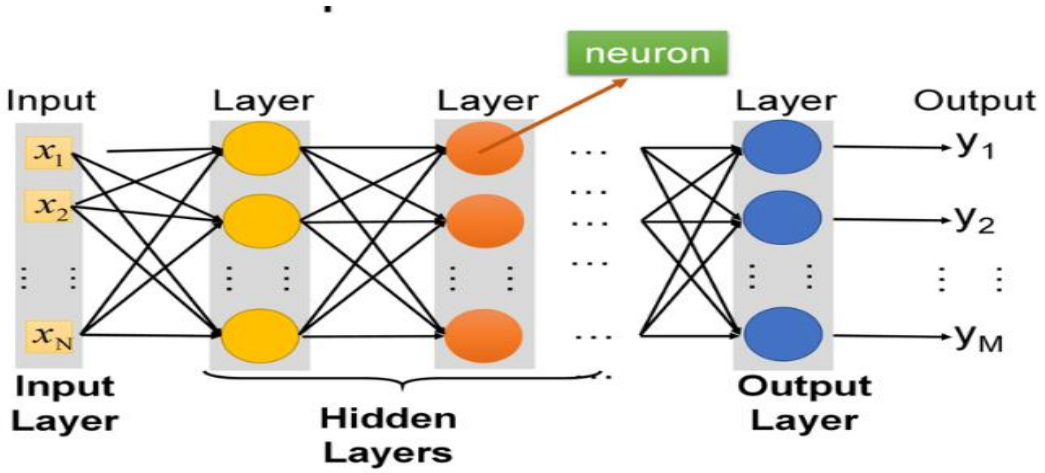


Figure 2.18: A typical architecture of DNN [91]

Each layer is always followed by a nonlinear function known as the activation function, which might be sigmoid, ReLU, or Tanh.

2.17 VGG ARCHITECTURE

Medical image processing uses leNet, AlexNet, and VGG16 CNNs. AlexNet improved CNNs in 2012 (CNNs). It topped the list of picture classification models. AlexNet is a good object-identification model. Almost immediately after its release, it won the ILSVRC. Overlapping pooling speeds up GPUs instead of the tanh function. Data augmentation and dropout rectified overfitting. Convolutional neural networks build VGG Nets (CNN). The VGG-16 contains 13 curve- and three fully linked layers. The architecture of VGG:

1. Images with a size of 224224 can be uploaded to the VGG Net. The designers of the model cut out these 224224 patches in the middle so that ImageNet would always get the same size image.
2. Convolutional Layers: The convolutional layers of VGG use a 3x3 receptive field to recognize movement up and down and left and right. 11 convolution filters change the input in a linear way.

   After this comes a ReLU unit, which is a big improvement to AlexNet that cuts down on training time.

ReLU is a piecewise linear function that gives back 1 if the input is positive and 0 if it is negative. After convolution, a 1-pixel convolution stride maintains spatial resolution (stride is the number of pixels shifts over the input matrix).

3. Hidden Layers: ReLU is used in VGG's hidden layers. Since it takes more time and memory, LRN is rarely used in VGG. It doesn't change the accuracy of the data.
4. Layers that are all linked: VGG Net has three layers that are all linked. The first two layers have 4096 channels each, while the third layer has 1000 channels, one for each class.

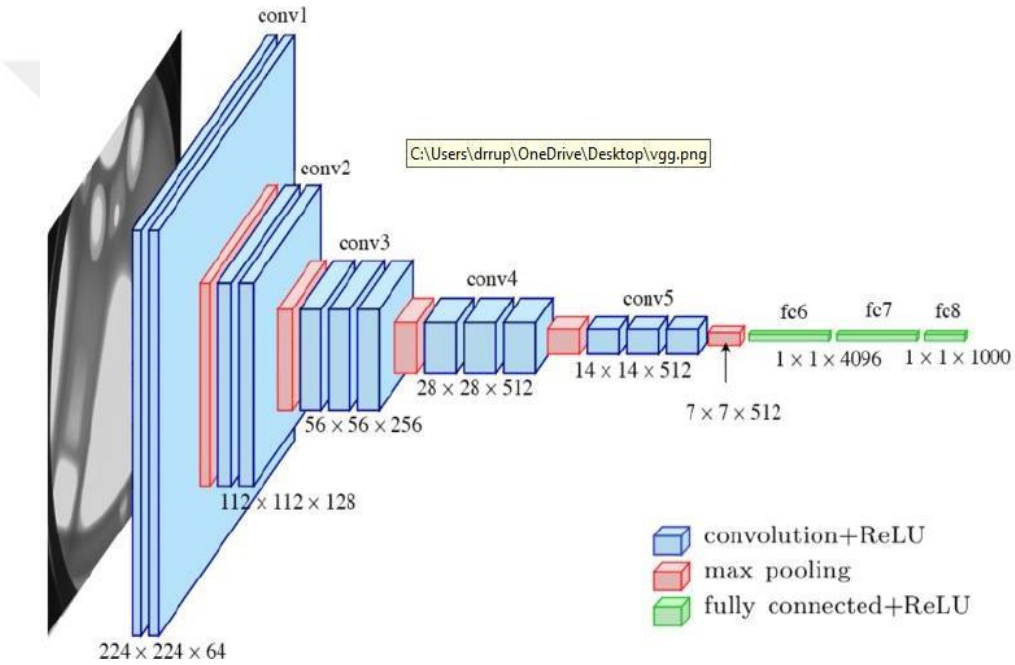


Figure 2. 19: VGG-16 Architecture proposed in 2014 by Karen Simonyan and [98].

VGGNet-16 is a 16-layer convolutional neural network with a highly attractive design. It features just 3x3 convolutions, like Alex Net, but a lot more filters. It takes two to three weeks to train on four GPUs. In the community, it is now the most used method for extracting characteristics from photos.

2.18 EVALUATION MEASURES

To calculate the number of Parameters and evaluating act of the system, may using equations following:

A. Calculating The Number Of Parameters

In each layer, the equation (2.29) is used to figure out the number of parameters. [99].

$$\text{No. parameters} = \text{output channels} \times (\text{input channels} \times \text{window size} + 1) \qquad (2.29)$$

B. Accuracy (Classification Rate):

It is obvious as the ratio between the number of true Classifications and the number of sample tests utilized during training and testing. Formal accuracy has been established [100].

$$Accuracy = \frac{TP+TN}{Total\ number\ of\ test\ samples} * 100 \qquad (2.30)$$

C. Recall:

Sensitivity is the ratio of correct positive predictions to total positive predictions [100].

$$\text{Recall} = \frac{TP}{TP+FN} * 100 \qquad (2.31)$$

D. Precision:

Specificity is denoted the ratio between the numbers of correct negative prediction and all number of negative predictions [100].

$$\text{Precision} = \frac{TN}{TN+FP} * 100 \qquad (2.32)$$

E. F1-score:

Is Precision plus Recall weighted. This score includes false positives and negatives. F1 is harder to use, but it's better if your classes aren't all the same size. When false positives and negatives cost the same, accuracy is good.

Precision and Recall should be included if there is a considerable difference in the cost of false positives and false negatives [101].

$$F1score = \frac{2*Recall*Precision}{Recall+Precision} \quad (2.33)$$

A. True Positives (TP): When a data point's actual and expected classes are both True, we have a true positive.

B. True Negatives (TN): True negatives are data points who's actual and expected classes are both False**.**

A. False Positives (FP): A data item is a false positive if its actual class is False while the expected value is True.

C. False Negatives (FN): False negatives have True actual class but **False** anticipated class.

3.THE PROPOSED SYSTEM IMPLEMENTATION

3.1 INTRODUCTION

As mentioned earlier in the previous chapters, lung disease like lung cancer and covid-19 causing an increase in deaths. Lung cancer is the most common kind of cancer in both men and women compared to all other types of cancer. Today, COVID-19 has had a significant worldwide effect and is contaminated by over 8 million people worldwide. This chapter illustrates the proposed lung cancer classification system design stages and the implementation requirements by using ML and DL algorithms to assist physicians in the diagnosis procedure. However, according to the high complexity of the patterns that appear in the dataset, which are used by the classifiers to extract the required knowledge, Data preparation may have a big impact on classifier performance. Accordingly, two different preprocessing techniques are evaluated in this study, to illustrate their influence on the performances of different classifiers.

As shown in Figure (3.1), This study looks into the effectiveness of two different classifiers in cancer diagnosis, as well as the impact of preprocessing approaches like standardization and normalization. The performance metrics of different combinations may then be used to compare the utility of these classifiers and preprocessing procedures in cancer diagnosis. The model's complexity is measured by the accuracy of the classifier's predictions and the time it takes it to correctly classify each input.

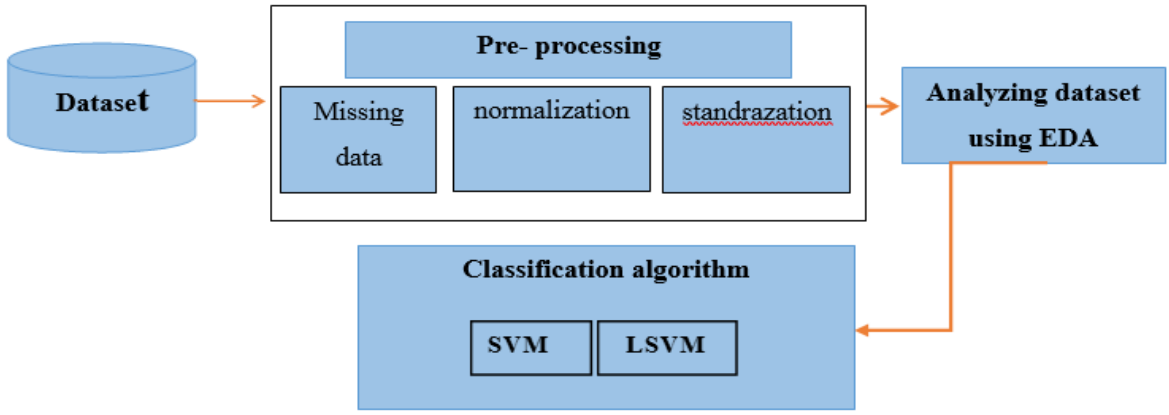


Figure 3.1: The proposed system model for SVC dataset

3.2 THE PROPOSED MODELS

The proposed system aims to classify lung diseases. This work includes two approaches. They are separate; each one is different from the other. They are applied on two different types of datasets.

Case1: The First Proposed Approach Is Classification with Machine Learning Technique

The first proposed approach based on three different algorithms of machine learning (SVM, LSVM) aim to recognize and classify both lung cancer disease. The proposed system has been recognized the cancer from datasets. A dataset serves as the system's input, while cancer categorization serves as the system's output. The proposed system consists of three main stages: dataset acquisition, preprocessing, and classification. Figure (3.1) illustrates the suggested system's overall block diagram for recognizing lung cancer from a given dataset svc.

3.3 DATA ACQUISITION STAGE (CSV LUNG CANCER DATASET)

The dataset of lung cancer disease used in this study is free to access obtained from the github website. The dataset has 25 attributes and 1000 instances. The dataset will be splatted into two parts the first (80%) is utilized to train lung cancer disease data and the second (20%) is utilized as test instances to test classification system performance. The data collection is divided into three cases of labeled (0) low, (1) medium and (2) normal.

3.4 DATA PRE-PROSSESSIING STAGE

Preprocessing (in ML algorithms) converts raw data into a format more appropriate for classifiers so they can quickly find patterns.

3.4.1 Missing Data Preprocessing for ML Algorithms

In real-life data, some of the attribute values may be missing, according to an error or lack of information being collected for a certain patient. Accordingly, it is important to investigate the existence of missing values in the dataset and replace them with a value that the classifier can handle, as by default classifiers cannot handle missing values.

In the proposed method, a missing value is replaced with the average value of that feature for the instances that belong to the same class. This approach ensures that no abnormal values

appear for instances at different classes, which can dramatically limit the performance of the classifier. Each value v in a feature F is replaced with a new value using Equation (3.1).

$$\hat{v} = \frac{v - min(F)}{max(F) - min(F)} \qquad (3.1)$$

3.4.2 Data Normalization

The feature values that are collected for different patients are different from each other, where these values characterize each input to the classifier. Moreover, the ranges of these values are different from one feature to another, depending on the nature of the attribute being used and its values. Data normalization defines a certain range for all the attributes, where the values in each attribute are normalized to the defined range, i.e., interval, to maintain relativity among these values. However, by using the same interval, the influence of the values of different features is balanced to the classifier, so that, the values with larger scales do not dominate the computations. Hence, the classifier can detect the required patterns easier and faster.

Input: Data

Output: New normalized data

Begin

Step1: Apply Z-Score normalization using the equation in (3.1)

Step2: determine the data set's minimum and maximum values, which are indicated by x minimum and x maximum.

Step 3: subtract the lowest value from the greatest value to get the range of the data set.

Range= $x_{maximum}$ and $x_{minimum}$

Step4: Determine how much more valuable the variable is when normalized from the minimal value by subtracting the variable's minimum value.

i.e., $x - x_{minimum}$

Step 5: By dividing the expression in step 3 by the expression in step 2, the formula for calculating the normalization of the variable x is generated.

Step 6: Find minimum and maximum values from the values results

Step7: Find the Rang

Rang= max-min

End

Figure 3.2: Algorithm normalization

3.4.3 Data Standardization

In data preprocessing, the difference between the value and the mean of the value in the feature is assessed in terms of standard deviation, as shown in Equation (3.2). Accordingly, the output of standardization has a mean value of zero, but contains positive and negative values, unlike data normalization.

The existence of negative and positive values, with a mean of zero, can significantly improve the performance of some classifiers, according to the computations that these classifiers use to extract the knowledge in the training dataset. Hence, both normalization and standardization are evaluated in this study to investigate their influence on the performance of different classifiers.

$$\hat{v} = \frac{v - mean(F)}{Standard\ Deviation\ (F)} \qquad (3.2)$$

Input: Data **Output**: New standardized data
Begin Step 1: Apply Z-Score normalizations using the equation (3.2) Step 2: A sample x's standard score is determined as follows: z = \frac{x - \mu} {\sigma} Step 3: The training samples' mean = zero, and their standard deviation = one. Step 4: By determining mean and standard deviation, we may center and scale each feature separately. Step 5: rescale features value with distribution value between o and 1 End

Figure 3.3: Algorithm standardization

3.5 DATA SPLIT

In supervised learning, a training set that includes labeled samples of the domain that the classifier is interacting with is required by the classifier to extract the required knowledge and use it to predict future, unlabeled, inputs. Furthermore, labeled data are necessary to assess the classifier's effectiveness by comparing the classifier's predictions to the inputs' actual classifications. However, the use of data instances that appear in the training set for evaluation is considered biased evaluation.

For instance, an over fitted classifier, to the training set, can produce high performance measures when the testing set is included in the training, whereas a more generalized classifier, which may be significantly better, produces lower performance measures.

A training and testing set is created by randomly selecting occurrences from the given data for each group.

In this research, 80% of the data has been used to teach and 20% was used to test. This procedure ensures unbiased evaluation, as the testing data are not included in the training data, while maintaining a significant ratio of the data for the evaluation purpose

3.6 CLASSIFIERS IMPLEMENTATION AND TRAINING

Two classification approaches are used in this study to evaluate their performance in cancer classification and the influence of preprocessing on their performances in such an application. The configurations of the implemented classifiers, training and how data are presented to the classifier are described in this section.

3.6.1 The Linear Support Vector Machine

Instead of the radial kernel, this model uses a linear kernel with the parameters shown in Table (3.1), which characterize the operation of this classifier as illustrated in Section (2.6.1.1). The implemented model is trained and evaluated using the training and testing datasets that are preprocessed using the two approaches described above.

Input: Training _set (Features function)

Output: Class name

Step 1: Begin
Step 2: Identify three classes and a training label for all training sets.

Step 3: Define the kernel function (linear kernel function), which was determined in the second chapter in equation (2.17) and is used to transfer the training set to kernel space using hyperplane computing (2.13)

Step 4: Data was separated into 1 against all methods.
Step 5: Passes SVM testing set based on hyperplane to identify class name.

Step 6: Return the class name

Step 7: End algorithm

Figure 3.4: Algorithm Multi-class linear Support vector machine classifier

Table 3.1: Parameters of the implemented Linear SVM classifier.

Parameter	Value
Penalty	**L2**
Loss function	**Squared hinge**
Optimization	**Dual**
Stopping tolerance	**1e-4**
Regularization inverse factor	**1**
Multi-class strategy	**Ovr**
Class weight	**Balanced**
Maximum number of iterations	**1000**

3.6.2 The Radial Support Vector Machines

As explained in Section (2.6.1.2), the SVM classifier can have different kernels, depending on the type of equations it uses to generate the hyperplanes that divide the instance in the domain space into their classes. For the radial kernel, the implemented model uses the value 1 for regularization and a third-degree polynomial function for its kernel.

A scaled gamma is used for the kernel coefficient, which is calculated as explained in Section (2.6.1.3). The implemented model is trained twice, using the different preprocessing techniques, using the training set, and evaluated using the testing sets, each corresponding to the preprocessing method used for the training set.

Input: Training _set (Features function)

Output: Class name

Step 1: Begin
Step 2: Create a training label for each training set and define 3 classes.
Step 3: Determine the kernel function ((RDF function), as derived in the second chapter in equation (2.15)) that will be used to transfer the training set to kernel space through hyperplane computing (2.13)
Step 4: Data was segregated to 1 against all methods.
Step 5: Passes SVM testing set based on the hyperplane to determine the class name.
Step 6: Return the class name
Step 7: End algorithm

Figure 3.5: Algorithm Multi-class non linear Support vector machine classifier

3.6.2.1 Svm training

SVM can usually distinguish two classes. It can distinguish more than two classes in this scenario. So, in chapter 2, section (2.6.1.2). Class A vs. the rest, Class B vs. the rest, etc. "Class n vs. the rest" is antonym. MSVM captures data for training so it may be used in any class.

Each SVM binary classifier is taught to use each vector row as a class inquiry characteristic. After training, the multi-class SVM model assigns the input features vector a class.

3.6.2.2 Svm testing

An independent test dataset was created, and the data was divided into groups according to how well the trained SVM model performed in its accuracy test. The suggested training model is tested and judged at random during the testing stage.

Each extracted feature vector row is categorized and unlabeled; in training, rows are labeled. The classification system used training and features vectors. Each sample feature is a column in a matrix. Due to several columns of attributes, the sample size must be divided.

Case2: The Scond Proposed Approach:(Classification with Deep Learning Techniques)

Figure (3.2) shows the second proposed solution, which uses convolutional neural network (CNN), deep neural network (DNN), and VGG-16. It also includes numerous stages that are dependent on the construction of this task. The input to the system image dataset and the output is classifying five cases for lung disease.

3.7 DATA ACQISITION STAGE (CT SACN IMAGES)

Different sources of CT images were accessed and compiled for our experiments. COVID-19 and lung cancer CT pictures, as well as normal scans, are included in this collection. For COVID-19, we first chose datasets from [101] SARS-CoV-2 CT scan dataset, which comprises (1252) CT scans for patients with (COVID-19) infection and (1230) CT scans for healthy people, for a total of (2482) CT scans. Secondly, lung cancer is the dataset's source CT images that are collected in Iraqi hospitals, totaling (1315) images that can be utilized to train our proposed deep learning system (CNN, DNN and VGG) to distinguish (Covid-19, NoCovid), lung cancer (Benign, Malignant, Normal).

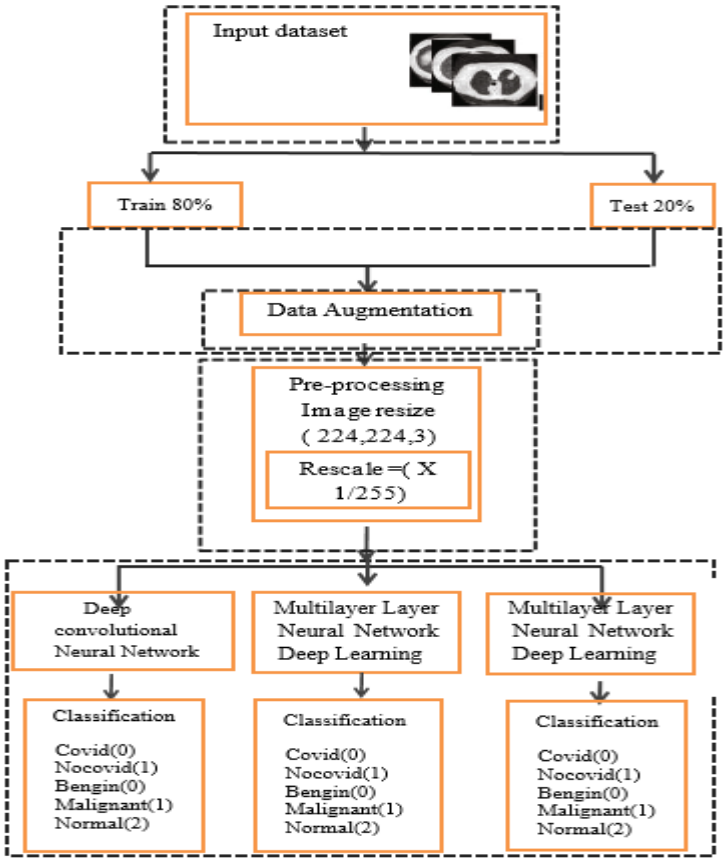


Figure 3.6: proposed system diagram of the (CNN , DNN ,VGG-16) models.

3.8 TRAINING MODELS

The flowing algorithms (3.5) illustrate the general steps of the used methods for training the (CNN) model and the same steps is repeated for Covid-19 dataset except the output will be (Covid(1) or No Covid (0)).

Input: CT scan image.
Output: Lung cancer (low (1) or medium (0) or high (2))
Step1: Splitting the data into two parts, 80% for training and 20% for thetesting.
Step2: Data augmentation To artificially enhance the size of an actual dataset,approaches generate multiple versions of it. such as (Rotation, resize, zooming, flipping, shear...) are applied to the images.
Step3: Pre-processing stage in it the image is resized from various sizes to (244* 244 *3). Also, the data will be normalized by divided each pixel into 255. So the value of data becomes between (0 and 1).
Step4: CNN design which consisting of several layers: a) Input Layer: lung cancer (images) 224*224 *3 size. b) Convolution layer: Multiple filters (layer1=32 filter, layer2=64 filter , layer3=128 filter, layer4=256 filter, layder5=512 filter) were usedwith size (3×3). c) Nonlinear layer (Activation layer): Using Rectified linear units(ReLU). d) Pooling layer: Using the Max-pooling layer used with size (2×2) and2 stride. e) Normalize layer: using batch normalization. f) Fully connected: Multiple node three-layer FC (Layer1=512, Layer2=128, Layer3=64) g) Softmax layer
Step5: Return the CNN with the optimum validation accuracy.
END

Figure 3.7: Algorithm CNN training algorithm to classify lung cancer.

3.8.1 Image Pre-processing Stage

The dataset will be augmented via several random, such as blur and rotate, to create new versions of existing images or frames. Transformations to make the most of the few dataset examples as explained earlier in chapter 2 (2.9). This ensures that the algorithm never sees the same image twice. Furthermore, the model generalizes more effectively, and it will prevent the system from overfitting.

3.8.2 Data Preparation For Deep Learning Algorithms

This section illustrates the main idea to have a more accurate result using all available data. Also, in some algorithms, the preprocessing operations are calculated over the entire

image. Any image in (CNN, DNN and VGG) should have a defined resolution (e.g., all input images must have a resolution of 224×224 pixels). The fixed image size provides us with more options in designing the model architecture. The samples are provided to the CNN as an image of fixed dimensions. The dimension of the network input is set to 224×224×3. Finding the right image size is one of the most difficult aspects of our proposed process. The choice of an effective scale is a trade-off between precision (a larger size) and classification speed (a smaller size). To put it another way, a bigger image size requires more data and, hopefully, greater precision. A smaller image size, on the other hand, means a smaller volume of detail in real-world implementation and, as a result, better classification because the large size of image mean more information input to network.

3.8.3 Convolutional Neural Networks (CNN)

The second part of the proposed model aims to recognize lung cancer and Covid-19 Using CT scans. To handle the amount of information in each input image, one of the most robust methods is used, which is Deep Learning. Which has become very popular in the literature; Deep learning, and particularly, the CNN is artificial multilayers neural network that is currently surpassing the performance of classical methods in several areas such as Pattern Recognition, image, and Object Detection. Thus, the CNN method has been selected for the following reasons:

A. High performance, accuracy, and outperform.
B. The CNN can be trained for an end-to-end architecture.
C. I/t requires no operations for preprocessing (except resize and normalize the input image).
D. It can classify the class of the cancer into (Benign, Malignant, Normal) for lung cancer dataset and (Covid, Nocovid) for Covid-19 dataset in almost real-time, after (few milliseconds) receiving the input image.

3.9 THE PROPOSED CNN ARCHITECTURE

After selecting the image size for our method. The second problem is to devise an appropriate CNN architecture. Therefore, firstly the input and the output structure of the network will be

defined. The ideal architecture will then be offered depending on the outcomes of the trials. The image size (CT) was selected as (224×224×3).

Nevertheless, in order to have a robust implementation, the input size must be fixed. In our technique, the output shape is the SoftMax function, which has two classes for Covid-19 and three classes for Lung cancer. As an output, a SoftMax layer is utilized to choose one of the classes. Although the highest value is chosen as the computed class in the output node, the SoftMax values take into account additional high probability hypotheses. The accuracy of the CNN's performance is affected by factors such as filter size and the number of filters employed in convolutional layers.

The number of feature maps is equal to the number of filters that are used; pooling filter size also influences network accuracy. The number of convolutional layers influences the outcome significantly. Our classifier models were built by cascading convolutional layers, as shown in figure (3.3). Convolution, non-linearity, and pooling layers comprise each convolutional network. And the pooling layer's output is linked to multiple fully connected levels before being linked to the final layer (classifier layer). The number of layers and parameters of the first network with the fully connected for the lung cancer and Covid-19 is as shown in the next chapter.

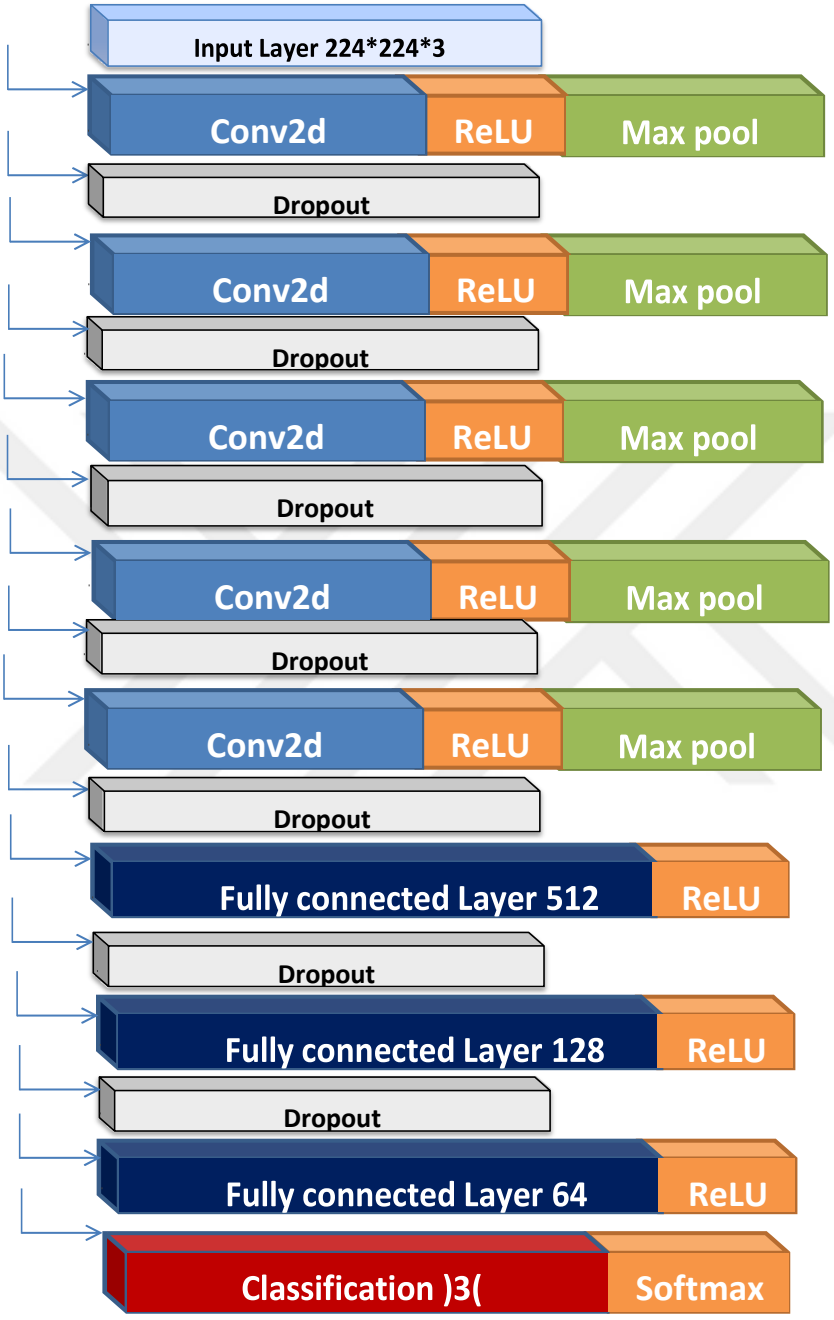


Figure 3.8: The structure of CNN for lung cancer dataset

CNNs are multilayer neural networks with convolution and fully linked layers. CNN has input, output, and hidden layers. Feed-forward neural networks have "hidden" intermediate layers.

The hidden layers include layers that produce convolution in a coevolutionary neural network. This usually involves a layer that performs a dot product with the layer input matrix in the convolution kernel. The activation of this product is usually the ReLU, and it is usually the Fresenius inner product. As the convolution kernel moves through the layer input matrix, it forms an options map, which in turn adds to the input of the subsequent layer. There are several different kinds of layers accessible, including pooling layers, fully connected layers, and normalizing layers.

3.9.1 Input Layer

Image data should be included in the CNN input layer. A 3-dimensional matrix is used to represent image data. These images are the images of lung cancer and Covid-19 diseases that entered in matrix form, or in other words layers, whose size has been reduced to the resize of (224×224×3).

3.9.2 Convolution Layer

Convolution layer extracts picture characteristics. First, a segment of the image is connected to the Convolution layer so the dot product between the receiving field (a region the same size as the filter) and the filter can be determined. Output volume has one number. So, integer output is 1. Repeat with the next responsive area. Finish the photo. Next-layer output is input.

Convolution extracts picture features. Convolution learns visual properties from small input squares to maintain pixel relationships. Image matrix and filter or kernel are inputs to a mathematical technique.

A convolution Neural Network uses several kernels, convolution process uses filters to get a perfect feature map.

CNN uses kernel convolution and smart vision techniques. It's a technique that takes a brief array of integers dubbed a kernel or filter.

These filters work on every section of the image it is searching for a similar feature throughout the image. Edge detection, blurring, and sharpening may all be accomplished through convolution with various filters. All filters which used an edge detection filters.

The equation from chapter 2 is used to calculate the values (2.10.1). The structure follows the same design present in architecture layers in the network lung cancer (5 Conv., 5 max pooling, and 3 FC layers). To determine how many filters should be used in each convolution layer, we evaluated the results of multiple tests to determine the optimal number of filters for each layer.

3.9.3 Non-Linear Layer (Activation Function)

This was covered in detail in chapter two, section (2.10.2). The CNN suggested method uses a batch normalization layer in the hidden layer; it uses a mini batch to build norms for any channel. This can help to reduce the sensitivity to differences in the outcomes. It solves the problem of fitting because it has a minor influence on regularization. Batch normalization improves the yield of the prior activation layer. To compute batch normalization, first determine the yield of the prior activation layer.

3.9.4 Pooling Layer

After the 2D convolution layer has been filtered, the pooling layer is another CNN building piece that is utilized to reduce the sample size of the function mappings generated. Following a coevolutionary layer is a pooling layer, which can be used to reduce feature map dimensions. Like convolutional layers, pooling layer algorithms take into account nearby pixels and are detailed in full in Chapter 2 in section (2.10.3). In the proposed system 2*2 Max-pooling is used with two strides, the "stride" is mean “the step scale to vertically and horizontally traverse the input”.

According to Figure (2.12) a and b, a maximum value in each (2×2) matrix is selected and therefore, the spatial dimensions are reduced to half of their initial size.

.

3.9.5 Normalization Layer

One of the most essential layers was explained in Chapter 2 of this text, section (2.10.6), as shown in algorithm (2.1). The CNN suggested method uses a batch normalization layer in the hidden layer; it uses a mini batch to build norms for any channel. This can help to reduce the sensitivity to differences in the outcomes. It solves the problem of fitting because it has a minor influence on regularization. Batch normalization improves the yield of the prior activation layer. For batch normalization, first determine the prior activation layer's yield, which was shown in the previous chapter in the equations (2.24), (2.25) and (2.26).

3.9.6 Fully Connected Layer

Deep convolutional neural networks use fully connected layers to classify data after convolutional and pooling layers (FC). Fully linked networks establish many network settings. As mentioned in chapter 2, section 2.10.4, this layer's neurons are directly connected to the layer above. This is shown in Figure (1.13). Over time, each neuron in the fully connected layers can learn input data, allowing it to forecast the proper SoftMax class value show Figure (3.3). From this step, training data and class count are retrieved as values for each class connected to the Fully Connected layer image. Last step uses SoftMax.

3.9.7 Softmax Activation Function

In neural networks, the widely used activation function is the SoftMax function, In classification issues, it's common to complete the convolution neural network with a SoftMax function. For binary classification, the results fall within the range of [0, 1]. The SoftMax function equation is described in the previous chapter in part (2.10.7) and equation (2.27).

In a fully linked step, values from two classes are retrieved and the class is SoftMax so that the class is picked based on the features collected from earlier layers, from which the photos of diseases have taken place.

.

3.10 CNN TRAINING

Convolutional layers and fully connected layers are trained by reducing the discrepancies between predictions and the truth labels for a training dataset. As one of CNN's most critical stages, this one involves the extraction of features from images of cancer disease in order for the network to learn from these features so that each image may be recognized based on its own unique features.

3.10.1 Layers Designing

The meant layers are the layers that are built in the construction of the CNN structure in the training stage, all of these layers mentioned earlier are also be passed in order to extract the features and learn from them to be classified through these features. All the data will be passed through the layers that structure for training.

3.10.2 Training Option (Training Algorithm)

In order to finish the training, the following choices must be utilized in the Python language by parameters established during the training process, among them:

A. **Stochastic Gradient Descent with AdaMax Optimizer**

In several fields of science and engineering, stochastic gradient-based optimization is of key practical significance. Many problems can be created in these areas as a way to maximize or minimize the scalar parameterized objective function for their parameters. The optimization equation is described in the previous chapter in part (2.12) and equation (2.11) and (2.12), As a black box, gradient descent is the favored method of optimizing neural networks and many other machine learning algorithms.

The AdaMax is a stochastic gradient descent adaptive approach based on the infinity norm and an Adam version. Adam, a first-order gradient optimization algorithm focused on adaptive estimations of lower-order moments for stochastic objective functions. This method is easy to implement, computer effective, with low memory requirements, invariant in diagonal gradient rescaling, and well suited to data and parameter-intensive problems. A non-stationary target and noisy gradients are also suited for this method. Intuitive representations of hyper-parameters usually involve little tuning.

Large datasets are prone to duplication since batch gradient descent must recalculate the same samples before changing parameters. By updating one at a time, SGD removes duplication. So, it's easy. SGD's frequent updates cause the goal function to vary greatly.

B. Max Epoch

In machine learning, an epoch is the number of times the machine learning algorithm has run over all of the data it has gathered so far. Currently, the datasets are organized into batches. Some people use the word iteration freely and refer to the model as an iteration for a batch. After testing and trying a lot of numbers of epochs that was explained in chapter two, section (2.14), the most effective maximum number of epochs for training in lung cancer and Covid-19 was 100, with a batch size of 32, as shown in the next chapter.

C. Shuffle

The data is shuffled to have different data for each batch, and this command shuffles the training data. In Python, there are several alternatives for shuffle datasets that was explained in chapter two, section (2.13). In the proposed system, " shuffle=(True)" is utilized for shuffling.

D. Loss Function

It's a way to see how well data is modeled on specific algorithms. If forecasts are too far from the actual outcomes, then loss function is too high.
The Loss Function equation is described in the previous chapter in part (2.11) and equations (2.28). The loss function progressively reduces the error in the estimation by means of a certain optimization function.No one size suits all loss functions of machine learning algorithms.

Diverse considerations are involved in the choice of a loss function for a particular problem, such as the kind of learning machine algorithm selected and the facility to calculate derivatives.

3.11 CNN TESTING

The model's performance can be assessed by comparing it to a set of observed data called the test set. As the dataset passes from (Lung cancer and Covid-19) diseases images on this neural network, the convolution neural network was used to train the algorithm, and

the features were derived by using the classes that were learned in the previous phase.

The dataset is also used that is allocated to the testing process when it is divided, based on this dataset and the training data for testing; the classification process for (Lung cancer and Covid-19) diseases images is done. Before the testing stage, there is the training stage. So, the main difference between training and testing, is test data used prediction the output to measure the accuracy, unlike training data that is labeled. In the proposed system, this function uses in Python, scores= model. evaluate (test data), That categorizes each row in a dataset into a training group. All array sizes must be the same for both training and sample arrays. The specific values determine groups, and each component determines which group the accompanying training row belongs to. The final step, which is the classification of both diseases, has been reached.

3.12 DEEP NEURALNETWORKS (DNN)

Multi-layered Deep Neural Networks algorithms extract higher-level information from raw input. This algorithm compares and fully comprehends the capabilities of the CNN model.

To provide a fair comparison, the input image was the same size (224x224x3) and all image pixels were used in the model, the CNN with DNN algorithms will be compared because the input to both algorithms is the complete image information without extracting the image details and using it as an input. Each neuron extracts a feature from the data it receives. Each of which is linked to all of these characteristics. Each neuron has its own set of weights that are applied to the features. During the network's training, you must choose weights for each of the neurons in such a way that the output supplied by the entire network is realistic. The structure and number of DNN layers for cancer are depicted in Figure (3.4) that have seven layers that are fully connected and a ReLU activation function.

The training and testing phases of a deep neural network algorithm are the most important. The data will be divided into two sets. The first is used to train a classification system on cancer data, while the second is used to test the system's accuracy. After being translated into numeric vectors, the outcomes of the class association rule algorithm are recorded in the database in this system. The DNN algorithm takes these vectors as input.

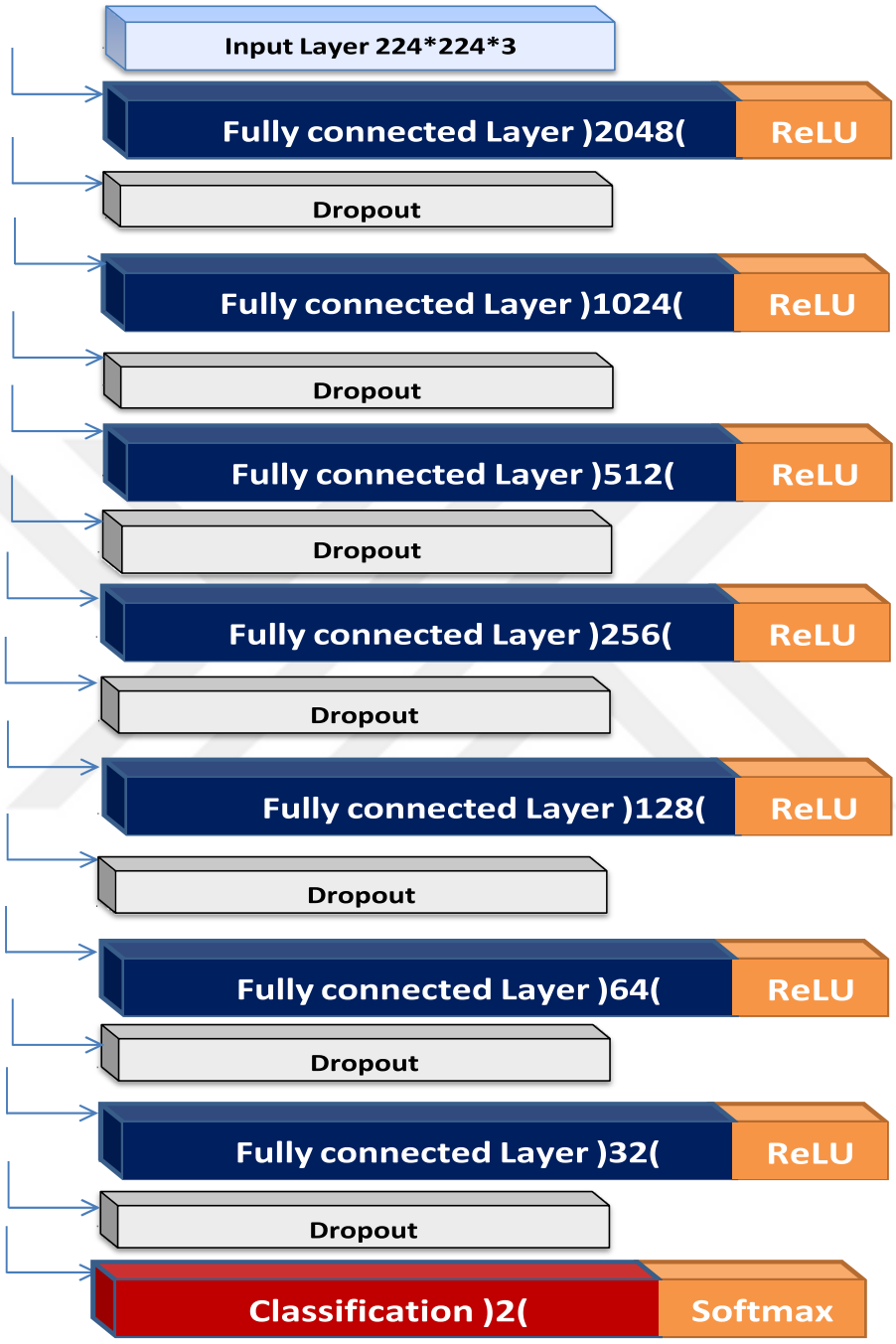


Figure 3.9: The structure of DNN for lung cancer

3.13 TRAINING PHASE

The DNNs training process is the most important part of the system. Forwarding propagation is used to start the learning process, with each neuron's inputs multiplied by its weight before the output is summed with a bias value. All neurons in the output layer

are computed at the end of the forward propagation. When the actual output does not meet the goal output, errors are computed into the output layer to alter the weights of each level according to the error gradient descent, and these mistakes are propagated back to the hidden levels and the input layer. When the mistakes in the output are reduced to an acceptable level, the procedure is complete algorithm (3.7) explains the steps of the DNNs classifier.

3.14 TESTING PHASE

A holdout dataset is used to assess the performance of the suggested training system, where only a fraction of the pictures is utilized for testing. In certain publications, the phrase "validation set" is used instead of "test set.". This step is critical because it allows us to classify all of the unlabeled items. The classifier follows the same approach that was utilized during the training phase.

Input: CT sacn image
Output: Lung cancer (Benign(0) or Malignant (1) or normal (2))
Begin
Step1: Splitting the data into two parts, 80% for training and 20% for the testing.
Step2: Data augmentation To artificially enhance the size of an actual dataset,approaches generate multiple versions of it. such as (Rotation, resize, zooming, flipping, shear...) are applied to the images.
Step3: Pre-processing stage in it the image is resized from various sizes to 50* 50 *3. Also, the data will be normalized by divided each pixel into 255. So the value of data becomes between (0 and 1).
Step4: DNN design which consisting of several layers: a)Input image of lung cancer (CT images) 224*224 *3 size. b)Flatten image into one dimensional consist (150,528) neurons. c)Dense layer: FC layer (Layer1=2048, Layer2=1024, Layer3=512, Layer4=265, Layer5=128, Layer6=64, Layer7=32). d) Nonlinear layer (Activation layer): Using Rectified linear units (ReLU). d) Softmax layer.
Step5: Return the DNN with the optimum validation accuracy.
END

Figure 3.10: Algorithm DNN training algorithm to classify lung cancer disease

3.15 VISUAL GEOMETRY GROUP (VGG-16)

A deep model that regarded as one of the best vision model architectures ever developed comprises of a VGG16 pre-trained model for utilizing CT to identify chest illnesses. The proposed model includes layers for input pictures, feature extraction, and classification.

.

3.16 TRAINING PHASE

VGG-16's input layer receives images (2242243). Convolution and ReLU layers are in CNN's initial block. Figure 2 illustrates the second block's two convolution layers, two ReLU layers, batch normalizing layer, maximum pooling layer, and dropout layer (2.19). A thick layer of 512 neurons precedes the dropout layer in categorization. The final output is created by a dense layer with two neurons and the SoftMax activation function, which sorts the output picture into one of four chest problems: (Covid-19, NOCOVID) for the Covid dataset and (Benign, Malignant, Normal) for the lung cancer dataset, which is broken into two groups; VGG Lung cancer has (35,475,139) trainable and (128) non-trainable characteristics. COVID contains (35,474,498) trainable and (128) non-trainable characteristics. Training changes trainable parameters over time. Non-trainable parameters don't change over time. If a model's parameters can't be trained, they must be set up ahead of time or given as inputs. So, being untrainable won't affect classification.

3.17 TESTING PHASE

The classifier in VGG -16 follows the same approach that was utilized during the training phase in algorithm (3.5) for the two dataset the models except the output step for Covid dataset will be (Covid(1),no covid(0)).

Input: CT scan image.
Output: Lung cancer (Benign(0) or Malignant (1) or normal (2))
Step1: Splitting the data into two parts, 80% for training and 20% for thetesting.
Step2: Data augmentation To artificially enhance the size of an actual dataset,approaches generate multiple versions of it. such as (Rotation, resize, zooming, flipping, shear...) are applied to the images.
Step3: Pre-processing stage in it the image is resized from various sizes to (244* 244 *3). Also, the data will be normalized by divided each pixel into 255. So the value of data becomes between (0 and 1).
Step4: VGG design which consisting of several layers: a) Input Layer: lung cancer images (224*224 *3) size. b) 8 Convolution layer with Multiple filters were used with size (3×3). c) Nonlinear layer (Activation layer): Using Rectified linear units(ReLU). d) Pooling layer: Using the Max-pooling layer used with size (2×2) and2 stride. e) Normalize layer: using batch normalization. f) Fully connected: Multiple node two-layer FC (Layer1=1024, Layer2=256) g) Softmax layer
Step5: Return the CNN with the optimum validation accuracy.
END

Figure 3.11: Algorithm VGG-16 training algorithm to classify lung cancer.

4. EXPERIMENTAL RESULTS AND EVALUATION

4.1 INTRODUCTION

In this section, we talk in detail about what happened when the suggested approach was used and tested. Based on the algorithms that are used, there are two different sub-approaches in this method. The results of the second approach, which is based on (CNN, DNN, VGG16) algorithms, are added to the results of the first proposed method, which was already explained. The results of these two methods will be looked at side by side. There will also be a comparison with similar works. In each of the two parts, a different set of data was used. Python is the language used to write the code for the proposed system.

4.2 IMPLEMENTATION ENVIRONMENT

Google Collaborator, or Google Colab in short, is the Integrated Development Environment (IDE) used in the process. Google Colab allows you to write and execute codes, save and exchange analyses, during a single runtime, there are 12.72 GB of RAM and 358.27 GB of hard drive space on Google Colab. After 12 hours, the runtime is reset, and the user must reconnect to the server. Cases are listed below. After opening a Google Colab file, you must choose a runtime type. There are 3 choices for the same,

1. None (uses user's CPU) 2. GPU 3. TPU

The selection box is found in Runtime all from the convenience of your browser. The dataset is downloaded and then uploaded to Google Drive in order to conduct the tests.

4.3 DATASET ACQUISTION

The suggested classification system was trained and evaluated on two datasets:

I. CSV for lung cancer dataset.
II. CT scan images for lung cancer and Covid-19 datasets.

Case1: The System Results for the First Proposed Approach

This case will display the results obtained from the implementation of preprocessing stage and first proposed approach. It includes the results of the classification of Lung cancer disease using

machine learning algorithm supervised vector machine (SVM), unsupervised vector machine (LSVM).

4.3.1 Pre-Processing Results

The process of applying the classification Lung cancer system in a correctly and integrated is needed to process the data before applying any proposed algorithm. The results of any phase will be the input to the next phase, so the data must be equipped with keeping the same specifications, in this phase process handle missing value and normalization are performed for the dataset. These processes will be explained as the following:

4.3.1.1 Data cleaning

It is essential to verify the dataset for missing values. In this thesis, the way used for handling the data is replacing each missing value by Mean value of that attribute using mean function that performs by using (2.1) equation.

4.3.1.2 Exploratory data analysis (EDA)

Exploratory data analysis (EDA) is a statistical method for summarizing data sets visually. EDA may or may not use a statistical model to go beyond formal modeling or hypothesis testing.

4.3.1.3 Data standardization results

An approach that uses z-score normalization to detect values that should be zero is explained here (3.1). The majority of machine learning methods depend on the training data distribution. Therefore, when the difference between the features means and variances are extreme, more training iterations will be needed. Hence, the machine learning algorithms will reach minimum error when the data is standardized, while it requires more training iterations when normalization is used. Therefore, standardization is considered better for the Gaussian distribution.

4.3.1.4 Splitting the dataset

Split the dataset 80% to train the classifier, and 20% to test classification system performance accuracy for each dataset.

4.4 CLASSIFICATION SYSTEM RESULTS

The outcomes of the suggested systems will be presented at this phase. It includes the results of the classification of lung cancer using the Support Vector Machines (SVM) and (LSVM) for svc data.

4.4.1 Supervised Vector Machine Result (SVM)

In this stage, the data of lung cancer are classified according to the SVM algorithm was previously explained in (3.4). Both the training and testing phases of the system have already been covered in detail.

After these two steps, lung cancer svc data can be detected and classified. A confusion matrix shows the model's accuracy.

4.4.1.1 Svm training

In the previous section (3.6.2.1), we talked about how to use RBF data to train nonlinear (SVM) models. During the training, phase used equations (2.12) and (2.13) to figure out the values for the hyper plane between the classes (2.13). The measurement of performance was based on what was said in section (2.6) of the second chapter and on equations (2.3), (2.4), (2.5), and (2.6).

4.4.1.2 Svm testing result

As was explained in the last chapter, unlabeled data from lung cancer is used to test how well the system can sort data into groups. At this point, the results were also given in the form of a "confusion matrix," which shows how well each class did in the testing stage for two different situations (model with applying normalization and with standardization). Figure shows the results in the form of a confusion matrix (4.1).

.

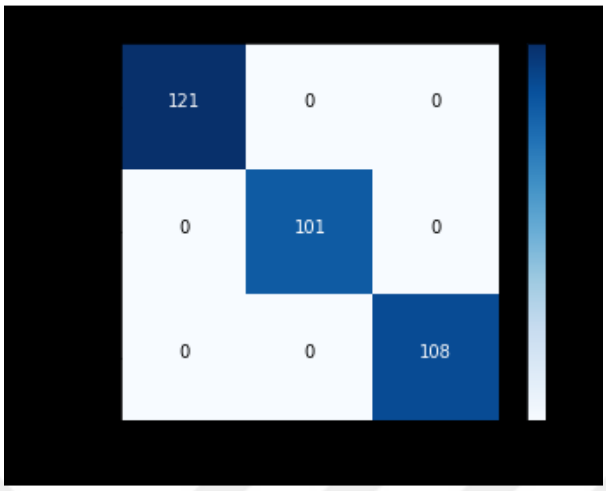


Figure 4.1: confusion matrix of (SVM) for CSV data

4.4.2 Results of Second Proposed Model (LSVM)

In this stage, lung cancer data are classified using the SVM algorithm (3.3). This operation was detailed in (3.6.2.1), where data is trained using a linear SVM kernel function. Phase computed the class hyper plane during training (2.13). (2.15).

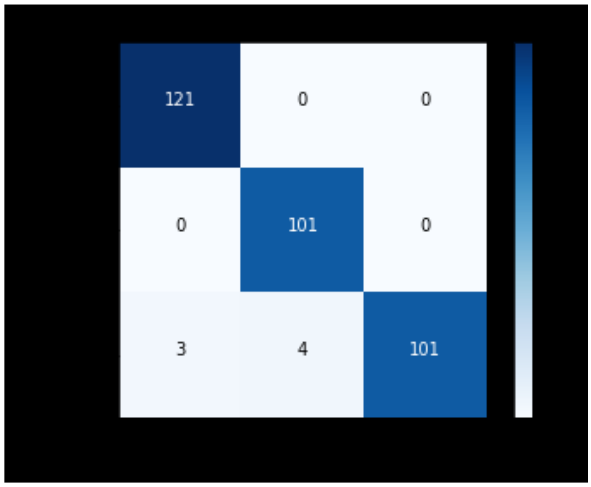


Figure 4.2: confusion matrix of (LSVM) for CSV dataset

4.5 COMPARSION BETWEEN THE PROPOSED (SVM, LSVM) TECHNIQUE

The comparison of the different methodologies is critical in highlighting the strengths and limitations of each one. Selecting a subject like lung cancer illness detection and classification, SVMs and LSVMs were applied. For the sake of comparison, we utilized the same dataset and divided the data in the same way for all algorithms, with 80% going to training and 20% going to testing, as shown in table (4.1).

Table 4.1: Comparison of accuracy between the proposed systems

Type	Accuracy	Time
SVM	**100.0%**	**0.013 second**
LSVM	**99.55%**	**0.051 second**

Case Two: The System Result for The Second Proposed Approach

It includes the results of the two phases (training and testing) of the classification for lung cancer and Covid-19 Diseases using Deep learning technique (CNN, DNN and VGG-16) .

4.6 CONVOLUTION NEURAL NETWORK RESULT

According to the CNN algorithm, CT scans of lung cancer are categorized at this stage, as previously discussed in section (3.3). When the network for training is set up it gives the weights randomly, the network trains these weights for several epochs until it becomes stable according to figures (3.5) and (3.6). The best training performance to lung cancer data is at epoch 100 as the figure (4.3) and (4.4),

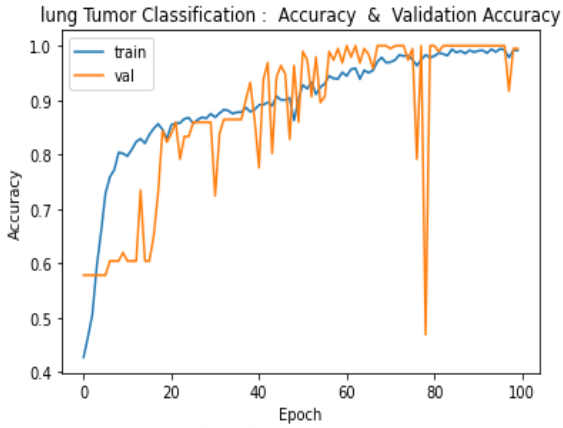


Figure 4. 3: CNN Accuracy for lung cancer

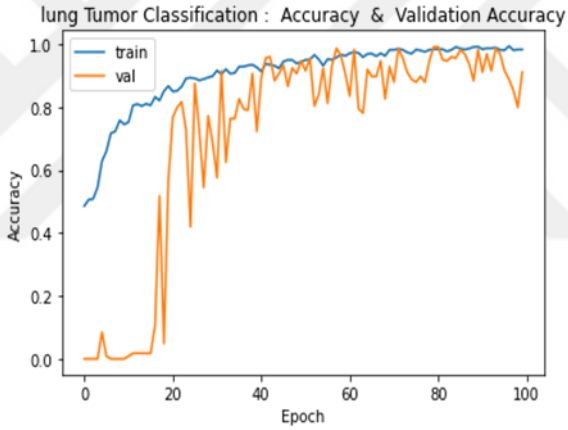


Figure 4.4: CNN Accuracy for Covid-19 dataset

The first layer of convolution consists of 3 channels 32 filters; the filter window is 3 x 3 and the step is one, which means that each 3x3 square is an input to the kernel. Such that the number of outputs in the layer corresponds to as many as inputs.

A maximum pooling layer is provided with the contribution from the first convolution layer, in which the 50x50 array has been separated into 2x2 strides two. The 50x50 array has now been an array 25x25, and 32 filters continue to exist. So, these processes repeat for layers (7 to 10).

Finally, in layer 11 the network is the flatten and takes inputs from the previous pooling layer 6×6×256= 9216, and this needs 256 nodes for the seventh denes layer, where 256 is equal to 29, while in the seventh denes needs 9 nodes for calculated classification by SoftMax function.

Accuracy, loss, validation accuracy, and validation loss represent the performance measurement of this model which explained previously in second chapter section (2.7.8), and the results of the training and testing process are presented in the tables, figures, and confusion matrices that show the results of five classes (Benign, Malignant, Normal) for lung cancer dataset and (covid, no Covid) for Covid-19 dataset.

80% of lung cancer and COVID19 datasets are used in training. All conceivable training and testing data rates have been attempted. Half of the training and testing data are gathered initially. Continue until 90% of training data and 10% of testing data are obtained. The largest percentage and best result were 80% for the training dataset and 20% for the testing dataset, notably for the testing procedure that relies on it for classifying and detecting lung cancer and COVID19 disorders, as shown in tables (4.2) and (4.3). Figure (4.5) A and B demonstrate how training periods were determined for categorization. This system's testing uses 20% of the lung cancer and COVID19 dataset. The model was evaluated on 1315 lung cancer pictures. Figures (4.6) and (4.7) illustrate the test data utilizing the confusion matrix to determine accuracy, responsiveness, specificity, of test results, as shown in figures (4.6) and (4.7).

According to equations (2.30), (2.31), (2.32), and (2.33) accuracy, Recall, $Precision$ and F1-score are computed for the CNN model and table (4.2) and (4.3) which shows The Results of CNN classification of the lung cancer disease and Covid-19 respectively.

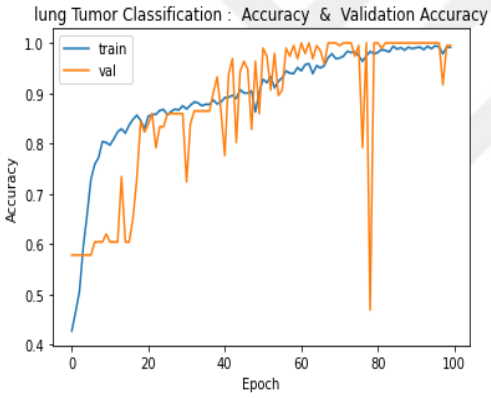
Table 4.2: Classification result of CNN lung cancer dataset

Classification	precision	recall	F1-score	support
covid	0.98	0.90	0.94	252
No covid	0.91	0.98	0.94	245
Accuracy			0.94	497
macro avg	0.94	0.94	0.94	497
weighted avg	0.94	0.94	0.94	497

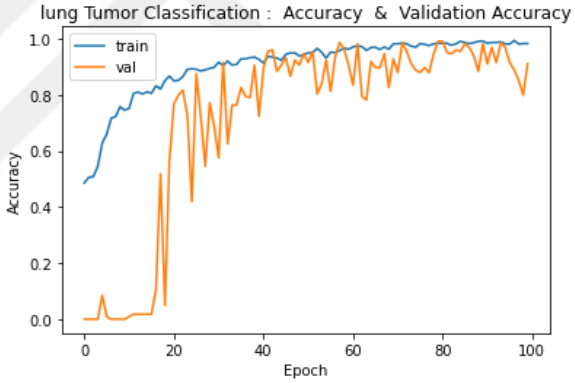
.

Table 4.3: Classification result of CNN covid-19 dataset

classification	precision	recall	F1-score	Support
l	0.96	1.00	0.98	26
M	1.00	1.00	1.00	111
H	1.00	0.99	0.99	83
Accuracy			1.00	220
Macro_avg	0.99	0.100	0.100	220



A-CNN accuracy for lung cancer dataset



B-CNN accuracy for Covid-19 dataset

Figure 4.5: Epochs versus accuracy for both datasets

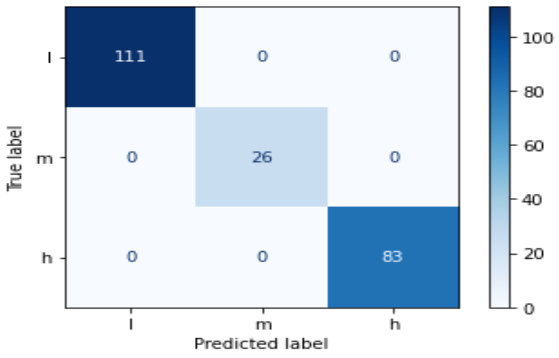


Figure 4.6: Confusion matrix for the proposed (CNN) Lung cancer dataset

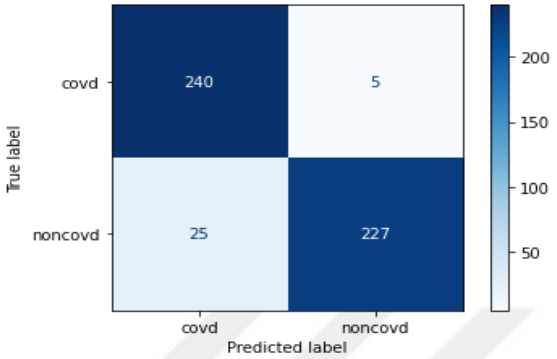


Figure 4.7: Confusion matrix for the proposed (CNN) Covid-19 datase

4.7 DEEP NEURAL NETWORK RESULT

In this step, lung cancer pictures and Covid-19 datasets are classified using the algorithm from chapter 3. (3.2).

According to equations (2.30), (2.31), (2.32), and (2.33), accuracy, Recall, Precision, and F1-score are obtained for the CNN model and tables (4.4) A and B, which illustrate CNN categorization of lung cancer illness and Covid-19.

It also features many parameters and hyper-parameters that may be adjusted to improve recognition (4.6).

The system requires training and testing. These two phases identify and categorize lung cancer and covid-19. The confusion matrix shows DNN accuracy.

In the training phase, 80% was used for training; this data results from the application of the DNN algorithm. When the network for training is set up it gives the weights randomly, the network trains these weights for several epochs until it becomes stable according to figure (3.4). The best training performance to both (lung cancer and covid19) data is at epoch 50. In this system, 20% remaining of the dataset is used as in figure (4.8) A and B.

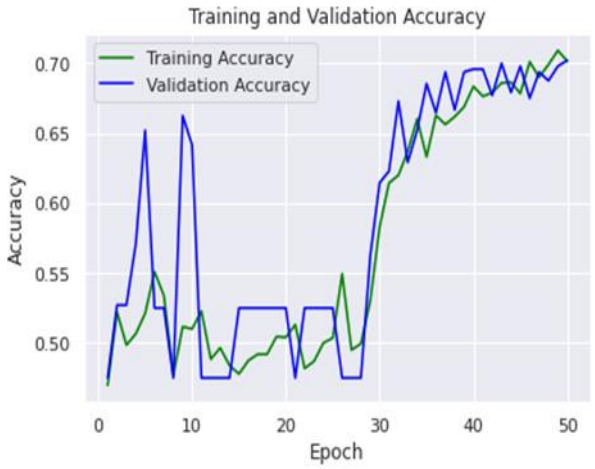
for the testing phase. Testing the system using unlabeled data is the first step in this phase of the experimentation process. At this stage, table (4.4) A and B shows the testing results of lung cancer and covid-19 Dataset by using DNN algorithm, as a consequence of this, a confusion

matrix is created to display the accuracy of each lung cancer test result as in figure (4.9) A and B.

Table 4.4: A: Classification report of DNN lung cancer B: Classification report of DNN covid19

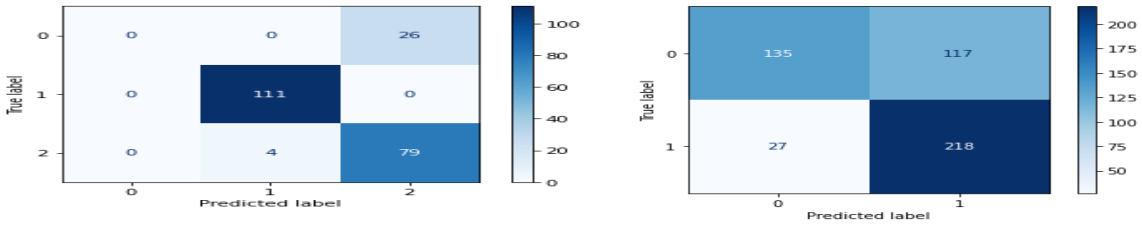
Classification	Precision	recall	F1-score	Support
0	0.00	0.00	0.00	26
1	0.97	1.00	0.98	111
2	0.75	0.95	0.84	83
Accuracy			0.86	220
Macro avg	0.57	0.65	0.61	220
Weighted-av	0.77	0.86	0.81	220

Classification	Precision	recall	F1-score	support
0	0.83	0.54	0.65	252
1	0.65	0.89	0.75	245
Accuracy			0.71	497
Macro avg	0.74	0.71	0.70	479
Weighted_avg	0.74	0.71	0.70	479



A-DNN accuracy for lung cancer dataset **B-DNN accuracy for Covid-19 dataset**

Figure 4.8: Epochs versus accuracy for both datasets



A: DNN confusion matrix for Covid-19dataset B: DNN confusion matrix for Covid-19 dataset

Figure 4.9: Confusion matrix of DNN for both datasets

Table 4.5: The parameters of the DNNs layers for Lung cancer dataset

Layer(type)	**Output Shape**	**Param #**
img_input (Input Layer)	[(None, 50, 50, 3)]	0
fc_1 (Flatten)	(None, 7500)	0
Layer_2(Dense)	(None, 2048)	15362048
lyaer_3 (Dense)	(None, 1024)	2098176
Dropout_1 (Dropout)	(None, 1024)	0
lyaer_4 (Dense)	(None, 512)	524800
Dropout_2 (Dropout)	(None, 512)	0
lyaer_5 (Dense)	(None, 256)	131328
dropout_3 (Dropout)	(None, 256)	0
lyaer_6 (Dense)	(None, 128)	32896
dropout_4 (Dropout)	(None, 128)	0
lyaer_7 (Dense)	(None, 64)	8256
dropout_5 (Dropout)	(None, 64)	0
lyaer_8 (Dense)	(None, 32)	2080
dropout_6 (Dropout)	(None, 32)	0
predictions (Dense)	(None, 2)	66
Total params: 18,159,683 Trainable params: 18,159,683 Non-trainable params:0		

Table 4.6: The parameters of the DNNs layers for covid19 data set

Layer(type)	**Output Shape**	**Param #**
img_input (Input Layer)	[(None, 50, 50, 3)]	0
fc_1 (Flatten)	(None, 7500)	0
lyaer_3 (Dense)	(None, 1024)	7681024
Dropout_1 (Dropout)	(None, 1024)	0
lyaer_4 (Dense)	(None, 512)	524800
Dropout_2 (Dropout)	(None, 512)	0
lyaer_5 (Dense)	(None, 256)	131328
dropout_3 (Dropout)	(None, 256)	0
lyaer_6 (Dense)	(None, 128)	32896
dropout_4 (Dropout)	(None, 128)	0
lyaer_7 (Dense)	(None, 64)	8256
dropout_5 (Dropout)	(None, 64)	0
lyaer_8 (Dense)	(None, 32)	2080
dropout_6 (Dropout)	(None, 32)	0
predictions (Dense)	(None, 2)	66
Total params: 8,380,450 Trainable params: 8,380,450 Non-trainable params:0		

4.8 VGG RESULT:

In all indicators, VGG-16 outperforms traditional image processing methods. It improves the accuracy of segmenting lung nodules and outperforms traditional image processing methods in a variety of performance indicators, allowing experts to effectively diagnose lung nodules.

A VGG-16 pre-trained model for diagnosing chest disorders on the basis of CT scans makes up a deep model widely recognized as one of the greatest vision model architectures ever built. The specifics of the proposed model comprise images, feature extraction, and classification layers. The input layer of the model receives (224x224x3) CT scans. In the first block of CNN, convolution and ReLU are both included. Second block has two convolution layers, two ReLU

layers, batch normalizing, maximum pooling, and dropout layers. The classification component has 512 neurons before the dropout layer. The final output is generated by a dense layer with two neurons and the SoftMax activation function, which classifies the output picture into one of the four groups of chest ailments.: (Covid-19, NOCOVID) for Covid dataset, (Benign, Malignant ,Normal) for lung cancer dataset which are split into two groups; The overall parameters are as follows: (35,475,139),trainable (35,475,011) with (128) non-trainable parameters for VGG Lung cancer dataset and (35,474,498) trainable parameters, (128) non-trainable parameters for COVID dataset.

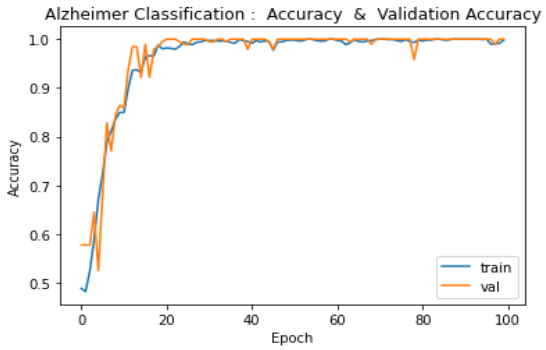
Training is essential to attain the best value for trainable parameters, which change over time, non-trainable parameters, on the other hand, do not change over time. Due to the inability to change or optimize non-trainable model parameters, they must be established or given as inputs before training can begin. As a result, the non-trainable will have no impact on categorization. Table (4.7) A and B shows the results of VGG-16 for both data sets used, figure (4.10) A and B shows the accuracy between the training and validation phases, with the number of epochs and figure (4.11) A and B shows the confusion matrix of VGG-16 for both datasets used.

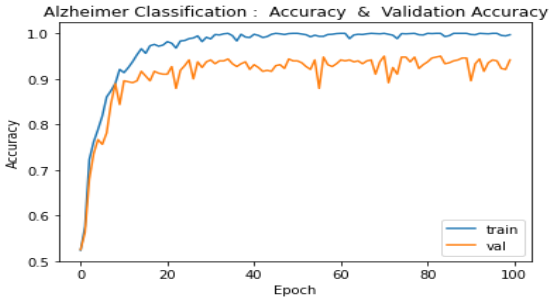
Table 4.7: A: Results of VGG Lung cancer dataset

Classification	precision	Recall	F1-score	Support
Benign	100	100	100	26
Malignant	100	100	100	111
Normal	100	100	100	83
Accuracy		100	100	220
Macro avg	100	100	100	220
weighted avg	100	100	100	220

B: Results of (VGG-16) for Covid-19 dataset

Classification	Precision	Recall	F1-score	support
Covid	0.94	0.98	0.96	252
Noncovid	0.98	0.94	0.96	245
Accuracy			0.96	497
Macro avg	0.96	0.96	0.96	497
weighted avg	0.96	0.96	0.96	497

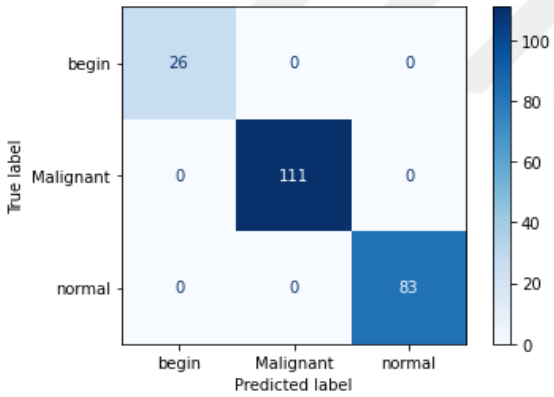


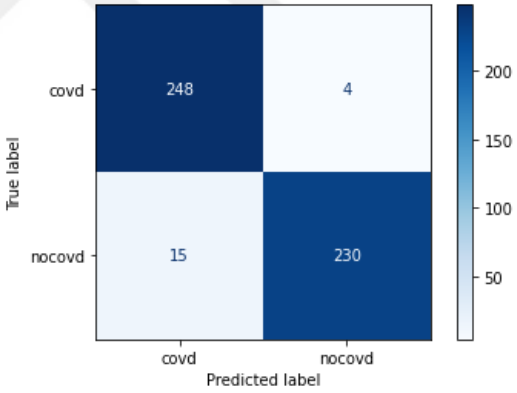


A-VGG-16 accuracy for lung cancer dataset B-VGG-16 accuracy for Covid-19 dataset

Figure 4.10: Accuracy between the training and validation phases, with the number of epochs for both datasets used

In both training and validation, the VGG16 design improves accuracy significantly. Accordingly, the VGG16 model performed well





A- (VGG-16) for lung cancer dataset B: (VGG-16) for covid-19 dataset

Figure 4.11: confusion matrix of the proposed VGG-16 model

4.9 RESULTS OF ALL MODELS

The number of accurate and wrong classification model predictions was also counted in the confusion matrix. The findings reveal that for the lung cancer dataset, the deep learning CNN and VGG models provide greater accuracies (100 percent),(94 percent)for COVID dataset and (100 percent) for both lung cancer and Covid-19 respectively, when compared to the accuracies of DNN model overfitting occurred when we increase epochs to (100) we notice that accuracy

became (50 percent) and less but when we decrease epochs to 50 then accuracy became (86 percent) for lung cancer dataset CT scan images and (75 percent) for Covid dataset CT scan images as specified in Table (4.8) . The epoch specifies how many times the data has been cycled. Loss is defined as the error over the training set, which is commonly expressed in terms of regression or classification.

Table 4.8: Results of all architectures

Models	Accuracy	Sensitivity
(CNN) Lung cancer	100%	1.00
(CNN) Covid	93%	0.94
(VGG)Lung cancer	100%	1.00
(VGG) Covid	100%	1.00
(DNN) Lung cancer	86%	86
(DNN) Covid	75%	75

4.10 PROPOSED ALGORITHM VS. RELATED WORKS

The proposed system for Ct scan images is implemented by using classification algorithms (CNN, DNN, VGG_16) and compared with related works. in Table (4.9) compares the planned and current systems.

Table 4.9: Comparison between other Existing works and the proposed work.

Work	Medical image	Accuracy (%)
Nascimento and al. [102]	**LIDC (73)**	**92.78%**
Krewer and al. [74]	**LIDC-IDRI (33)**	**90.91%**
Dandil et al. [103]	**Private (128)**	**90.63%**
Kuruvilla and Gunavathi, 2014 [104]	**LIDC (110)**	**93.30%**
Gupta and Tiwari [105]	**Private (120)**	**90%**
Kumar and al. [106]	**LIDC (4323)**	**75.01%**
da Silva [107]	**LIDC-IDRI (8296)**	**82.3%**
CNN[108]	**LIDC-IDRI (5024)**	**84.15%**
DNN [108]	**LIDC-IDRI (5024)**	**82.37%**
SAE [108]	**LIDC-IDRI (5024)**	**82.59%**

5.CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

Several studies have been established in the field of Lung cancer and COVID-19 detection and classification and find the best solutions to overcome these types of diseases. This work was conducted with the aim of investigating the different approaches and methods of detecting lung cancer and COVID19 diseases. However, the proposed model distinguishes itself from the previous methods by employing best machine learning algorithms and deep learning techniques to classify lung cancer and COVID-19 in suspected patients.

The simulation runs two different datasets to test and evaluate the performance of the proposed model. The models classify COVID-19 (COVID, NOCOVID) as well as lung cancer (Benign, Malignant and Normal) A test dataset is used to test images and their accuracy, and the expected accuracy for each model is given. Performance measures like precision, recall, and F1-score are used to judge how well the model works.

For the lung cancer dataset, CNN and VGG16 have the highest accuracy at 100%, while DNN has the lowest accuracy at 85%. However, the proposed method has a flaw in that it does not detect COVID-affected lung regions. A larger dataset, as well as new preprocessing approaches and pre-trained models in transfer learning, must be reviewed to determine which parts of the lungs are damaged. This will increase the accuracy of the recommended models.

5.2 FUTURE WORKS AND RECOMMENDATIONS

In the future, this work is hoped to be continued in the direction of detection and prevention to various diseases that affect the proper function of human lung. Consider a large dataset and employ preprocessing techniques and pre-trained models in transfer learning to increase the suggested models' accuracy.

With a temperature precision of around 0.6 degrees Fahrenheit, the vision-based infectious disease thermal tracking application can scan up to 30 persons at a distance of 3 meters, or about 10 feet. (0.3° C). Installed at major entrances, checkpoints, or other places with a lot of people, it can do mass screenings and check the temperature of people who might be

infected. The most important thing about this app is that it uses thermal camera sensors and an algorithm based on artificial intelligence (AI) to analyze temperature.

.

REFERENCES

[1] Z. Ma, H. Zhong, L. Xie, Q. Xia, and C. Kang, "Month ahead average daily electricity price profile forecasting based on a hybrid nonlinear regression and SVM model: an ERCOT case study," *Journal of Modern Power Systems and Clean Energy*, vol. 6, no. 2, pp. 281–291, 2018, doi: 10.1007/s40565-018-0395-3.

[2] C. T. Lin, C. M. Yeh, S. F. Liang, J. F. Chung, and N. Kumar, "Support-vector-based fuzzy neural network for pattern classification," *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 1, pp. 31–41, 2006, doi: 10.1109/TFUZZ.2005.861604.

[3] J. Cui, M. Shan, R. Yan, and Y. Wu, "Aero-engine fault diagnosis using improved local discriminant bases and support vector machine," *Math Probl Eng*, vol. 2014, 2014, doi: 10.1155/2014/283718.

[4] L. B. Nascimento, A. C. de Paiva, and A. C. Silva, "Lung nodules classification in CT images using Shannon and Simpson diversity indices and SVM," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 2012, pp. 454–466.

[5] E. Dandıl, M. Çakiroğlu, Z. Ekşi, M. Özkan, Ö. K. Kurt, and A. Canan, "Artificial neural network-based classification system for lung nodules on computed tomography scans," in *2014 6th International conference of soft computing and pattern recognition (SoCPaR)*, 2014, pp. 382–386.

[6] J. Kuruvilla and K. Gunavathi, "Lung cancer classification using neural networks for CT images," *Comput Methods Programs Biomed*, vol. 113, no. 1, pp. 202–209, 2014.

[7] B. Gupta and S. Tiwari, "Brain tumor detection using curvelet transform and support vector machine," *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 4, pp. 1259–1264, 2014.

[8] W. Shen, M. Zhou, F. Yang, C. Yang, and J. Tian, "Multi-scale convolutional neural networks for lung nodule classification," in *International conference on information processing in medical imaging*, 2015, pp. 588–599.

[9] G. da Silva, A. Silva, A. de Paiva, and M. Gattass, "Classification of malignancy of lung nodules in CT images using convolutional neural network," in *Anais do XVI Workshop de Informática Médica*, 2016, pp. 2481–2489.

[10] Q. Song, L. Zhao, X. Luo, and X. Dou, "Using deep learning for classification of lung nodules on computed tomography images," *J Healthc Eng*, vol. 2017, 2017.

[11] D. Nagajyothi, R. Addagudi, T. Gunda, and S. santhoshi Logitla, "Detection of lung cancer using SVM classifier," *International Journal*, vol. 8, no. 5, 2020.

[12] P. Katiyar and K. Singh, "A Comparative study of Lung Cancer Detection and Classification approaches in CT images," in *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2020, pp. 135–142.

[13] P.-P. Ypsilantis and G. Montana, "Recurrent convolutional networks for pulmonary nodule detection in CT imaging," *arXiv preprint arXiv:1609.09143*, 2016.

[14] S. R. Jena and S. T. George, "Morphological feature extraction and KNG- CNN classification of CT images for early lung cancer detection," *Int J Imaging Syst Technol*, vol. 30, no. 4, pp. 1324–1336, 2020.

[15] M. G. Nakrani, G. S. Sable, and U. B. Shinde, "A Review: Automatic and Semi-Automatic Detection of Pulmonary Lung Nodules in Computed Tomography Images".

[16] L. M. Pehrson, M. B. Nielsen, and C. Ammitzbøl Lauridsen, "Automatic pulmonary nodule detection applying deep learning or machine learning algorithms to the LIDC-IDRI database: a systematic review," *Diagnostics*, vol. 9, no. 1, p. 29, 2019.

[17] D. Moitra and R. K. Mandal, "Classification of non-small cell lung cancer using one-dimensional convolutional neural network," *Expert Syst Appl*, vol. 159, p. 113564, 2020.

[18] C. Lu, Z. Zhu, and X. Gu, “An intelligent system for lung cancer diagnosis using a new genetic algorithm based feature selection method,” *J Med Syst*, vol. 38, no. 9, pp. 1–9, 2014.

[19] M. Ajin and L. Mredhula, “Diagnosis of interstitial lung disease by pattern classification,” *Procedia Comput Sci*, vol. 115, pp. 195–208, 2017.

[20] S. Qiu *et al.*, “Highly selective colorimetric bacteria sensing based on protein-capped nanoparticles,” *Analyst*, vol. 140, no. 4, pp. 1149–1154, 2015.

[21] S. Singh, Y. Singh, and R. Vijay, “An evaluation of features extraction from lung CT images for the classification stage of malignancy,” *IOSR J. Comput. Eng.(IOSR-JCE)*, pp. 2278–8727, 2015.

[22] S. Mukherjee and N. Sharma, “Intrusion detection using naive Bayes classifier with feature reduction,” *Procedia Technology*, vol. 4, pp. 119–128, 2012.

[23] L. al Shalabi and Z. Shaaban, “Normalization as a preprocessing engine for data mining and the approach of preference matrix,” in *2006 International conference on dependability of computer systems*, 2006, pp. 207–214.

[24] C. M. Salgado, C. Azevedo, H. Proença, and S. M. Vieira, “Noise versus outliers,” *Secondary analysis of electronic health records*, pp. 163–183, 2016.

[25] D. Morgan and G. Ceder, “Data mining in materials development,” in *Handbook of Materials Modeling*, Springer, 2005, pp. 395–421.

[26] J.-P. O. Li *et al.*, “Digital technology, tele-medicine and artificial intelligence in ophthalmology: A global perspective,” *Prog Retin Eye Res*, vol. 82, p. 100900, 2021.

[27] R. R. R. Robinson and C. Thomas, “Ranking of machine learning algorithms based on the performance in classifying DDoS attacks,” in *2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, 2015, pp. 185–190.

[28] P. A. R. Kumar and S. Selvakumar, “Distributed denial of service attack detection using an ensemble of neural classifier,” *Comput Commun*, vol. 34, no. 11, pp. 1328–1341, 2011.

[29] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[30] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, “Machine learning: a review of classification and combining techniques,” *Artif Intell Rev*, vol. 26, no. 3, pp. 159–190, 2006.

[31] T. O. Ayodele, “Types of machine learning algorithms in New Advances in Machine Learning. Croatia, Rijeka.” InTech, 2010.

[32] Y. Sakumura *et al.*, “Diagnosis by volatile organic compounds in exhaled breath from lung cancer patients using support vector machine algorithm,” *Sensors*, vol. 17, no. 2, p. 287, 2017.

[33] N. J. Nilsson, “Introduction to machine learning. An early draft of a proposed textbook,” 1996.

[34] Z. Wu and L. Chen, “Similarity-based method with multiple-feature sampling for predicting drug side effects,” *Comput Math Methods Med*, vol. 2022, 2022.

[35] E. Osuna, R. Freund, and F. Girosi, “An improved training algorithm for support vector machines,” in *Neural networks for signal processing VII. Proceedings of the 1997 IEEE signal processing society workshop*, 1997, pp. 276–285.

[36] M. Mrówczyńska, “The SVM method as an instrument for the classification of vertical displacements,” *Reports on Geodesy and Geoinformatics*, vol. 98, 2015.

[37] A. Zisserman, “Lecture 2: The svm classifier,” *C19 Machine Learning (Hilary Term 2015). Available online at: http://www. robots. ox. ac. uk/~ az/lectures/ml/lect2. pdf (accessed March 23, 2019)*, 2015.

[38] X. Lin, L. Gao, Z. Tang, X. Hu, and X. Lin, "Identification of embedded mathematical formulas in PDF documents using SVM," in *Document recognition and retrieval xix*, 2012, vol. 8297, pp. 93–100.

[39] T. Rao, "Crop Coverage Data Classification using Support Vector Machine," *Global Journal of Computer Science and Technology*, 2016.

[40] L. D. van Khoa and Z. Chen, "Incremental learning with online SVMs on LiDAR sensory data," *arXiv preprint arXiv:2101.01667*, 2020.

[41] I. Sharif and D. Chaudhuri, "A multiseed-based SVM classification technique for training sample reduction," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 27, no. 1, pp. 595–604, 2019.

[42] M. Kejriwal and D. P. Miranker, "Semi-supervised instance matching using boosted classifiers," in *European semantic web conference*, 2015, pp. 388–402.

[43] T. Herawan, R. Ghazali, and M. M. Deris, "Recent Advances on Soft Computing and Data Mining: Proceedings of The First International Conference on Soft Computing and Data Mining (SCDM-2014) Universiti Tun Hussein Onn Malaysia, Johor, MalaysiaJune 16th-18th, 2014," 2014.

[44] T. N. F. Novo, "MisInfoBot: fight misinformation about COVID on social media," 2021.

[45] K. S. Durgesh and B. Lekha, "Data classification using support vector machine," *J Theor Appl Inf Technol*, vol. 12, no. 1, pp. 1–7, 2010.

[46] J. P. Lewis, "A short SVM (support vector machine) tutorial," *CGIT Lab/IMSC, University Southern California*, 2004.

[47] M. Awad and R. Khanna, "Machine learning and knowledge discovery," in *Efficient learning machines*, Springer, 2015, pp. 19–38.

[48] Z. Ma, H. Zhong, L. Xie, Q. Xia, and C. Kang, "Month ahead average daily electricity price profile forecasting based on a hybrid nonlinear regression and SVM model: an ERCOT case study," *Journal of Modern Power Systems and Clean Energy*, vol. 6, no. 2, pp. 281–291, 2018.

[49] M.-Y. Chen and D. A. Linkens, "Rule-base self-generation and simplification for data-driven fuzzy models," *Fuzzy Sets Syst*, vol. 142, no. 2, pp. 243–265, 2004.

[50] M. Oujaoura, B. Minaoui, M. Fakir, R. el Ayachi, and O. Bencharef, "Recognition of isolated printed tifinagh characters," *Int J Comput Appl*, vol. 85, no. 1, 2014.

[51] P. S. Kohli and S. Arora, "Application of machine learning in disease prediction," in *2018 4th International conference on computing communication and automation (ICCCA)*, 2018, pp. 1–4.

[52] R. Burbidge and B. Buxton, "An introduction to support vector machines for data mining," *Keynote papers, young OR12*, pp. 3–15, 2001.

[53] R. Huerta, F. Corbacho, and C. Elkan, "Nonlinear support vector machines can systematically identify stocks with high and low future returns," *Algorithmic Finance*, vol. 2, no. 1, pp. 45–58, 2013.

[54] V. Jain and J. L. Crowley, "Head pose estimation using multi-scale gaussian derivatives," in *Scandinavian Conference on Image Analysis*, 2013, pp. 319–328.

[55] X.-Y. Wang, H.-Y. Yang, Y.-W. Li, W.-Y. Li, and J.-W. Chen, "A new SVM-based active feedback scheme for image retrieval," *Eng Appl Artif Intell*, vol. 37, pp. 43–53, 2015.

[56] M. Sabzekar, M. GhasemiGol, M. Naghibzadeh, and H. S. Yazdi, "Improved DAG SVM: A New Method for Multi-Class SVM Classification.," in *IC-AI*, 2009, pp. 548–553.

[57] J. Cui, M. Shan, R. Yan, and Y. Wu, "Aero-engine fault diagnosis using improved local discriminant bases and support vector machine," *Math Probl Eng*, vol. 2014, 2014.

[58] S. B. Belhaouari and A. Islam, “Deep learning in healthcare,” in *Multiple Perspectives on Artificial Intelligence in Healthcare*, Springer, 2021, pp. 155–168.

[59] M. I. Razzak, S. Naz, and A. Zaib, “Deep learning for medical image processing: Overview, challenges and the future,” *Classification in BioApps*, pp. 323–350, 2018.

[60] Y. Guo and Y. Guo, “Deep learning for visual understanding,” 2017.

[61] R. Jafari-Marandi, S. Davarzani, M. S. Gharibdousti, and B. K. Smith, “An optimum ANN-based breast cancer diagnosis: Bridging gaps between ANN learning and decision-making goals,” *Appl Soft Comput*, vol. 72, pp. 108–120, 2018.

[62] H. H. Aghdam, E. J. Heravi, and D. Puig, “A practical approach for detection and classification of traffic signs using convolutional neural networks,” *Rob Auton Syst*, vol. 84, pp. 97–112, 2016.

[63] M. I. Razzak, S. Naz, and A. Zaib, “Deep learning for medical image processing: Overview, challenges and the future,” *Classification in BioApps*, pp. 323–350, 2018.

[64] K. Han, D. Yu, and I. Tashev, “Speech emotion recognition using deep neural network and extreme learning machine,” 2014.

[65] M. Wąsowicz *et al.*, “Computed aided system for separation and classification of the abnormal erythrocytes in human blood,” in *Biophotonics—Riga 2017*, 2017, vol. 10592, pp. 49–55.

[66] S. Zhang, Q. Cheng, D. Chen, and H. Zhang, “Image target recognition model of multi-channel structure convolutional neural network training automatic encoder,” *IEEE Access*, vol. 8, pp. 113090–113103, 2020.

[67] S. Albawi, O. Bayat, S. Al-Azawi, and O. N. Ucan, “Social touch gesture recognition using convolutional neural network,” *Comput Intell Neurosci*, vol. 2018, 2018.

[68] M. Liu, S. Li, S. Shan, and X. Chen, “Au-inspired deep networks for facial expression feature learning,” *Neurocomputing*, vol. 159, pp. 126–136, 2015.

[69] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[70] L. Li, L. G. Wang, F. L. Teixeira, C. Liu, A. Nehorai, and T. J. Cui, "DeepNIS: Deep neural network for nonlinear electromagnetic inverse scattering," *IEEE Trans Antennas Propag*, vol. 67, no. 3, pp. 1819–1825, 2018.

[71] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," *IEEE Trans Industr Inform*, vol. 14, no. 4, pp. 1606–1615, 2017.

[72] G. Papandreou, I. Kokkinos, and P.-A. Savalle, "Untangling local and global deformations in deep convolutional networks for image classification and sliding window detection," *arXiv preprint arXiv:1412.0296*, 2014.

[73] S. Ganapathy and V. Peddinti, "3-D CNN models for far-field multi-channel speech recognition," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2018, pp. 5499–5503.

[74] A. Agrawal and N. Mittal, "Using CNN for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy," *Vis Comput*, vol. 36, no. 2, pp. 405–412, 2020.

[75] T. Steinmetzer, I. Bönninger, M. Reckhardt, F. Reinhardt, D. Erk, and C. M. Travieso, "Comparison of algorithms and classifiers for stride detection using wearables," *Neural Comput Appl*, vol. 32, no. 24, pp. 17857–17868, 2020.

[76] Z. Wei, H. Hu, H. Zhou, and A. Lau, "Characterizing rock facies using machine learning algorithm based on a convolutional neural network and data padding strategy," *Pure Appl Geophys*, vol. 176, no. 8, pp. 3593–3605, 2019.

[77] L. Zaniolo and O. Marques, "On the use of variable stride in convolutional neural networks," *Multimed Tools Appl*, vol. 79, no. 19, pp. 13581–13598, 2020.

[78] D. Xie, L. Zhang, and L. Bai, “Deep learning in visual computing and signal processing,” *Applied Computational Intelligence and Soft Computing*, vol. 2017, 2017.

[79] A. Narin, C. Kaya, and Z. Pamuk, “Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks,” *Pattern Analysis and Applications*, vol. 24, no. 3, pp. 1207–1220, 2021.

[80] T. Murata, K. Fukami, and K. Fukagata, “Nonlinear mode decomposition with convolutional neural networks for fluid dynamics,” *J Fluid Mech*, vol. 882, 2020.

[81] S. Albawi, T. A. M. Mohammed, and S. Alzawi, “Layers of a convolutional neural network,” *Ieee*, 2017.

[82] H. Su and C. Jung, “Perceptual enhancement of low light images based on two-step noise suppression,” *IEEE Access*, vol. 6, pp. 7005–7018, 2018.

[83] M. L. Muammer and O. I. Alsaif, “Employing X-ray images Characteristics For Early Corona Virus Detection Based on Deep Learning Techniques,” *NeuroQuantology*, vol. 20, no. 8, pp. 5664–5673, 2022.

[84] O. Abdel-Hamid, L. Deng, and D. Yu, “Exploring convolutional neural network structures and optimization techniques for speech recognition.,” in *Interspeech*, 2013, vol. 2013, pp. 1173–1175.

[85] J. X. Chen, P. W. Zhang, Z. J. Mao, Y. F. Huang, D. M. Jiang, and Y. N. Zhang, “Accurate EEG-based emotion recognition on combined features using deep convolutional neural networks,” *IEEE Access*, vol. 7, pp. 44317–44328, 2019.

[86] M. Hashemi, “Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation,” *J Big Data*, vol. 6, no. 1, pp. 1–13, 2019.

[87] A. G. Schwing and R. Urtasun, “Fully connected deep structured networks,” *arXiv preprint arXiv:1503.02351*, 2015.

[88] W. M. D. W. Zaki, M. M. Daud, A. H. Saad, A. Hussain, and H. A. Mutalib, "A Mobile Solution for Lateral Segment Photographed Images based Deep Keratoconus Screening Method," *International Journal of Integrated Engineering*, vol. 13, no. 5, pp. 18–27, 2021.

[89] R. Mazzon and A. Cavallaro, "Multi-camera tracking using a multi-goal social force model," *Neurocomputing*, vol. 100, pp. 41–50, 2013.

[90] Y. Guo and Y. Guo, "Deep learning for visual understanding," 2017.

[91] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448–456.

[92] S. Joseph, "Australian Literary Journalism and 'Missing Voices' How Helen Garner finally resolves this recurring ethical tension," *Journalism Practice*, vol. 10, no. 6, pp. 730–743, 2016.

[93] G. Ciaburro, "Sound event detection in underground parking garage using convolutional neural network," *Big Data and Cognitive Computing*, vol. 4, no. 3, p. 20, 2020.

[94] F. B. M. Suah, "Preparation and characterization of a novel Co (II) optode based on polymer inclusion membrane," *Anal Chem Res*, vol. 12, pp. 40–46, 2017.

[95] D. Swain, S. Pani, and D. Swain, "Diagnosis of coronary artery disease using 1-D convolutional neural network," *Int. J. Recent Technol. Eng.(IJRTE)*, vol. 8, no. 2, 2019.

[96] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," *Adv Neural Inf Process Syst*, vol. 31, 2018.

[97] J. Feng, X. He, Q. Teng, C. Ren, H. Chen, and Y. Li, "Reconstruction of porous media from extremely limited information using conditional generative adversarial networks," *Phys Rev E*, vol. 100, no. 3, p. 033308, 2019.

[98] D. P. Kingma and P. Dhariwal, “Glow: Generative flow with invertible 1x1 convolutions,” *Adv Neural Inf Process Syst*, vol. 31, 2018.

[99] A. Klein, “klein.” Google Patents, 1909.

[100] Y. Cao, J. He, Y. Zhang, G. Lu, S. Liu, and X. Wu, “DnM3Net: Multi-Scale & Multi-Level Shuffle-CNN Via Multi-Level Attention for Image Denoising,” in *2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS)*, 2019, pp. 286–292.

[101] L. M. Azizah, S. F. Umayah, S. Riyadi, C. Damarjati, and N. A. Utama, “Deep learning implementation using convolutional neural network in mangosteen surface defect detection,” in *2017 7th IEEE international conference on control system, computing and engineering (ICCSCE)*, 2017, pp. 242–246.

[102] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation functions: Comparison of trends in practice and research for deep learning,” *arXiv preprint arXiv:1811.03378*, 2018.

[103] Y. Li, L. Liu, C. Shen, and A. van den Hengel, “Mining mid-level visual patterns with deep CNN activations,” *Int J Comput Vis*, vol. 121, no. 3, pp. 344–364, 2017.

[104] H. A. Khan, W. Jue, M. Mushtaq, and M. U. Mushtaq, “Brain tumor classification in MRI image using convolutional neural network,” *Math. Biosci. Eng*, vol. 17, no. 5, pp. 6203–6216, 2020.

[105] S. Oviatt, B. Schuller, P. R. Cohen, D. Sonntag, G. Potamianos, and A. Krüger, *The Handbook of Multimodal-multisensor Interfaces: Language Processing, Software, Commercialization, and Emerging Directions*. Association for Computing Machinery and Morgan & Claypool, 2019.

[106] H. Alaeddine and M. Jihene, “Deep network in network,” *Neural Comput Appl*, vol. 33, no. 5, pp. 1453–1465, 2021.

[107] E. Soares, P. Angelov, S. Biaso, M. H. Froes, and D. K. Abe, “SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification,” *MedRxiv*, 2020.