Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Software Department

Texture and Shape Content Based Image Retrieval Using Deep Neural Network

A Dissertation

Submitted to the Council of the College of Information Technology for Postgraduate Studies of the University of Babylon in Partial Fulfilment of the Requirements for the Degree of Doctor of Philosophy in Information Technology / Software

By

Elaf Ali Abbood Hassan

Supervised by

Prof. Dr. Tawfiq Abdulkhaleq Abbas Abdulreda

2022 A.C. **1444 A.H.**

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

نَرْفَعُ دَرَجَٰتٍ مَّن نَّشَآءُ ۗ
وَفَوْقَ كُلِّ ذِى عِلْمٍ عَلِيمٌ ﴿٧٦﴾

صَدَقَ اللَّهُ الْعَلِيُّ الْعَظِيمُ

سورة يوسف

Dedication

To the pure soul of my father

To my lover's mother

To my family

Dear husband and my darling children

My sister and brothers

Acknowledgment

In the name of Allah, Praise and thanks to God Almighty, without God's care and help, I would not have been able to complete this research.

I would like to express my thanks and gratitude to my supervisor Prof. Dr. Tawfiq A. Al-Assadi for his valuable instructions, scientific, intellectual, and technical guidance to support and evaluate the work and follow up accurately throughout the period of preparation and completion this dissertation.

My sincere appreciation and gratitude to my beloved mother for her encouragement, help and moral support. I also dedicate my thanks and gratitude to my family: my husband, my children, my sister, and my brothers for their endurance, patience and encouragement for me during the period of work.

Also, I would like to thank all my colleagues and to whole staff of the College of Information technology/ University of Babylon for their supportive efforts and knowledge of technical guidance of during courses of this study.

Abstract

Recently, using medical information in hospitals has become an indispensable necessity and increasing rapidly with presenting a huge number of diagnostic imaging data. For this reason, information systems need to be developed to obtain, collect, manage, and process this information in arranged datasets. Content Based Image Retrieval (CBIR) is one of the important systems designed to assist in managing the process of retrieving similar images to the query.

This dissertation proposes a CBIR system that depends on various extracted features graded from low-level features including texture, shape, and topological relations to the high-level features extracted using deep learning methods. The proposed system includes three stages. The first stage is texture feature extraction which consists of many steps. The first step is proposing a non-uniform quantization method to convert a gray image into three versions of quantized images with 16, 32, and 64 gray levels. This method depends on optimal selection for threshold values considering PSNR optimizing between the quantized image compared with the original image. The method uses one of the Greedy programming algorithms which is the Divide and Conquer strategy. The second step includes constructing Grey Level Co-occurrence Matrices (GLCMs) creating for the three quantized images using three distances 1, 2, and 3 and different directions of context. Resulted GLCMs matrices are flattened and concatenated, then reduced using Principle Component Analysis (PCA) method to result three vectors with different lengths of GLCMs statistical information. The third step includes entering three GLCMs vectors into the proposed multi inputs one dimension convolution neural network (1D CNN) model containing three pipelines with different architectures to produce finally one vector of features to represent the image.

The second stage is the shape and topological feature extraction. This stage proposes a convolution layer that uses a random distribution for some constrained weights (trainable parameters) in the kernel. Weights are located randomly in specific locations considering the desired percentage specified by the user. This layer is introduced to extract a special type of feature considering the local shape of a sub-image (window) and topological relations between a group of pixels. This layer is implemented using a proposed two dimension convolution neural network (2D CNN) model architecture that uses proposed convolution layers as well as traditional CNN layers.

The third stage is the image retrieval process. This stage includes extracting features of dataset images using two previous stages in the offline phase. Then, concatenating these features in one vector for each image in the dataset and storing them in one database. Also, in the online phase, extracting the query features using the same previous two stages. Then, similarity matching is performed between database features and query features to retrieve the more similar ranked images to the query.

In experimental results, the proposed quantization method outperforms the other traditional methods in terms of PSNR and SSIM metrics for different gray levels. The proposed CBIR system is implemented and evaluated using three types of medical datasets specified for COVID-19 disease, each separately. The proposed 1D CNN and 2D CNN models are implemented using these three datasets. The models prove the extracted features' efficiency, and outperform the pre-trained DNNs and state-of-art methods. The proposed CNN models achieve accuracy of about 98%, 89%, and 93% for the three datasets, respectively. Moreover, the proposed CBIR system obtains efficient and acceptable results and achieves top-10 ranked results of 99%, 94%, and 93% in terms of mean average precision (mAP) metric for the three datasets, respectively.

Publications Associated with this Thesis

Some of the works presented in this dissertation are published as listed following:

1. **Published paper:** Abbood, Elaf A., and Tawfiq A. Al-Assadi. "Gray Image Quantization Method Based on New Threshold Optimizing Technique." In 2021 1st Babylon International Conference on Information Technology and Science (BICITS), pp. 86-91. IEEE, 2021. DOI: https://doi.org/10.1109/BICITS51482.2021.9509881
2. **Published paper:** Abbood, Elaf Ali, and Tawfiq A. Al-Assadi. "GLCMs Based multi-inputs 1D CNN Deep Learning Neural Network for COVID-19 Texture Feature Extraction and Classification." Karbala International Journal of Modern Science, vol. 8, no. 1, pp. 28-39, 2022. https://doi.org/10.33640/2405-609X.3201
3. **Accepted paper:** Abbood, Elaf Ali, and Tawfiq A. Al-Assadi. "A New Convolution Neural Layer Based on Weights Constraints" International conference on Data science and intelligent computing 2022 ICDSIC, IEEE, 2022

Table of Contents

Table of Figures

List of Tables

List of Algorithms

List of abbreviations

Abbreviation	**Description**
1D CNN	one dimension convolution neural network
2D CNN	two dimension convolution neural network
Adam	Adaptive Moment Estimation
AI	Artificial Intelligent
ANN	Artificial Neural Network
AWDO	adaptive wind driven optimization
BBHE	Brightness Preserving Bi-Histogram Equalization
BCE	Binary cross entropy
CBIR	Content Based Image Retrieval
CBMIR	Content Based Medical Image Retrieval
CCE	Categorical cross entropy
CCL	Proposed 2D constrained convolution layer
CLAHE	Contrast Limited Adaptive Histogram Equalization
COVID-19	Coronavirus disease 2019 pandemic
CT	Computer Tomography
D1	SARS-CoV-2 CT-scan dataset
D2	COVID-CT dataset
D3	DLAI3 Hackathon COVID-19 Chest X-Ray dataset
DCNN	Deep Convolution Neural Network
DECAPS	Detail-Oriented Capsule Networks
DeTraC	Decompose, Transfer, and Compose
DNN	Deep Neural Network
DR	Dimensionality reduction
EME	Effective Measure of Enhancement
GANs	Generated Adversarial Networks
GLCMs	Grey Level Co-occurrence Matrices
GLRLM	Gray Level Run Length Matrix

GLRM	Gray Level Run Matrix
HE	Histogram Equalization
HRO	Chinese hybrid rice method
ICA	Independent Component Analysis
IR	Information retrieval
ISOMAP	Isometric mapping
KNN	K- Nearest Neighbours
LBGLCM	Local Binary Gray Level Co-occurrence Matrix
LBP	Local binary pattern
LDA	Linear discriminant analysis
LDA	Linear Discriminant Analysis
LSTM	long short-term memory
mAP	mean average precision
MDS	Multi-Dimensional Scaling
MRI	Magnetic Resonance Imaging
PCA	Principle Component Analysis
PSNR	Peak to Signal Noise Ratio
ReLU	Rectified Linear Unit
RNNs	recurrent neural networks
SBIR	Semantic-Based Image Retrieval
SCNN	Siamese CNN
SFTA	Segmentation-based Fractal Texture Analysis
SGD	Stochastic Gradient Descent
SIFT	Scale-invariant feature transform
SM	similarity measures
SSIM	Structural Similarity Index Measure
SURF	Speeded up robust features
SVD	Singular Value Decomposition
SVM	Support Vector Machine
Tanh	Hyperbolic Tangent
TBIR	Text-based image retrieval
TCL	traditional convolution layer
VLAD	Vocabulary learning
xDNN	eXplainable DNN

Chapter One

Introduction

1.1 Introduction

Many teams and researchers collect datasets and use Artificial Intelligent (AI) techniques to help hospitals and medical scientific research centers for a correct and rapid diagnosis. An image retrieval system is one of the most useful AI techniques in this field. The image retrieval system that depends on the extracted features for the images rather than name, annotation, or description is called the Content-Based Image Retrieval (CBIR) system [1].

The coronavirus disease 2019 pandemic (COVID-19) is one of the recent virus diseases spread and caused many injuries and deaths in the world. Computer Tomography (CT) scans and X-Ray images consider non-invasive (potentially bedside) tools to test and monitor the progress of the disease [2].

A CBIR technique is considered an effective medical diagnosis and assistance treatment method for diverse diseases. CBIR is an efficient management tool for handling a large amount of data [3]. The development of an effective medical image retrieval system is required to aid clinicians in browsing large medical datasets. To facilitate the process of production and management of such large medical image databases, many algorithms for the automatic analysis of medical images have been proposed in the literature [4].

The main and important stage in CBIR is the texture feature extraction algorithms that identify the best features to represent the image and contain fewer parameters [5]. Feature extraction using Grey Level Co-occurrence Matrix (GLCM) method is a popular and powerful statistical texture-based feature extraction approach. It can describe useful and efficient image features, thus, a successful retrieval task [6].

The resultant GLCM matrix dimensions depend on the number of image grey levels. If the grey levels of the image are large, the GLCM matrix dimensions will be quite large, and therefore it will be time and memory-consumption during its process [7]. For this reason, quantization is a useful process to reduce image gray levels.

Quantization is used in many applications and image processing tasks to perform more fast performance and more efficient results with large datasets. Also, applications such as image retrieval have a good effect if it contains a quantization as a pre-processing [8] [9] [10]. Principle component analysis (PCA) is a widely used dimensionality reduction methods that present its effectiveness with many applications and algorithms [11]. To increase the CBIR performance, image enhancement is one of the pre-processing methods that are useful in these tasks. Indirect contrast enhancement is one of the enhancement techniques that gets an efficient effect on medical images by modifying the histogram of the image and thus increases contrast [12] [13].

Image feature extraction using Convolution neural networks (CNNs) is one of the powerful methods used in many image processing applications. Deep CNN captures the low-level features in the earlier convolution layer and progresses to get the high-level features in later convolution layers [14]. There are many types of CNN networks according to the filter dimensions used; 1D, 2D, and 3D CNN. The 2D convolution layer is the standard type of CNN that contains appropriate dimensions of filters and slide size. 3D CNN requires more trainable parameters than 2D CNN usually used with video data. 1D CNN involves a lower number of trainable parameters than other CNN types and it is usually used with audio and 1D serial data [15] [16].

1.2 Problem Statement

The strength of the CBIR system depends mainly on the accuracy and efficiency of features extracted from the input data (images) and their significant impact on increasing the powerful system. The medical images especially chest images infected with COVID-19, are very complex and contain detailed information that interferes with the symptoms of the disease. Thus, using one type of feature extraction method depending on one type of image details will not describe the image accurately and precision form, thus, not giving the required performance in the retrieval process. Also, the main problem of using Deep Neural Network (DNN) methods in retrieval systems is how to take advantage of the low-level features such as texture, shape, and topological relations in earlier CNN layers model and strengthen them with high-level features that usually get in later CNN layers.

CBIR system, especially in medical fields, needs to utilize the diversity of the extracted features to get powerful, accurate, and efficient resultant retrieved images. The performance of the retrieval system is determined by the resultant images that contain accurate information that help to accurately diagnose and monitor the disease's progress.

1.3 Related Works

The proposed system includes many contributions achieved by proposed many techniques and methods. This dissertation deals with the following topics: gray image quantization, Texture features extraction using GLCM, developing CNN layers, 1D CNN and 2D CNN model architectures, and CBIR system. In the following, some of the previous works related to the proposed system consider these topics.

In the context of quantization methods, there are many works deal with this method:

1. Kotte Sowjanya et al. (2018) [17] introduced a multi-thresholding gray levels method based on adaptive wind-driven optimization (AWDO) algorithm and applied it to Magnetic Resonance Imaging (MRI) brain images. The method produced the optimum threshold values by maximizing the popular objectives between class variance (Otsu method) and Kapur's entropy. Otsu method is a process to find the optimum threshold separation values for k class (intensities) using an intra-class variance. The results obtained are near to 48% and 88% in terms of PSNR and SSIM metrics, respectively.
2. R. Srikanth and K. Bikshalu (2020) [18] used Otsu's method on Energy Curve and Harmony Search method as a substitute for the histogram to find the best gray levels. The method found the optimum by computing the maximum variance between regions of the image. Also, to find threshold values, the Harmony Search method using an objective function that maximize the inter-class variance was used. Results closed to 17% as average for 5 gray levels in terms of PSNR metric.
3. Wei Liu et al. (2020) [19] introduced a method that adapted a breeding mechanism of the Chinese hybrid rice method (HRO) to find the optimal gray level values. They benefited from Renyi's entropy as a fitness function its values ranged from superior to interior to optimize the results. HRO, as a metaheuristic function, accelerates the optimization process to find the optimum threshold values compared with other optimization functions. Results obtained for 8 gray levels are nearby 17% and 68% for PSNR and SSIM metrics, respectively.

Compared with previous methods, the quantization method proposed in this work depends on another mechanism to find the optimum threshold values to produce the quantized image. The proposed method finds that the quantized gray levels gradually started from two gray levels until the desired number of gray levels. The method adopts a divide and conquer strategy for fast and accurate results, considering optimizing PSNR during the division process.

In the context of using GLCM texture features to represent the medical images as well as COVID-19 CT and X-Ray images, there are many recent types of research dealing with this field and introducing methods to process it.

4. Saban and Bayram (2018) [20] used many texture feature extraction methods such as GLCM, LBP, LBGLCM, GLRLM, and SFTA. They compared the classification results using these algorithms. They used large size medical histopathological images and partition them into pieces with smaller sizes. They used the GLCM properties like entropy, homogeneity, and dissimilarity as texture features as well as the other texture features methods. Their method collects all features from texture feature extraction methods in one feature matrix and evaluates its efficiency using many classification methods like SVM, KNN, LDA, and Boosted Tree approaches. They found that SFTA get best accuracy with 94.3% using boosted tree method.
5. K. Shankar et al. (2021) [21] introduced a method to classify chest X-Ray COVID-19 dataset using handcraft feature extraction approaches. Weiner filtering enhancement method was used as a pre-processing for the dataset images. They fused features extracted using three types of texture feature extraction techniques: GLCM, GLRM, and LBP after selecting the optimum set using the salp

swarm algorithm. The collected features are classified using a traditional artificial neural network (ANN) to distinguish between infected and healthy patients. They get accepted classification results with 95.1% and 956% on binary and multiple classes, respectively.

6. Saban Ozturk et al. (2021) [22] analyzed the X-Ray and CT scan images using two steps of image enhancement. They used the data augmentation technique to treat the deficient and unbalanced in the dataset. To analyze the images, they used handcraft feature extraction methods such as GLCM, LBGLCM, GLRLM, and SFTA approaches and combined these features in one feature vector. Then, they used synthetic minority over-sampling technique as the second step of enhancement. PCA and stacked auto-encoder are used to reduce the dimension of the feature vector and finally classified it using the SVM technique. They get classification accuracy with 94%.
7. Yifan H. and Yefeng Z. (2019) [23] extracted the GLCM matrix from the information of irregular regions of MRI and CT scan images. They applied the CNN model and utilized the GLCM as one of the CNN inputs as well as the 2D image. GLCM matrix was used as a 2D matrix and constructed for one distance and four directions only. The CNN models used to implement classification tasks were ResNet-18 and VGG-11 pre-trained CNN models. They get best accuracy with 87.5 % using ResNet18 and GLCM method.
8. Jiaxing T. et al. (2019) [24] introduced a CNN for 3D polyp diagnosis Information analysis. First, they converted the 3D poly information to many 2D images with k gray levels. Then, they constructed many GLCM matrices in tensor, each matrix represents one of the generated 2D images (sampling view of 3D polyp

diagnosis). they presented the CNN model with sequences of convolution and pooling layers for the classification task and to evaluate the extracted features. They achieved 93% in terms of AUC metric.

The first three related works 4, 5, and 6 used GLCM properties to extract the dataset images' texture features and don't utilize the raw GLCM statistics to feature extraction purposes. The two related works 7 and 8 used a 2D GLCM matrix as input to the CNN model rather than its properties. However, the constructed GLCM is not generated in many distances and directions. Furthermore, as well known, GLCM is a sparse matrix and contains many zeros values. In this dissertation, efficient use for GLCM matrices is presented by generating them in many distances and directions. Furthermore, special processing for sparse values problem is introduced to get more efficient performance, and thus, accurate CBIR system results.

In the context of developing CNN layers, there are many types of research that deal with this purpose:

9. Belhassen Bayar et al. (2018) [25] introduced a type of CNN called constrained convolutional layer for image manipulation detection for forensic tasks. They adaptively learn the kernel weights as image manipulation traces. The constraint used in their approach included set -1 to the weight located in the center of the kernel and the sum of the rest weights is equal to 1. This constraint is applied during training and the error computing is constrained with the constrained kernels in this layer. This layer made the CNN model tend to do a specific task, but the number of training weights is still not changed. They get accuracy with 99.97% in manipulation detection.

10. Ephy R. et al. (2021) [26] introduced a topology CNN layer that used manifold relationships metrics to parameterize the kernels' weights values. They used two types of manifold metrics; first localized the weights concerning the location of the circle in addition to using the traditional CNN locality metric. Second, they localized layer kernels weights for the Klein bottle manifold metric. They set kernel weights to zero if the slices' layer weights are greater than the threshold to the two manifolds metrics and initialize the rest layers' kernels weights randomly. The method is proven to reduce the number of weights, but in only specific two forms of topological relations and that is constrained by the produced features in a limited form direction. They get accuracy with 97% using Klein filter method.
11. Juan P. et al. in (2021) [27] introduced a CNN layer for regularization purposes. The layer applied a random rotation process with a small rate on some feature maps after applying the convolution process. They rotated the feature map at an oriented rate selected randomly and this was done during the training phase without increasing the size and number of rotated feature maps. They get accuracy with 97% using the introduced method.

In this dissertation, the proposed convolution layer suggests a different constrained distribution for the weights in each kernel of the convolution layer. Instead of training all weights in kernels, the trainable weights will be located randomly in specific locations with a particular percentage determined by the user. The rest locations of kernels will contain zero weights and not change during a training phase.

In the context of using DNNs to implement feature extraction COVID-19 dataset images, there are diverse methods introduced and presented to this task.

12. Eduardo Soares et al. (2020) [28] built a public COVID-19 dataset called SARS-CoV-2 CT and made it a baseline for later research that deals with developing AI techniques in this field. They introduced the eXplainable DNN (xDNN) method which is considered prototype-based learning to test and evaluate this dataset. First, the method converts the image to one vector with 4096 values using VGG-16 pretrained method. Then, processed the resultant information using density local peaks and resultant probability distributions for each class separately. They get accuracy with 97.3% using SARS-CoV-2 dataset.
13. Matteo Polsinelli et al. (2020) [29] introduced a DNN model to classify the COVID-19 CT scan dataset by building a custom CNN model based on SqueezeNet pre-trained DNN that has many squeeze convolution layers called fire modules. They added the Transpose Convolutional layer to the last fire module and expand the feature maps 4 times. Then, concatenated these feature maps with the second custom fire module followed by weighted sum, convolution layer, and global average pooling layers. They get accuracy with 84.5% using COVID-19 CT scan dataset.
14. Aryan Mobiny et al. (2020) [30] introduced a DNN model called "Detail-Oriented Capsule Networks (DECAPS)" applied to COVID-19 CT scan images. The method merged Capsule Networks by using "Inverted Dynamic Routing" method to prevent the information passing from undescriptive segments. Then, they employed a Peekaboo training function, drop strategy, and activation maps to crop the patches and focus on regions of interest as well as merge the large and small details that represent the image. Furthermore, they used data augmentation to increase the number

of images in the dataset, thus improving the model performance. They get accuracy with 87.6% using COVID-19 CT scan dataset.

15. Xuehai He et al. (2020) [31] introduced a DNN model a Self-Trans method that merged contrastive self-supervised learning and transfer learning in one model to prevent the overfitting problem. They defined different transfer learning strategies and compared between them using a collected COVID-19 dataset to distinguish the infected and normal image cases. The self-trans method gets the intrinsic patterns and image features and not depends on human-provided labeling. The method used auxiliary processes include building a dictionary on-the-fly based on the contrastive loss. They get accuracy with 86% using COVID-19 CT scan dataset.
16. Pedro Silva et al. (2020) [32] introduced a DNN method to classify the COVID-19 CT scan image dataset based on a voting approach. They utilized the pre-trained EfficientNet weights by copying them into a new model and adding an additional layer on top of the model. Then, they train the model randomly initialized to the learning process according to the loss function and optimization method. They reconstructed two COVID-19 CT scan datasets using a voting method and used it to evaluate the DNN model. They get accuracy with 87.6% using COVID-19 CT scan dataset.

The proposed 1D CNN and 2D CNN models introduced in this dissertation are evaluated using the datasets described in some of the previous related works. The proposed CNN models get efficient performance and outperform some of these works as well as pre-trained DNN that are transfer learning methods as explained in experimental results in chapter

In terms of medical image retrieval systems;

17. Adnan Qayyum et al. (2017) [3] introduced a CBMIR system that used DCNN model classification purposes. The system extracted medical image features using a 2D CNN neural network. The retrieval process depended on extracted features and predicted class from the classification phase. They used and trained five types of convolution layers and three types of max pooling layers as well as three types of fully connected layers to extract image features. They get average classification accuracy about 99.7% and mean average precision about 69% for retrieval task.
18. Yu-An Chung and Wei-Hung Weng (2017) [33] introduced a deep Siamese CNN (SCNN) architecture that applied to binary image pair information. They used two identical CNNs sharing the same weights that are built using ResNet-50 architecture with the ImageNet pre-trained weight. They used 25% dropout for regularization and used batch normalization, rectified linear units (ReLU) nonlinearity activation function layers for all layers. They get 66% and 77% as retrieval results in terms of mean average precision and mean reciprocal rank metrics, respectively.
19. Haripriya P and Porkodi R (2019) [34] presented DCNN neural network model to extract image features and then used these features to retrieve similar images. They used metadata information as well as images and convert it from RGB to grayscale. The DCNN model contained five convolutions, three max pooling, and three fully connected layers. They get precision 80% with 22 class for medical images dataset.
20. Pradnya Maske and J. A. Kendule (2020) [35] retrieved medical images using two stages; first: they encoded the input medical image into a set of features using pre-trained VGGNet parameters as initial weights followed by the decoder which reconstructs the

input medical image from the encoded features. The second stage makes use of the extracted features by the encoder for index matching and retrieval tasks. The presented CNN architecture contained the basic building blocks of the CNN networks that consist of a combination of convolution and rectified linear unit (ReLU layer) as well as a max-pooling layer. They get accuracy 99% for retrieval task.

All these methods are introduced a CBIR system for different types of datasets with various DNN techniques that utilized to extract details in low-level image features as well as high-level features which is the main goal of using DNN in retrieval systems.

Table (1.1) explains the brief report for related works and their methods and results.

Table (1.1) The related works brief report

Ref.	year	Specialization	The methods used	results
[17]	2018	Quantization	multi-thresholding gray levels method based on adaptive wind-driven optimization (AWDO) algorithm and applied it to Magnetic Resonance Imaging (MRI) brain images	PSNR: 48% SSIM: 88%
[18]	2020		Otsu's method on Energy Curve and Harmony Search method as a substitute for the histogram to find the best gray levels.	PSNR: 17% of 5 grey levels
[19]	2020		adapted a breeding mechanism of the Chinese hybrid rice method (HRO) to find the optimal gray level values	PSNR: 17% SSIM: 68%
[20]	2018	GLCM with medical datasets	texture feature extraction methods such as GLCM, LBP, LBGLCM, GLRLM, and SFTA using many classification methods like SVM, KNN, LDA, and Boosted Tree approaches	Accuracy: 94%
[21]	2021		Weiner filtering enhancement method and fused features extracted using three types of texture feature extraction techniques: GLCM, GLRM, and LBP after selecting the optimum set using the salp swarm algorithm	Accuracy: 95.1% and 956% on binary and multiple classes
[22]	2021		analyzed the X-Ray and CT scan images using two steps of image enhancement. Used GLCM, LBGLCM, GLRLM, and SFTA approaches and PCA and stacked auto-encoder	Accuracy: 94%

[23]	2019	GLCM & CNN	They applied the CNN model and utilized the GLCM as one of the CNN inputs as well as the 2D image	Accuracy: 87.5%
[24]	2019		introduced a CNN for 3D polyp diagnosis Information analysis	AUC: 93%
[25]	2018	Developing CNN layer	introduced a type of CNN for image manipulation detection for forensic tasks	Accuracy: 99.7%
[26]	2021		introduced a topology CNN layer using manifold relationships metrics to parameterize the kernels' weights values	Accuracy: 97%
[27]	2021		introduced a CNN layer for regularization by a random rotation process with a small rate on some feature maps after applying the convolution process	Accuracy: 97%
[28]	2020	DNN for COVID-19	built a public COVID-19 dataset called SARS-CoV-2 CT and introduced the eXplainable DNN (xDNN) that considered prototype-based learning	Accuracy: 97%
[29]	2020		building a custom CNN model based on SqueezeNet pre-trained DNN that has many squeeze convolution layers called fire modules	Accuracy: 84%
[30]	2020		introduced a DNN model called “Detail-Oriented Capsule Networks (DECAPS)” applied to COVID-19 CT scan images	Accuracy: 87.6%
[31]	2020		introduced a DNN model a Self-Trans method that merged contrastive self-supervised learning and transfer learning	Accuracy: 86%
[32]	2020		introduced a DNN method based on a voting approach and utilized the pre-trained EfficientNet weights	Accuracy: 87.6%
[3]	2017	CBIR	introduced a CBMIR system used DCNN model classification purposes and extracted medical image features	mAP: 69%
[33]	2017		introduced a deep Siamese CNN (SCNN) architecture that applied to binary image pair information	mAP: 66% mRR: 77%
[34]	2019		presented DCNN neural network model to extract image features	Precision: 80%
[35]	2020		encoded the input medical image into a set of features using pre-trained VGGNet parameters and used encoder for index matching and retrieval tasks	Accuracy: 99%
Proposed system		CBIR system	Proposed CBIR system based on texture features using GLCM and 1D CNN and shape features using 2D CNN with proposed convolution layer	Classification accuracy: 98%, 89%, and 93% mAP:99%, 94%, and 93% for three datasets

1.4 Dissertation Aim

The aim of this study is proposing the CBIR system retrieve accurate and efficient images more similar to the query one. The retrieval process depends on the images’ context, shape, and topological relations features using different feature extraction methods and utilized from CNN models to extract high-level features. The performance of the proposed system is represented in increasing the efficiency and accuracy of descriptive features of the medical images and therefore improving the performance and power of the retrieval process. Moreover, this dissertation introduces a new direction in DNN work by proposing a 2D convolution layer that is utilized to focus on the local shapes and topological relations in earlier layers of the CNN model. Also, the aim of proposing this method is to reduce the number of trainable parameters of the convolution layer used in the 2D CNN model compared with the traditional CNN layer.

1.5 Challenges

There are many problems and issues are faced with CBIR system's work specially the feature extraction stage. The methods and techniques used to extract features are different according to features type and level. The challenges faced the work and design CBIR system include:

1. Most of the existing CBIR methods that are based on deep learning algorithms use traditional feature extraction methods or apply the DNN models on images directly with simple preprocessing steps. These methods are not utilizing the diversity of the extracted features that are increased the powerful and efficient CBIR system according to different types of image features.
2. Most of the existing CBIR methods that use GLCM as a statistical feature extraction method are not look at the GLCM matrix as a whole. They used GLCM properties to extract the texture features of images and do not utilize the raw GLCM statistics. The raw GLCM contain many diverse texture features can be extracted using DNN techniques. In this dissertation, we generate a multi GLCMs matrices in different distances and directions and enter these matrices into the 1D CNN

model with particular additional processes as well as a specific quantization method.

3. The dimensions of extracted GLCM matrices depend on the image gray levels, thereby, the multi GLCMs matrices with these dimensions will be very big to be input to the CNN model. Therefore, this dissertation uses a new gray level quantization method to reduce image gray levels at different desired levels.
4. The diversity of the extracted features, it’s useful to use another type of images features that is the local shape and topological pixel relations based on the proposed CNN layer. The proposed layer excludes the percentage of weights in each filter in the 2D convolution layer and trains the rest part. To increase the features’ accuracy and precision, the image enhancement method as preprocessing step are used.

1.6 Contributions

The contributions of this dissertation are illustrated as:

1. Propose a non-uniform quantization method to optimize the quantized gray level values of the images uses Divide and Conquer strategy that depends on optimal selection for threshold values considering the PSNR metric.
2. Propose a method that extracts efficient and perfect texture features using GLCMs matrices constructed in three different distances and many different directions. These matrices are applied to three quantized image versions from each image in the dataset.
3. Propose a 1D CNN model architecture with three different input layers that required a little number of parameters. The model is trained on the image GLCMs statistical matrices only after flattening, concatenating, and reducing using PCA to produce good represented features for the retrieval task.

4. Propose a 2D CNN layer with a specific distribution for weights in each convolution filter. The proposed method includes a new suggestion for the filters' form in the convolution layer. The weights in these filters concentrate on the local shape and topological relations of the sub-image (window).
5. Propose a 2D CNN model architecture using successive two convolution layers followed by one pooling layer. The proposed 2D CNN model uses several of the proposed convolution layers as well as the traditional convolution layers. Also, an enhancement method as preprocessing on images is used to increase the accuracy and efficiency of medical image features.

1.7 Organization of dissertation

After presenting this chapter, which contains a brief description of the work as a whole, the remaining chapters of this dissertation can be described as follows:

- Chapter Two "Theoretical background": This chapter describes the techniques and methods utilized to explain the work of the proposed system.
- Chapter Three "The Proposed System": This chapter explains the proposed system stages, block diagrams, and algorithms used to implement this dissertation.
- Chapter Four "Results and discussions": This chapter presents performed results and the evaluation produced in each stage of the system. As well as, this chapter explains the system stages discussions to prove the efficiency of the proposed techniques.
- Chapter Five "Conclusions and Future Works": In this chapter, conclusions obtained from this dissertation and suggestion for a future works are introduced.

Chapter Two

Theoretical Background

2.1 Introduction

This chapter provides a theoretical background with respect to the research work described in this dissertation. Firstly, there is need to explain the different forms of features to represent the image. The texture feature extraction methods including the GLCM method, furthermore, shape, topology, and learning feature extraction techniques are explained. Secondly, presenting the CNN network architecture, layer types, training process, parameters, and hyperparameters as well as 1D CNN. Then, illustration for the information retrieval system, especially CBIR is introduced. Next, image quantization process and image enhancement techniques are explained. Also, there is explanation and definition for dimensionality reduction techniques including specific clarification for the PCA technique used in this dissertation. The last section includes the evaluation metrics for different techniques used in this dissertation including; image quality metrics, CNN evaluation metrics, and CBIR evaluation metrics.

2.2 Image Feature Extraction

Image features are defined as a vector of values collected using one or set of functions each specialized in a specific property of the image. There are many types of features represent the descriptive form for the significant characteristics of the image like color, texture, shape, topological relations, and transform features [36]. Feature extraction is a process of transforming the significant content of the image to the reprehensive and efficient features that describe it to be used in different tasks like classification, detection, pattern recognition, data mining, image retrieval, diagnosis, and other computer vision systems. Feature extraction methods are classified in many ways considering several aspects.

The performance and accuracy of image processing techniques, including retrieval systems, depend primarily on the image feature extraction methods. In many types of research, several feature extraction methods are developed that includes advantage and disadvantages. An efficient feature extraction technique provides appropriate descriptive characteristics [37] [38].

The popular classification manner for feature extraction methods is to classify methods into two main classes; general and domain-specific features. The general features contain all extracted features from the context of the image on different levels that includes; color, texture, shape, topological relations, and learning features. Figure (2.1) illustrates a summary of the classification of image feature extraction methods [39].

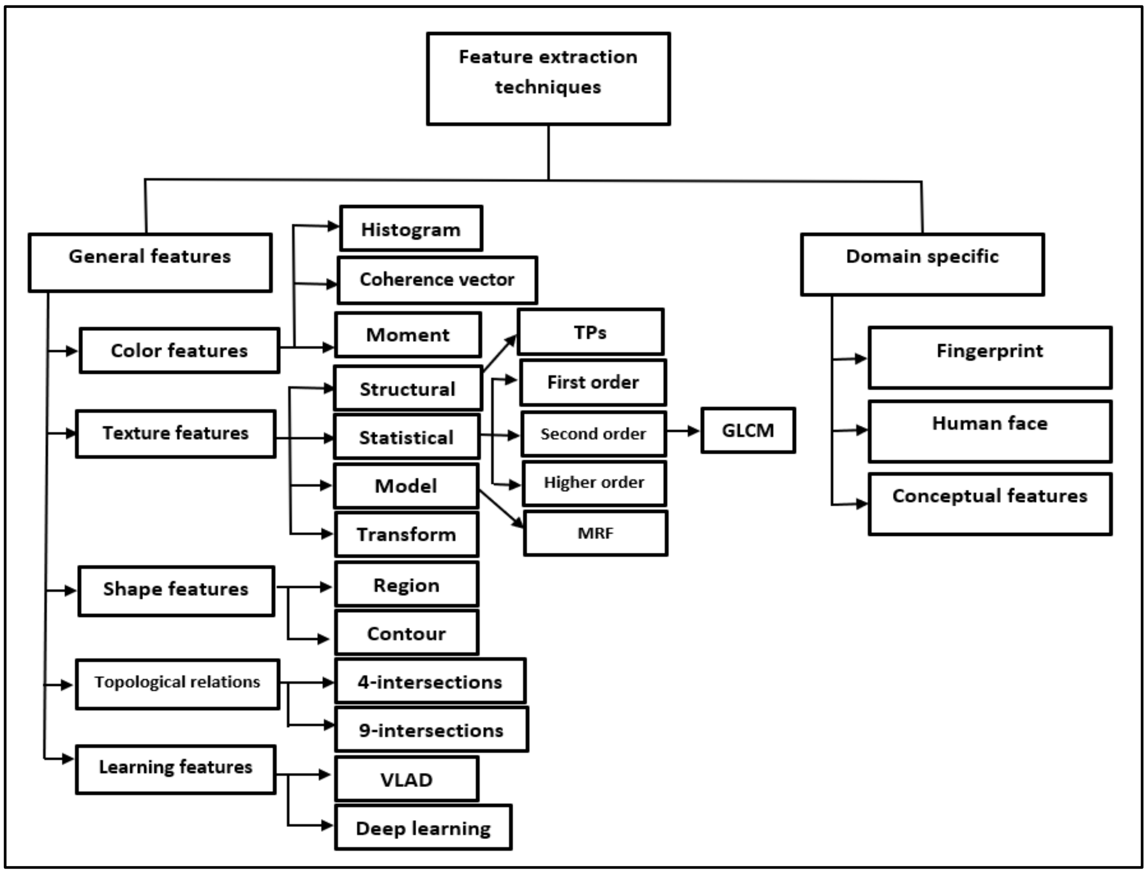


Figure (2.1) Summary for classification of feature extraction techniques [39]

Each of these classes of features is classified another classification according to the level of dealing with images into three branches [36] [40]:

- **Pixel level features:** contains the features computed considering each pixel in the image, e.g. color, statistical texture, morphology features.
- **Local level features:** includes the features extracted from blocks, regions, or segments of the image, e.g. local shape, Scale-invariant feature transform (SIFT), and Speeded Up Robust Features (SURF).
- **Global level features:** contains features computed for all the images a whole or sub-image, e.g. contour shape, global texture descriptors.

2.2.1 Texture Features Extraction

The texture is a group of pixels that describes the image structure or is defined as prearranged repeated pixels patterns in a specific form. The texture is complex visual patterns that are collections of pixels, or objects in patterns that have characteristics of good effects, colors, and shapes.

Also, texture can define as patterns of information or order of the structure with consistent intervals. In general, texture denotes the surface appearances and characteristics of an object described in size, shape, density, and arranging, of its basic parts. Generally, human visual systems use texture for distinguishing and identification. While color is typically a property of pixels, the texture is measured from the number of pixels formed in groups [37] [41].

As mentioned, some texture feature extraction methods proposed, and some of them and examples are presented in Figure (2.1) [39]. The methods are divided based on the scheme of processing data and the domain used to extract features. The techniques broadly include structural, statistical, model, and transform-based methods and each of them has advantages and disadvantages [42] [43].

2.2.1.1 Gray Level Co-occurrence Matrix (GLCM)

Texture statistical approaches are used to analyze the spatial allocation of gray values by calculating local features at each point in the image. Thus, the gray levels' spatial distribution is considered one of the defining texture qualities. The texture statistical characteristics features include density, uniformity, and directivity of the image. Statistical methods are considered adaptable, robust, and simple feature extraction methods, but they require a lot of statistical calculations and consume a large size of memory [39] [43]. Statistical methods are classified into first-order (one pixel), second-order (two pixels), and high-order statistics considering the number of pixels used to define the spatial features of an image [44].

The Grey level co-occurrence matrix (GLCM) approach presented by Haralick [45] is considered one of the second-order statistics techniques which consider the relationship among pixels or groups of pixels (usually two). GLCM is a squared matrix G with N×N size, where N refers to the number of gray levels of the image. G (i, j) denotes the number of occurring pairs of pixels with gray levels i and j in special considerations using spatial relations between each other. The relation between the two gray levels is defined by distance d and direction (orientation) Ɵ which are denoted as a polar coordinate (d, Ɵ). Therefore, there are many GLCM matrices in different polar coordinates that can be generated using modified spatial relationships [46] [47]. The GLCM elements can defined as Eq. (2.1) [48] [36]:

$$G(i,j)_{(d,\theta)} = num\begin{Bmatrix} [(a1,b1),(a2,b2)] \in X \times Y \\ |f(a1,b1)=i, f(a2,b2)=j \end{Bmatrix} \qquad (2.1)$$

Where, $G(i,j)_{(d,\theta)}$ denoted the GLCM matrix element located in index *i* and *j* with distance *d* and *Ɵ* direction between two pixels (a1, b1) and (a2, b2). *num(x)* defines the function for the number of elements in set *x*. *f* indicates the gray image with size $X \times Y$.

Once, all frequencies of each two gray levels $G(i,j)_{(d,\theta)}$ in image *f* is computed, the final GLCM matrix is used after normalizing its elements using Eq. (2.2) [49]:

$$GLCM(i,j) = \frac{GLCM(i,j)}{\sum_{v=0}^{N}\sum_{w=0}^{N} GLCM(v,w)} \qquad (2.2)$$

Where N×N is the size of the GLCM matrix that denoted the number of gray levels in the image. Figure (2.2) illustrates a simple example of the GLCM computing process [50].

It is clear that GLCM size depends on the number of gray levels of the image, therefore, it is considered memory consumption and computation time is large, as well as, contains sparse values. This explains why many applications use a limited number of distances and directions to generate GLCM. For that, usually, it is useful to quantize the gray levels number to decrease the dimensions of the GLCM matrix. Generally, the number of quantized gray levels are differing values such as 8, 16, 32, 64, 128, and 256 [43] [47].

It is worth mentioning, if the pairs of pixels are extremely correlated, the GLCM values are focused along diagonal of it, the cause makes GLCM very sensitive to the size of processed textures. For many image tasks, d must be selected carefully including the patterns of the texture and keeping the local details of the spatial components. As mentioned before, the disadvantage of the GLCM is that the matrix has high dimensionality and consumes time and memory. Despite this, the GLCM method is easy to implement and can get very efficient outcomes in many application fields [51].

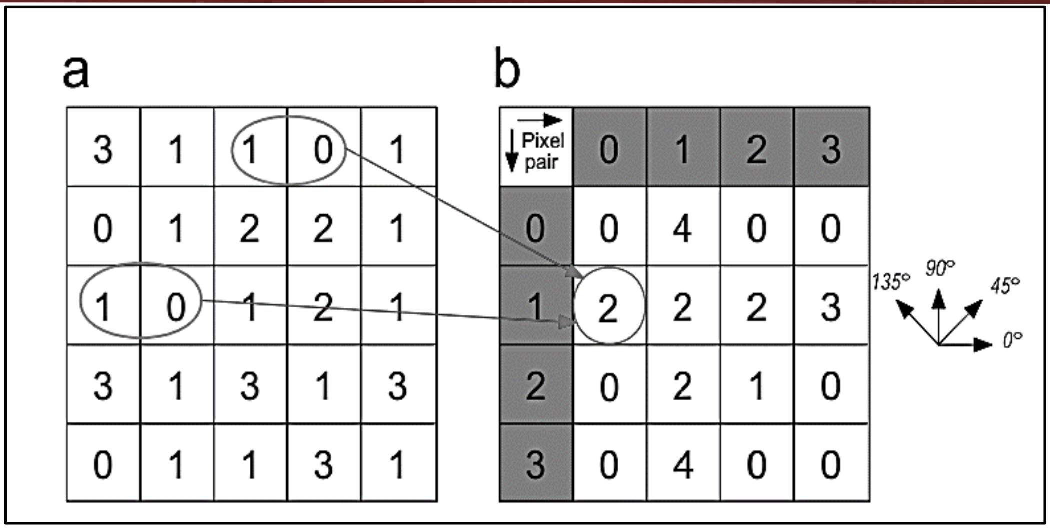


Figure (2.2) Example of GLCM computing [50]

GLCM matrix can be used directly to represent the image features and distinguish the texture as presented in many methods such as GLCM-CNN used in [23] [24]. However, there are several features (also called properties) introduced by Haralick [45] that can be extracted from the GLCM matrix some of them are efficient for many tasks. These features include; inertia (contrast), angular second moment, correlation coefficient, entropy, linearity correlation degree, and inverse variance [52].

These features and a lot of other linear combinations of GLCM elements can be generated and be useful to represent image texture features. Nevertheless, it has not been theoretically proven that these features give optimal results for tasks used. So, the entire elements of GLCM matrix is entered as an image to predictor or convolution neural network (CNN) to derive the efficient features instead of using the GLCM statistical features. Also, when comparing the CNN features with the traditional GLCM features it is found that CNN optimizes the loss function, thus, can get more optimum and efficient features than traditional features. Furthermore, traditional features are generated using linear function only but CNN features are generated using some non-linear functions represented in some types of layers, such as convolution, pooling, and activation layers [23].

2.2.2 Shape Feature Extraction

The shape is an important and basic geometrical visual feature that give human a descriptive form of the object or part of it. It considers one of the features used to recognize and identify the image content that represents the primitive features. Shape feature extraction is one of the useful and efficient features used in image retrieval tasks. The shape feature is represented using a shape descriptor which is defined as a vector of values obtained in different ways to characterize the shape features. A descriptor describes the shape as representative of human intuition [42] [53].

Shape feature extraction techniques are mainly classified into two essential branches. First, the contour-based methods depend on extracting the features on the boundary of objects. Second, the region-based methods depend on the whole contents of a described object including interior information as well as the boundary. Each class is subdivided based on the part of the object used to extract the shape. These two sub-divisions are; structural methods that deal with shape primitives or segments. Also, there are global methods that deal with the shape features as a whole [54]. Figure (2.3) illustrates the more popular classification for shape feature extraction methods [55].

All these traditional methods extract the shape features with many aspects. However, with the evolution of deep learning feature extraction methods, there is an enhancement in the ability to extract more efficient features and get an acceptable degree of feature quality. The deep learning algorithms include many techniques such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) [56]. Deep learning methods are very appropriate to describe the local contour shape features rather than the global shape information. That is performed in the former convolution layers as well as the intermediate layers of the CNN model [57].

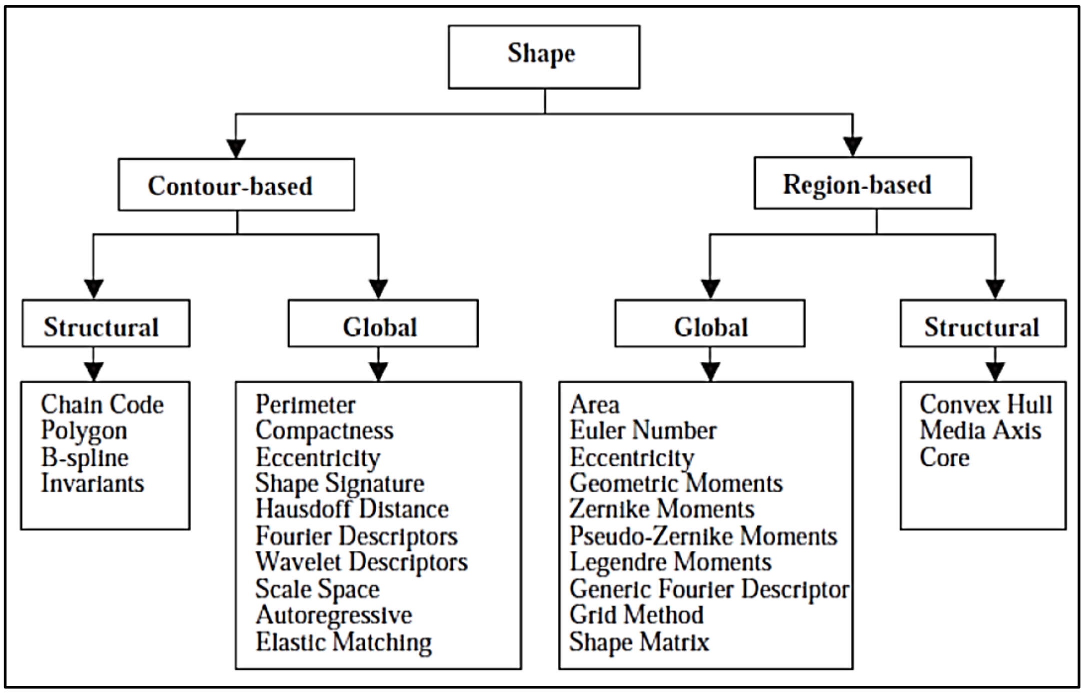


Figure (2.3) classification of shape feature extraction methods [55]

2.2.3 Topological Relations Feature Extraction

Topological relations are considered one of the spatial object properties described by an arbitrary set of elements to preserve the spatial relations between objects. Topological relations consist of several operators which define transformations applied to a single spatial object or among many spatial objects. The relationships between objects in space can be any spatial geometrical primitives such as point (0D), line (1D), and area (2D) spaces for different types of deformations [58]. There are several models presented for topological relations extraction; the Allen model [59], Egenhofer region model, and Egenhofer line [60].

The two Egenhofer models adapted the topological algebra to describe the geometrical primitives in various dimensions. 4-intersections and 9-intersections models are examples of popular topological relations models that are based on spatial relations between objects A and B. These relations are represented by intersections (∩) between interior (°), exterior (¯), and boundary (∂) of these objects. The possible possibilities between these topological primitives can be described in Eq. (2.3) [61]:

$$R(A,B)=\begin{pmatrix} A^{\circ}\cap B^{\circ} & A^{\circ}\cap \partial B & A^{\circ}\cap B^{-} \\ \partial A\cap B^{\circ} & \partial A\cap \partial B & \partial A\cap B^{-} \\ A^{-}\cap B^{\circ} & A^{-}\cap \partial B & A^{-}\cap B^{-} \end{pmatrix} \tag{2.3}$$

The elements in the R matrix have been a value if the topological relation is presented (1), or else it will be (0) indicating no relation. Figure (2.4) illustrates the eight topological relations with 9 (3×3) intersections matrices [62]. Considering all 9-intersections combinations that are 2^9 is equal to 512 relations. Nevertheless, not all of them are possible [61].

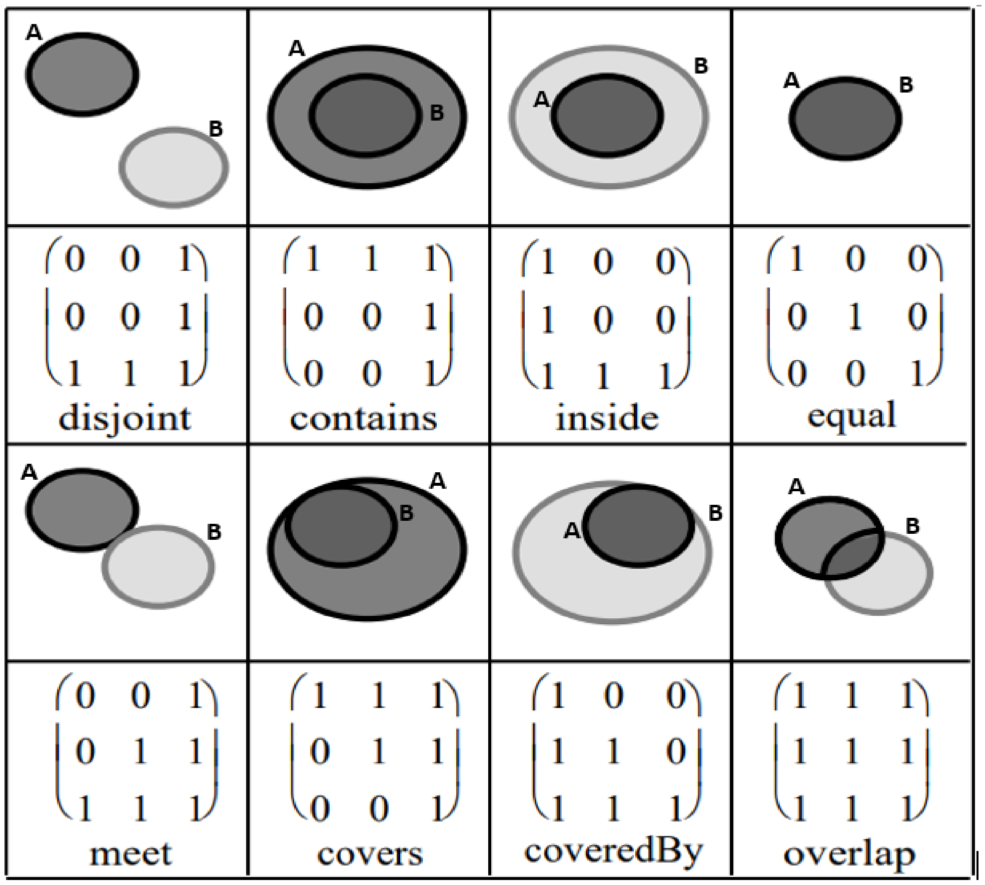

A
B
$\begin{pmatrix}0&0&1\\0&0&1\\1&1&1\end{pmatrix}$
disjoint
A
B
$\begin{pmatrix}1&1&1\\0&0&1\\0&0&1\end{pmatrix}$
contains
A
B
$\begin{pmatrix}1&0&0\\1&0&0\\1&1&1\end{pmatrix}$
inside
A
B
$\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$
equal
A
B
$\begin{pmatrix}0&0&1\\0&1&1\\1&1&1\end{pmatrix}$
meet
A
B
$\begin{pmatrix}1&1&1\\0&1&1\\0&0&1\end{pmatrix}$
covers
A
B
$\begin{pmatrix}1&0&0\\1&1&0\\1&1&1\end{pmatrix}$
coveredBy
A
B
$\begin{pmatrix}1&1&1\\1&1&1\\1&1&1\end{pmatrix}$
overlap

Figure (2.4) Eight types of topological relations with their 3×3 intersections matrices [62]

It is worth mentioning that\, topological relations can be computed in several methods and their possibilities can give good accuracy in many applications. Therefore, the topological feature can be extracted automatically using machine learning, especially in deep learning techniques [26] [63] [64].

2.2.4 Learning Feature Extraction

In recent years, a new type of feature extraction method has emerged which uses learning methods and predicted the appropriate features to represent images. This type of image feature is classified into many categories depending on the techniques used to learn the model. Some of these categories explain in the following [43]:

a. **Vocabulary learning (VLAD):** includes learning for the visual dictionary which trains the visual words using learning methods such as K-mean or Gaussian mixture methods. Then, construct local descriptors and rearranged them in the visual dictionary. The learning of a dictionary requires key points detection. Finally, the local descriptors are encoded and pooled to construct the global image descriptor dictionary that contains the reprehensive features for the images. This method has many examples of applications such as 2D and 3D texture features and encoding of their geometric and photometric structures [65].
b. **Deep learning features:** these features are obtained from deep learning networks that are usually convolutional neural networks CNN especially in computer vision tasks. The features extracted using CNN prove its efficiency and excellent capability to produce representative features for the image datasets [43].

   The advantage of these features is that CNN can extract different levels of features automatically gradient from low, intermediate, and high-level features depending on the layers used to obtain the feature descriptors.

However, the limitations of deep learning feature notably CNN models summarized in two points. First, deep learning techniques are considered computationally expensive methods depending on the number of training stages and the amount of training part of the dataset. Second, the earlier deep learning methods use one level of features that are obtained from one type of these models without using the hybrid or aggregated methods to merge many types of features [66].

The deep learning features can be applied in many tasks such as image classification, retrieval, object detection, and pattern recognition. In the context of image retrieval, deep learning features are used at different levels and performed effectively compared with other traditional feature extraction methods [67]. Several methods use deep features in retrieval tasks considering the way that is dependent. Following there are some of these methods:

- Deep learning features as a descriptor vector extracted from different fully connected layers to represent features [3] [33].
- Deep learning features obtained from fully connected layer then compressed using PCA or aggregated using VLAD method [67].
- Deep learning features are obtained from the last convolution layer followed by the sum-pooling layer to encode and reduce the representative features [68] [69].
- Deep learning features represented by the last output layer of the CNN model that contained one neuron referred to the predicted class of image [70] [71].

2.3 Convolution Neural Network (CNN)

Deep learning is also called hierarchal learning consider one of the common branches of machine learning. Deep Neural Networks (DNN) inspired a new direction of processing the high dimensions and huge datasets. DNN converts the computationally expensive concept of multiple layers models to a more abstract and concise form. DNN can be classified into three types considering the specific use. First, supervised models that use data with class labels e.g. AlexNet and InceptionNet architectures. Second, unsupervised models that learned with no labels for data e.g. Belief autoencoder architecture. Third, hybrid DNN is intermediate between the previous two classes e.g. Generated Adversarial Networks (GANs) architecture [72].

Convolution Neural Network (CNN) is a particular type of DNN that is spatialized to process the complex problems of visual data such as images, and videos, as well as other data types such as text, audio, and other multimedia forms. The main difference between CNN and traditional neural networks is that the CNN is designed to reduce the number of trainable parameters (weights), and the number of neurons, therefore, reducing the complexity of the model. Thus, the CNN models have a high generalization and are appropriate for high dimensional data [73]. CNN deal with 2D data such as image usually called 2D CNN. In general, the purpose of CNN is to extract efficient features automatically in different levels of the model. Practically, CNN in a popular form consists of several layers of difference in functions and works. These layers include; convolution, batch normalization, activation, pooling, flattening, and fully connected layers. Figure (2.5) illustrates a general architecture for CNN and explains some of its layers [74].

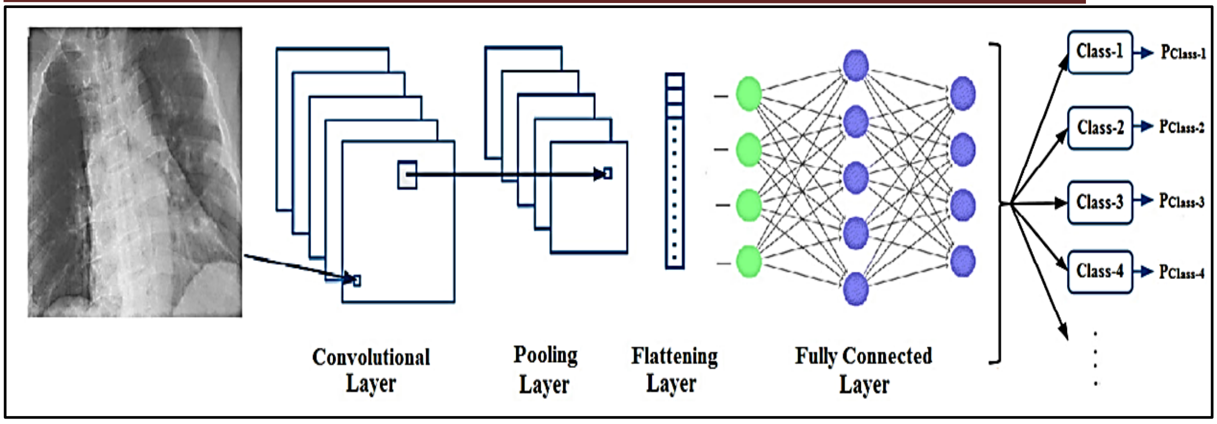


Figure (2.5) General architecture for CNN [74]

2.3.1 2D CNN Architecture

As mentioned, 2D CNN consists of several layers to construct 2D CNN model to perform a specific function. The design of 2D CNN model depends on repeating and arranging these layers and depends on parameters and hyperparameters specified in the designed model. The details of 2D CNN layers are explained in the following.

a. 2D Convolution Layer

The convolution layer which also called Conv considered the building block of any CNN model. The Conv layer is performed using a convolution process. This process is utilized to abstract the feature information throughout the sequences of layers. The filters in the earlier convolution layers are dedicated to extract low-level and local features such as edges of local shape etc. In the intermediate layers, the filters extract the mid-level features that are constructed by a combination of low-level features such as more general shapes and edges. In the later convolution layers, the filters are designed to extract high-level semantic features that include the global information and overall objects and information in the image. In other words, as the convolution layers progressed, the level of abstraction increased [75] [76].

For each convolution layer, the input of the layer is the image channels (in the first convolution layer) or feature maps (in the output of the previous layer). The size of input feature maps is referred to as (X, X, A) where A is the number of features maps each one has X×X size.

Since the feature maps input to the convolution layer with X×X with depth A, the kernel for the convolution layer (filter or weights) is a square matrix k×k (typically k= 3, 5, 7, …) with depth A contains the weights which convolved over the input image using dot product calculation that is convolution process to produce the feature map. The depth of kernels is the number of kernels for each convolution layer referred to as F that is also referred to as a number of feature maps output from the convolution layer. For that, the number of weights for each layer is k×k×A×F (number of parameters is explained in detail in Section 2.3.3) [75].

Moreover, padding is an important notation in the convolution layer. It is referred to the number of pixels (usually zero) added around the input feature map to make sure accomplishes the convolution process on the boundary pixels of the feature maps. The padding size depends on the kernel size. Furthermore, stride refers to the number of steps of move (convolve) kernels over the feature map, typically, it is one. In other words, it refers to the jump size of moving kernels. As the stride is large, the size of the jump is bigger, thus, the produced feature map will be smaller [77].

As mentioned, for each current convolution layer *l*, the input feature maps have $X \times X \times A^l$ dimension and have F^{l+1} ($l+1$ is the next layer) kernels with size k×k×A^l. To produce one location for output feature map, each k×k sub-image from the input feature map is convoluted with kernel matrix k×k for all depth A^l. The sum values for all results from A^l is computed and located in one location of output feature map. The kernel is moved with stride size (typically one) and repeats the convolution process. The scan is from top to bottom and from left to right of the feature map. Figure (2.6) illustrates an example of this process of convolution in CNN [78]. To produce all output feature maps (depth F^{l+1}), the convolution process is repeated for depth F^{l+1} of different kernels with size $k \times k \times A^{l}$. For that, the total number of weights for each convolution layer is $k \times k \times A^{l} \times F^{l+1}$ [78].

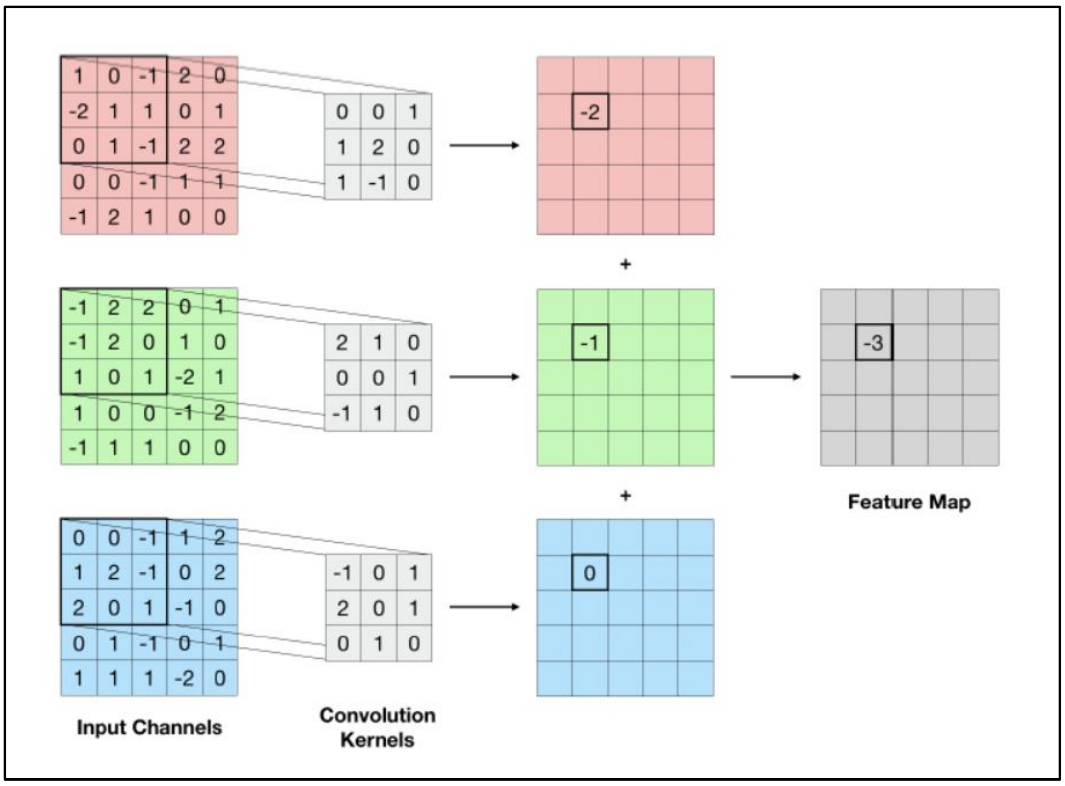


Figure (2.6) Example of convolution process for 2D CNN [78]

b. 2D Pooling Layer

Each feature map produced from the convolution layer mostly has the same size (X×X) with different depth (depending on the layer depth). The pooling process also called subsampling or down-sampling is reducing the size of feature maps along the CNN layers, thus, reducing the network as a whole. The pooling layer focuses on the extracted information contained in feature maps by obtaining the more useful and efficient features in many statistical ways. The pooling process is computed for each pooling window that is a square window from the feature map with a specific size denoted by the part which processed using the pooling function. The stride of pooling is the number of steps to move the pooling window (usually the same size as pooling window. In general, the pooling layer is calculated using Eq. (2.4) [79]:

$$X_n^l = \sigma(\rho_n^l \ . \ pool(X_n^{l-1}) + b_n^l \qquad (2.4)$$

Where, X_n^l is the output of the pooling process for the feature map n in the layer l, X_n^{l-1} is the feature map n in the current layer l-1, ρ_n^l and b_n^l are the multiply and addition biases for the pooling layer, respectively, pool(.) is the pooling function that computed for each pooling window, σ is the non-linear activation function.

There are several methods of the pooling functions that can be explained three types of them in the following:

1. **Max pooling layer:** is the most popular pooling used in CNN and used in the first layers in the architectures due to sparse values contained in feature maps. The result is computed by calculating the maximum values in the pooling window.
2. **Average pooling layer:** mostly used in the later layers when all values of feature map are important and contain useful information. The result is computed by getting the average of the pooling window.
3. **Global average pooling:** it's also used mostly in the later layers and computed by finding the average of all values of the feature map [80].

Figure (2.7) shows a simple example of max and average pooling process [80].

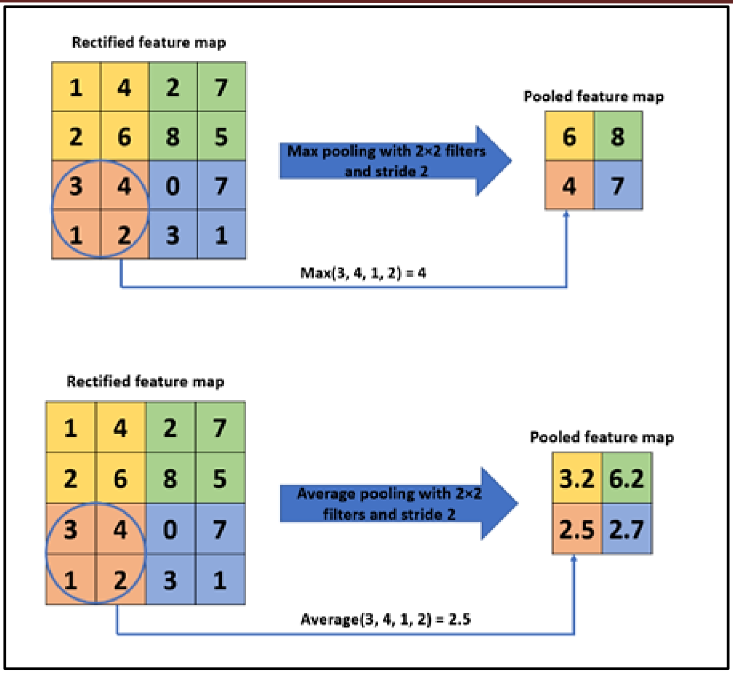


Figure (2.7) Example of Max and Average pooling process [80]

c. Batch Normalization layer

This layer is used between layers of CNN architecture to raise the stability and efficiency of the performance of the resulting features in the training phase. This process includes normalizing the feature map values with zero mean and 1 standard deviation. Batch normalization is utilized to recover the overfitting problem stronger than the dropout method. Furthermore, Batch normalization can get a faster training process, especially when used with complex networks [76]. The mathematical formula for the batch normalization process is explained in Eq. (2.5) [81]:

$$\widehat{N} = \frac{N-\mu(N)}{\sqrt{V(N)}} \qquad (2.5)$$

Where, $\widehat{N}$ is the output of layer, N is the input to batch normalization process, μ and V are the mean and variance for N that are explained in Eq. (2.6) and Eq. (2.7), respectively [81].

$$\mu(N) = \frac{1}{x}\sum_{i=1}^{x} N(i) \qquad (2.6)$$

$$V(N) = \frac{1}{x}\sum_{i=1}^{x}(N(i) - \mu(N))^2 \qquad (2.7)$$

Further, the process may include bias parameter for scale and shift process to obtain a linear transform to the results as explained in Eq. (2.8) [81]:

$$Y = \widehat{N}.\beta + \rho \qquad (2.8)$$

Where, β and ρ are bias parameters for scale and shift, respectively.

d. Activation layer

The activation function is a basic and important layer in CNN. The main purpose of this function is present the nonlinearity of feature maps values that increase the model efficiency to learn the complex data for several tasks. It is a critical and important issue to choose the appropriate activation function because of its impact on reducing the loss function during tuning the trainable parameters in backpropagation using gradient descent [77]. There are many types of activation functions:

- **Sigmoid:** used in binary classification tasks that compute the probability of prediction by converting the input values to the range [0, 1]. In some research, this function is called a logistic function used in the final of CNN model and is a time-consumption function [82].
- **Hyperbolic Tangent (Tanh):** used to convert the input values to smoothness form in the range [-1, 1]. It is considered the sigmoid function scaled version and also time-consuming, but its gradient is more stable than sigmoid [76].
- **Rectified Linear Unit (ReLU):** this function converts the negative values to 0 and the positive values remain the same. Thus, this function passes many input values in contrast with the other activation functions [76]. ReLU solves the vanishing gradient problem that is happen when the gradient of the neuron arrives to zero then the neuron is never learned and is useless [77].

Figure (2.8) illustrates a curve representing sigmoid, Tanh, and ReLU activation functions [83]. The mathematical formula of sigmoid, Tanh, and ReLU activation functions are explained in Eq. (2.9), Eq. (2.10), and Eq. (2.11), respectively [84].

$$sigmoid(n) = \frac{1}{1+e^{-n}} \qquad (2.9)$$

$$Tanh(n) = \frac{e^{n}-e^{-n}}{e^{-n}+e^{n}} \qquad (2.10)$$

$$ReLU(n) = \max(0, n) \qquad (2.11)$$

Where, n is the input values vector is wanted to be activated.

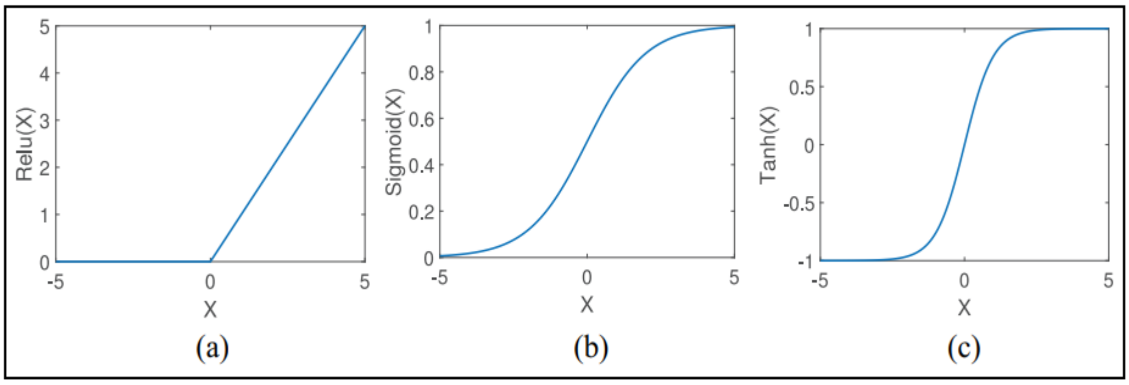


Figure (2.8) Activation functions: (a) ReLU, (b) Sigmoid, and (c) Tanh [83]

- **Softmax:** This function transforms the layer values produced from the fully connected layer from numeric to probabilistic form, that is ranged between 0 and 1, and its summation equal to 1 to make the final decision as a classifier. Figure (2.9) shows a curve representing the behavior of Softmax function [85]. The mathematical formula of this function is explained in Eq. (2.12) [85]:

$$y_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \qquad (2.12)$$

Where, y and z are the input and output of Softmax function, respectively, and n is the number of inputs (neurons) in the Softmax activation layer.

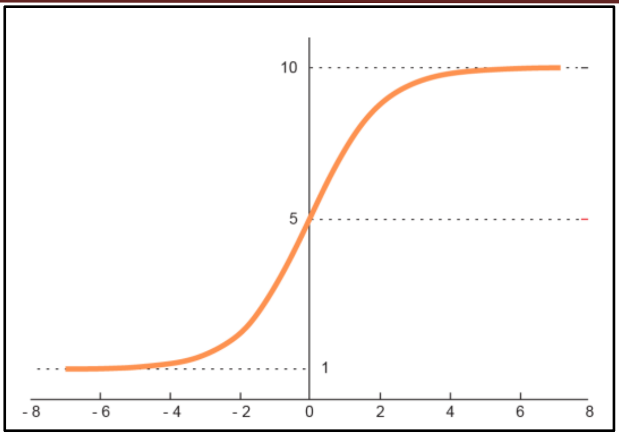


Figure (2.9) Softmax activation function [85]

e. Flattening layer

Flattening is the procedure of converting the resulted 2D feature map into a 1D vector to prepare it to be input to the next layer; a fully connected layer. Figure (2.10) illustrates an example of flattening process [82].

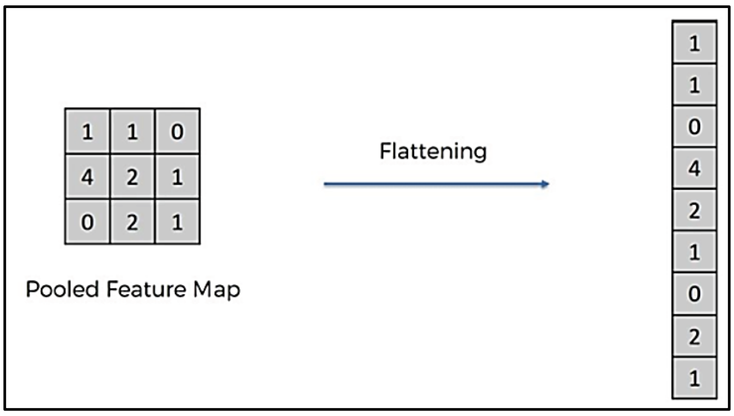


Figure (2.10) Flattening layer example [82]

f. Fully connected layer

The work of a fully connected layer, which is also called a dense layer, is the same hidden layer in the conventional neural networks. This layer is located at the end of CNN architecture and ends with Softmax activation function as a classifier. This layer works by connecting all neurons in the previous layer with all neurons in the current layer. The purpose of the fully connected layer is to combine resulted features from previous layers and compressed them into the final representative form [77] [78].

2.3.2 CNN Training

The training process of CNN includes two main phases; a forward phase that contains the mapping process for the inputs to the outputs using layer parameters, and a backward phase that contains the error computing and trainable parameters updating. This mechanism of training network parameters is called backpropagation that used in enhancing the performance of neural network [86].

The loss function also called the cost function is an important concept in the machine learning field, especially deep learning. The loss function is used at the end of the CNN model to minimize the error between predicted and actual values of the training data. Depending on this function, the CNN model parameters are adjusted and weights are tuned. In the context of the loss function, there are many algorithms used to optimize the trainable parameters, such as Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (Adam), and Mini-Batch Gradient Descent that are considered as some types of gradient-based algorithms [87]. There are many types of loss functions based on the derived distribution used such as cross-entropy function that is Categorical cross entropy (CCE) and Binary cross entropy (BCE). These functions are considered popular loss functions used for optimizing tasks that determine the difference between the probability distributions of training data CNN computed class (predicted) and actual labels. BCE mathematical formula explained in Eq. (2.13) [88] [89].

$$loss_{(BCE)} = -\beta \, . \log\big(\sigma(\gamma)\big) + (1-\beta)\log(1-\sigma(\gamma)) \qquad (2.13)$$

Where, $loss_{(BCE)}$ is loss function using BCE, β and γ are actual and predicted output of the CNN, $\sigma(.)$ is the sigmoid activation function defined in Eq. (2.9).

The initialization weights, forward, and backward phases for convolution layer are explained in the following.

- Kernel weights initialization: in convolution layer, the weights in kernels are initialized randomly of uniform distribution using the Eq. (2.14) [90]:

$$W_m^l = Uni[-\frac{1}{\sqrt{k}} \ ,\frac{1}{\sqrt{k}}] \qquad (2.14)$$

Where, W_m^l is kernel m in layer l, Uni [-a, a] is the uniform distribution function to generate random values in the range [-a, a], and k is the size of the kernel.

- Forward phase: the convolution process for traditional convolution layer is defined as Eq. (2.15) [79]:

$$o_m^l = \sigma((W_m^l * a^l) + b_m^l) \qquad (2.15)$$

Where o_m^l is the output of convolution operation between kernel m and input feature maps of the layer l, W_m^l is kernel weights, a^l is the layer inputs (feature maps), b_m^l is the bias parameter, σ is a non-linear activation function.

- Backward phase: In convolution layer, the weights in the kernel are affected by the loss function of computing the error values and updating of weights in the backward phase. Eq. (2.16) represents the weights kernel updating function using gradient descent error function that computes the error [73]:

$$^{*}W_m^l = W_m^l - Lr * \partial W_m^l \qquad (2.16)$$

Where, $^{*}W_m^l$ is the new weights kernel number m for layer l, W_m^l is old weights kernels, ∂W_m^l is the gradient of error function computed of each weights kernel, Lr is the learning rate that is determined by a very small value in the range [0.1,0.0001].

2.3.3 CNN Parameters and Hyperparameters

The basic parameters in the CNN are the weights in kernels of convolution layer and neurons weights in fully connected layer. Also, Bias is a parameter represented by one value that is added or multiplied in many types of layers.

These parameters are also called ultimate parameters trained and adjusted during the training phase of the model. The final values of ultimate parameters are saved with the final convergent model structure. There is a mechanism for calculating the number of ultimate parameters (weights and biases) for convolution layer using Eq. (2.17) [91]:

$$p_l = (k \times k \times A + 1) \times F \qquad (2.17)$$

Where, p_l is the number of parameters for the current layer l, A is the number of feature maps input to layer l, and F is the number of feature maps computed for layer l, adding one for the bias parameter. Also, F is the number of kernels each with size k×k×A; as mentioned in Section (2.3.1).

Hyperparameters are some other important parameters that control ultimate parameters and training of entire CNN. Choosing these parameters is affects the training process and reaches the convergence state. Hyperparameters include many types such as learning rate, padding, stride, number of iterations (epoch), number of kernels, number of feature maps, and activation functions [77].

2.3.4 1D Convolution Neural Network (1D CNN)

CNN is divided into many categories according to the dimension of input data and thus the dimension of produced feature maps and dimension of kernels used in CNN. CNN contains types such as 1D CNN, 2D CNN, and 3D CNN. The input to 1D CNN is a 1D vector of data (sequential data) that is can be raw data or processed. There are many researches such as [92] [93] [94] used and proved that using 1D CNN with 1D data is reduced the model complexity in size, process, and parameters especially when using sequential data [79].

1D CNN has the same concept of working used in 2D CNN except for the difference in dimension of input data, feature maps, and used kernels. Since the input to 1D CNN is a vector with X length, the resulted feature maps are also 1D vector with length X for a number of feature maps A. Therefore, the kernels used in convolution process have k×A dimensions where k is the length of kernel and A is the depth (the same number of input feature maps). In pooling layer, the size of pool window is 1D that represents the length of values entered into the pooling process [95]. Figure (2.11) explains the 1D CNN architecture that contains the same type of layers contained in 2D CNN with different dimensions such as input, convolution, pooling, activation, flattening, fully connected, and output [96].

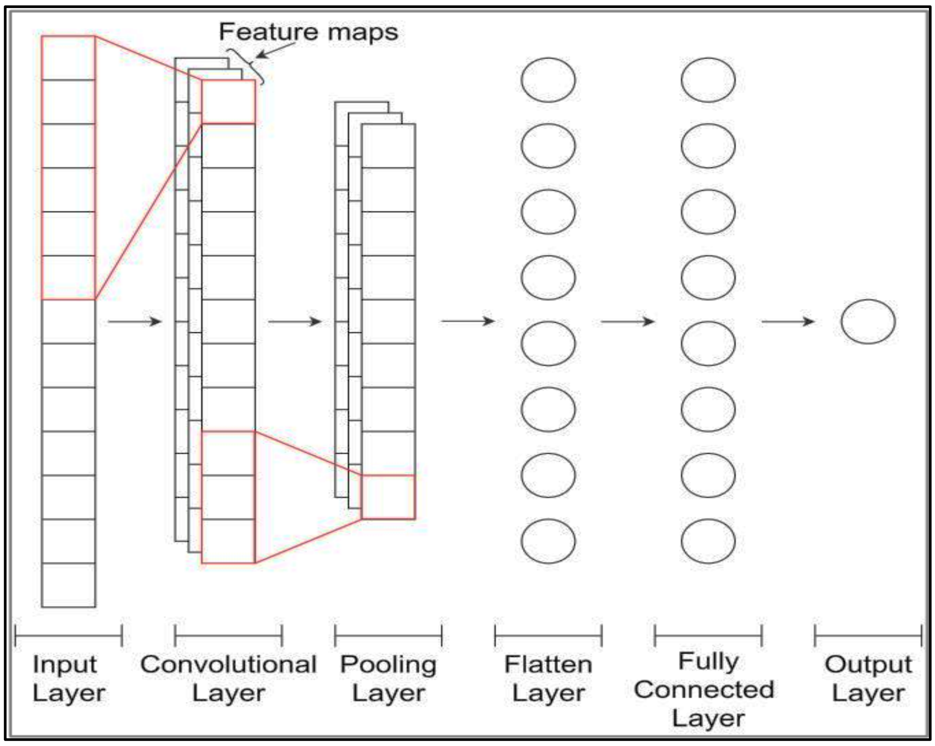


Figure (2.11) 1D CNN architecture [96]

2.4 Image Retrieval System

Information retrieval (IR) system is one of fields of artificial intelligence that include searching for information resources relevant to the information needed to look for. IR deal with searching for many types of information such as text, image, video, sound, as well as metadata. The need for an IR system is increased with the increasing a huge number of collected data and the problem of locating a specific item of data in this huge repository [97].

In the context of image, the retrieval system is classified into three classes considering the type of input query and the mechanism used to retrieve images. The classes of image retrieval illustrated in Figure (2.12) [97] with method examples for each class that are include:

1. Text-Based Image Retrieval (TBIR): it is the oldest concept of retrieval. In TBIR, the images are described using text in annotation process. This process includes a collection of keywords, notations, and captions. The retrieval process is employed on the collected set of words between query image and dataset information. TBIR is considered a difficult and complex retrieval process due to its time-consuming. TBIR requires a lot of preparation operations for the mounting of huge numbers of images [98].

2. Semantic-Based Image Retrieval (SBIR): this type of retrieval system is used to address the semantic gap between low-level features extracted from the image and high-level features that are distinguished and visualized by the human. SBIR process retrieved similar images depending on feature extraction methods to describe the efficient details and visual features of image. Then, SBIR depends on semantic feature extraction methods and is added to the extracted features database [99].

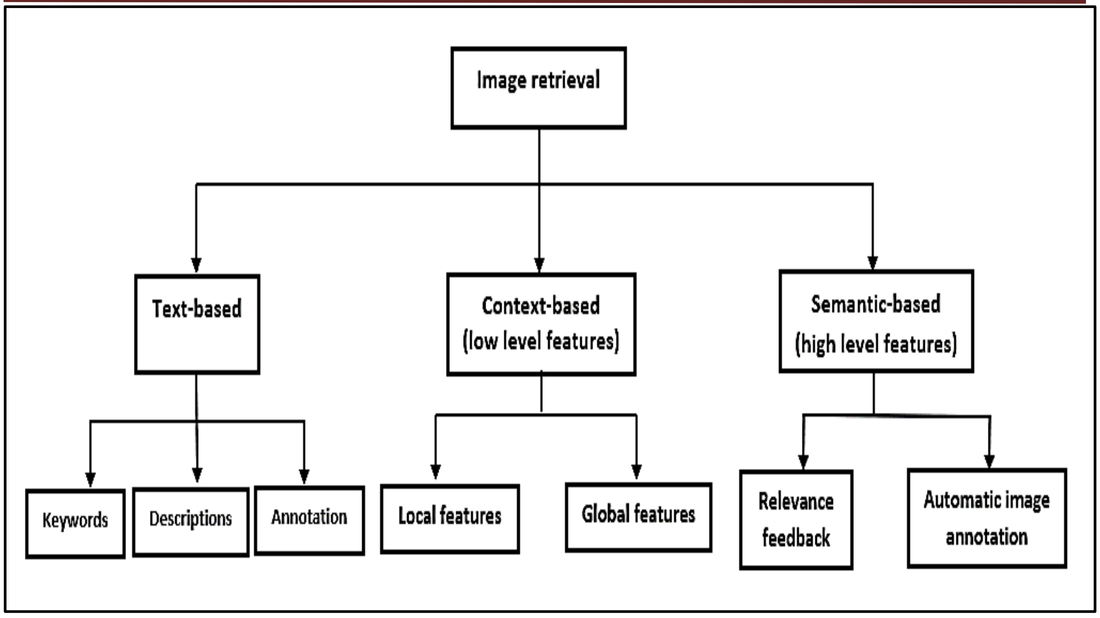


Figure (2.12) Image retrieval methods classification [97]

3. Content Based Image Retrieval (CBIR): the process of retrieving more similar images to the query depending on several features extracted from the image such as color, texture, or shape using diverse techniques. CBIR is including image processing techniques for preprocessing and feature extraction processes that are usually implemented automatically. Then, retrieve images that are similar to the query image by matching the extracted features vectors that represent color, texture, or shape features.

CBIR is used in many applications and several fields such as image historical research, fingerprint identification, disease diagnosis, social media, and web research [100].

Generally, CBIR passes through steps all the images entering the system pass through it including query image. CBIR steps include feature extraction, indexing and similarity matching, and retrieving process. Figure (2.13) illustrates the general CBIR system framework and components [97]. In CBIR, query image is passed through all the steps of processing in a stage called online phase, all dataset images are also passed through them and in the same order and called offline phase [101].

Feature extraction is a most important step that extracts the representative and efficient features and stores them in database. The database repository must be updateable to contain the most recent collection of images. To get more efficient extracted features, image processing techniques can be used as preprocessing before applying feature extraction step. Feature extraction methods produce a vector of features that describe the image such as color, texture, shape, and local features (feature extraction methods are described in details in Section (2.2)) [102].

When the size of dataset is very big and it is difficult to find the most similar feature vectors, indexing techniques for the stored features database are applied. Indexing techniques are methods for sorting the database of extracted features and finding the more similar results to query image features in less time than if using linear searching time. Indexing methods include R-tree indexing and hashing techniques [103].

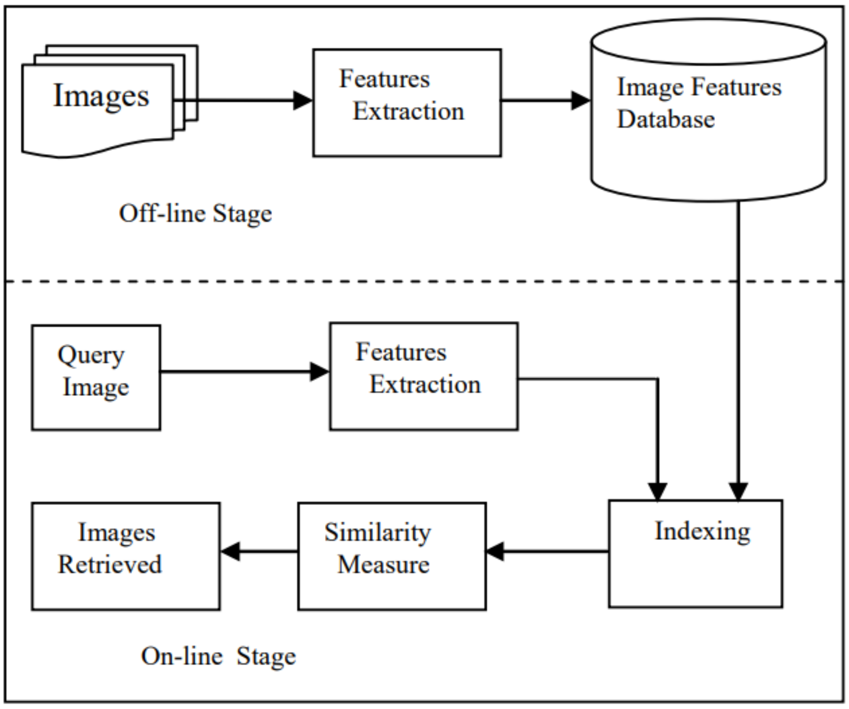


Figure (2.13) General CBIR system framework [97]

Similarity matching is a process of matching between the query image features and the database of all images features vectors and finding the more similar images to the query one. The similarity matching process is performed using one of the similarity measures (SM).

SM is important and critical tools effects on CBIR system performance. Using the SM in CBIR system depends on the composition of the input feature vector values. Furthermore, choosing an appropriate SM results a high performance for CBIR system by retrieved images more similar and relevant to the query one [102]. SM includes many types of similarity/distance measures such as Euclidean, Manhattan, and Mahalanobis Distance these are considered as distance metrics that are measure dissimilarity [104] [105].

Whereas, the cosine distance (Cos) is considered a similarity metric that is used to measure closeness between two values. The resulted value of Cos is ranged between 0 and 1and the larger value of Cos, the greater similarity between two image features. Cos metric that is calculated between two vectors A and B is defined in Eq. (2.18) [105]:

$$Cos(A,B) = \frac{\sum_{i=1}^{L} a[i]b[i]}{\sqrt{\sum_{i=1}^{L} a[i]^2}\sqrt{\sum_{i=1}^{L} b[i]^2}} \qquad (2.18)$$

Where L is the length of vectors A and B.

2.5 Quantization Technique

Quantization is a process of reducing the number of colors in images without effect on the quality of descriptive information and the global appearance of the image. The gray image contains at most 2^8 gray levels are equal (256) different gray levels values each gray level takes 8 bits of the memory [106]. In other words, if the image contains L gray levels, then there is a need to $\log_2 L$ bits to describe the gray levels. That means L is equal to 2^N , where N refers to number of bits need to represent image gray levels [107].

The quantization process is effective when using the GLCM method for texture feature extraction. This method reduces the resulting GLCM dimensions and decreases the computation time as well as the efficiency of resulted features in different quantized gray levels [108]. Also, it is worth noting there is a trade-off between the number of reduced gray levels and the amount of information lost. As the number of quantized gray levels decreases, there are more important details in the image may be lost especially in medical images such as CT scans and X-Ray images [109]. For that, it’s very important to find an efficient quantization method that minimizes the quantized error and loss of important information. There are several quantization methods presented over decades:

2.5.1 Uniform Quantization

Uniform quantization divides the image gray levels into equal range spaces as the desired number of quantized gray levels. It’s also considered the simplest type of zero memory quantizer that is defined as the quantization method when the resulting gray levels are independent of the original image gray levels. Figure (2.14) illustrates the uniform quantization process [107]. If the number of gray levels for the original image is M and the number of gray levels for the quantized image is L, then each threshold value used as quantized gray level T computed in Eq. (2.19) [107]:

$$T = \frac{M(k-1)}{L}, k = 1, \ldots L \qquad (2.19)$$

The lower the number of color values, the greater the loss of spatial details and appearing addition false contour lines. False contouring refers to artificial boundaries that appear because abrupt gray levels change from their real values [110] [111].

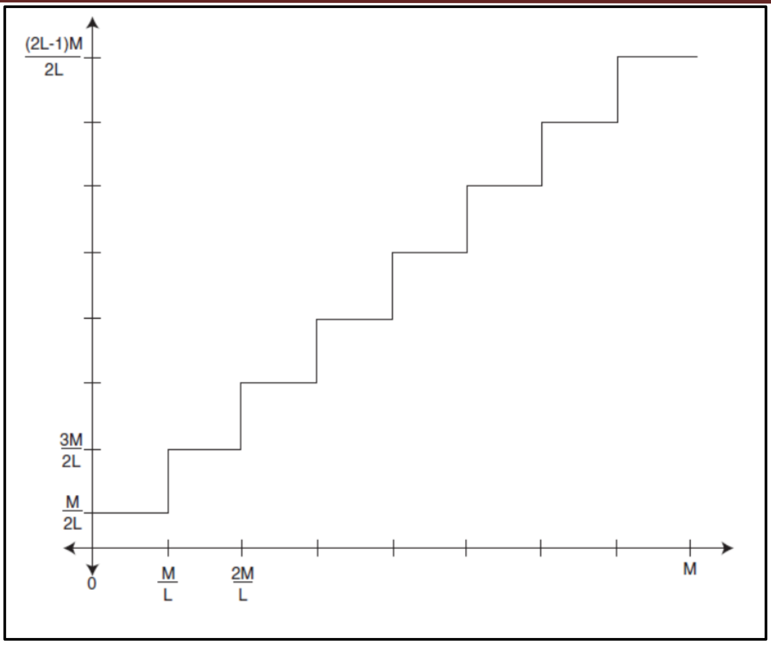


Figure (2.14) Uniform quantization gray levels [107]

2.5.2 Non-uniform Quantization

This method depends on choosing an optimal quantizer that minimizes the distortion of the quantized image instead of depending on equal regions spaces of gray levels. There are many algorithms presented for this process:

a. **Popularity algorithm:** this method depends on selecting threshold values to obtain quantized gray levels in the regions of the histogram that contain most of the gray levels of the original image [111].

b. **K-mean algorithm:** K-mean is a clustering technique that is adapted to use for a quantization process. K-mean is widely used to select the gray levels to minimize the difference between the original and the quantized image. This method is also named as iterative technique. The method uses three methods to determine the initial cluster centers; random color space, random image gray levels, and uniform partition of the gray level values [112]. It is worth noting, K-mean considers a non-uniform quantization method depending on the Lloyd algorithm [113] as a concept of iterative technique to minimize the mean square error to select the cluster centers (threshold values) [108].

2.6 Image Enhancement

Image enhancement is a process of improving the visibility of images in different fields like medical, satellite, and aerial images and different types of images suffer from noise or low contrast. The image enhancement techniques can be classified into two classes based on their implementation space domain [114]. Spatial domain methods that are applied directly to image pixel values or statistical calculations of the spatial image values and are considered fast and simple methods. On the other hand, the frequency domain is a method of enhancing the image based on many transformations using the mathematical function of operations implemented on the transform coefficients of the image [115].

Image contrast is defined as the measure of illuminance and color distribution to distinguish the object features from the background details. In certain applications, medical images can have unbalanced distribution in the dynamic range of gray levels or noise because of the low quality of acquisition devices. Thus, medical images suffer from low contrast which occurs due to inconsistency in lighting and many other effects [116] [117]. In this context, image enhancement techniques can be classified based on: direct and indirect techniques.

a. Direct Techniques

The improvement of image contrast is established on the image pixels directly. Some of these methods adapt fuzzy logic by computing the fuzzy entropy from the constructed fuzzy domain of the image [118]. Some other methods use adaptive neighborhood techniques or multiscale measures on the wavelet domain by enhancing the details of the image on various scales [12].

b. Indirect Techniques

Generally, these techniques use the cumulative density function using a histogram as a basic platform to contrast the enhancement process rather than the image pixels directly [119]. Indirect contrast enhancement methods include:

1. Histogram Equalization (HE)

This method updates the cumulative density function and rearranged the probability distribution of the image gray levels by stretching it in a uniform arrangement. This method suffers from artificial distortions which give an unnatural look to the enhanced image [120].

2. Brightness Preserving Bi-Histogram Equalization (BBHE)

To preserve the local brightness and enhance the contrast, the image histogram is divided into two sub-ranged gray levels. The split process depends on the mean of intensity (brightness) and each part of the histogram is equalized separately [121].

3. Contrast Limited Adaptive Histogram Equalization (CLAHE)

Another version of the histogram equalization process. To overcome the problem of artificial noise caused by equalization for the overall image histogram, CLAHE is presented [122]. This method divides the image into non-overlapped sub-regions and a histogram for each region is calculated. Then, the desired limit of contrast amplification is clipped from the histogram frequencies. It is advantageous not to discard the part of the histogram that exceeds the clip limit but to redistribute it equally among all histogram bins. Figure (2.15) illustrates the clipping step of the CLAHE process [123].

The redistribution will push some bins over the clip limit again (region shaded gray in the figure), resulting in an effective clip limit that is larger than the prescribed limit and the exact value of which depends on the image. If this is undesirable, the redistribution procedure can be repeated recursively until the excess is negligible [124].

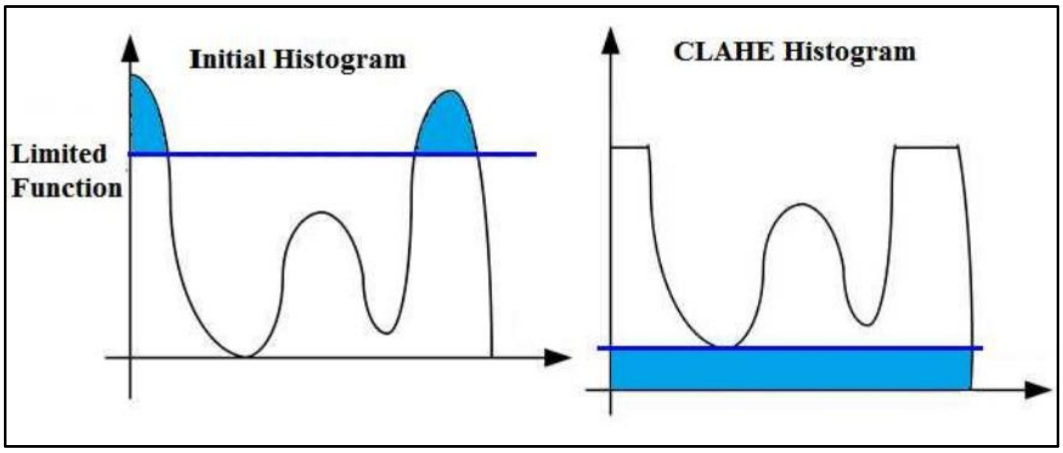


Figure (2.15) clipping step of CLAHE method [123]

2.7 Dimensionality Reduction

Datasets, especially sets of images, contain many of features that are represented in many dimensions to be as input to predictive models. This big size of information and features may affect the time computing and accuracy of results especially when high dimensional spaces data contain noises, unmeaningful, and spare data that are called a curse dimensionality [125].

Dimensionality reduction (DR) refers to the process of reducing the dimension of features during maintaining the most meaningful qualities as much as possible. The DR techniques are classified into many different categories considering the nature of processed data and the mathematical functions used in these techniques. Figure (2.16) illustrates the classification of DR techniques and some examples of methods [126].

The DR techniques can be classified, considering the purpose of their performance into two classes; feature selection and feature extraction. In the feature selection methods, the useful features are selected and the redundant data is discarded directly from the original data. Feature extraction methods refer to transforming the raw features from high dimensional space to lower dimensional space while preserving well-defined and meaningful features to represent the images [127].

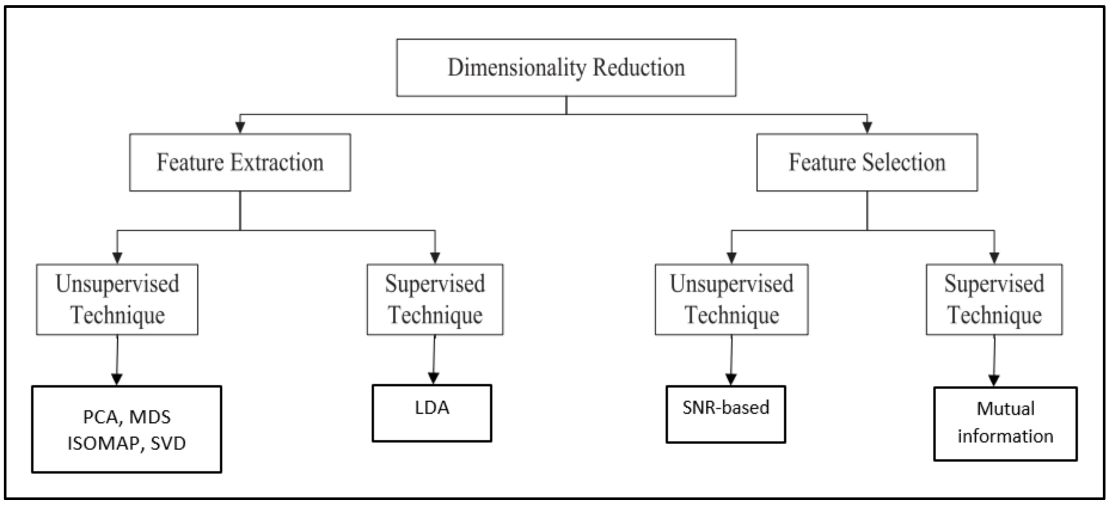


Figure (2.16) Dimensionality reduction techniques classification [126]

In the following, some DR techniques widely used in image feature reduction and their uses and applications are explained:

1. **Linear Discriminant Analysis (LDA)**

This method is considered a supervised method that needs the class label of data with normal distribution for each class. Many applications used this method as multiclass prediction techniques. This method is a linear process that defines a new space for features by maximizing the separation between classes [128].

2. **Independent Component Analysis (ICA)**

ICA is a linear supervised DR method that searches in directions independent of each other in a given dataset. It is considered a feature extraction method to compute new features and reduce the high-order and second-order information in the dataset [129].

3. **Isometric Mapping (ISOMAP)**

ISOMAP is a nonlinear unsupervised classical scaling process. It computes the distance between data points using geodesic distance and finds the shortest path using Dijkstra’s algorithm [125].

4. Singular Value Decomposition (SVD)

SVD is considered a precise representation of the data described in a matrix form have high dimensions. This method is a linear unsupervised process that rearranges the input matrix into new ranked values and larger k-singular values represent the new form of the data [130].

5. Multi-Dimensional Scaling (MDS)

MDS computes the similarity among data points, closes the similar data, and keeps different data away. This method is considered a nonlinear unsupervised process classified into metric and non-metric MSD. Furthermore, it’s used in data visualization applications and has expensive computational complexities [131].

6. Principle Component Analysis (PCA)

A linear unsupervised method transforms the high-dimensional data and reduces its dimensions to the new extracted features without the need to class label the testing dataset [132]. This method is considered the most popular DR technique used in image retrieval system classification, and machine learning (ML) context. It converts a set of correlated components P to smaller uncorrelated linear K values (K<P) called principle components using a set of orthogonal transformations [133].

PCA process is started by constructing the covariance matrix from the input matrix. Then, applying linear Eigen decomposition schema on the resulted covariance matrix to find eigenvectors that represent the data direction and its corresponding eigenvalues that represent the data magnitude. Finally, transform the original data to the top k sorted eigenvectors according to their eigenvalues [134].

2.8 Evaluation Metrics

In this dissertation, there are several evaluation metrics are used to evaluate the methods used in parts of the system. These metrices includes evaluation for; quality of image, CNN models, and image retrieval system. In the following subsections, the definitions and mathematical equations of evaluation metrics used in this dissertation are explained.

2.8.1 Image Quality Evaluation Metrics

There are many popular quality metrics that used to evaluate the image processing techniques and interpret the affect and changes on processed image. Some of these metrics used in this dissertation are explained as following:

a) Peak to Signal Noise Ratio (PSNR): is an objective metric often considered as a similarity metric between two images. A higher value of PSNR, the good quality, and more similar images. PSNR reflects the human perception for the quality of images. PSNR measures the ratio between the power of maximum value in image and power of the deforming noise measured by mean square error (MSE) metric illustrated in Eq. (2.20). The PSNR values are ranged between [0,1] and the accepted values for PSNR are equal or larger than 25. PSNR is represented using Eq. (2.21) [115].

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[I(i,j) - I^{\prime}(i,j)]^2 \qquad (2.20)$$

$$PSNR = 10log_{10}\frac{X_I^2}{MSE} \qquad (2.21)$$

Where, I is the original image with size M×N, $I^{\prime}$ is the processed image, $X_I$is the maximum pixel value in I.

b) Structural Similarity Index Measure (SSIM): is an objective metric that measure the similarity between two images. SSIM results values in range [0,1], that is a higher SSIM value, a more similarity measured. This metric is correlated with the human visual system quality perception. SSIM is a grouping of three parameters; correlation loss (CL) Eq. (2.23), luminance distortion (LD) Eq. (2.24), and contrast distortion (CD) Eq. (2.25). the SSIM values ranges are [0,1] and the mathematical formula of the SSIM quality metric is explained in Eq. (2.22) [135]:

$$SSIM(I,I') = CL(I,I') \times LD(I,I') \times CD(I,I') \qquad (2.22)$$

$$CL(I,I') = \frac{v_{II'}+C3}{v_I v_{I'}+C3} \qquad (2.23)$$

$$LD(I,I') = \frac{2m_I m_{I'}+C1}{m_I^2+m_{I'}^2+C1} \qquad (2.24)$$

$$CD(I,I') = \frac{2v_I v_{I'}+C2}{v_I^2+v_{I'}^2+C2} \qquad (2.25)$$

$$C1 = (x1 \times L)^2 \qquad (2.26)$$

$$C2 = (x2 \times L)^2 \qquad (2.27)$$

$$C3 = \frac{C2}{2} \qquad (2.28)$$

Where, *I* and *I´* are original and processed images, respectively. $v_{II'}$ means the covariance between *I* and *I´*. *m* and *v* are the mean and standard division for the *I* and *I´* images. *C1, C2,* and *C3* are constants to stabilized the denominator of division and defined in Eq. (2.26), Eq. (2.27), and Eq. (2.28), respectively. *L* is the dynamic range of image pixels. *x1* and *x2* are positive small constants.

c) Effective Measure of Enhancement (EME): computes the average contrast of the image by dividing image into nonoverlapping blocks, finding a measure based on minimum and maximum intensity values in each block, and averaging them [121]. This metric is computed for one image, not between two images and this is why it is used in unsupervised AI methods when there are no images to compare with. The range of EME values are [0,1], a large value of EME, a high and good image contrast improvement. The mathematical formula of EME metric is explained in Eq. (2.29) [12]:

$$EME = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} 20\log\left(\frac{Max[I(i,j)]}{Min[I(i,j)]}\right) \qquad (2.29)$$

Where, *I* is the enhanced image, *M* and *N* are the number of blocks on the height and width of the image, respectively, *Max* and *Min* return the maximum and minimum values in a given block.

2.8.2 CNN Model Evaluation Metrics

The evaluation of CNN model or any model in DNN is an important part of proving performance efficiency and to compare with other models and methods to perform a specific task. Usually, using several types of performance metrics is essential to a fairly adjudge on the model [136]. Evaluation metrics for CNN model were determined using a number of model results considering four terms:

TP: “True Positive” correctly predicted for positive class.

TN: “True Negative” correctly predicted for negative class.

FP: “False Positive” incorrectly predicted for positive class.

FN: “False Negative” incorrectly predicted for negative class.

There are many evaluation metrics are used to evaluate CNN models all have values ranged [0,1] and here are some of popularly used metrics.

a) Accuracy (AC): is the percentage between the number of right predictions to the total amount of input examples. The mathematical formula of AC metric is explained in Eq. (2.30) [32]:

$$AC = \frac{TP+TN}{TP+FP+TN+FN} \qquad (2.30)$$

b) Recall (R): represents the sensitivity parameter of the model results. R is defined using Eq. (2.31) [136]:

$$R = \frac{TP}{TP+FN} \qquad (2.31)$$

c) Precision (P): rate of correctly positive predictive among all positive predictive results. Eq. (2.32) explain P formula [137]:

$$P = \frac{TP}{TP+FP} \qquad (2.32)$$

d) F1 score: metric combines P and R in the harmonic mean that is defined using Eq. (2.33) [138]:

$$F1 = 2 \times \frac{P \times R}{P+R} \qquad (2.33)$$

2.8.3 CBIR System Evaluation Metrics

The metrics used to evaluate any information retrieval system including CBIR are basically depending on terms:

Retrieved: number of all images produced for a query in retrieval process.

Relevant: number of images relevant to the query in the test set.

Relevant retrieved: number of images relevant to query and retrieved as results for retrieval system (intersection between relevant and retrieved).

Using these terms, the metrics used to evaluate the CBIR system includes (all values metrics are ranged in [0,1] values) [3]:

a) Precision (Pr): is the ratio between relevant retrieved image and retrieved as defined in Eq. (2.34):

$$Pr = \frac{\text{Relevant retrieved}}{\text{Retrieved}} \qquad (2.34)$$

b) Recall (Rr): is the rate between relevant retrieved images and relevant images that defined in Eq. (2.35):

$$Rr = \frac{\text{Relevant retrieved}}{\text{Relevant}} \qquad (2.35)$$

c) Average Precision (AP): for K ranked retrieval resultant images of one query image, *AP* is the average of precision for each s ranked images (s= 1, 2, 3, …,K) for that query image. *AP* is defined using Eq. (2.36) [102]:

$$AP = \frac{1}{K}\sum_{s=1}^{K} \Pr(s) \times b \qquad (2.36)$$

Where, *K* is the number of ranked images retrieved from the system, *s* is the retrieved image order in ranked list, b is a binary value that refers to the state of image if it relevant or not.

d) Mean Average Precision (mAP): is a standard metric in CBIR. It is calculated for a number of queries Q for K ranked retrieved resultant images from the CBIR. mAP represents the average of AP for many Q images. The formula for mAP is explained in Eq. (2.37) [139]:

$$mAP = \frac{1}{Q}\sum_{d=1}^{Q} AP(d) \qquad (2.37)$$

Chapter Three

The Proposed System

3.1 Introduction

The proposed system constructs to retrieve medical images considering the features extracted in three stages. The two stages are feature extraction and the third is for retrieval stage. The extracted features are collected to be as input to the retrieval process. The extracted features different considering the extraction methods and effect on strength the efficiency of the retrieval system. The dataset images features are extracted and saved in one database in off-line phase. The query (test) image feature is extracted in the online phase and matched with saved database images features to retrieve the most similar and near images to the query one using a similarity metric.

3.2 Description of Proposed System Stages

In general, the proposed system consists of three stages; two stages for features extraction and the third stage for retrieval process, and each stage consists of many steps. Figure (3.1) illustrates the block diagram of the three stages of the proposed system.

Stage 1: Texture feature extraction contains many steps for texture feature extraction using GLCMs matrices and 1D CNN model. This stage consists of the following steps:

Step 1: Quantization: A quantization method for gray medical image is proposed to reduce number of image gray levels. The Divide and Conquer strategy is adapted to divide the gray levels range into non-uniform periods and select the optimum threshold value considering a heuristic function.

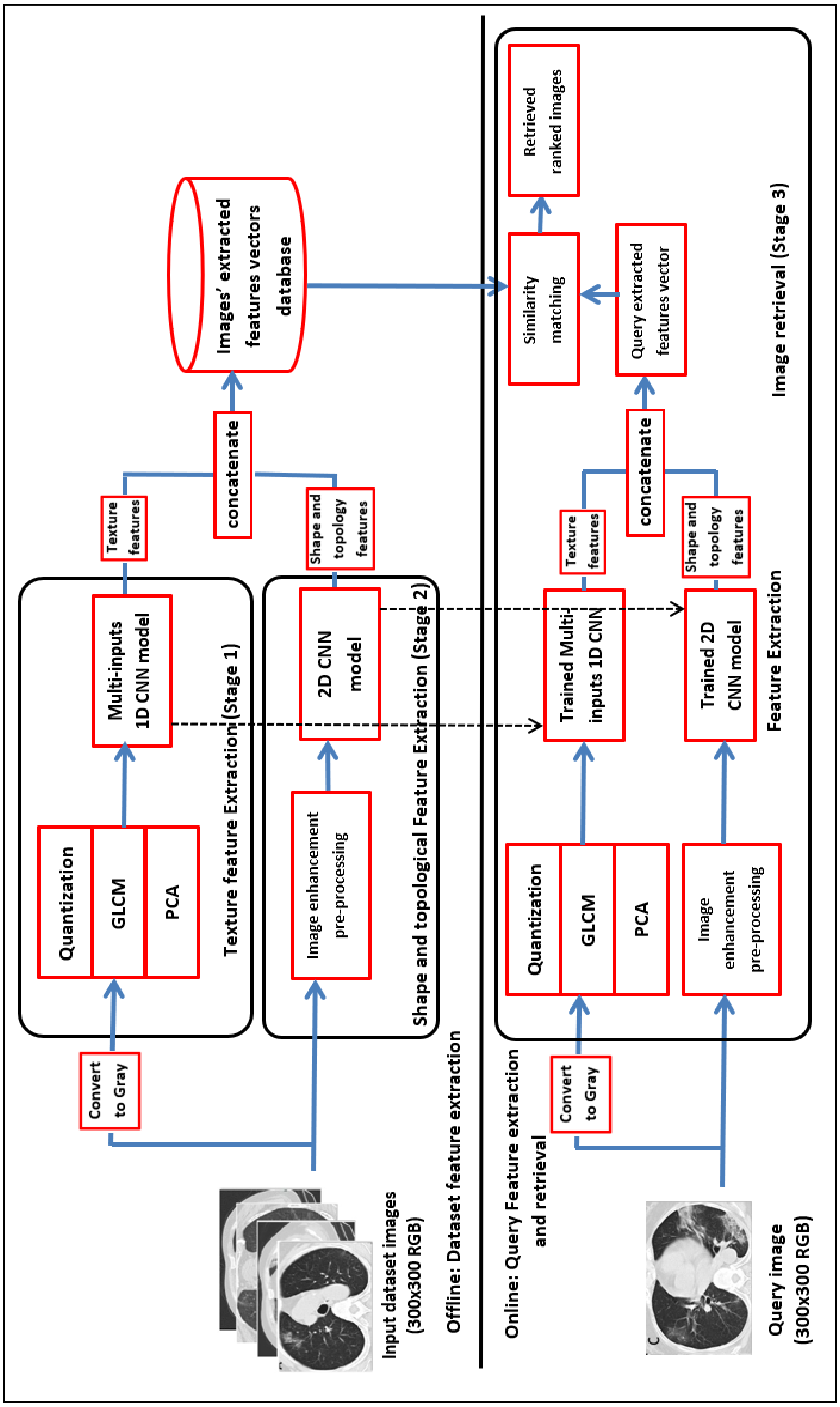

Input dataset images
(300x300 RGB)
Convert to Gray
Quantization
GLCM
PCA
Multi-inputs 1D CNN model
Texture feature Extraction (Stage 1)
Texture features
Image enhancement pre-processing
2D CNN model
Shape and topological Feature Extraction (Stage 2)
Shape and topology features
concatenate
Images' extracted features vectors database
Offline: Dataset feature extraction
Online: Query Feature extraction and retrieval
Query image
(300x300 RGB)
Convert to Gray
Quantization
GLCM
PCA
Image enhancement pre-processing
Trained Multi-inputs 1D CNN
Trained 2D CNN model
Feature Extraction
Texture features
Shape and topology features
concatenate
Query extracted features vector
Similarity matching
Retrieved ranked images
Image retrieval (Stage 3)

Figure (3.1) Block diagram of the proposed system

Step 2: Grey level Co-occurrence matrix (GLCM): GLCMs matrices are generated at three different distances and different directions for three quantized versions for each image. The generated GLCMs matrices are flattened to be vectors for each quantized image version used.

Step 3: Principal Component Analysis (PCA): whereas, the GLCMs matrices have a big size to be as an input to the next stage (1D CNN model) and GLCM natural is sparse, therefore, it's useful to use one of the dimensionality reduction methods to achieve more efficient GLCMs vectors.

Step 4: Multi inputs 1D CNN model: To obtain the final extracted features from images in this stage, a proposed multi inputs 1D CNN classification model architecture is built. The input layer of the model contains three pipelines according to this three GLCMs vectors produced from the previous step.

Stage 2: Shape and topology feature extraction: consists of many steps constructed to extract special type of features including local shape and pixels topological features. This stage contains the following steps:

Step1: Preprocessing: image enhancement process is adapted to enhance and increase the efficiency of extracted features.

Step 2: Proposed 2D constrained convolution layer (CCL): To extract local shape and topological features of the images, especially medical images, a 2D constrained convolution layer (CCL) is proposed. CCL reduces the number of trainable parameters (weights) in layer kernels considering desired percentage.

Step 3: 2D CNN model: a 2D CNN classification model architecture including CCL layer as well as traditional convolution layer (TCL) is proposed.

Stage 3: Image retrieval: contains a retrieval process that includes extracting images features and matching process. This stage consists of the following steps:

Step 1: Dataset feature extraction: features of images dataset are extracted using previous stages in offline and are concatenated to store as vectors in one database to represent the images features.

Step 2: Query image feature extraction: features are extracted to query image in online using the same stages used to extract the dataset features.

Step 3: Matching and retrieval: contains a similarity matching between stored dataset images features and query image features. Then, return the most similar ranked images to the query image considering a similarity metric.

3.3 Proposed System Construction

The system is implemented in two main phases; offline phase and online phase. These two phases include preparing datasets, learning, and testing to the proposed feature extraction models. Also, these phases include extraction of the features for whole dataset in offline phase and for query image in online phase. Algorithm (3.1) illustrates implementation steps of the proposed system. The proposed system's two main phases can be explained as following:

1. **Offline phase:** This phase is implemented just in the first run for the system and consists of the following sequence steps:

- Implement the first stage (Texture feature extraction) that includes:
    1. quantization process for all dataset (training and testing sets).
    2. Generate GLCMs matrices and prepare them for the 1D CNN model.
    3. Train, test, and save the proposed 1D CNN model (including feature extraction and classification parts).

4. Separate the classification part from the feature extraction part in the trained model.

- Implement second stage (shape and topology feature extraction) that includes:
    1. Preprocessing for all dataset (training and testing sets) using enhancement method.
    2. Train, test, and save the proposed 2D CNN model (including features extraction and classification parts).
    3. Separate the classification part from the feature extraction part in the trained model.
- Extract features for all dataset images using features extraction part of the trained 1D CNN model.
- Extract features for all dataset images using feature extraction part of the trained 2D CNN model.
- Concatenate the two vectors of features for each image in dataset.
- Store the total extracted features as one database.

2. Online phase:

This phase is implemented for image retrieval process and includes the following sequence steps:

- The system asks the user to enter a query image.
- The features of query image are extracted using first and second stages of the proposed system in the same manner that used to extract the dataset images features and have the same resultant dimension vector.
- Similarity matching metric is used to retrieve the most similar images to the query one.
- Ranking the resultant similar images from most similar to far.
- Retrieve and display to the user that most N images are similar to the query image.

```
Algorithm (3.1): Construction of The Proposed System
Input: D: Images dataset, Q: query image
Output: N: set of images in D are most similar to Q
Begin
    For each image I in D Do
        Generate three versions of quantized image of I with 16, 32, and
        64 gray levels using proposed quantization method.
        For each quantized image Do
            Generate 24 GLCMs matrices in three distance and different
            angle directions.
            Flatten the 24 GLCMs matrices to be 24 vectors
            Reduce GLCMs vectors to one vector using PCA
        End for
    End for
    Train, test, and save 1D CNN model using classification function.
    Separate the classification part in trained model from feature
    extraction part.
    Extract features for all D images in V1.
    Enhance each image in D with the CLAHE enhancement method.
    Train, test, and save 2D CNN model using classification function.
    Separate classification part from feature extraction part in trained
    model
    Extract features for all D images in V2.
    Concatenate extracted features V1 and V2 and save in one database.
    Extract features of Q in same manner used to extract dataset features.
    Match Similarity between Q extracted features and all saved database
    Retrieve N images which features are more similar to Q features
    Display the ranked N similar images to Q
End
```

3.4 Texture Feature Extraction Stage

The first stage in proposed system includes the extraction to the texture features. Figure (3.2) shows the block diagram for the first stage of the proposed system. The input for this stage is dataset of gray images after converting it from color images. The output of this stage are vectors with 4 values of texture extracted features for entered dataset images; one vector for each image. This stage consists of the following steps:

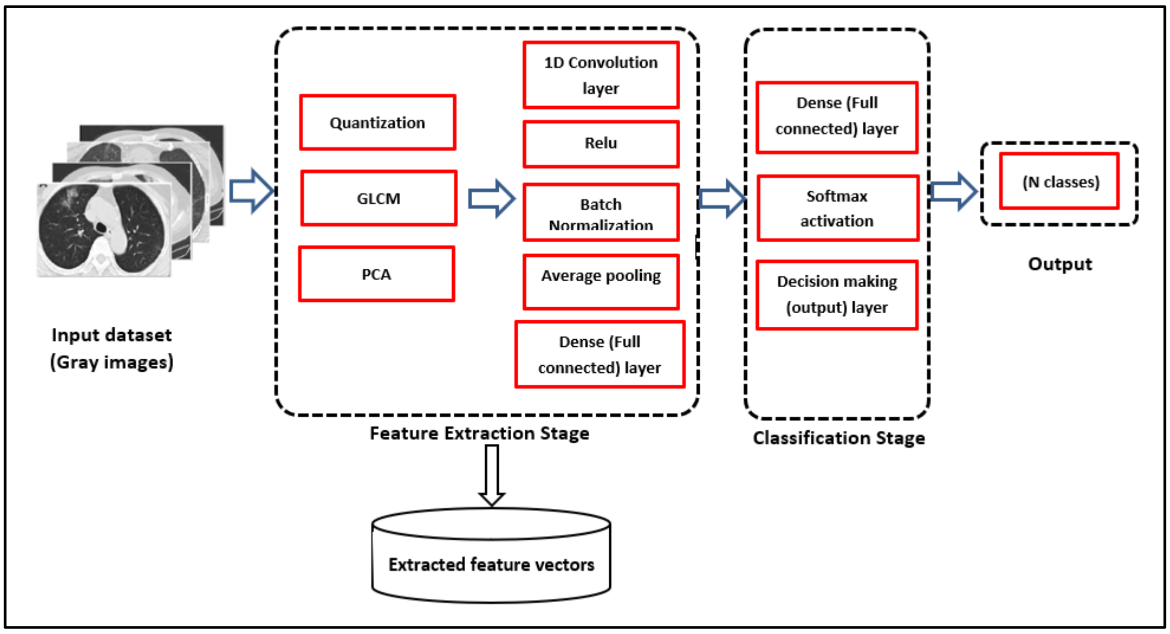


Figure (3.2) Block diagram of the texture feature extraction stage

3.4.1 Quantization Step

The first step for the first stage of proposed system is a proposed quantization method. Figure (3.3) illustrates the quantization step. Each image in dataset is converted to three different quantized images with 16, 32, and 64 gray levels.

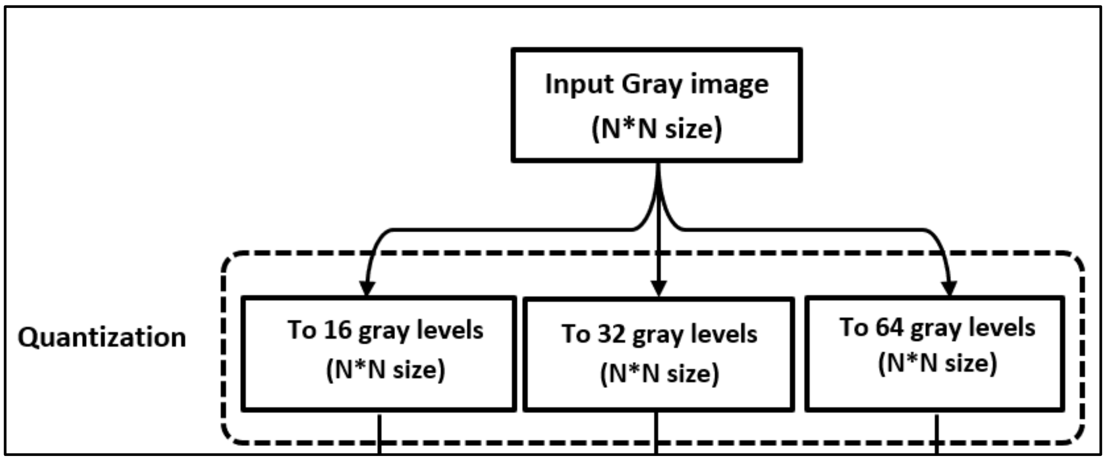


Figure (3.3) Quantization step

The proposed method includes quantizing the gray image by reducing the number of gray levels from 256 gray levels to the desired number of gray levels. The quantized gray levels are generated by finding threshold values that divided the range of gray levels and are selected depending on optimization criteria.

The main idea of the proposed method is motivated by the uniform quantization of gray images that reduces the number of gray levels of the image by dividing the range of gray levels into equal periods. The resulted gray levels from the uniform process are distributed uniformly and may be not the real gray levels in the image.

The proposed method considers the reduction of gray levels in the image will depend on non-uniformed quantization that divided the actual range of gray levels into unequal periods or regions split by threshold values selected depending on a heuristic function that optimizes each threshold value before selecting it. To find these threshold values, a greedy programming method that uses Divide and Conquer strategy has been adopted.

First, we start with generating a quantized image with one gray level only (higher gray level in image) then we generate a quantized image with 2 gray levels (one threshold value) then a quantized image with 3 gray levels (two threshold values) etc. The chosen threshold values will be the values that give an optimal division to the image gray levels range. The optimal dividing is depending on which threshold value (gray level) in the image gray level range gets a maximum PSNR between the original image and quantized image that resulted after adding the chosen threshold dividing values.

Also, the proposed method can be considered a multi-thresholding problem for the image histogram. Let I the original image, the range of gray levels is [L, H] (e.g. [4, 243]) and the desired number of threshold values is M (1 to $2^k - 1$), then the threshold values will be (th_1, th_2, th_3,…, th_m), where th_1=L, th_m= H and other th_i are gray level values selected from image gray levels and distributed in range [L, H] (k is number of bit required to represent each gray level).

The proposed method selects the threshold values using a hieratical division wherein each level. The procedure of proposed method is applied recursively on each division part. The block diagram that introduces the method is illustrated in Figure (3.4). The proposed procedure contains the following steps:

Step 1: start with a quantized image with H gray level only.

Step 2: select the threshold value Th from gray levels range [L, H] that represents the division position for the origin image gray levels range. We test all gray levels from L to H in the original image and find the gray level (Th) that after applying it to quantized image gets higher PSNR compared with the original image.

Step 3: add the selected threshold value Th to the quantized image to be have two gray levels [Th, H]. Where, all the pixels that have gray levels less than or equal to Th will have Th value, otherwise will have H gray level.

Step 4: If the number of threshold values is equal to desired number of quantized gray levels M then stop. Else, repeat steps 2 and step 3 with a new quantized image after adding (Th) and two new gray level ranges; [L, Th-1] and [Th, H] to find two other threshold values.

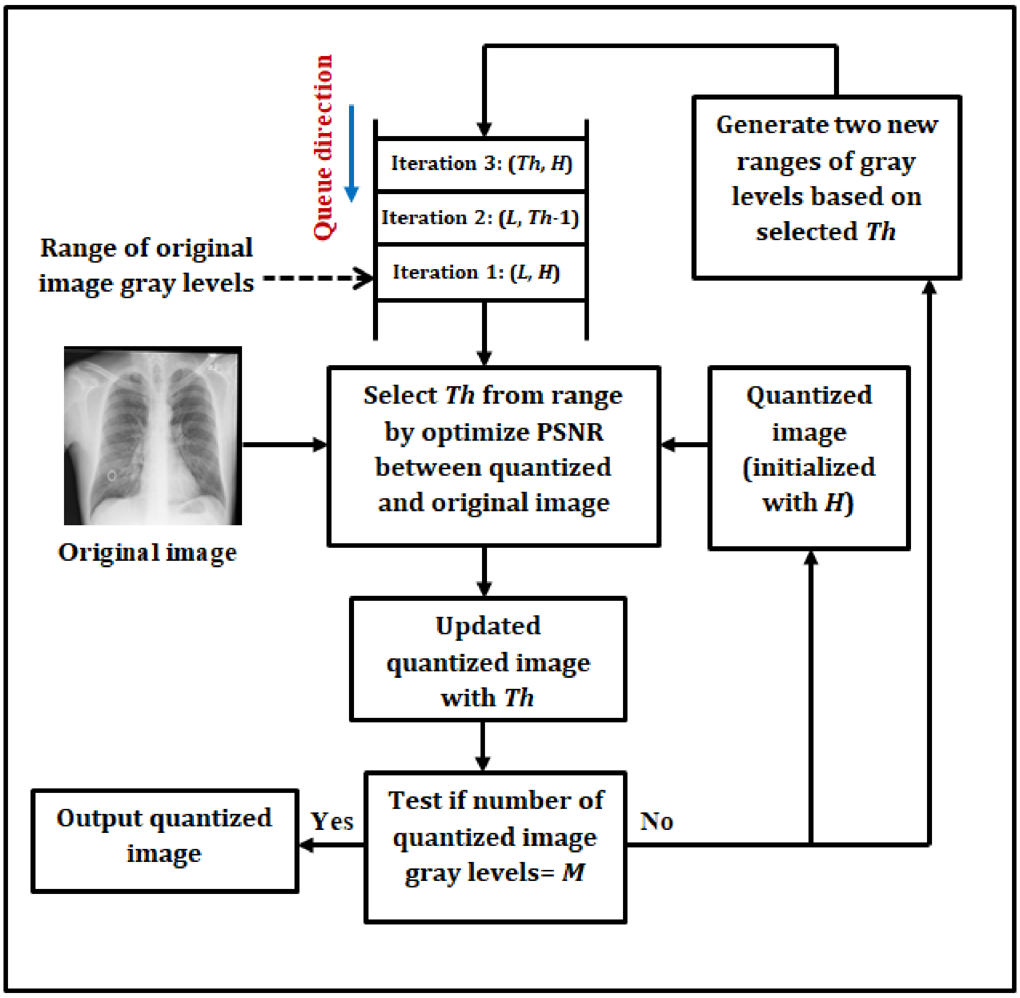

Queue direction
Iteration 3: (Th, H)
Iteration 2: (L, Th-1)
Iteration 1: (L, H)
Range of original image gray levels
Generate two new ranges of gray levels based on selected Th
Original image
Select Th from range by optimize PSNR between quantized and original image
Quantized image (initialized with H)
Updated quantized image with Th
Output quantized image
Yes
Test if number of quantized image gray levels= M
No

Figure (3.4) Block diagram of the proposed quantization method

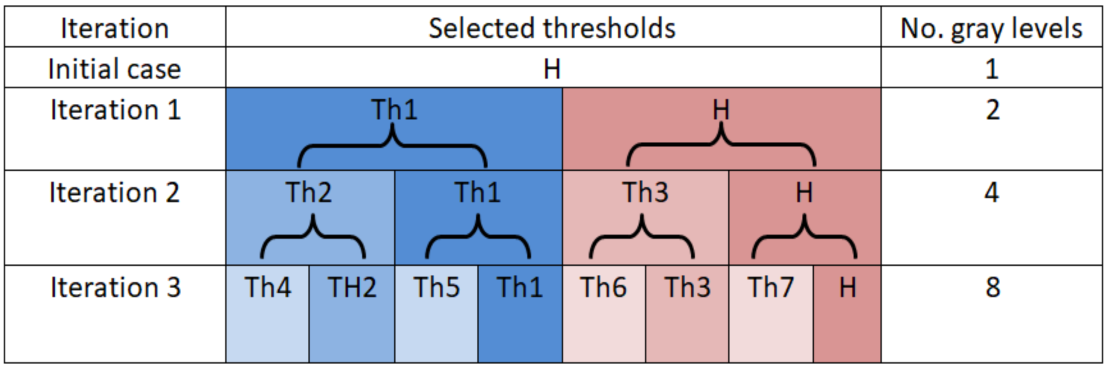
Algorithm (3.2) shows the proposed quantization method steps more accurately. Figure (3.5) illustrates the hierarchy of the first three iterations of finding the first eight threshold values that generate the quantized image from 2 to 8 gray levels.

Algorithm (3.2) Proposed Quantization Method

Inputs: A: Gray image, **M**: desired number of quantized gray level

Output: QA quantized gray image with **M** gray levels

```
Begin
    Create queue (2D array) LH (each row contains range of gray level)
    L ⟵ minimum gray level in A
    H⟵ maximum gay level in A
    LH [0,0] ⟵ H
    LH [0,1] ⟵ L
    Initialize inc with 1 // start count of HL queue
    Initialize outc with 0 // end count of HL queue
    Initialize QA image with H gray level and same dimension of A
    Initialize U with 1 // number of QA gray levels
    While (length (U) < M) Do
        H⟵ LH[outc, 0]
        L ⟵ LH[outc, 1]
        outc ⟵ outc + 1
        Create A2 copy of QA
        MaxIndex ⟵ H // the new threshold value
        For i =L to H Do
            For m=1 to height of A Do
                For n=1 to width of A Do
                    A2[m,n]⟵ i if ((A(m,n)>= L) &(A(m,n)<= i))
                    A2[m,n]⟵H if ((A(m,n) > i) &(A(m,n) <= H))
                End for
            End for
            Compute PSNR P1 between A and A2
            Compute PSNR P2 between A and QA
            if (P1 > P2) Then
                QA ⟵ A2
                MaxIndex = i
            End if
        End for
        if (MaxIndex ≠ H) Then
            LH[inc, 0] ⟵ L        // add new range [L, MaxIndex] in queue
            LH[inc, 1] ⟵ MaxIndex
            inc ⟵ inc + 1
            LH[inc, 0] ← MaxIndex+1 // add new range [MaxIndex+1, H]
            LH[inc, 1] ←  H              // in queue
            inc ← inc + 1
        End if
        U ← number of QA gray levels
    End while
    Return QA
End
```



Iteration	Selected thresholds								No. gray levels
Initial case	H								1
Iteration 1	Th1				H				2
Iteration 2	Th2		Th1		Th3		H		4
Iteration 3	Th4	TH2	Th5	Th1	Th6	Th3	Th7	H	8

Figure (3.5) The hierarchy of the first three iterations for proposed method

To illustrate the method, an X-Ray image as an example to apply the proposed method is used. The original image and the histogram of it with the first eight quantized gray levels compared with uniform division of histogram are illustrated in Figure (3.6). Worth noting, the method is not used a histogram but just to illustrate the distribution of image gray levels and the position of selected threshold values.

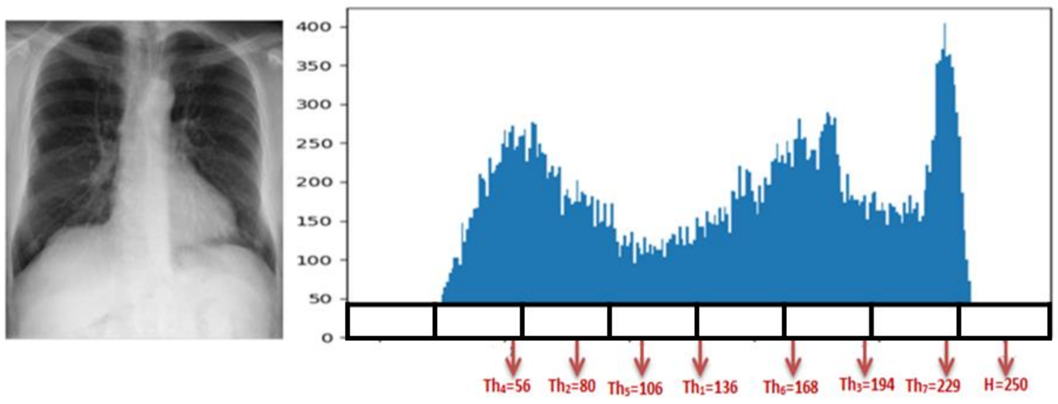


Figure (3.6) The original image and histogram of the x-ray with the first quantized eight gray levels compared with uniform division of histogram

Each grey image in experiment datasets is inserted into the three quantized quantization processes and resulted in three quantized images with 16, 32, and 64 grey levels values. Each version of the resulting images has features that are differ depending on the number of grey levels.

3.4.2 Grey Level Co-occurrence Matrix (GLCM) Step

This step includes extraction for texture statistical features depending on GLCM method. There are many different features that can be extracted considering GLCM parameters diversity due to the difference in context and the statistical characteristics of the image texture.

The GLCM parameters involve distance and direction. Distance is the offset between pixels i and j in the image and direction is the angle of that offset distance. In the GLCM step, we find many GLCM matrices in three different distances and different angle directions in each distance.

These matrices are considered second-order statistical features that can find a relation between two pixels almost be neighbours in the image. The power and efficiency as well as the simplicity of GLCM matrices in the way we use are the reason behind using this method instead of other methods. For each distance d, the amount of increase between angle directions is computed as shown in Eq. (3.1):

$$Inc\_Angle = \frac{360}{d \times 8} \qquad (3.1)$$

As is known, GLCM is a square matrix its dimensions depend on the number of gray levels of the input image. For that, the dimension of generated GLCM matrices depends on the three different quantized images with 16, 32, and 64 gray levels constructed in the previous step (quantization step). Figure (3.7) illustrates the construction of GLCM matrices for one quantized image with N gray levels values.

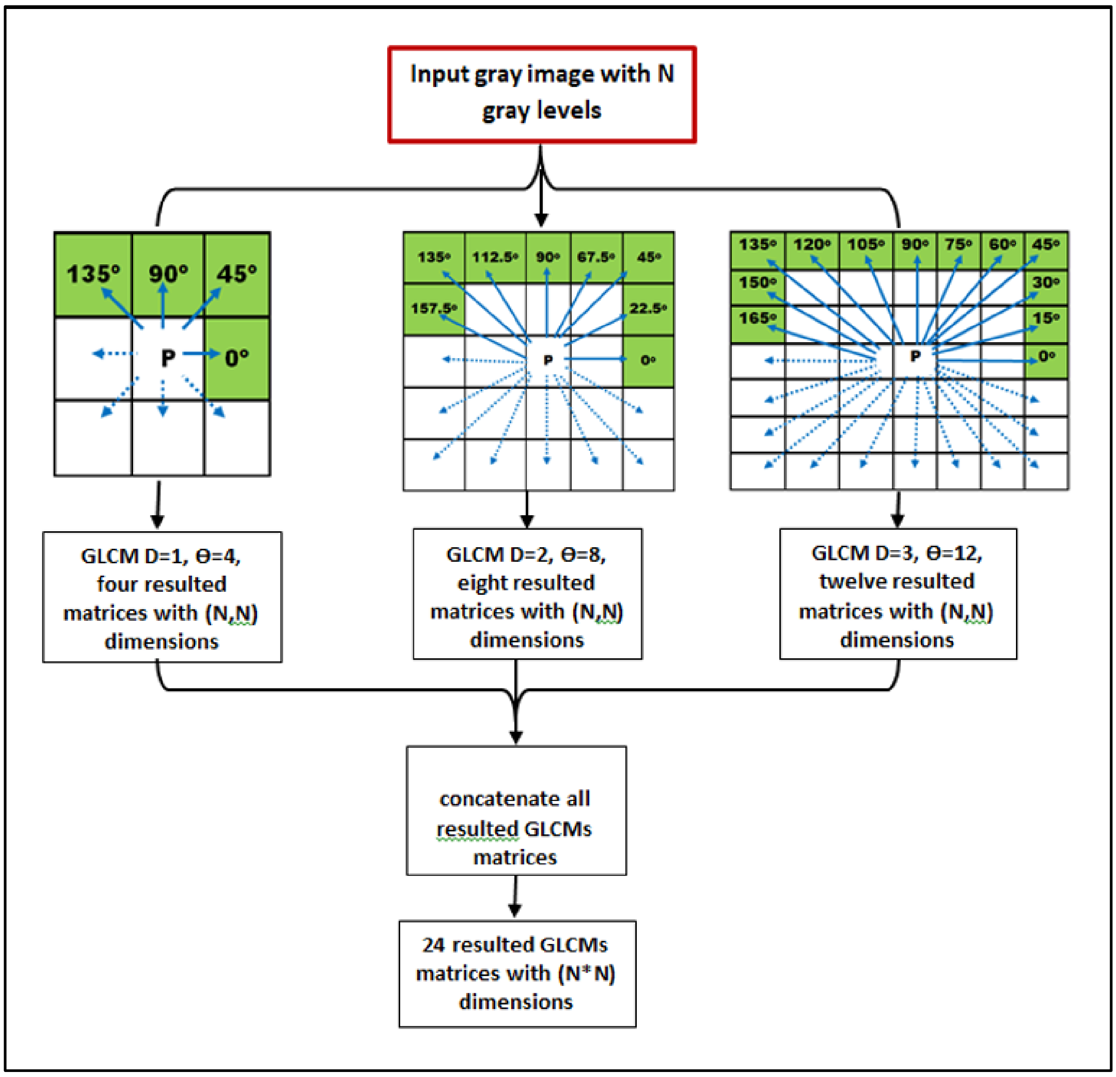

Input gray image with N gray levels
135° 90° 45°
P 0°
135° 112.5° 90° 67.5° 45°
157.5° 22.5°
P 0°
135° 120° 105° 90° 75° 60° 45°
150° 30°
165° 15°
P 0°
GLCM D=1, Ө=4, four resulted matrices with (N,N) dimensions
GLCM D=2, Ө=8, eight resulted matrices with (N,N) dimensions
GLCM D=3, Ө=12, twelve resulted matrices with (N,N) dimensions
concatenate all resulted GLCMs matrices
24 resulted GLCMs matrices with (N*N) dimensions

Figure (3.7) GLCMs matrices generation for one quantized image with N gray levels

As mentioned, each GLCM matrix is generated based on the context that depends on the distance between two pixels and the direction angle. Figure (3.8) illustrates contexts extracted using three distances and different directions (angles) for constructing GLCM matrices in this work.

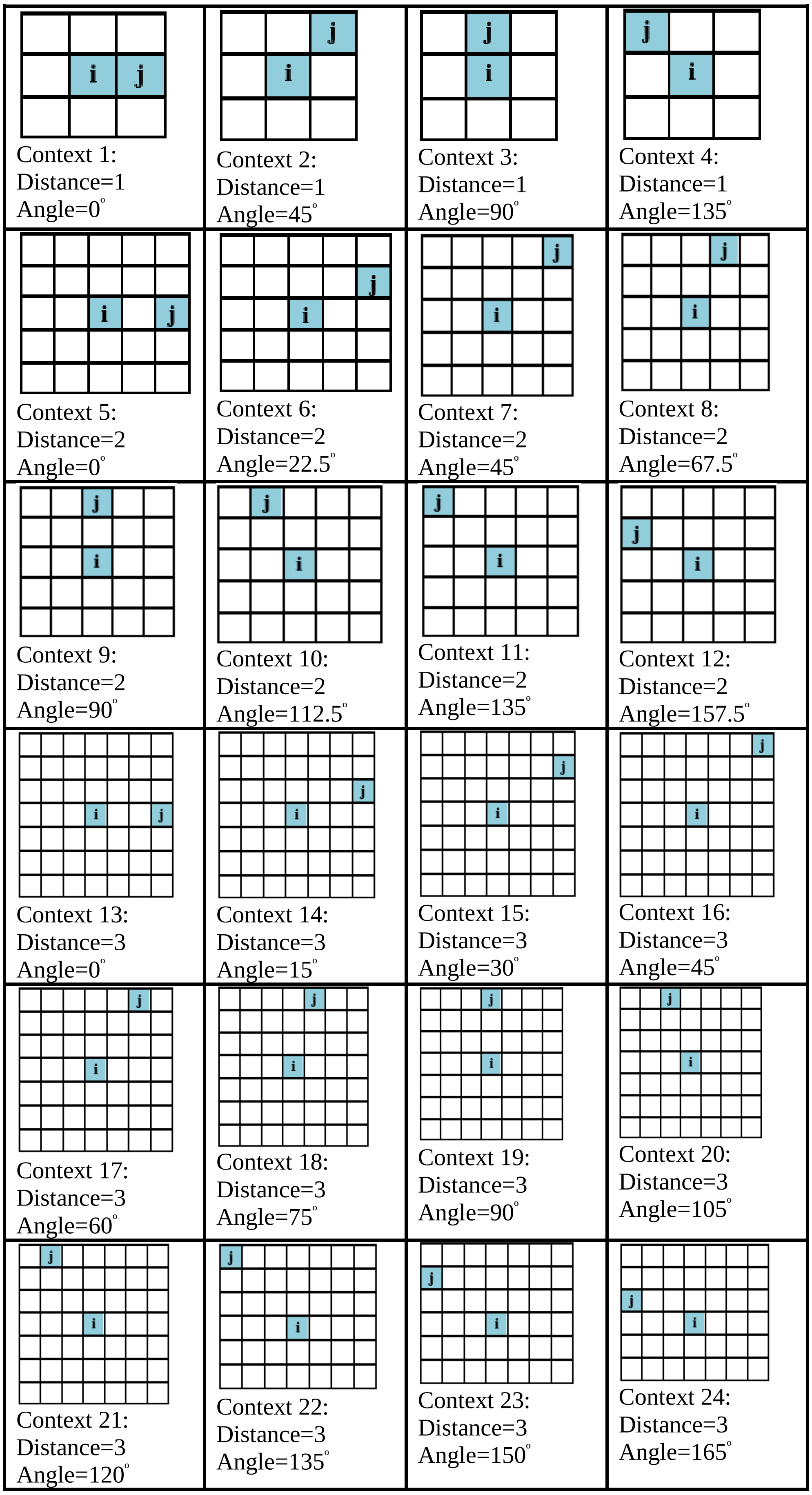

Context 1: Distance=1 Angle=0°
Context 2: Distance=1 Angle=45°
Context 3: Distance=1 Angle=90°
Context 4: Distance=1 Angle=135°
Context 5: Distance=2 Angle=0°
Context 6: Distance=2 Angle=22.5°
Context 7: Distance=2 Angle=45°
Context 8: Distance=2 Angle=67.5°
Context 9: Distance=2 Angle=90°
Context 10: Distance=2 Angle=112.5°
Context 11: Distance=2 Angle=135°
Context 12: Distance=2 Angle=157.5°
Context 13: Distance=3 Angle=0°
Context 14: Distance=3 Angle=15°
Context 15: Distance=3 Angle=30°
Context 16: Distance=3 Angle=45°
Context 17: Distance=3 Angle=60°
Context 18: Distance=3 Angle=75°
Context 19: Distance=3 Angle=90°
Context 20: Distance=3 Angle=105°
Context 21: Distance=3 Angle=120°
Context 22: Distance=3 Angle=135°
Context 23: Distance=3 Angle=150°
Context 24: Distance=3 Angle=165°
i
j

Figure (3.8) 24 contexts for GLCM matrices based on distances and angles

GLCMs matrices generation process consists of the following steps:

1. Four GLCMs are generated by finding the occurrence of pixel P with green pixels in Figure (3.7) with distance d=1 and directions ϴ= (0º, 45º, 90º, 135º). The angles for directions are generated using Eq. (3.1). These matrices are stacked in one 3D matrix with dimensions (4×N×N). Where N is the number of image’s gray levels.
2. Eight GLCMs are constructed with d = 2 and ϴ= (0º, 22.5º, 45º, 67.5º, 90º, 112.5º, 135º, 157.5º) and then stacked in one 3D matrix with dimensions (8×N×N).
3. Generate 12 GLCMs with d=3 and ϴ= (0º, 15º, 30º, 45º, 60º, 75º, 90º, 105º, 120º, 135º, 150º, 165º) and grouped in one 3D matrix with dimensions (12×N×N).
4. After GLCMs generation, we need to prepare the resulted data for the next step (PCA). For that, the resulted matrices for each quantized image are flattened and concatenated to produce a 1D matrix. This step is done by merging and reshaping the GLCM matrices (4×N×N), (8×N×N), and (12×N×N) to a 2D matrix with length (4+8+12) that is (24×N×N).
5. Repeat three previous steps for three different quantized gray images resulted from previous step (quantization) with 16, 32, and 64 gray levels. These three quantized images are entered into GLCMs step to produce (24×16×16), (24×32×32), and (24×64×64), respectively. The resulted matrices are merged and flattened as illustrated in Figure (3.9) to produce 2D matrices with 2D dimensions (24×256), (24×1024), and (24×4096), respectively.

Algorithm (3.3) shows the GLCMs matrices generation for different distance and different angle directions. As illustrated in Algorithm (3.3), the whole procedure is repeated for three quantized versions of medical image.

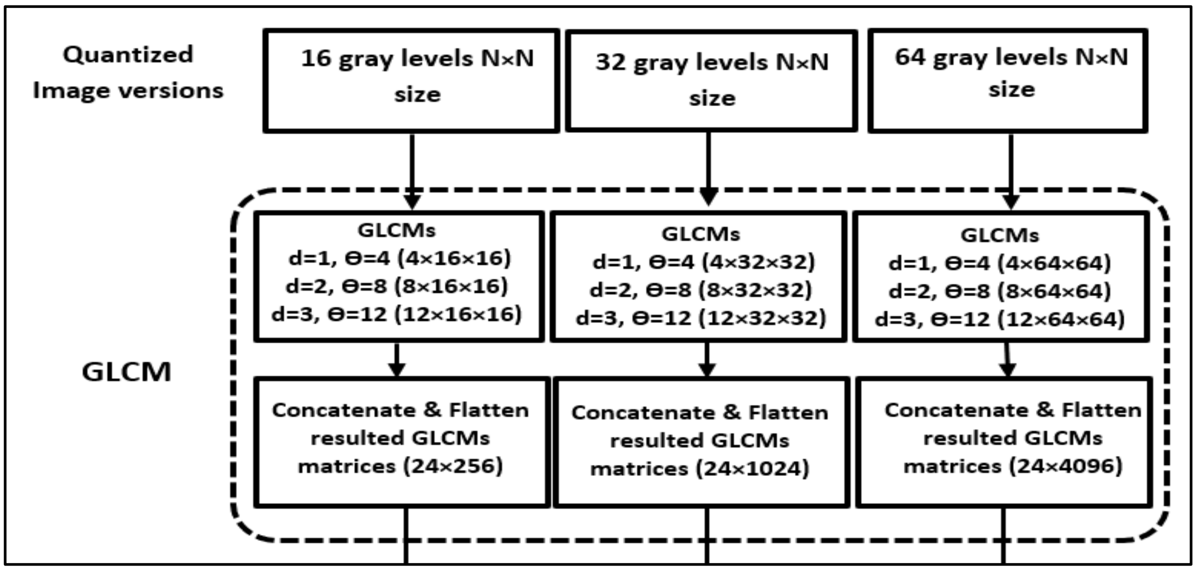


Figure (3.9) Concatenating and flattening GLCMs matrices

Algorithm (3.3): Creation of GLCMs Matrices

Input: *QA* Quantized medical gray image with *N* gray levels.

Output: *glcms* 2D GLCMs for 3 different distances and different angles

```
Begin
   Create glcms as an empty 2D array with 24, N×N size
   For d=1 to 3 Do // 3 distances to compute GLCMs
      Inc_Angle = 360 / d × 8 // Find angle increment amount for d using Eq. (3.1)
      Θ = 0
      For k=1 to d×4 Do // d×4 is a number of angles for current distance d
         For i=1 to N Do    //  N×N is GLCM dimension
            For j=1 to N Do
               G (i, j)= i and j context occurrences in QA image with d and Θ
                         using Eq. (2.1)
            End for
         End for
         GN = apply normalization for G using Eq. (2.2)
         GNF=flatten GN to 1D vector with N×N length by concatenating rows
         Add GNF to glcms as a row with index k
         Θ = Θ +Inc_Angle
      End for // Angles for
   End for // distance for
   Return glcms
End
```

3.4.3 Principal Component Analysis (PCA) Step

It is beneficial and efficient to reduce the matrices dimensions that will be as an input layer to the next stage (1D CNN) and decrease the computational complexity as well as get a more effective feature extraction process. Figure (3.10) illustrates the PCA step for reducing GLCMs matrices.

As shown in the previous step, the dimension of resultant 2D GLCMs matrices is (24×16×16), (24×32×32), and (24×64×64) or (24×256), (24×1024), and (24×4096). These matrices have high dimensions to be an input layer to the 1D CNN (next feature extraction stage). Also, the natural of GLCM is a sparse matrix that contains a lot of zero values. Thus, it is very useful and efficient to use one of the dimension reduction methods to reduce the matrices dimensions that will be as an input layer to the CNN model and decrease the computational complexity as well as get a more effective feature extraction process.

In this step, PCA explained in Section (2.7) is used as an efficient dimension reduction method to extract the most significant values of data and reduce the 2D GLCMs matrices. Each GLCM matrix with (24×N×N) dimension is entered into the PCA procedure and one principal component is selected. The output of the PCA procedure is one vector with (1×N×N) dimension. For three 2D GLCMs matrices (24×256), (24×1024), and (24×4096), the resultant one principal component will be 1D vectors with (1×256), (1×1024), and (1×4096) dimensions, respectively.

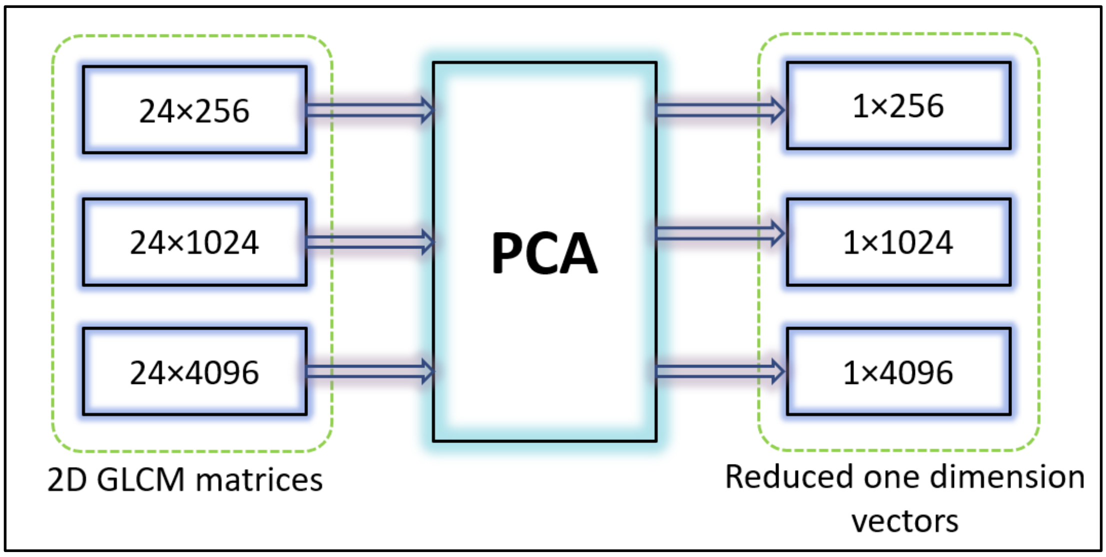


Figure (3.10) Principal Component Analysis (PCA) step to reduce GLCM matrices dimension

3.4.4 Multi Inputs 1D CNN Model Building Step

In this stage, feature extraction for reduced GLCMs matrices resulted from previous step processes is implemented by proposing a new multi inputs 1D CNN model. The main objective of the proposed model is to extract the most important features from three different raw reduced GLCMs vectors. The proposed CNN model consists of two stages; The first is features extraction stage to extract features from reduced GLCM vectors and the second is classification stage that is used for training and extracted features evaluation.

The proposed model contains three pipelines of cascading different types of layers according to the three inputs GLCM vectors. Layers model are used to do different functions depending on the particular objective of each layer.

The proposed architecture of each pipeline in the model depends on the input GLCM vector that represents the input layer to the model and its dimension. Figure (3.11) illustrates the model architecture and the dimension depth of data in each layer.

Since the input layers are 1D vectors, there is very effective in this model to use 1D Convolution (1D Conv) layers instead of 2D convolution layers. In the proposed 1D CNN model architecture, all the 1D CNN layers have:

- Convolution function with filter size (3×1) represents the trainable parameters of the layers (weights).
- Followed by non-linear activation function that is represented by Rectified Linear Units (ReLU) function.
- Then batches normalization layer.

The multi inputs 1D CNN model architecture layers are constructed in three pipelines explained in the following steps:

1. First pipeline: For each image in dataset, a (1×256) vector resulted from previous steps is entered into the first pipeline of the model as input layer. Then passed through one step of convolution represented by two 1D convolution layers with 64 filters to produce 64 feature maps with dimension 256. This step is followed by one average pooling layer with (3×1) pool size to be 64 vectors with dimension 85. Then passed to fattening and dense layer to produce one vector with dimension 1000 features.
2. Second pipeline: For each image in dataset, a (1×1024) vector resulted from previous steps is entered into the second pipeline of the model as input layer. Then passed through two steps of convolution; the first is represented by two 1D convolution layers with 32 filters to produce 32 feature maps with dimension 1024 followed by one average pooling layer with (3×1) pool size to be 32 vectors with dimension 341. The second is represented by two 1D convolution layers with 64 filters to produce 64 feature maps with dimension 341. This step is followed by one average pooling layer with (3×1) pool size to be 64 vectors with dimension 113.

Then passed to fattening and dense layers to produce one vector with dimension 1000 features.

3. Third pipeline: For each image in dataset, a (1×4096) vector resulted from previous steps is entered into the third pipeline of the model as input layer. Then passed through three steps of convolution; the first is represented by two 1D convolution layers with 32 filters to produce 32 feature maps with dimension 4096 followed by one average pooling layer with (3×1) pool size to be 32 vectors with dimension 1365. The second is represented by two 1D convolution layers with 64 filters to produce 64 feature maps with dimension 1365. This step is followed by one average pooling layer with (3×1) pool size to be 64 vectors with dimension 455. The third is represented by two 1D convolution layers with 64 filters to produce 64 feature maps with dimension 455. This step is followed by one average pooling layer with (3×1) pool size to be 64 vectors with dimension 151. Then passed to fattening and dense layers to produce one vector with dimension 1000 features.
4. Combine the three features resulted from three pipelines by concatenating them to produce one vector with dimension 3000. After that, passes through a dense layer to produce one vector with dimension of 1000 features, then passes through dense layer to produce one vector with 4 features. In this step, feature extraction function is finished and the resulted vector represents the extracted features of the current image.
5. To train and test the proposed model and thus get efficient extracted features, classification function is used on the resulted concatenated vector. For this reason, dense fully connected layer with 2 neurons and Softmax layers are used for classification purposes. The Softmax layer is considered as an activation function that used a

normalized exponential function to normalize the network output and find a probability distribution over predicted the output number of classes considering the used dataset.

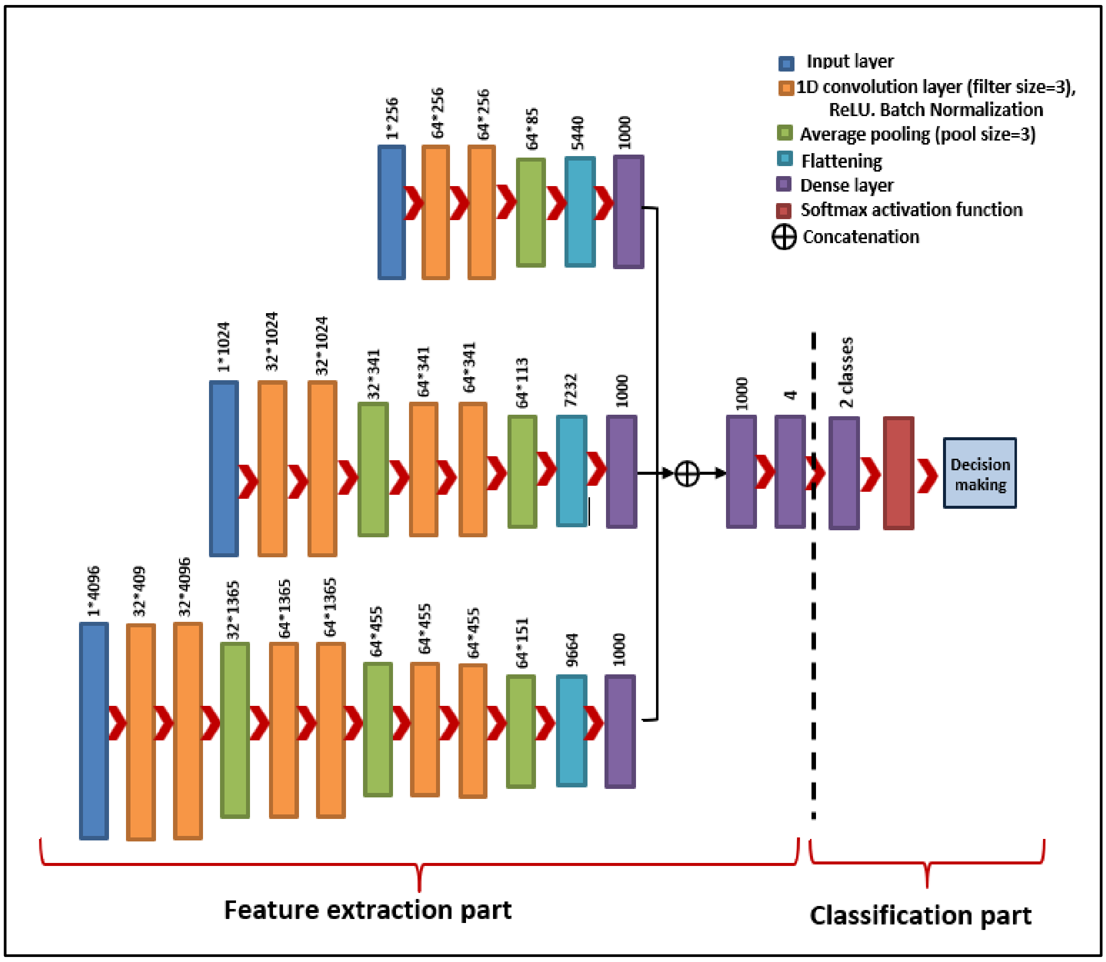


Figure (3.11) Multi inputs 1 D CNN model architecture for GLCM features extraction

3.5 Shape and Topology Feature Extraction Stage

This stage introduces a development in CNN work by proposing a convolution layer with constrained weights that focuses on a special type of features such as local shape and topological features of sub image. Furthermore, this stage proposed an architecture for the DNN model including the proposed convolution layers as well as the traditional CNNs layers. The CNN model is preceded by enhancement as preprocessing step to increase the model performance and accuracy of the extracted features. The input to this stage is colored RGB images dataset and the output vectors of images extracted features; one vector for each image with 4 features. Figure (3.12) illustrates block diagram for shape and topology features extraction stage. This section introduces a description and explanation of the proposed CNN layer as well as the proposed CNN model architecture applied for shape and topology feature extraction and use classification for the model learning purpose.

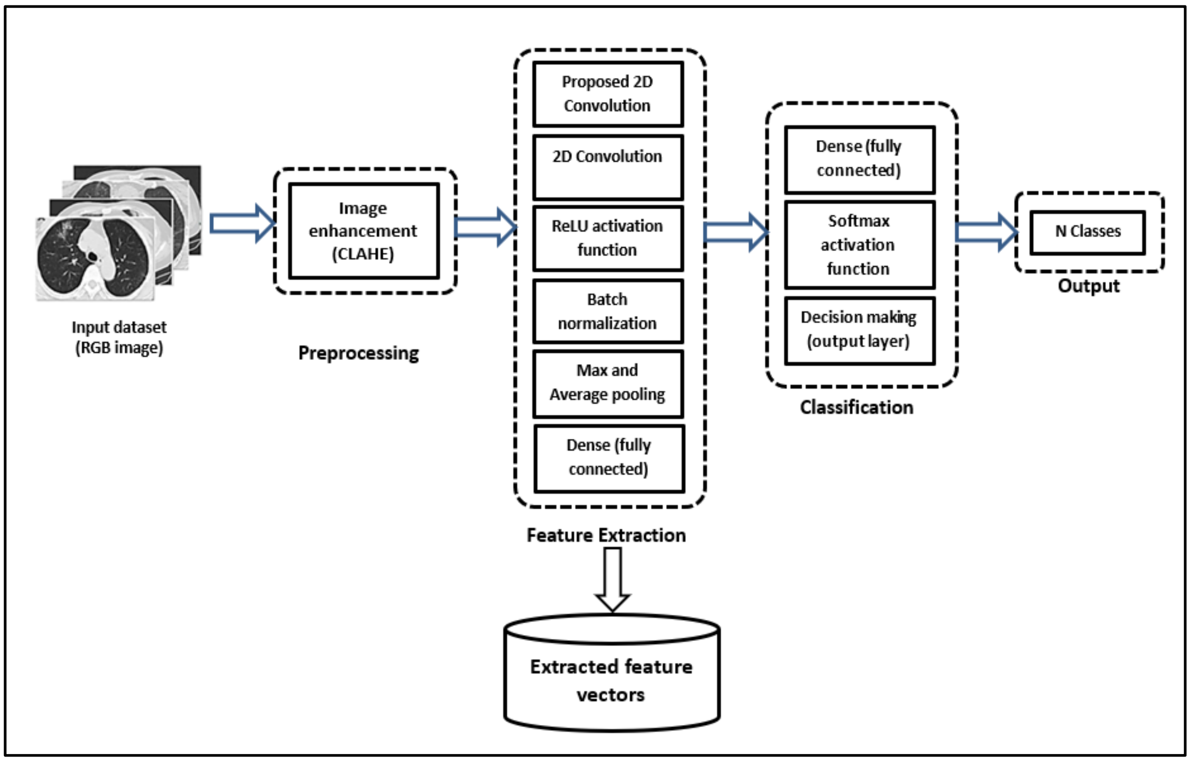


Figure (3.12) Block diagram of shape and topology feature extraction stage

3.5.1 Preprocessing Step: Image Enhancement

Preprocessing step includes an image enhancement process to improve the CNN model performance and increase the classification accuracy results, thus, extracting efficient and descriptive images features. The enhancement method is applied on the three implemented datasets in their two parts: train and test. Contrast Limited Adaptive Histogram Equalization (CLAHE), explained in Section (2.6), is one of efficient indirect contrast image enhancement methods that work well especially with medical CT scan and X-ray images to increase the contrast and appearing details. CLAHE method is applied to all RGB images in dataset.

3.5.2 Constrained Weights 2D Convolution Layer (CCL)

The convolution layer is the base construction block in the deep neural network architecture. The convolution process includes a dot product between a sub-image matrix (window) also called the receptive field and another matrix containing values that represent the trainable parameters (weights) called kernel or filter.

As explained in Section (2.3.1.a), in the Traditional Convolution Layer (TCL), the input of the layer (feature map that is the output of the previous layer) is X×X×A where A is number of features maps each one has X×X size. Also, each kernel is 3D dimension with a square matrix have a size of k×k (e.g. k= 3, 5, 7…) and depth A (same number of feature maps in current layer). That means each kernel has (k, k, A) dimensions containing k×k×A trainable parameters. Also, number of kernels with size k×k×A that is specified to the layer is denoted by F; where F represents the layer output (number of feature maps produced from current layer). For that, the total number of weights for layer is k×k×A×F, and thus, the output of the layer will be X×X×F (consider the hyperparameters are the same padding and one stride).

CCL suggests diverse distribution for weights of kernels in each convolution layer rather than the traditional square distribution. Figure (3.13) illustrates the difference between TCL and CCL work for one feature map convolved with N kernels each has a 3×3 size (to simplify the example comprehend we use 2D kernels and one 2D feature map). The main idea of CCL is to determine the percentage of trainable parameters (weights) distribution within each k×k kernel for depth A in the convolution layer and the rest non-trainable parameters set to zeros values. The trainable parameters form specific shapes and topological relations to extract the local features in these shapes. Furthermore, the distribution of the trainable parameters in specific (random) locations in each kernel with a particular percentage will determine only the useful and efficient local features related to those shapes and topological relations. In other words, the method specifies the weights (trainable parameters) in each kernel with a particular percentage in some locations and allocated zero values to weights (non-trainable parameters) in rest locations. Thus, this method reduces the number of trainable parameters in kernels.

Figure (3.14) illustrates examples of kernel shapes and their appropriate parts on some sub-images. As explained in Figure (3.14), each kernel in the proposed layer contains a particular shape represented by weights considering a specific percentage. To more explain the understanding of CCL layer, the initialization, forward, and backward training phases are explained in the following:

- **Kernels initialization:** in TCL, the weights in kernels are initialized randomly using uniform distribution using the Eq. (2.14).

In CCL layer, the initial values of kernel weights are defined as Eq. (3.2):

$$W_{f(i,j,a)}^{l} = \begin{Bmatrix} Uni\left[-\frac{1}{\sqrt{k}}\ ,\frac{1}{\sqrt{k}}\right]\ and\ set\ trainable\ property = 1\ ,\quad if\ loc_{f(i,j,a)}^{l} = 1 \\ 0 \quad and\ set\ trainable\ property = 0\ ,\quad if\ loc_{f(i,j,a)}^{l} = 0 \end{Bmatrix}$$

(3.2)

Where, $W_{f(i,j,a)}^{l}$ is weight with (i, j, a) index in kernel number f ($f \in F$) with size $k \times k \times A$ in layer *l*, Uni [-a, a] is the uniform distribution function to generate random values in range [-a, a], *trainable property* is property for each weight in kernel give trainable ability to the weights, $loc_{f(i,j,a)}^{l}$ is index of one location in *loc* matrix that contains 0 or 1 random values are generated using a random function with particular percentage for appearing 1 in it. *loc* matrix has the same dimension as W matrix. Eq. (3.3) explains generation of each $k \times k$ *loc* matrix in depth A for F kernels.

$$loc = Genrate\_0\_1\_random\ (k \times k\ , \mathrm{per}) \qquad (3.3)$$

Where *Genrate_0_1_random (.)* is random generation function to generate random values [0 or 1]. *per* is the percentage of appearance of 1 in the random generation function.

loc matrix with 0/1 values forms a specific shape of weights in kernel instead of a square shape where 1's values represent the trainable weights and 0 values represent non-trainable weights in kernels.

In this form of kernels, the convolution operation will take the effect of sub-image positions that is convolved with actual weights and get a zero effect on the zeros weights values in each kernel.

- Forward phase: the convolution process for TCL is defined in Eq. (2.15). For CCL, the convolution process is staying the same, where the data in locations multiplying with the zero weights are neglected (not added in convolution process).

- Backward phase: The weights in kernel are affected by the loss function of computing the error values and updating of weights in the backward phase. In TCL, Eq. (2.16) represents the weights kernel updating function using gradient descent error function that computes the error.

In CCL, since the zeros weights within the kernel have zero error ($trainable\ property = 0$), for that, it is not updated in the backward phase. Thus, the weights in kernel are updated using gradient descent error function as in Eq. (3.4):

$$W^{*l}_{f(i,j,a)} = \begin{cases} W^{l}_{f(i,j,a)} - Lr * \partial W^{l}_{f(i,j,a)} & if\ W^{l}_{f(i,j,a)}\ trainable\ property = 1 \\ W^{l}_{f(i,j,a)} & if\ W^{l}_{f(i,j,a)}\ trainable\ property = 0 \end{cases} \qquad (3.4)$$

Where, $W^{*l}_{f(i,j,a)}$ is the updated weight in index *(i, j, a)* for kernel number $f\ (f \in F)$ with size k×k×A of layer l. $W^{l}_{f(i,j,a)}$ is old weight in kernels, $\partial W^{l}_{f(i,j,a)}$ is the gradient of error function computed of each weight in kernel, Lr is the learning rate.

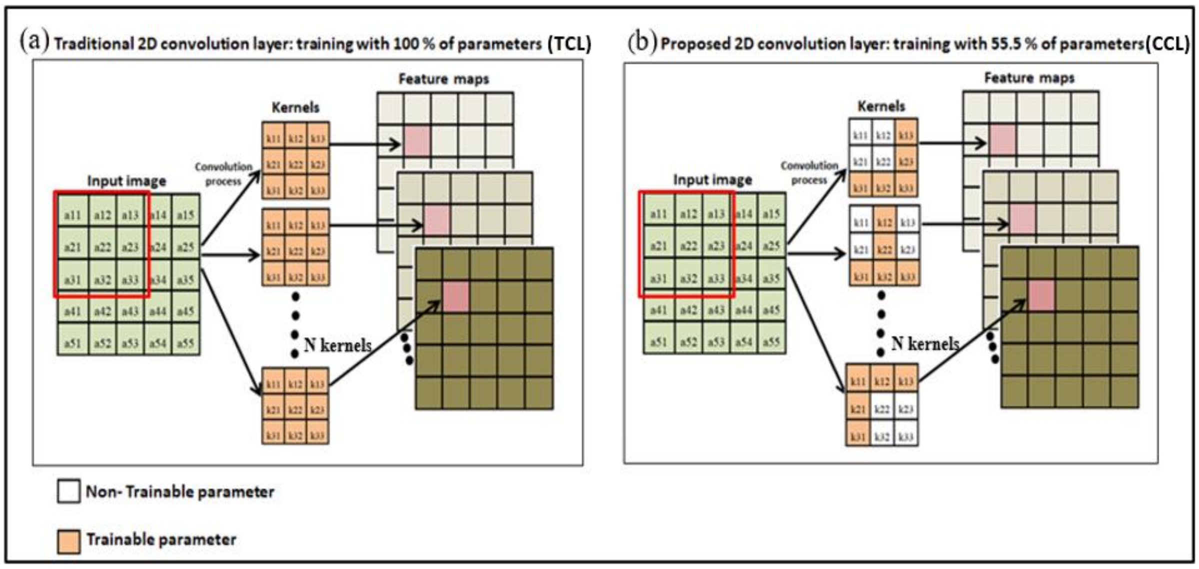

(a) Traditional 2D convolution layer: training with 100 % of parameters (TCL)
(b) Proposed 2D convolution layer: training with 55.5 % of parameters(CCL)
Input image
Convolution process
Kernels
Feature maps
N kernels
Non- Trainable parameter
Trainable parameter

Figure (3.13) The convolution layer work; a: traditional (TCL) and b: proposed (CCL) convolution layers

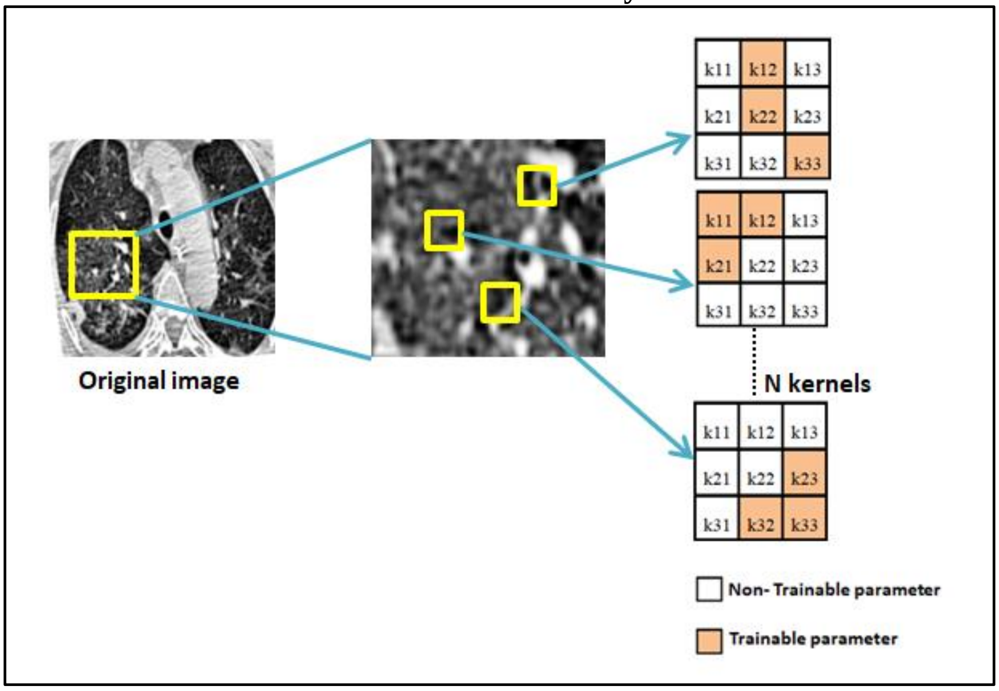


Figure (3.14) Examples of kernels shapes and its appropriate parts on some sub images

The pseudocode of CCL layer is illustrated in the Algorithm (3.4). The weights of the layer are selected randomly and in specific indices in each kernel. The weights with the trainable property are updated in the backward phase but the weights with the non-trainable property are not changed and are still with zeros values.

Algorithm (3.4): Initialization Weights For CCL Layer

Inputs: input feature maps (*A*), output feature maps (*F*), kernels size (*k*) and percentage of trainable weights (*per*)

Output: weights matrix with initial values (*Weight*)

```
Begin
    Set the dimension of Weight and loc matrices with [k, k, A, F]
    Initialize Weight with random uniform values ranged [0-1] using Eq. (2.14)
    Initialize loc with random (0 or 1) values with per using Eq. (3.3)
    For f =1 to F
      For a =1 to A
        For i=1 to k   // applying Eq. (3.2)
          For j =1 to k
            If (loc [i, j, a, f] =0) then
              Weight [i, j, a, f] =0
              Set Weight [i, j, a, f] trainable property with 0
            Else
              Set Weight [i, j, a, f] trainable property with 1
            End if
          End for
        End for
      End for
    Set Weight matrix as convolution layer weights
End
```

As mentioned, CCL layer reduces trainable parameters by the desired percentage. To explain the difference between the CCL and TCL layer, the trainable parameters are computed for these two types of layers. In TCL, the trainable parameters are computed in Eq. (2.17). If we enter

F parameter (refer to the number of feature maps output from layer) in the parentheses, the equation will be as Eq. (3.5)

$$P_t = k \times k \times A \times F + F \qquad (3.5)$$

Where P_t is the number of trainable parameters in TCL layer, $k \times k \times A$ is the kernel size, and we add F for the bias parameter that used F times. In contrast, the trainable parameters in CCL layer are computed as Eq. (3.6):

$$P_p = (k \times k \times per) \times A \times F + F \qquad (3.6)$$

Where, P_p is number of trainable parameters in CCL layer, *per* is the desired percentage of trainable parameter in each $k \times k$ kernel size with depth A. For more explanation, Figure (3.15) illustrates Example for some kernels with different sizes formed according to different percentages.

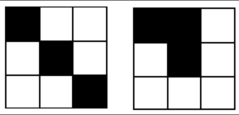
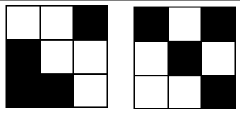
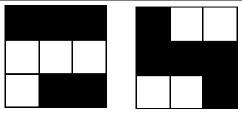
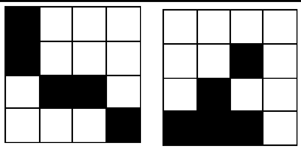
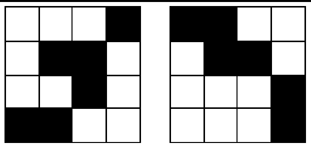
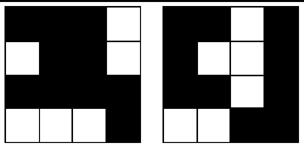
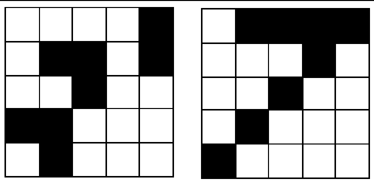
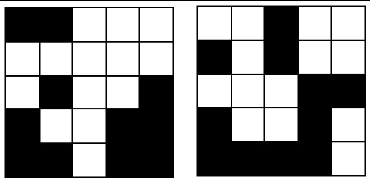
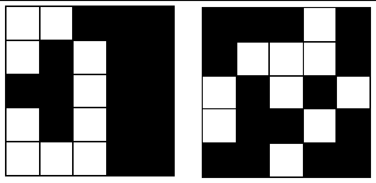
		
Kernel size =3×3=9 Percentage=0.33 Trainable weights =9×0.33=2.97~3	Kernel size =3×3=9 Percentage=0.44 Trainable weights =9×0.44 =3.96~4	Kernel size =3×3=9 Percentage=0.60 Trainable weights =9×0.55 =4.95~ 5
		
Kernel size =4×4=16 Percentage=0.33 Trainable weights =16×0.33=5.28~5	Kernel size =4×4=16 Percentage=0.4 Trainable weights =16×0.4=6.2~6	Kernel size =4×4=16 Percentage=0.60 Trainable weights =16×0.60=9.6~10
		
Kernel size =5×5=25 Percentage=0.33 Trainable weights =25×0.33=8.25~8	Kernel size =5×5=25 Percentage=0.44 Trainable weights =25×0.44=11	Kernel size =5×5=25 Percentage=0.60 Trainable weights =25×0.60=15

Figure (3.15) Example of some kernels with different sizes formed according to different percentages

3.5.3 2D CNN Proposed Model Architecture Building Step

In this section, feature extraction for local shape and topological relations are applied by proposing a new CNN model architecture. The objective of building a CNN model is to prove the efficiency and powerful of the new 2D CNN layer that is one of the earlier layers in the proposed model architecture. Figure (3.16) illustrates the 2D CNN model architecture with a number of kernels and feature maps dimensions inputs for each layer in the model.

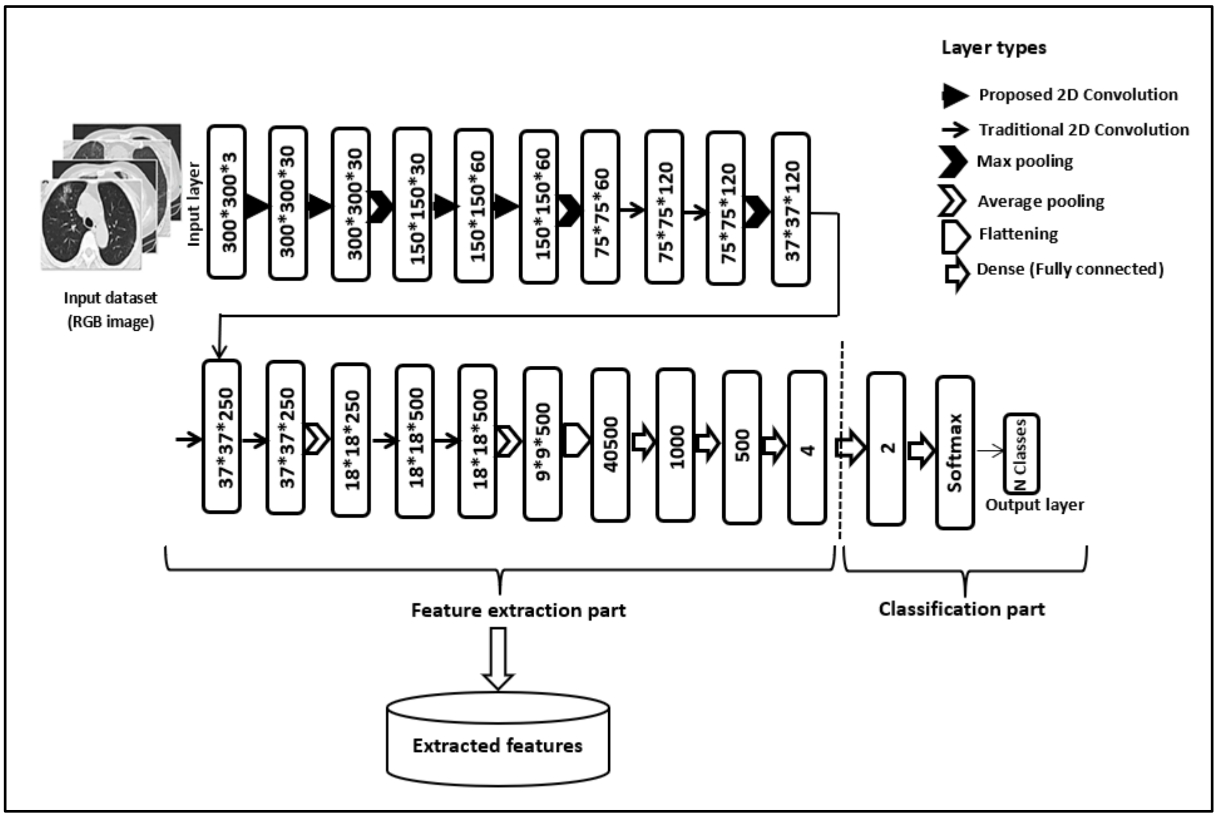


Figure (3.16) The proposed 2D CNN model architecture

The proposed model architecture is motivated from the design of transfer learning networks that includes many successive of convolution layers followed by pooling layer. In convolution layers, including CCL layer, have *an A* number of feature maps with $a \times b$ dimensions and produced *an F* number of feature maps with $a \times b$ dimensions depending on the number of kernels in that layer.

The model starts by input layer with three bands of image (RGB image) with size 300×300 pixels. Then, feature maps produced in each convolution layer are increase progressively and be 30, 30, 60, 60, 120, 120, 250, 250, 500, and 500 feature maps. The size of produced feature maps is decreases progressively as they pass through pooling layer (max and average pooling are used) to be (300×300), (150×150), (75×75), (37×37), (18×18), and (9×9).

In this model, all convolution layers kernels have a size of 3×3 (9 trainable weights) with depth A (number of input feature maps) except CCL layers kernel have 3×3×*per* (9 × per trainable weights) with depth A, where per is the desired percentage appearing weights in a kernel. Also, strides of moving the kernel on feature maps are 1 and padding is the same on left, right, up, and down that returning the same size of the feature maps dimension.

For more explanation, the building of 2D CNN model architecture takes the following steps:

a. Model starts with input layer by three image bands and size 300×300 pixels.
b. Adding two of CCL layers with percentage=33% and both have 30 kernels and produce 30 feature maps with size 300×300. In these layers, low level features for local shape and topological relations of pixels are extracted considering the size for kernels and percentage of trainable parameters in it. These two layers followed by ReLU activation function and batch normalization layers. Then, feature maps pass through one max pooling with 2×2 pool size that considered as a down-sampling for feature maps by reducing its dimensions to the half and convert its size to 150×150.

c. After that, feature maps passed through two CCL layers with 60 kernels and produce 60 feature maps followed by max pooling layer with 2×2 pool size to reduce feature maps dimensions to 75×75.
d. Then, feature maps pass through two TCL layers with 120 kernels and produce 120 feature maps followed by max pooling layer with 2×2 pool size to reduce feature map size to 37×37.
e. Then, two TCL layers are constructed with 250 kernels and produce 250 feature maps followed by 2×2 pool size average pooling layer, max, and average pooling to reduce feature maps size to 18×18.
f. Two convolution layers with 500 kernels and produce 500 feature maps followed by average pooling layer with 2×2 pool size to reduce feature map size to 9×9.
g. 500×9×9 feature maps are flattened using flattening layer to be as one vector with 40500 features. These features are passed through dense layer and is reduced to 1000 values and then dense layer to reduce it to 500 then 4 features. This step finishes the model layers for feature extraction part.
h. Finally, and for learning the model, classification task is used for that purpose. The model uses dense (full connected) layer with 2 neurons and then the Softmax activation function at the end of the model architecture for classification task.

3.6 Image Retrieval Stage

The main objective for this dissertation is to build a CBIR system for medical images considering texture, shape, and topological features of it. The retrieval stage includes achieving this purpose and retrieve set of ranked medical images from part of dataset according to the features extracted using the previous two stages matched a query image entered by the user. As explain in Figure (3.1), this stage contains many steps in off-line and on-line phases that explained in detail as follows:

3.6.1 Dataset Feature Extraction Step

This dissertation includes using many types of datasets of medical images. The test part of the dataset images is passed through the system feature extraction stages. The resultant features are stored in one database for each dataset. Each image in dataset passes through:

The first stage of the system including: quantization process, GLCMs matrices construction, PCA process, and then, multi inputs1D CNN model. The resultant features are collected as one vector for each image with 4 features.

The second stage of the system is also applied on dataset images that consist of: enhancement (preprocessing), 2D CNN model. The resultant features are formed as one vector for each image with 4 features.

Then, features extracted form first and second stages are concatenated to be one vector with 8 features for each image in dataset. Dataset images features are stored in one database in off-line phase to be as a reference for retrieval process for query image that be in on-line phase. Algorithm (3.5) illustrates the dataset feature extraction and storing step.

Algorithm (3.5): Dataset Feature Extraction

```
Input: D: test part of dataset with N images
Output: Database DB with N vectors for D images features
Begin
    Start with Empty database DB
    For each image A in D with N images
            Convert A from RGB color to Ag gray image
            Implement quantization for Ag using Algorithm (3.1)
            Generate GLCMs matrices using Algorithm (3.2)
            Implement PCA process on GLCMs matrices
            Extract texture features vector V1 with 4 features for GLCMs
            using trained 1D CNN model after clip classification layers
            Enhance the image A to AC using CLAHE method
            Extract shape and topology features vector V2 with 4 features
            for AC using trained 2D CNN model after clip classification
            layers
            Concatenate texture features vector V1 and shape and topology
            features vector V2 in one vector V with 8 features
            Add vector V to DB
    End for
    Return DB
End
```

3.6.2 Query Image Feature Extraction Step

In on-line phase, user enters a query image using interactive interface. The selected query image passes through proposed system stages for extracting texture, shape, and topology features and get the resultant features in one vector Vq with 8 features. Algorithm (3.6) explain the steps for query image feature extraction.

```
**Algorithm (3.6): Query Image Feature Extraction**
Input: Q: query image
Output: Vq vector with 8 features of Q
Begin
    Convert Q from RGB color to gray image Vg
    Implement quantization for Vg using Algorithm (3.2)
    Generate GLCMs matrices using Algorithm (3.3)
    Implement PCA process on GLCMs matrices
    Extract texture features vector V1 with 4 features for GLCMs
    matrices using trained 1D CNN model after clip classification layers
    Enhance the image Q to QE using CLAHE method
    Extract shape and topology features vector V2 with 4 features for QE
    using trained 2D CNN model after clip classification
    Concatenate texture features vector V1 and shape and topology
    features vector V2 in one vector Vq with 8 features
    Return Vq
End
```

3.6.3 Matching and Retrieval Step

In this step, a similarity matching between stored database images features and query image features is performed. The test part of the dataset is used in this step to return the most similar ranked images to the query image considering a similarity metric. Cosine metric described in Eq (2.18) is the similarity measurement used to find the most similar images to the query image. If similarity between query features and test image features are equal or more than 0.75, then the test image is returned as retrieved image. Algorithm (3.7) explain the image retrieval process.

Algorithm (3.7): Image Retrieval Process and Evaluation

Input: *DB*: database contains features of test dataset images, *Vq* are features of query image, *K*: number of ranked more similar images to the query.

Output: Display *N* ranked most similar test dataset images to query image

```
Begin
    Create index_images as empty list to save indices of similar images
    Create similarity_measures as empty list for retrieved image cosine
    similarity Create test_images_label as empty list for retrieved images labels
    Initialize relevant number of images have the same label of query image
    Initialize retrieve number of images resulted as similar images (retrieved)
    Initialize retrieve_relevant number of images that relevant and retrieved
    query_label is the label of Vq
    For i =1 to number of DB Do // number of test database images features
        V= DB[i] // features vector of image i in test part
        test_label is the label of V
        if (query_label==test_label) Then // that mean it is relevant
            Compute cos similarity metric between Vq and V using Eq. (2.18)
            if (cos > = 0.75) Then // that mean it will be retrieved
                retrieve = retrieve + 1
                Append i index to index_images list
                Append cos to similarity_measures list
                Append test_label to test_images_label list
            End if
        End if
    End for
    Build a database Db for similar images with three columns;
    similarity_measures, index_images, and test_images_label lists
    Sort Db ascending according to Similarity_measures column
    Display images have index_images in the first K elements in Db
End
```

To evaluation the retrieval process, mAP used as retrieval accuracy metric described in Eq. (2.37). Therefore, there is need to find **Retrieved,** and **Relevant retrieved** quantities that are explained in Section (2.8.3). To find mAP metric there is need to find AP metric explained in Eq. (2.36) for each query image (30 query images in this experimental). Then mAP is computed by finding the average of AP for all query images used in the evaluation process. Algorithm (3.8) explains the retrieval evaluation process using mAP metric.

Algorithm (3.8): Retrieval Evaluation Process

```
Input: DB: features database, K: number of ranked more similar images to the query, Qn: list of N query images features.
Output: Display mAP for ranked K images similar to query image.
Begin
    Initialize AP as empty list for all queries AP metric
    For j = 1 to N // Find AP for each query image using Eq. (2.36)
        retrieved_relevant=0 // count for retrieved and relevant images
        Vq= Qn[j]
        query_label is the label of Vq
        Initialize P as empty list for precision
        Call for Algorithm (3.7) and find number of retrieve images
        if (retrieve => K) Then
            For v =1 to K Do
                test_lable is the label of DB[v]
                if (query_label == test_lable) Then//that mean it is relevant retrieved
                    retrieved_relevant= retrieved_relevant+1
                    L= retrieved_relevant / v
                    Append L to P list
                End if
            End for
            C = sum of P elements / retrieved_relevant
            Append *C* to *AP* list
        **End if**
    **End for**
    $mAP$ = mean average of $AP$ list // Eq. (2.37)
    print ($mAP$)
**End**
```

Chapter Four

Results and Discussions

4.1 Introduction

This chapter presents system hardware specification and four types of medical datasets used as a case studies when implements the system and some samples of these images. Also, in this chapter, experimental results and discussions produced from proposed system stages are introduced. The presented experimental results will be for the proposed quantization method for different quantized gray levels and time complexities. Furthermore, 1D multi inputs CNN model evaluation using different datasets are explained compared with the other pretrained DNN and state-off-art methods. The effect of enhancement as a preprocessing step is shown using case study datasets of images samples. Furthermore, the proposed 2D CNN model evaluation in some aspects and different parameters are presented in this chapter. Experimental results for medical image retrieval system evaluation for ranked N retrieved images are introduced.

4.2 Proposed System Specification

To implement the machine learning techniques and Deep learning models as well as big images datasets, there are some hardware and software requirements must be available to deal with time and processing demands of these techniques. For that, the proposed system is applied using the following requirements:

1. Central processing unit (CPU): Core (TM): i7-8750H, CPU: @2.20GHz 2.21 GHz, Intel(R).
2. Ram: Samsung 16.0 GB (15.9 GB usable) ram
3. Hard Disk:128GB SSD + 1 TB HHD
4. Graphics processing unit (GPU): NVIDIA GeForce RTX 2070 8 GB with Max Q-Design
5. Operating system: windows 11 64 bit
6. Programming language: python 3.8, PyCharm 2022.1.3

4.3 Datasets Description

In this dissertation, to prove the efficiency of the proposed system, three medical image datasets are considered. Datasets are collected for the coronavirus disease 2019 pandemic (COVID-19) that is a virus disease spread and caused many injuries and deaths around the world. The three datasets are described a Computer Tomography (CT) scans and X-Ray images that considered are test and monitor tools for disease progression. To evaluate the proposed CNN models and the proposed retrieval system as a whole, the three types of datasets are divided into train and test parts in a specific percentage according to the known protocol for each dataset. Each division part contains images with COVID-19 infection (positive) and non-COVID-19 (negative) images. The division of each dataset is explained in Table (4.1). The description of the three datasets is presented as follows:

Table (4.1) Three types of datasets divide into training and testing parts as used in the proposed system

	D1 dataset		**D2 dataset**		**D3 dataset**	
	positive	**negative**	**positive**	**negative**	**positive**	**negative**
Train part	1001	983	244	277	125	551
Test part	251	246	105	120	118	290
Total	1252	1229	349	397	243	841

1. SARS-CoV-2 CT-scan Dataset

The first dataset is called “SARS-CoV-2 CT-scan” (D1) dataset [28] contains 2481 CT scan images belonging to 120 patients, 1252 images with COVID-19 infection, and 1229 images with non-COVID-19, but have other lung infections. Data was composed by Sao Paulo hospitals in Brazil and it is obtained from “https://www.kaggle.com/datasets/plameneduardo/sarscov2-ctscandataset”. The dataset image sizes differ and range between 104 to 153 pixels for the smallest size and 484 to 416 for the largest one. Figure (4.1) illustrates some samples of this dataset images with their class and size.

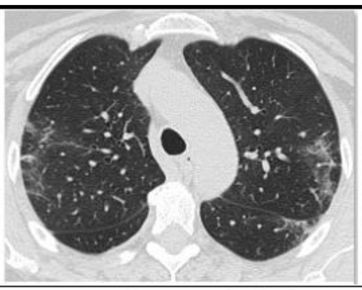
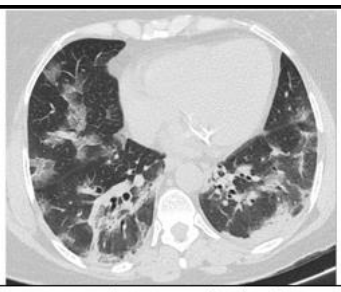
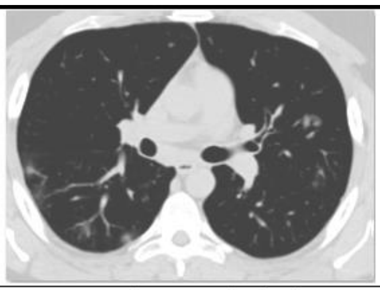
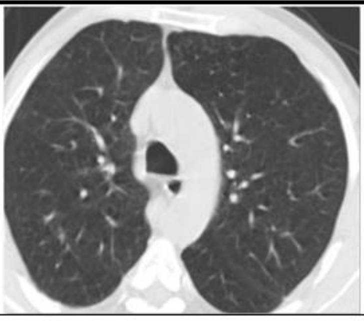
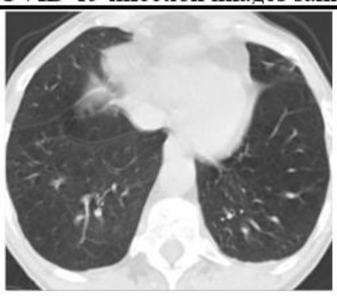
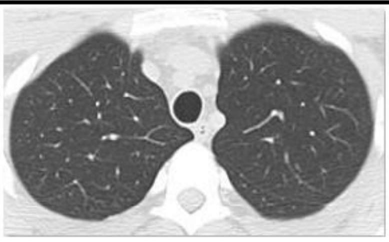
		
PNG (256×202) pixels	PNG (392×327) pixels	PNG (274×201) pixels
COVID-19 infection images samples		
		
PNG (391×338) pixels	PNG (415×355) pixels	PNG (272×162) pixels
Non-COVID-19 images samples		

Figure (4.1) D1 dataset images samples

2. COVID-CT Dataset

The second dataset is the COVID-CT dataset (D2) [31]. This dataset includes CT scan images that contain a total of 746 CT scan images, 349 images with COVID-19 infection and 397 images with non-COVID-19. The dataset is grouped from technical researches published in the medRxiv and biRxiv impactful journals. Moreover, set of images defined in laboratories (“http://medicalsegmentation.com/COVID-19/”). The dataset contains Metadata collected manually for these images: patient age, gender, location, medical history, scan time, the harshness of COVID-19, and medical report. The D2 dataset is obtained from “https://github.com/UCSD-AI4H/COVID-CT”. The images in this dataset are very different in size that ranged in heights between 153 to

1853 pixels and ranged in widths between 124 to 1458 pixels. Figure (4.2) illustrates some images samples for this dataset.

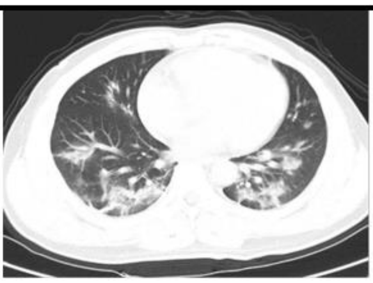
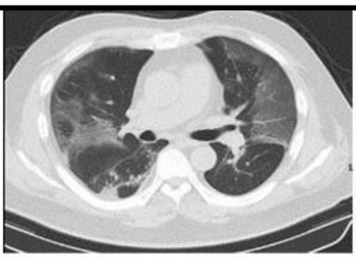
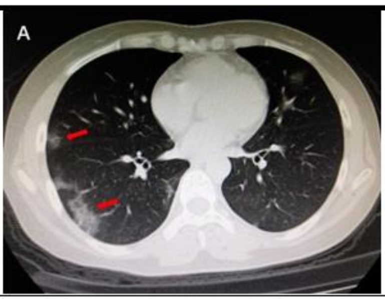
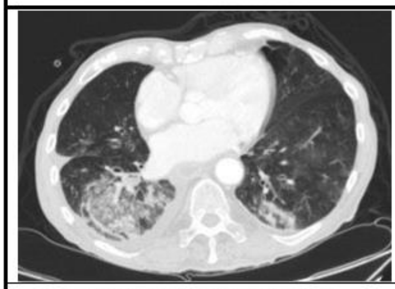
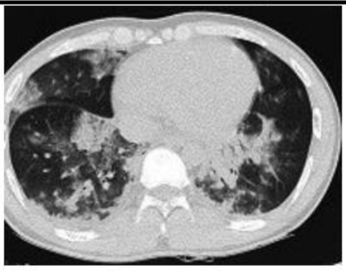
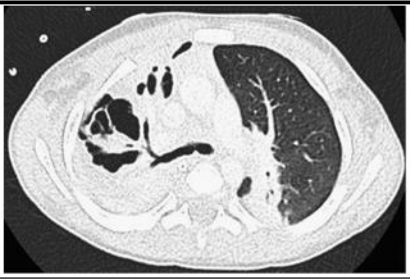
		
PNG (724×529) pixels	PNG (287×202) pixels	PNG (627×473) pixels
COVID-19 infection images samples		
		
PNG (384×280) pixels	PNG (196×150) pixels	PNG (778×518) pixels
Non-COVID-19 images samples		

Figure (4.2) D2 dataset images samples

3. DLAI3 Hackathon COVID-19 Chest X-Ray Dataset

The third dataset is called "DLAI3 Hackathon COVID-19 Chest X-Ray" (D3) dataset [140]. D3 is downloaded from the Kaggle website that introduced a "DLAI3 Hackathon COVID-19 Chest X-Ray" challenge and it is obtainable in "https://www.kaggle.com/c/dlai3/data".

The dataset contains 1135 X-Ray images with COVID-19 and Non-COVID-19 (normal) divided into train and test sets. Train set contains 125 images for with COVID-19 and 551 images with non-COVID-19 states. Test set contains 118 images with COVID-19 and 290 images with non-COVID-19 states. The size of images is different and it is varies in contrast and brightness. The larger dataset was from the CSC532 Machine Learning project of Puttipong Thammachart & Vasin Virasak. Also, a part of this dataset is from "https://github.com/ieee8023/covidchestxray-dataset". Figure (4.3) illustrates some images sample for this dataset.

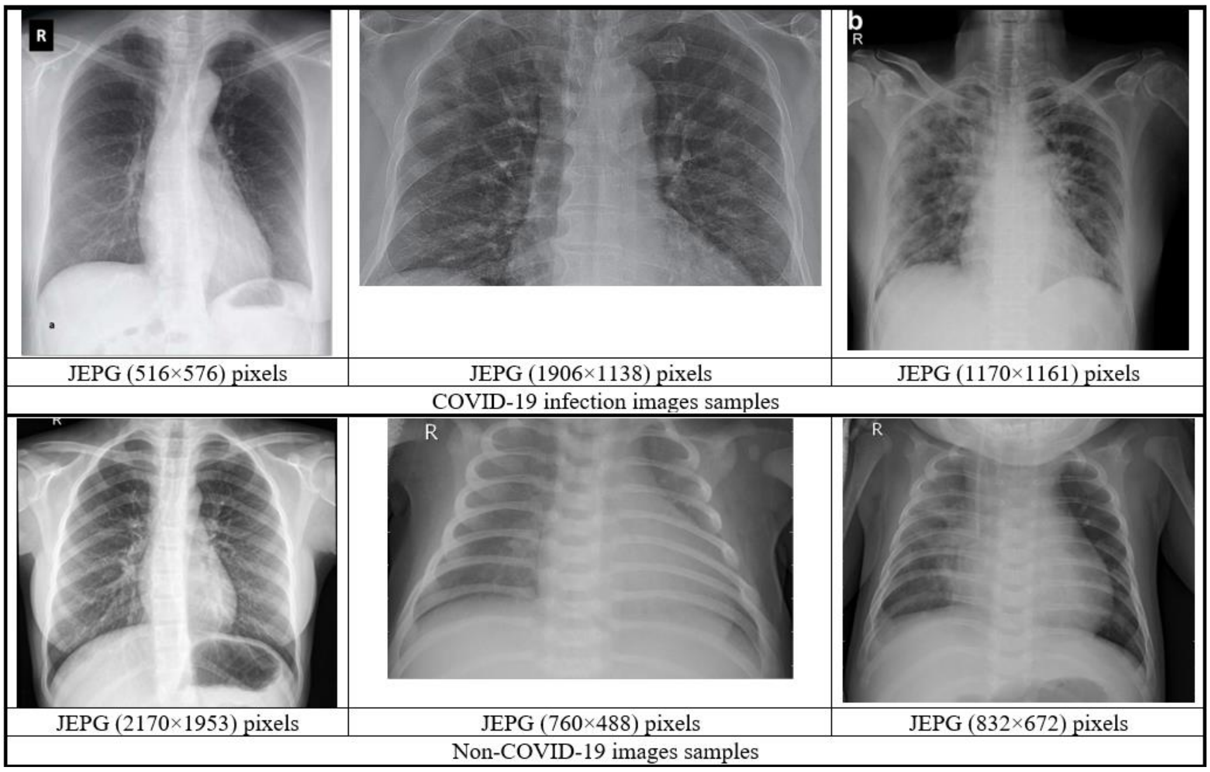


Figure (4.3) D3 dataset images samples

4.4 Texture Feature Extraction Stage

The results and discussions of the texture features extracted using GLCMs matrices and multi inputs 1D CNN model are presented in this section. The experimental results are explained for the quantization step for different medical images compared with other traditional quantization methods. Furthermore, this section presents results and discussions of evaluation the resultant texture features for GLCMs matrices using multi inputs 1D CNN model architecture for the three datasets compared with other pre-trained DNN and state-of-art methods.

4.4.1 Quantization Step

As explain in chapter three, the dimensions of GLCMs matrices depend on the number of image gray levels. Hence, the collected and reduced GLCMs matrices will be use as inputs for the CNN model. Also, medical images contain very important details necessary for the diagnosis of diseases. For these reasons, there was a need to propose a new quantization method to reduce the image gray levels and obtain the efficient and most accurate quantized image. Also, producing three versions of quantized images to get many GLCMs matrices allows the system to get more diverse texture features for each input medical image.

4.4.1.1 Quantization Experimental Results and Discussions

The original and resultant quantized images for three samples images for D1, D2, and D3 medical datasets used in this dissertation are illustrated in Figures (4.4), (4.5), and (4.6), respectively, considering different quantized gray levels.

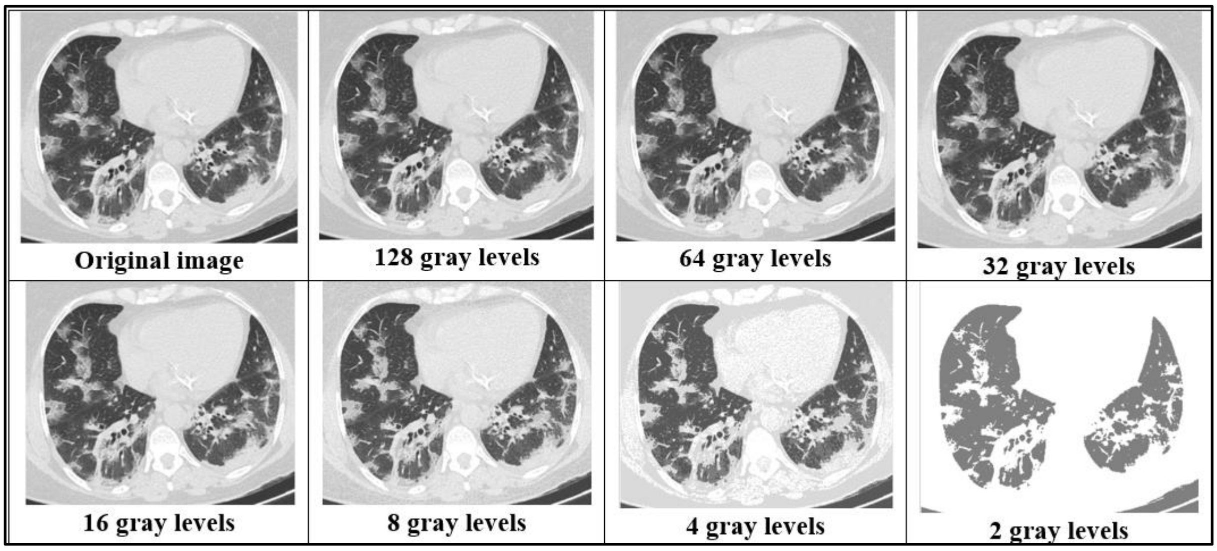


Figure (4.4) Original and quantized sample medical image from D1 dataset using proposed quantization method

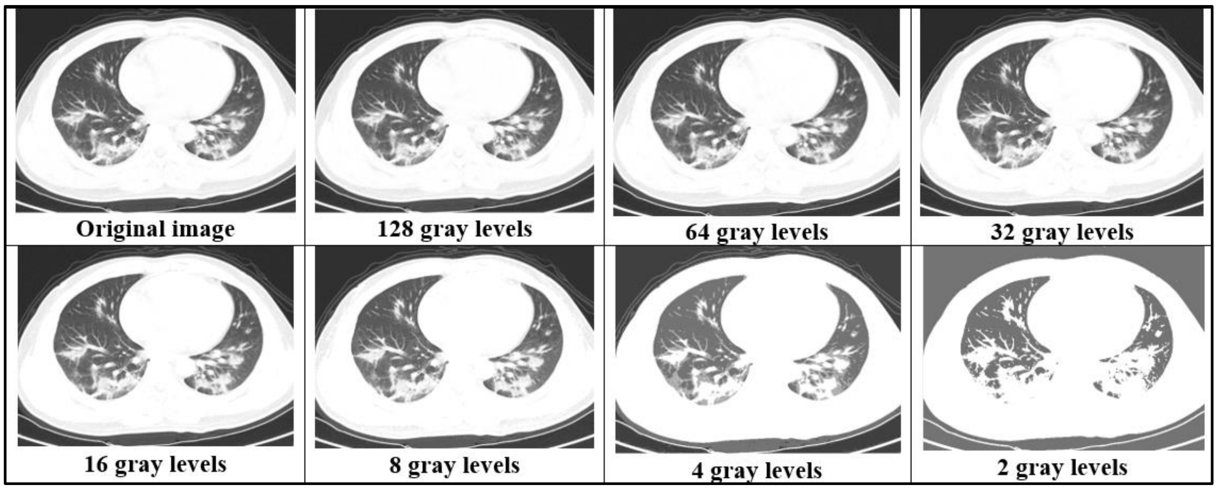


Figure (4.5) Original and quantized sample medical image from D2 dataset using proposed quantization method

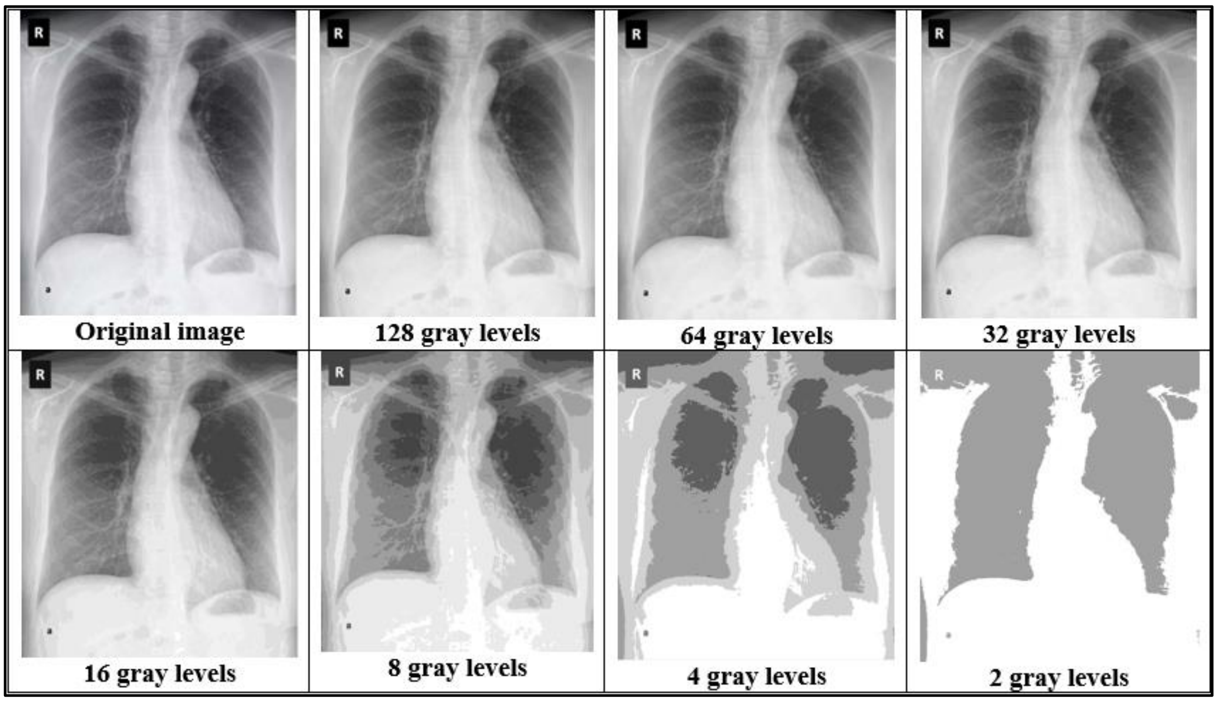


Figure (4.6) Original and quantized sample medical image from D3 dataset using proposed quantization method

To prove the efficiency of the proposed quantization method, PSNR and SSIM accuracy metrics described in Eq. (2.21) and Eq. (2.22), respectively, between the original and quantized images are used to evaluate the method. Table (4.2) shows the PSNR metric for three samples medical images from D1, D2, and D3 datasets used in this dissertation. Results in Table (4.2) show the proposed quantization method PSNR compared with standard K-mean clustering and uniform quantization methods for different quantized gray levels. Furthermore, the results of SSIM measure are illustrated in Table (4.3) resulted using the proposed method for different gray levels compared with K-mean clustering and uniform quantization methods the three samples of medical images from D1, D2, and D3 datasets used in this dissertation.

As shown in Table (4.2), the proposed method outperforms other methods like uniform and K-mean quantization methods with gray levels 2, 4, 8, 16, 32, 64, and 128 in terms of PSNR metric.

Also, Table (4.2) shows that the proposed method is outperforms the uniform method in a large difference in small gray levels as 2, 4, and 8 and the difference is decreasing as the gray levels are bigger like in 32, 64, and 128. From Table (4.3), it is obvious that all experimental results are shown that the proposed method achieves an efficient accuracy and outperforms the other methods in terms of SSIM quality measure for different quantized gray levels.

In general, although the proposed method superior the other method in terms of PSNR and SSIM metrics, the difference range between the compared methods is not stable or fixed. This not stability in metrics between proposed method and others is due to nature of images, its size, and distribution of gray levels in images.

Table (4.2) Comparison of proposed quantization (proposed) with other methods in terms of PSNR

Tested images	**Quantization methods**	**PSNR for different number of quantized gray levels**						
		2 gray levels	**4 gray levels**	**8 gray levels**	**16 gray levels**	**32 gray levels**	**64 gray levels**	**128 gray levels**
D1 dataset Sample 392x327	Uniform	9.564	14.296	19.854	26.118	32.028	37.728	44.060
	K-mean	14.334	21.313	22.430	29.168	32.424	36.313	40.591
	Proposed	**19.709**	**22.085**	**24.439**	**30.565**	**36.282**	**42.685**	**49.699**
D2 dataset Sample 724x529	Uniform	7.896	14.324	20.008	26.158	32.134	38.172	43.829
	K-mean	18.372	20.250	24.148	32.518	33.793	38.415	41.938
	Proposed	**21.798**	**24.564**	**27.049**	**32.979**	**38.862**	**45.379**	**53.002**
D3 dataset Sample 516x576	Uniform	10.143	15.120	20.261	26.612	32.286	38.040	44.346
	K-mean	15.589	18.474	20.530	26.862	30.224	33.806	39.002
	Proposed	**19.151**	**21.818**	**24.784**	**30.293**	**36.536**	**43.327**	**51.462**

Table (4.3) Comparison of proposed quantization (proposed) with other methods in terms of SSIM

Tested images	**Quantization methods**	**SSIM for different number of quantized gray levels**						
		2 gray levels	**4 gray levels**	**8 gray levels**	**16 gray levels**	**32 gray levels**	**64 gray levels**	**128 gray levels**
D1 dataset Sample 392x327	Uniform	0.3474	0.5708	0.7755	0.9284	0.9808	0.9953	0.9988
	K-mean	0.3851	0.4923	0.6884	0.8065	0.9075	0.9532	0.9843
	proposed	**0.7086**	**0.7851**	**0.8681**	**0.9570**	**0.9835**	**0.9954**	**0.9990**
D2 dataset sample 724x529	Uniform	0.4334	0.7058	0.8464	0.9187	0.9752	0.9922	0.9980
	K-mean	0.6850	0.7188	0.7736	0.9136	0.9268	0.9616	0.9822
	proposed	**0.8037**	**0.8490**	**0.8910**	**0.9595**	**0.9856**	**0.9958**	**0.9992**
D3 dataset sample 516x576	Uniform	0.5037	0.6851	0.7376	0.8667	0.9515	0.9875	0.9971
	K-mean	0.6771	0.7124	0.6823	0.7392	0.7926	0.8606	0.9380
	proposed	**0.7108**	**0.7601**	**0.7670**	**0.8864**	**0.9582**	**0.9878**	**0.9974**

4.4.1.2 Quantization Time Complexities Analysis

In the proposed method, the search process for the optimum PSNR is applied on desired gray levels M that determined by the user. However, the search process adapts the divide and conquers strategy that divides the search space into two ranges in each iteration. Therefore, The time complexity of finding optimum PSNR is O(log M). For more explaination, each search process requires PSNR computing and compares it with the original image. This process scans the entire image with size A×B. Therefore, the time complexity for the proposed method is O(A×B×log M). Figures (4.7) and (4.8) illustrate the time changing with different gray levels values and different images size used in experiment results and three big datasets for equal size images described in Section (4.3), respectively.

The time complexity is computed using implementation requirments explained in Section (4.2). As shown in Figure (4.7), the time consumption is relatively for small size images such as image from D1 dataset. By contrast, the time consumption for big size images proportionately big such as images from D2 and D3 datasets. That's because proposed method time complexity depends on the size of the processed image.

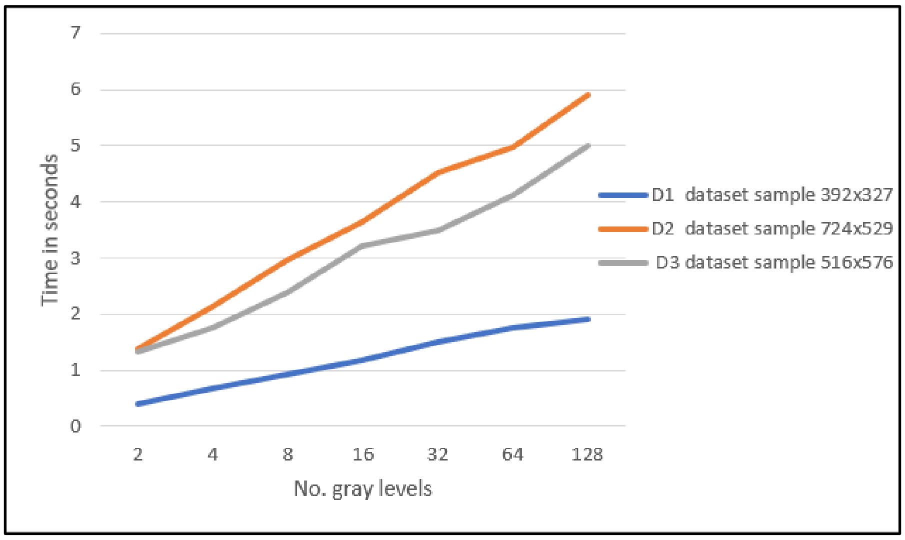


Figure (4.7) Time consumption for apply proposed method on different types of images with different size and different quantized gray levels

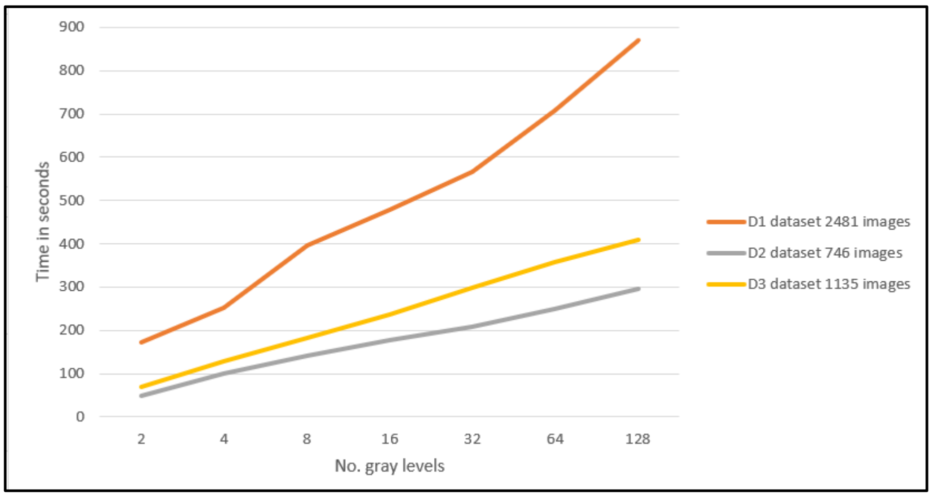


Figure (4.8) Time consumption for apply proposed method on three different size datasets and different quantized gray levels

Also, we observed that the time is increased as the number of quantized gray levels is increased. That's because increasing number of quantized gray levels will increase number of division iterations, and time consumption in applying the method is increase. Furtheremore, as observed from Figure(4.8), there is ability to implement the proposed method on big datasets and get acceptable time consumption and it is increase as increasing the number of dataset images.

4.4.2 Multi Inputs 1D CNN Model Evaluation

To evaluate the efficiency of extracted GLCM features for three quantized images and different distances and directions for context, experimental results are performed for the proposed multi inputs 1D CNN model. Accuracy and F1 score are used as accuracy measures are used to evaluate the model.

The proposed model is trained using an SGD optimizer, 0.001 learning rate, 0.9 momentum, and decay rate with learning (rate/epochs) rate. The model trains for 70 epochs with 32 batch sizes. The size of images used in experiments is (300×300) pixels.

As mentioned before, the experimental results are applied on three different medical images datasets for COVID-19 disease. Different pre-trained DNN models and state-of-arts methods are compared with the proposed model. For a fair comparison and because the different aspects for each dataset, we illustrate each dataset experimental results individually in the following sub sections.

4.4.2.1 1D CNN Model Evaluation on D1 Dataset

In training mode, the proposed model is trained on train part of the dataset and evaluate reached the model to convergence and minimum amount of error rate using the test part of the dataset. To explain the behavior of the learning process and evaluation of train and test datasets parts during training mode, accuracy defined in Eq. (2.30) and loss defined in Eq. (2.13) metrics are plotted during progressing of training process for this dataset with 70 epochs as illustrated in Figure (4.9).

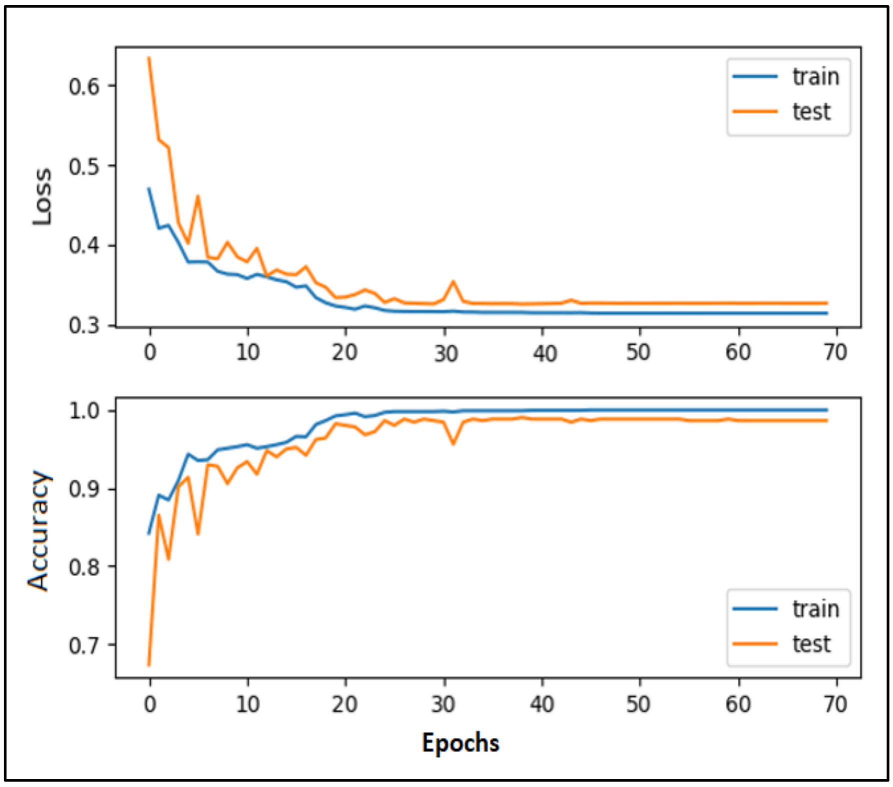


Figure (4.9) Accuracy and loss metrics plots for proposed 1D CNN model training using D1 dataset

In testing mode and to evaluate the proposed model performance, comparison is performed between the proposed model with other pre-trained (black-box) DNN approaches including; ResNet, GoogleNet, VGG-16, and AlexNet networks. Also, the model is compared with the xDNN algorithm introduced by Soares et al. [28] (explained in Section (1.3). Table (4.4) illustrates the comparison results between proposed model (1D multi inputs CNN) and other methods in terms of accuracy defined in Eq. (2.30) and F1 score defined in Eq. (2.33) metrics. Figure (4.10) illustrates the results in visual form.

From the experiment results in Table (4.4), it is clear that proposed 1D CNN model depended on GLCMs reduced matrices outperforms the pre-trained DNN and other previous methods. The proposed method presents enhanced results of accuracy with 98.994% and F1 score

98.981%. Table (4.5) shows the more evaluation measures for the proposed model such as accuracy, recall, precision, and F1 score that defined in Eq. (2.30), Eq. (2.31), Eq. (2.32), and Eq. (2.33), respectively.

Table (4.4) Proposed models results compared with other pre-trained DNNs and state-of-art approaches applied on D1 Dataset

Methods	Accuracy	F1 score
ResNet-50	94.96%	95.03%
GoogleNet	91.73%	91.82%
VGG-16	94.96%	94.97%
AlexNet	93.75%	93.61%
xDNN [28]	97.38%	97.31%
1D multi inputs CNN	**98.994%**	**98.981%**
2D CNN with TCL	**98.195%**	**98.114%**
2D CNN with CCL	**98.597%**	**98.595%**

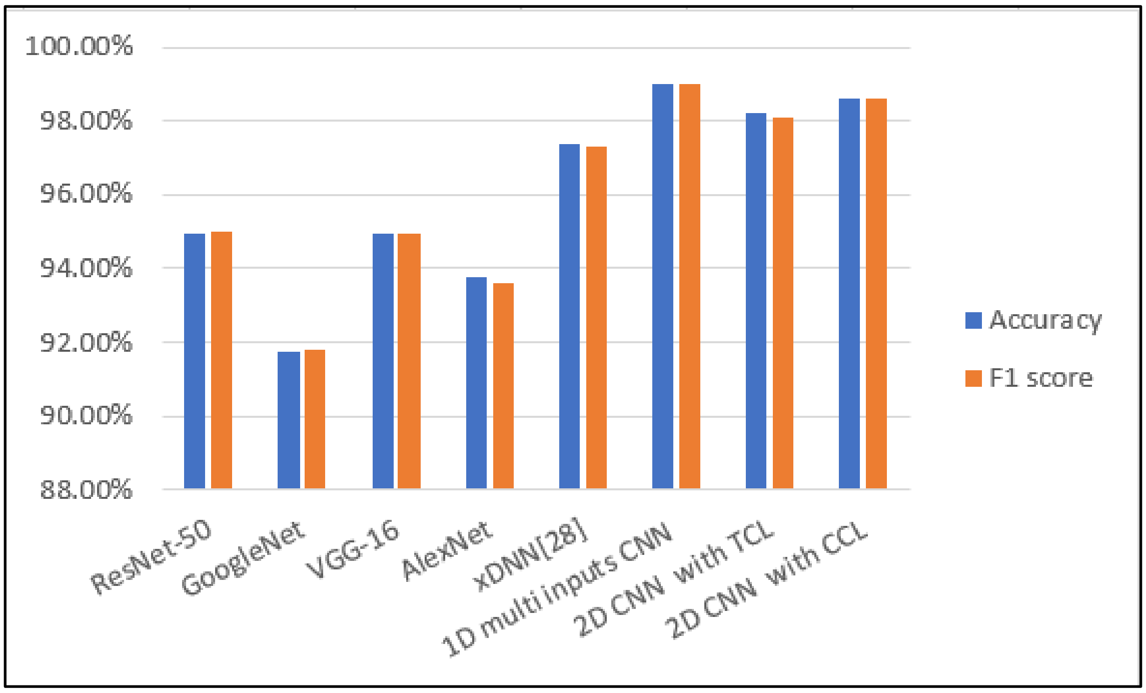


Figure (4.10) Proposed models results compared with other pre-trained DNNs and state-of-art approaches applied on D1 dataset

Table (4.5) The evaluation measures values of 1D CNN model on D1 dataset

Evaluation measure	Value %
Accuracy	**98.994**
Precision	**99.183**
Recall	**98.781**
F1 score	**98.981**

4.4.2.2 1D CNN Model Evaluation on D2 Dataset

The D2 dataset is considered a more challenging and popular dataset that is used by many researchers for evaluation purposes. In training mode, the proposed model is trained on D2 Dataset using train part of the dataset. The accuracy and loss metrics are plotted during progressing of training process for this dataset as illustrated in Figure (4.11) to explain the behavior of the learning process and evaluation the model using train and test datasets parts during training mode with 70 epochs.

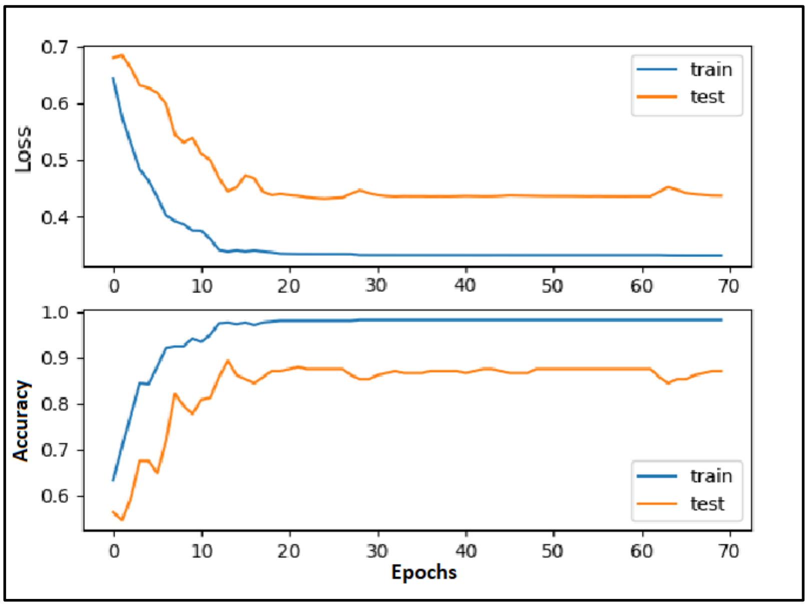


Figure (4.11) Accuracy and loss metrics plots for proposed 1D CNN model training using D2 Dataset

To apply proposed model on this dataset, we use the division dataset explained in Table (4.1) and divide the dataset to 70% train and 30% test. The experiment results of the proposed model are compared with pre-trained DNNs that trained on large-scale datasets, including ImageNet and the Lung Nodule Malignancy (LNM) datasets. These pre-trained models include VGG16, ResNet18, ResNet50, DenseNet-121, and DenseNet-169.

Also, the results of proposed model are compared with other state-of-art approaches included methods presented by Mobiny et al. [30], Polsinelli et al. [29], He et al. [31], and Pedro Silva et al. [32] (explained in Section (1.3)). It is important to note that some of the previous methods as in [30] and [29] are also used a data augmentation and merge more than one dataset that increases the number of images in the train set and therefore causes different accuracy results.

Table (4.6) illustrates the results of proposed model compared with the other methods in terms of accuracy and F1 score performance metrics in testing mode. Figure (4.12) shows these results visually.

Although the pre-trained DNNs are trained on large datasets and have deeper and complex architectures, the proposed model performs better than these networks. The proposed model improves compared with pre-trained DNNs in terms of accuracy and F1 score metrics with 88.444% and 89.166%, respectively.

Table (4.6) Proposed models results of D2 Dataset compared with other pre-trained DNNs and state-of-art approaches

Methods	**Accuracy**	**F1 score**
ResNet-18	74%	73%
ResNet-50	80%	81%
DenseNet-121	79%	79%
DenseNet-169	83%	81%
VGG-16	76%	76%
Polsinelli et al. [29]	84.56%	83.98%
Mobiny et al. [30]	87.6%	87.1%
He et al. [31]	86%	85%
Pedro et al. [32]	87.6%	86.19%
1D multi inputs CNN	**88.444%**	**89.166%**
2D CNN with TCL	**88.495%**	**88.401%**
2D CNN with CCL	**89.666%**	**89.583%**

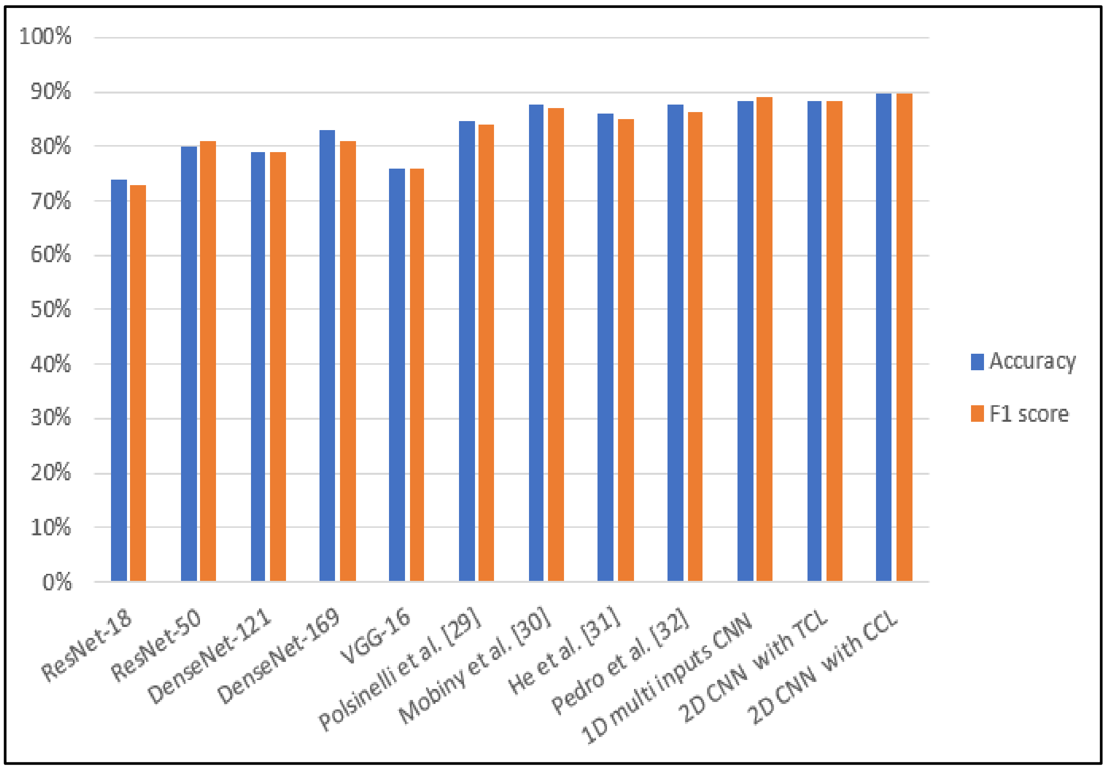


Figure (4.12) Proposed models results compared with other pre-trained DNNs and state-of-art approaches applied on D2 Dataset

As shown in Table (4.6), the proposed model outperforms the other state-of-art previous methods in [29, 30 , 31 , 32] in terms of accuracy and F1 score with an improved performance. Table (4.7) shows more evaluation measures including accuracy, precision, recall, and F1 score for the proposed model.

Table (4.7): The evaluation measures values of applying proposed 1D CNN model on D2 Dataset

Evaluation measure	**Value %**
Accuracy	**88.444**
Precision	**89.166**
Recall	**89.256**
F1 score	**89.166**

4.4.2.3 1D CNN Model Evaluation on D3 Dataset

The D3 dataset is obtained from the Kaggle website that introduced a challenge applied using D3 dataset [140]. The D3 dataset [140] introduced a baseline of 92% for accuracy metric. The accuracy and loss metrics are plotted during progressing of training process for this dataset as illustrated in Figure (4.13) with 70 epochs.

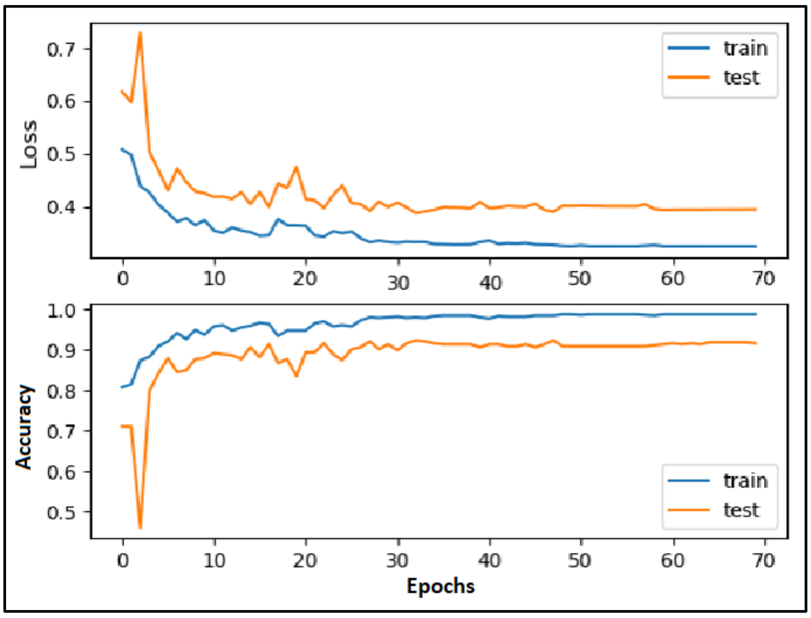


Figure (4.13) Accuracy and loss metrics plots for proposed 1D CNN model training using D3 dataset

In test mode, we evaluate the proposed model by comparing the accuracy and F1 score results with the baseline [140] and with the pre-trained DNNs included; VGG16, AlexNet, VGG19, Dense-169 networks as shown in Table (4.8). Figure (4.14) shows these results visually.

As shown in Table (4.8), the proposed model outperforms the baseline accuracy and other pre-trained DNNs networks. The proposed model gets better results and exceeded the pre-trained DNNs networks and baseline applied in [140] in terms of accuracy and F1 score metrics with 93.627% and 93.064%, respectively. Table (4.9) illustrates the evaluation measures for applying the proposed model on the D3 dataset.

Table (4.8) Proposed models results using D3 dataset compared with other pre-trained DNNs and baseline approaches

Methods	Accuracy	F1 score
DenseNet-121	90.17%	88.08%
DenseNet-169	90.08%	87.41%
VGG-16	86.27%	82.95%
VGG-19	78.92%	76.85%
Baseline [140]	92%	-
1D multi inputs CNN	**93.627%**	**93.064%**
2D CNN with TCL	**93.20%**	**92.848%**
2D CNN with CCL	**93.302%**	**93.054%**

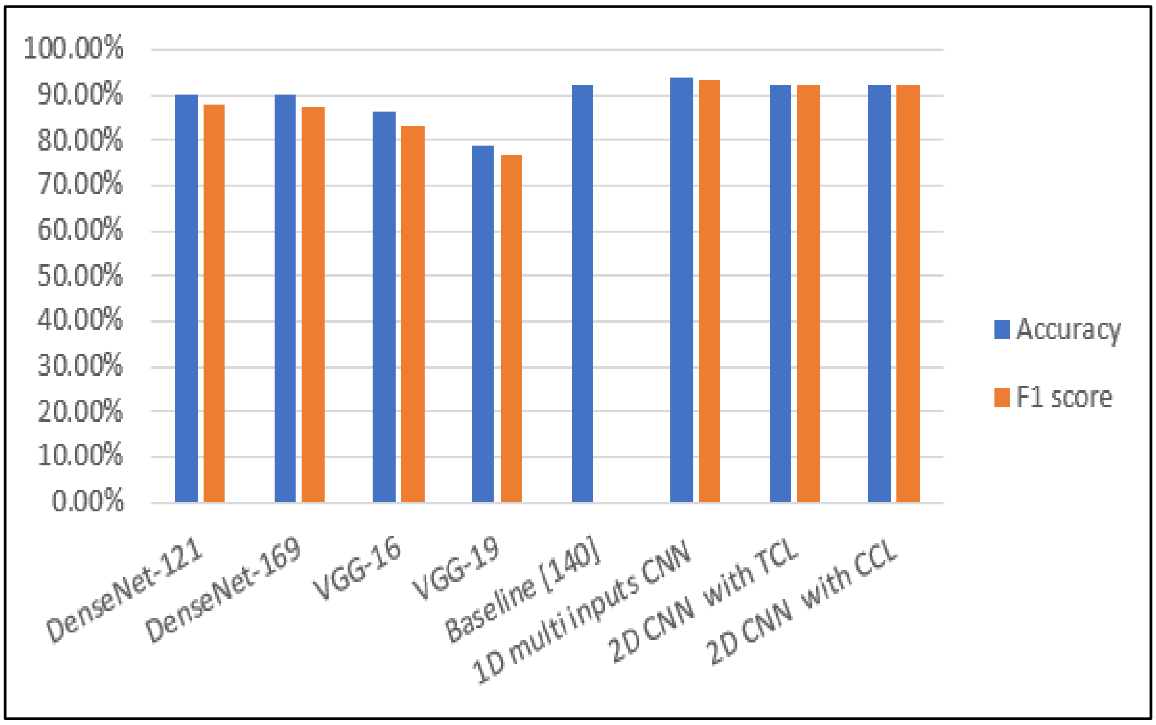


Figure (4.14) Proposed models results compared with other pre-trained DNNs and state-of-art approaches applied on D3 Dataset

Table (4.9) The evaluation measures values for applying proposed 1D CNN model on D3 dataset

Evaluation measure	Value%
Accuracy	**93.627**
Precision	**92.802**
Recall	**87.288**
F1 score	**93.064**

4.5 Shape and Topology Feature Extraction Stage

To evaluating the local shape and topology features extracted using proposed 2D CNN model, experimental results and discussions are introduced to prove the efficiency of features. As mentioned before, the experimental results are performed on three types of medical images datasets for CT scan and X-Ray images for COVID-19 disease. First, the experimental results of applying enhancement preprocessing step for different types of medical images is presented. Then, the 2D CNN model with new convolution layer is evaluated compared with other pre-trained DNN networks and state-of-art methods.

4.5.1 Enhancement Preprocessing Step

CLAHE is the enhancement method that used as preprocessing step to increase proposed 2D CNN model performance by increase and enhance the images contrast. Figure (4.15) three enhanced samples images from three datasets using CLAHE method compared with the original images and HE enhancement method. Table (4.10) illustrates evaluation of CLAHE method compared with original image and image enhanced using HE method in terms of EME metric defined in Eq. (2.29).

Table (4.10) EME contrast image metric for original datasets samples image compared with contrast enhanced images using HE and CLAHE methods

Image	Original image	Enhanced image using HE	Enhanced image using CLAHE
D1 dataset sample	9.198%	18.097%	**22.371%**
D2 dataset sample	8.328%	12.487%	**14.625%**
D3 dataset sample	3.374%	7.294%	**9.198%**

EME metric is used to compute the contrast enhancement and improvement in image and a higher EME measure means a high contrast and more clearly information in the image. As clear in Figure (4.15) and

Table (4.10), CLAHE method enhances the contrast of image and outperforms the original image and image enhanced using HE method. The enhancement method is performed on all images in dataset in its two parts; train and test images before entered to the proposed 2D CNN model.

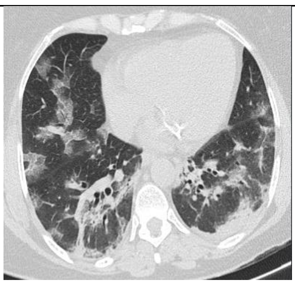
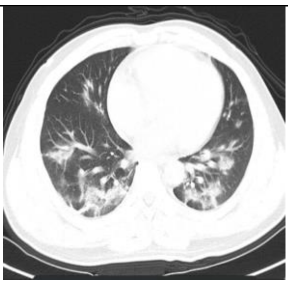
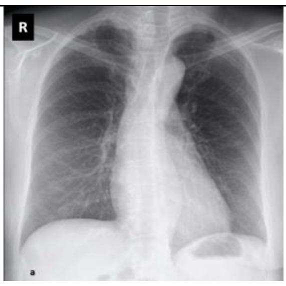
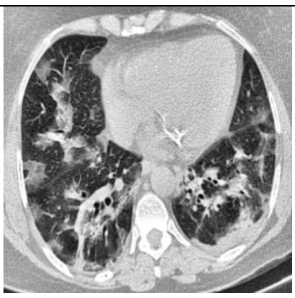
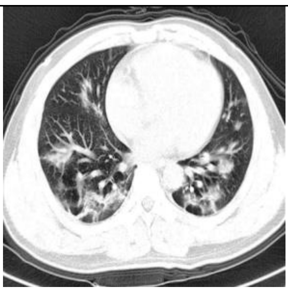
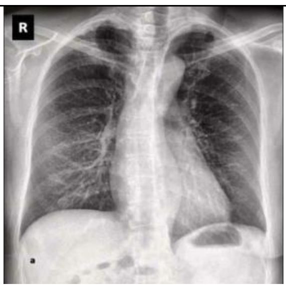
	D1 dataset sample	**D2 dataset sample**	**D3 dataset sample**
Original image			
CLAHE enhanced image			

Figure (4.15) The original and CLAHE enhanced images for three datasets samples images

4.5.2 The Proposed 2D CNN Model Evaluation

To evaluate the efficiency of extracted shape and topology features for enhanced images, experimental results are performed for the proposed 2D CNN model architecture. Accuracy and F1 score explained in Section (2.8) are used as accuracy measures to evaluate the model. As declared before, three different medical images datasets for COVID-19 disease are implemented on proposed model and are applied the experimental results.

The proposed 2D CNN model is trained using an SGD optimizer, 0.01 learning rate, 0.9 momentum, and decay rate with learning (rate/epochs) rate. The model trains for 40 epochs with 32 batch sizes. The size of images used in experiments is (300×300) pixels.

The proposed 2D CNN model is applied in two manners; first with CCL as well as TCL layers and second with TCL layers only. As will explain later the difference in performance is not large although the different in number of trainable parameters used in earlier convolution layers, as well as, extracted specific features specialized for local shape topological pixels relations.

As mentioned in Section (3.4.2) and according to Eq. (3.5) and Eq. (3.6), CCL layer reduces the trainable parameters (weights) of the model. Table (4.11) illustrates a comparison for number of parameters for earlier convolution layers used in the proposed 2D CNN model. First for proposed model using CCL layers and second for the same model architecture using TCL layers (the rest model layers have the same number of parameters). As clear in Table (4.11), the trainable parameters are reduced almost to the desired percentage (in experimental results per=0.33) of the kernel size in CCL layer compared with TCL layers. Fewer trainable parameters faster learning processes and fewer saved model weights are required.

Table (4.11) Number of trainable parameters for earlier convolution layers for the proposed 2D CNN model using CCL layers and using TCL layers

Earlier convolution layers in proposed 2D CNN model	**Number of trainable parameters with kernel size =3×3**	
	CCL layer computed using Eq. (3.6) with per =0.3	**TCL layer computed using Eq. (3.5)**
Layer 1: A=3, F=30	(3×3×0.3) ×3 ×30+30=**273**	3×3×3×30+30=840
Layer 2: A=30, F=30	(3×3×0.3) ×30 ×30+30=**2460**	3×3×30×30+30=8130
Layer 3: A=30, F=60	(3×3×0.3) ×30 ×60+60=**4920**	3×3×30×60+60=16260
Layer 4: A=60, F=60	(3×3×0.3) ×60 ×60+60=**9780**	3×3×60×60+60=32,460
Total number of parameters	**17433**	57690

Different pre-trained DNN models and state-of-arts methods are compared with the proposed model. As mentioned before, for a fair comparison and because the different aspects for each dataset and different preprocessing needed in each dataset, we illustrate each dataset experimental results and discussions individually in the following sub sections.

4.5.2.1 2D CNN model evaluation on D1 dataset

In training mode, the enhanced train part images of the dataset are entered to the proposed model considering the division of datasets explained in Table (4.1). To explain the behavior of the learning process and evaluation of train and test datasets parts during training mode, accuracy and loss metrics are plotted during the training process for this dataset with 70 epochs as illustrated in Figure (4.16).

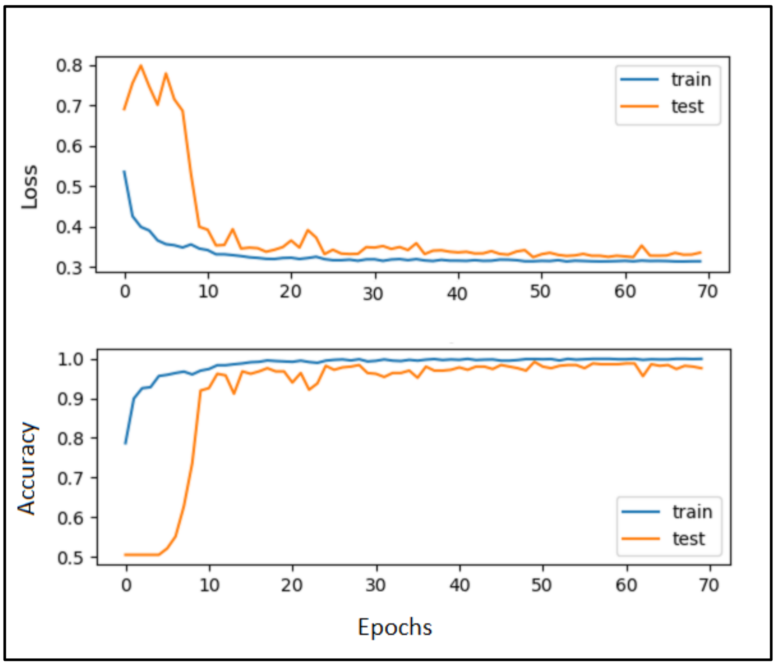


Figure (4.16) Accuracy and loss metrics plots for proposed 2D CNN model training using D1 dataset

In test mode, and to evaluate the proposed 2D CNN model used CCL layers, we present experimental results for prediction of test part of the enhanced dataset in Table (4.4). The proposed model using CCL layers is compared with the same proposed 2D CNN model architecture but using only TCL layers in terms of accuracy and F1-score metrics. Also, the proposed model performance is compared with other pre-trained DNN models like ResNet, GoogleNet, VGG-16, and AlexNet networks, as well as, compared with the xDNN algorithm introduced by Soares et al. [28]. We use the same protocol of division used in the proposed 1D CNN model that presented in [28] and presented in Table (4.4). Figure (4.12) illustrates the results in visual form. As explained in Table (4.4), the experimental results of proposed model using CCL layers is exceed the same model using TCL layers by a small amount in terms of accuracy metric with 98.597% and F1 score metric with 98.595%.

Also, we find that the proposed model gets an efficient performance compared with other pre-trained DNN models like ResNet, GoogleNet, VGG-16, and AlexNet networks, as well as, the xDNN algorithm introduced by Soares et al. [28]. Table (4.12) illustrates the evaluation measures for applying the proposed model with new convolution layers on the D1 dataset.

Table (4.12) Evaluation measures for applying the proposed 2D CNN model on the D1 dataset

Evaluation measure	**Value %**
Accuracy	**98.597**
Precision	**98.7903**
Recall	**98.5935**
f1	**98.5951**

4.5.2.2 2D CNN Model Evaluation on D2 Dataset

Using the same division protocol explained in Table (4.1), in training mode, the proposed model is learned using train part of the COVID-19 dataset part. Figure (4.17) illustrates the behavior learning process and evaluation in terms of accuracy and loss metrics using train and test datasets parts for 70 epochs.

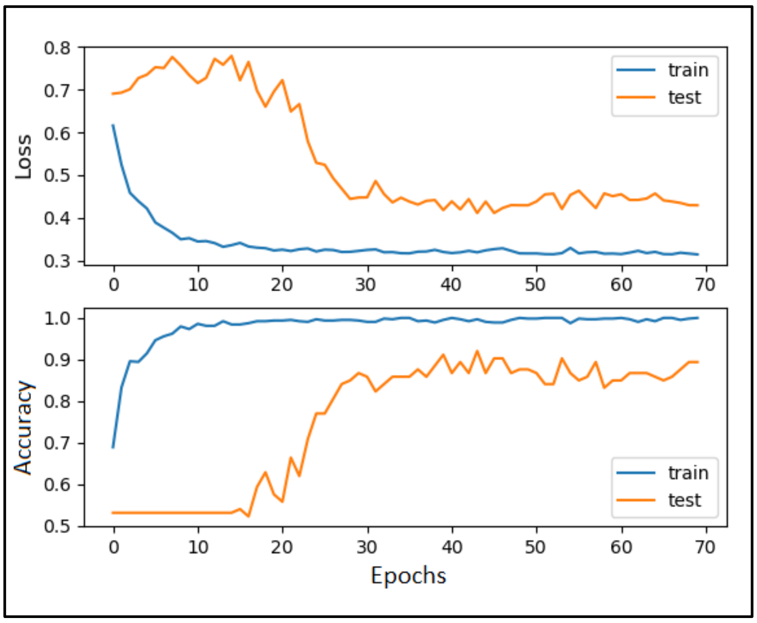


Figure (4.17) Accuracy and loss metrics plots for proposed 2D CNN model training using D2 Dataset

Table (4.6) explains the experiment results of the proposed 2D CNN model used CCL layers on this dataset with accuracy 89.666% and F1 score 89.583% compared with the same proposed model architecture used TCL layers with accuracy 88.495% and F1 score 88.401%. The two models get very close results despite the different in number of trainable parameters.

Also, as clear in Table (4.6), the proposed 2D CNN model used CCL layers outperforms other pre-trained DNN models included VGG16, ResNet18, ResNet50, DenseNet-121, and DenseNet-169 and achieves an efficient performance in terms of accuracy and F1 score.

Further, the results of proposed model get superior performance in terms of accuracy and F1 score compared with other state-of-art approaches included methods presented by Mobiny et al. [30], Polsinelli et al. [29], He et al. [31], and Pedro Silva et al. [32]. Figure (4.14) shows these results visually. Table (4.13) shows more evaluation measures for the proposed model.

Table (4.13): The evaluation measures values for proposed 2D CNN model applied on D2 Dataset

Evaluation measure	Value%
Accuracy	**89.666**
Precision	**89.6**
Recall	**93.33**
F1 score	**89.583**

4.5.2.3 2D CNN Model Evaluation on D3 Dataset

The accuracy and loss metrics are plotted during progressing of training process for this dataset as illustrated in Figure (4.18) with 70 epochs. As explained in Table (4.8), the proposed 2D CNN model using CCL layers get very close results with 93.302% for accuracy and 93.054% for F1 score compared with the same 2D CNN model architecture used TCL layers only although the difference in number of trainable parameters. Also, as shown in Table (4.8), the proposed model gets superior performance by the accuracy and F1 score metrics compared with the baseline challenge [140] and with the pre-trained DNNs included; VGG16, AlexNet, VGG19, Dense-169 networks. Figure (4.16) shows these results visually. Table (4.14) illustrates evaluation measures for applying proposed 2D CNN model on D3 dataset.

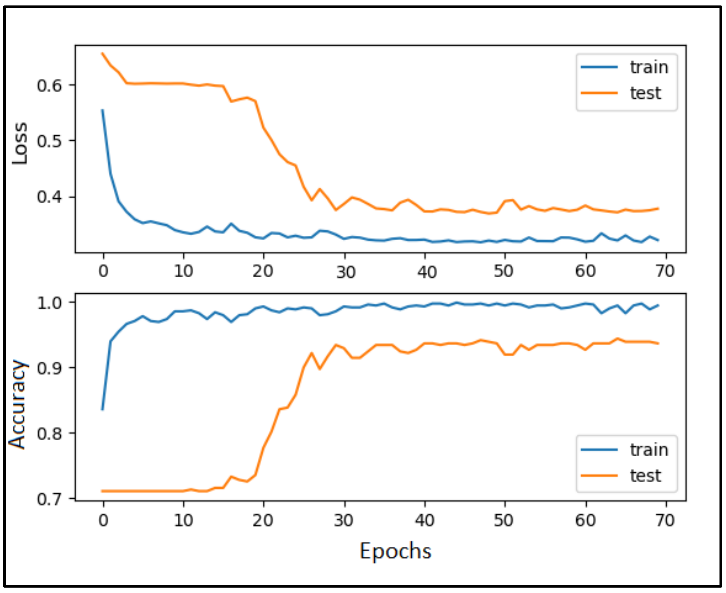


Figure (4.18) Accuracy and loss metrics plots for proposed 2D CNN model training using D3 dataset

Table (4.14): The evaluation measures values for applying 2D CNN model on D3 dataset

Evaluation measure	Value%
Accuracy	**93.302**
Precision	**91.071**
Recall	**86.44**
F1 score	**93.054**

4.6 Medical Image Retrieval Stage Evaluation

In this section, the evaluation for medical image retrieval system is presented. For each image in test part of dataset, there are two features vectors are extracted using first and second stages of the proposed system. The two features vectors are concatenated to get one features vector for each image and store all test images features vectors in one database. Also, in this stage, the features vector of the query image is extracted in the same manner of the test images features vectors extraction. Then, the similarity matching between query features and test images feature is done using cosine similarity metric explained in Eq. (2.18). The system retrieves ranked k more similar images to the query one (k is taken for different values).

The retrieval process is implemented on test part of the dataset images. We select, randomly, set of test images contains 30 images as queries. The mean average precision (mAP) defined in Eq. (2.37) is used as a metric to evaluate the retrieval system. Average precision (AP) as defined in Eq. (2.36) is computed for each query image and mean of them is represent the mAP metric. The experimental results are explained for different ranked k retrieved images, as well as, for all retrieved images in the study cases. The following sections illustrate the experimental results of each dataset separately.

4.6.1 Retrieval System Evaluation Using D1 Dataset

The test part of this dataset contains 497 test images. we select 30 of them randomly as query images. The cosine similarity metric used in this dataset is equal or larger than 0.75. Figure (4.19) illustrates one sample query image with COVID-19 class and top-10 images are retrieved from D1. As clear in Figure (4.19), the top 10 ranked resultant images are similar in an acceptable percentage to the query and are relevant (COVID-19 class). Table (4.15) explains the evaluation of the proposed retrieval system using mAP metric defined in Eq. (2.37) for D1 dataset with different top-k similar retrieved images, as well as, all the retrieved images. As clear in Table (4.15), the result of mAP is high in top- 10 retrieved image about 99.5% and is not 1 because there are some queries among the 30 queries are get mAP percentage less than 1. As the number of ranked retrieved images increases, the mAP result decreases. When computing all retrieved images, the result is giving lowest mAP percentage about 98.5%.

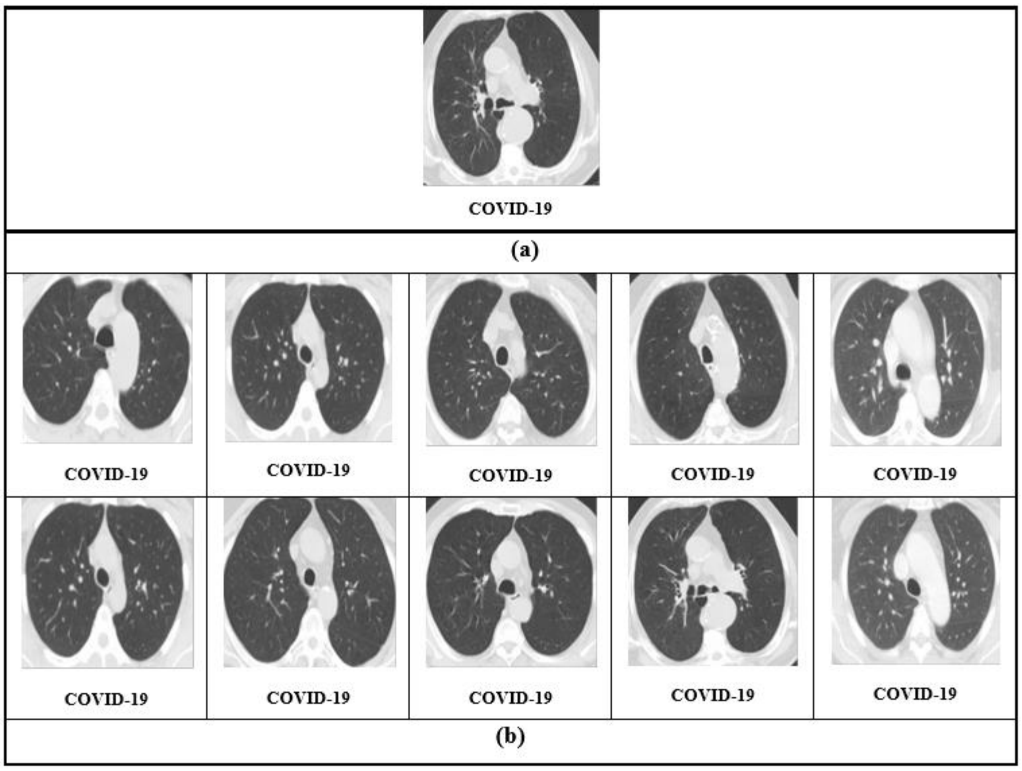


Figure (4.19) Retrieval results for COVID-19 image from D1 dataset: (a) query image, (b) top-10 ranked retrieved images

Table (4.15) Evaluation of the proposed retrieval system for D1 dataset in terms of mAP metric and different top-k

Different top-k similar retrieved images	mAP metric%
10	99.4019
25	99.2953
50	99.1215
100	98.9954
All retrieved	98.5115

2.4.3 Retrieval System Evaluation Using D2 Dataset

The test part of this dataset contains 225 test images. we select 30 of them randomly as query images. In D2 dataset, the cosine similarity metric used is equal or larger than 0.75. Figure (4.20) illustrates one sample query image with COVID-19 class and top-10 retrieved images from D2. As clear in Figure (4.20), the top-10 retrieved images are very similar with a good percentage to query and are relevant (COVID-19 class).

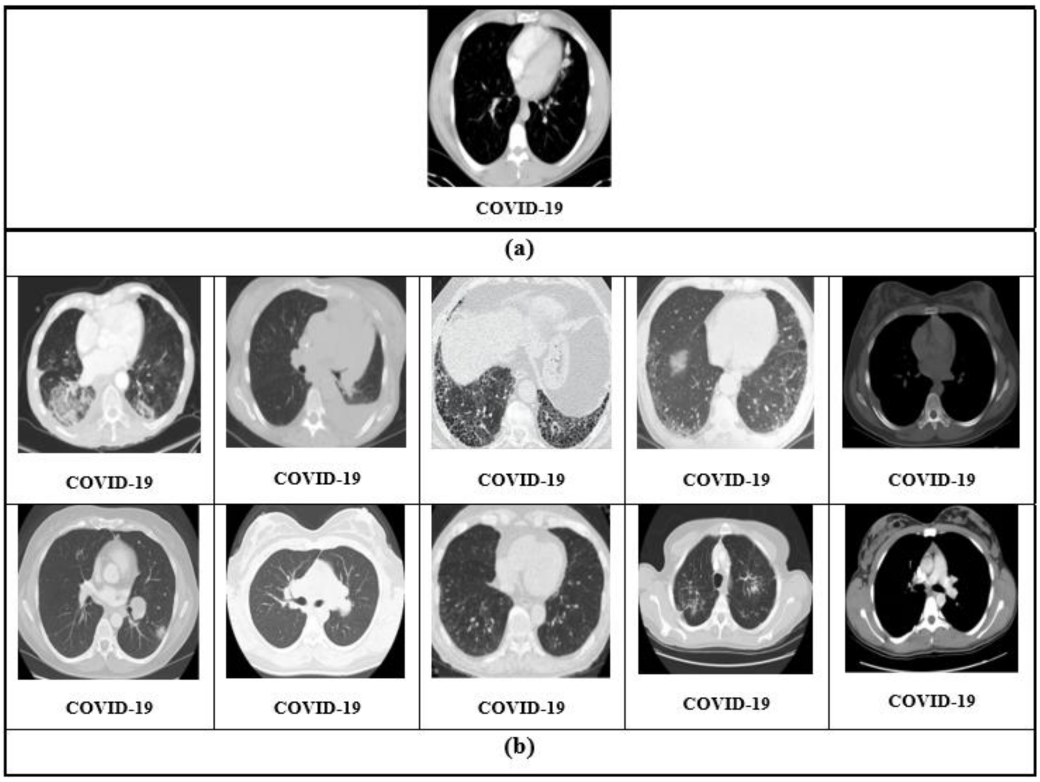


Figure (4.20) Retrieval results for COVID-19 image from D2 dataset: (a) query image, (b) top-10 ranked retrieved images

Table (4.16) explains the evaluation of the proposed retrieval system using mAP metric for D2 dataset with different top-k similar retrieved images, as well as, mAP for all the retrieved images. As clear in Table (4.16), top- 10 retrieved images give a higher result in terms of mAP metric about 94.5% and is not equal to 1 because there are some computed query images are getting percentage less than 1 in terms of mAP. When compute all retrieved images, the mAP results are giving lowest mAP percentage about 84.3%.

Table (4.16) Evaluation of the proposed retrieval system for D2 dataset in terms of mAP metric and different top-k

Different top-k similar retrieved images	mAP metric%
10	94.5559
25	91.0774
50	88.2803
100	85.1595
All retrieved	84.3803

4.6.3 Retrieval System Evaluation Using D3 Dataset

The test part of this dataset contains 408 test images. We select 30 of them randomly as query images. The cosine similarity metric used in this dataset is equal or larger than 0.75. Figure (4.21) illustrates one sample query image with normal class and top-10 retrieved images from D3. As clear in Figure (4.21), all top-10 retrieved images are very similar with acceptable percentage to query and are relevant (normal class).

Table (4.17) explains the evaluation of the proposed retrieval system using mAP metric for D3 dataset with different top-k similar retrieved images, as well as, mAP for all the retrieved images.

As clear in Table (4.17), top- 10 retrieved images archive a higher percentage about 93% in terms of mAP metric and is not equal to 1 because there are some calculated results for query images are getting mAP percentage less than 1. When compute all retrieved images, the results are giving lowest mAP percentage about 85.3%.

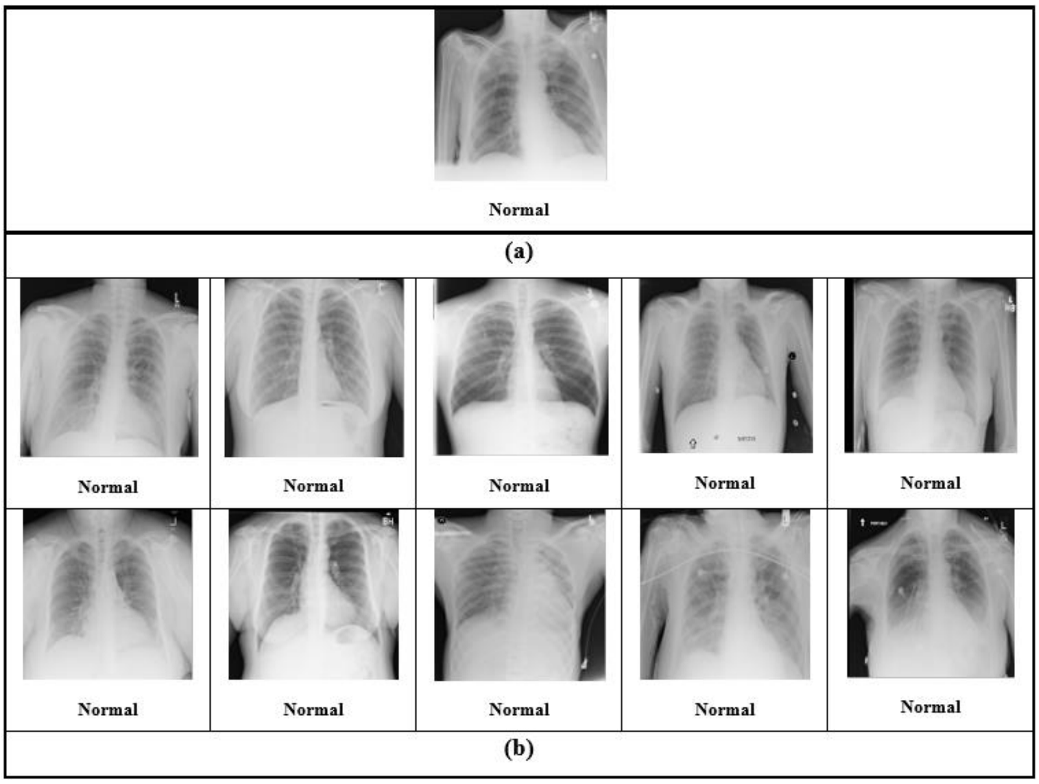


Figure (4.21) Retrieval results for Normal image from D3 dataset: (a) query image, (b) top-10 ranked retrieved images

Table (4.17) Evaluation of the proposed retrieval system for D3 dataset in terms of mAP metric and different top-k

Different top-k similar retrieved images	**mAP metric %**
10	93.0877
25	90.3904
50	89.8866
100	87.5114
All retrieved	85.3191

Chapter Five

Conclusions and Future Works

5.1 Conclusions

During implementation of proposed CBIR system, there are many observations and points are listed as follows:

1. The implementation of the proposed CBIR system extracts features graded from low level features extracted in earlier CNN models until achieving the high-level features extracted in the later layers of CNN models.
2. The proposed system constructs diverse GLCMs matrices with different considerations. Thus, obtaining stack of GLCMs matrices difference in extracted context and texture features to describe each image. These differences include:
   a. Creating three versions of quantized images with 16, 32, and 64 gray levels using proposed quantization method and constructing GLCMs matrices from them rather than one image with 256 gray levels. This leads to a diversity in extracted features from the original image.
   b. Constructing GLCMs matrices in three distances and various directions to obtain different forms of textures ranging from small to large contexts of texture.
3. Concatenating and flattening the GLCMs matrices for quantized images and reducing them dimensions using PCA method make it vectors possible to be inputs to 1D CNN rather than entered directly to 2D CNN. Entering GLCMs matrices to 2D CNN doesn’t useful and doesn’t get good results because GLCM is not a raw image but it is a statistical information for the image. For that, the system proves efficiency and usefulness of using 1D CNN.

4. Using multi inputs 1D CNN with three different pipelines architectures for three different GLCMs vectors increasing the ability to extracting different features for each image.
5. In shape and topology feature extraction stage, whereas, the earlier CNN layers extract the low-level features, dedicating the proposed convolution layer to strengthening this task. The proposed convolution layer specifies number of weights in kernel to extract a specific local shapes and pixel topological relations.
6. The proposed convolution layer reduces the weights number (trainable parameters) in earlier convolution layers kernels for the proposed 2D CNN model. This construction increases the model performance and gives efficient results despite decreasing trainable parameters number.
7. The proposed convolution layer uses a new concept to deal with kernels. Instead of allocating weights for all square kernel matrix, the weights are reduced to form a specific shape. This process simulates the dropout process for the kernels but it applied on weights in kernel rather than the kernel as a whole.
8. In proposed 1D CNN and 2D CNN models, the final extracted features are obtained after passing through some dense fully connected layers. This is useful to reduce the resulted features vector rather than using feature obtained from flattening layer that are very long vectors.
9. Collecting all features in one vector represents an abstraction to describe the images. for that, the system stores short vectors that reduce the memory requirements as well as get a fast and efficient retrieval system.
10. Proposed CBIR system can be implemented on noisy medical images that is using enhancement method, thus increasing the efficiency of 2D CNN model consequently the CBIR system.

11. Proposed CBIR system implemented on three types of medical images datasets. The system achieves accuracy and efficiency with better performance compared with pre-trained DNNs and state of art methods. That is appeared the system ability to deal with various types of medical images.

5.2 Future Works

There are several future ideas and works can be suggested to develop the work of this dissertation and improving the performance of methods used including:

1. Extracting more than one type of texture features such as statistical, structural, transform, and local features and make them as input to the CNN model.
2. Testing the efficiency of the proposed convolution layer with another deep learning applications such as object detection, video tracking, and recommendation system.
3. developing CBIR system using DNN model specialized to enhancing and segmentation the medical images before inputs to the DNN model rather than using handcraft enhancement methods.
4. Developing CBIR system using efficient indexing method for faster retrieval process.

References

References

[1] D. Toussie, N. Voutsinas, M. Finkelstein, M.A. Cedillo, S. Manna, S.Z. Maron, A. Jacobi, M. Chung, A. Bernheim, C. Eber, J. Concepcion, Z.A. Fayad, Y.S. Gupta, "Clinical and chest radiography features determine patient outcomes in young and middle-aged adults with COVID-19," Radiology, vol. 297, pp. e197-e206, 2020.

[2] M.Y. Ng, E.Y.P. Lee, J. Yang, F. Yang, X. Li, H. Wang, M.M.S. Lui, C.S.Y. Lo, B. Leung, P.L. Khong, C.K.M. Hui, K.Y. Yuen, M.D. Kuo, "Imaging profile of the covid-19 infection: Radiologic findings and literature review", Radiol. Cardiothorac. Imaging. vol. 2, pp. 1-9, 2020.

[3] Qayyum, Adnan, Syed Muhammad Anwar, Muhammad Awais, Muhammad Majid, "Medical image retrieval using deep convolutional neural network," Neurocomputing, vol. 266, pp. 8-20, 2017.

[4] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," Medical Image Analysis, vol. 42, pp. 60–88, 2017.

[5] S. Singh, D. Srivastava, S. Agarwal, "GLCM and its application in pattern recognition," in: 5th Int. Symp. Comput. Bus. Intell. ISCBI 2017, pp. 20-25, 2017.

[6] F. Albregtsen, B. Nielsen, H.E. Danielsen, "Adaptive gray level run length features from class distance matrices," Proceedings 15th International Conference on Pattern Recognition, vol. 15, pp. 738-741, 2000.

[7] S.S. Sastry, T.V. Kumari, C.N. Rao, K. Mallika, S. Lakshminarayana, H.S. Tiong, "Transition temperatures of thermotropic liquid crystals from the local binary gray level cooccurrence matrix," Adv. Condens. Matter Phys., 2012, pp. 1-9, 2012.

[8] K. Meskaldji, S. Boucherkha, S. Chikhi, "Color quantization and its impact on color histogram based image retrieval accuracy," in: 2009

1st Int. Conf. Networked Digit. Technol. NDT 2009, pp. 515-517, 2009.

[9] M. Ponti, T.S. Nazaré, G.S. Thumé, “Image quantization as a dimensionality reduction procedure in color and texture feature extraction,” Neurocomputing, vol. 173, pp. 385-396, 2016.

[10] G.H. Liu, J.Y. Yang, Z.Y. Li, “Content-based image retrieval using computational visual attention model,” Pattern Recognit, vol. 48, pp. 2554-2566, 2015.

[11] M. Mustafa, M.N. Taib, Z.H. Murat, S. Lias, “GLCM texture feature reduction for eeg spectrogram image using PCA,” in: Proceeding, 2010 IEEE Student Conf. Res. Dev. - Eng. Innov. Beyond, SCOReD 2010, pp. 426-429, 2010.

[12] Akila, K., L. S. Jayashree, A. Vasuki.: “Mammographic image enhancement using indirect contrast enhancement techniques–a comparative study,” Procedia Computer Science, vol. 47, pp. 255-261, 2015.

[13] Pisano, Etta D., Shuquan Zong, Bradley M. Hemminger, Marla DeLuca, R. Eugene Johnston, Keith Muller, M. Patricia Braeuning, Stephen M. Pizer.: “Contrast limited adaptive histogram equalization image processing to improve the detection of simulated spiculations in dense mammograms,” Journal of Digital Imaging vol. 11, No. 4, PP. 193-200, 1998.

[14] P. Simon, V. Uma, “Deep Learning based Feature Extraction for Texture Classification,” in: Procedia Comput. Sci., 171 (2020), pp. 1680-1687,2020.

[15] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, D.J. Inman, “1D convolutional neural networks and applications: A survey,” Mech. Syst. Signal Process, vol. 151, pp. 1-21, 2021.

[16] J. Li, R. Cui, B. Li, R. Song, Y. Li, Q. Du, “Hyperspectral image super-resolution with 1D-2D attentional convolutional neural network,”

Remote Sens. Vol. 11, pp. 1-21, 2019.

[17] Kotte sowjanya, Pullakura Rajesh Kumar, and Injeti S. Kumar,” Optimal Multilevel Thresholding Selection for Brain MRI Image Segmentation based on Adaptive Wind Driven Optimization”, Measurement, Vol. 130, pp. 340-361, 2018.

[18] R. Srikanth and K. Bikshalu, “Multilevel thresholding image segmentation based on energy curve with harmony Search Algorithm”, Ain Shams Engineering Journal, Elsevier BV on behalf of Faculty of Engineering, 2020.

[19] Wei Liu, Yongkun Huang, Zhiwei Ye, Wencheng Cai, Shuai Yang, Xiaochun Cheng and Ibrahim Frank, “Renyi’s Entropy Based Multilevel Thresholding Using a Novel Meta-Heuristics Algorithm” , MDPI, Applied sciences, Vol. 10, Issue 9, 2020.

[20] Ş. Öztürk, B. Akdemir, “Application of Feature Extraction and Classification Methods for Histopathological Image using GLCM, LBP, LBGLCM, GLRLM and SFTA,” in: Procedia Comput. Sci., vol.132, pp. 40-46, 2018.

[21] K. Shankar, E. Perumal, P. Tiwari, M. Shorfuzzaman, D. Gupta, “Deep learning and evolutionary intelligence with fusion-based feature extraction for detection of COVID-19 from chest X-ray images,” in: Multimed. Syst., Special Issue Paper, pp.1-3, 2021.

[22] Ş. Öztürk, U. Özkaya, M. Barstuğan, “Classification of Coronavirus (COVID-19) from X-ray and CT images using shrunken features,” Int. J. Imaging Syst. Technol. Vol. 31, pp. 5-15, 2021.

[23] Hu, Yifan, and Yefeng Zheng. “A GLCM embedded CNN strategy for computer-aided diagnosis in intracerebral hemorrhage.” arXiv preprint arXiv: pp. 1906.02040, 2019.

[24] Tan, Jiaxing, Yongfeng Gao, Weiguo Cao, Marc Pome’roy, Shu Zhang, Yumei Huo, Lihong Li, and Zhengrong Liang. “GLCM-CNN:

gray level co-occurrence matrix based CNN model for polyp diagnosis." In 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), pp. 1-4, 2019.

[25] Bayar, Belhassen, Matthew C. Stamm.,"Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection," IEEE Transactions on Information Forensics and Security, vol.13, no. 11, PP. 2691-2706, 2018.

[26] Love, Ephy R., Benjamin Filippenko, Vasileios Maroulas, Gunnar Carlsson.: "Topological deep learning", arXiv preprint arXiv: PP. 2101.05778, 2021.

[27] Vigueras-Guillén, Juan P., Joan Lasenby, Frank Seeliger.: "Rotaflip: A New CNN Layer for Regularization and Rotational Invariance in Medical Images," arXiv preprint arXiv: pp. 2108.02704, 2021.

[28] Soares, P. Angelov, S. Biaso, M.H. Froes, D.K. Abe, "SARSCoV-2 CT-scan dataset: a large dataset of real patients CT scans for SARS-CoV-2 identification," medRxiv eprint, vol. 143767, pp. 1-8, 2020.

[29] M. Polsinelli, L. Cinque, G. Placidi, "A light CNN for detecting COVID-19 from CT scans of the chest," Pattern Recognit. Lett. Vol. 140, PP. 95-100, 2020.

[30] Aryan Mobiny, Pietro Antonio Cicalese, Samira Zare, Pengyu Yuan, Mohammad sajad Abavisani, Carol C. Wu, Jitesh Ahuja, Patricia M. de Groot, Hien Van Nguyen, "Radiologist-level covid-19 detection using ct scans with detail-oriented capsule networks," arXiv. preprint 07407, pp. 1-11, 2020.

[31] X. He, X. Yang, S. Zhang, J. Zhao, Y. Zhang, E. Xing, P. Xie, "Sample-efficient deep learning for covid-19 diagnosis based on ct scans," medRxiv vol. 1, pp. 1-10, 2020.

[32] P. Silva, E. Luz, G. Silva, G. Moreira, R. Silva, D. Lucio, D.

Menotti, “COVID-19 detection in CT images with deep learning: A voting-based scheme and cross-datasets analysis,” Informatics Med. Unlocked, vol. 20, pp.1-9, 2020.

[33] Chung, Yu-An, and Wei-Hung Weng. “Learning deep representations of medical images using siamese CNNs with application to content-based image retrieval.” arXiv preprint arXiv: pp. 1711.08490,2017.

[34] Haripriya P, Porkodi R, “An Insight Framework for Content based Medical Image Retrieval using Deep Convolutional Neural Network”, International Journal of Engineering and Advanced Technology, vol. 9, pp.172- 179, 2019.

[35] Maske, Pradnya, and J. A. Kendule. “Medical Image Retrieval using End-to-End Convolution Neural Network,” Journal of Xi'an University of Architecture & Technology, Volume XII, Issue IX, pp.199-206, 2020.

[36] CHORAS, Ryszard S. “Image feature extraction techniques and their applications for CBIR and biometrics systems,” International journal of biology and biomedical engineering, vol. 1, no. 1, pp. 6-16, 2007.

[37] Alazawi, Sundos Abdulameer, Narjis Mezaal Shati, and Amel H. Abbas. “Texture features extraction based on GLCM for face retrieval system.” Periodicals of Engineering and Natural Sciences (PEN) vol. 7, no. 3, pp. 1459-1467, 2019.

[38] K. Xiao, A. L. Liang, H. B. Guan and A. E. Hassanien, “Extraction and application of deformation-based feature in medical images”, Neurocomputing, vol. 120, pp. 177-184, 2013.

[39] Salau, Ayodeji Olalekan, and Shruti Jain. “Feature extraction: a survey of the types, techniques, applications.” In 2019 International Conference on Signal Processing and Communication (ICSC), pp. 158-164. IEEE, 2019.

[40] Lisin, Dimitri A., Marwan A. Mattar, Matthew B. Blaschko, Erik G. Learned-Miller, and Mark C. Benfield. “Combining local and global image features for object class recognition.” In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)-Workshops, pp. 47-47. IEEE, 2005.

[41] Medjahed, Seyyid Ahmed. “A comparative study of feature extraction methods in images classification.” International journal of image, graphics and signal processing vol. 7, no. 3, pp. 16, 2015.

[42] Ping Tian, Dong. “A review on image feature extraction and representation techniques.” International Journal of Multimedia and Ubiquitous Engineering vol. 8, no. 4, pp. 385-396, 2013.

[43] Humeau-Heurtier, Anne. “Texture feature extraction methods: A survey.” IEEE access vol. 7, pp. 8975-9000, 2019.

[44] Pathak, Biswajit, and Debajyoti Barooah “Texture analysis based on the gray-level co-occurrence matrix considering possible orientations.” International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering vol. 2, no. 9, pp.4206-4212, 2013.

[45] R. M. Haralick and K. Shanmugam, “Textural features for image classification,” IEEE Transactions on systems, man, and cybernetics, no. 6, pp. 610-621, 1973.

[46] Gadelmawla, E. S. “A vision system for surface roughness characterization using the gray level co-occurrence matrix.” NDT & e International, vol. 37, no. 7 , pp. 577-588, 2004.

[47] De Siqueira, Fernando Roberti, William Robson Schwartz, and Helio Pedrini. “Multi-scale gray level co-occurrence matrices for texture description.” Neurocomputing vol. 120, pp. 336-345, 2013.

[48] Mukherjee, A. (2016), “Content based image retrieval using GLCM”, International Journal of Innovative Research in Computer and

Communication Engineering, Vol. 4 No. 11, pp. 20142-20149, 2016.

[49] Sebastian V, Bino, A. Unnikrishnan, and Kannan Balakrishnan. “Gray level co-occurrence matrices: generalization and some new features.” arXiv preprint, pp. 1205.4831, 2012.

[50] Verma, Manisha, Balasubramanian Raman, and Subrahmanyam Murala, “Local extrema co-occurrence pattern for color and texture image retrieval,” Neurocomputing, vol. 165, pp. 255-269, 2015.

[51] Mehri, M., Héroux, P., Gomez-Kramer, P., Mullot, R. “Texture feature benchmarking and evaluation for historical document image analysis.” International Journal on Document Analysis and Recognition (IJDAR), vol. 20 no. 1, pp. 1-35, 2017.

[52] Conners, R. W., Harlow, C. A. “A theoretical comparison of texture algorithms.” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 3, pp. 204-222, 1980.

[53] Andaló, Fernanda A., Paulo AV Miranda, R. da S. Torres, and Alexandre X. Falcão. “Shape feature extraction and description based on tensor scale.” Pattern recognition vol. 43, no. 1, pp. 26-36, 2010.

[54] Mingqiang, Yang, Kpalma Kidiyo, and Ronsin Joseph. “A survey of shape feature extraction techniques.” Pattern recognition vol. 15, no. 7, pp. 43-90, 2008.

[55] Upadhyaya, Nikita, and Manish Dixit. “A review: relating low level features to high level semantics in CBIR.” International Journal of Signal Processing, Image Processing and Pattern Recognition vol. 9, no. 3, pp. 433-444, 2016.

[56] Razzak, Muhammad Imran, Saeeda Naz, and Ahmad Zaib. “Deep learning for medical image processing: Overview, challenges and the future.” Classification in BioApps pp. 323-350, 2018.

[57] Baker, Nicholas, Hongjing Lu, Gennady Erlikhman, and Philip J. Kellman. “Local features and global shape information in object

classification by deep convolutional neural networks." Vision research vol. 172, pp. 46-61, 2020.

[58] Brahim, Lejdel, Kazar Okba, and Laurini Robert. "Mathematical framework for topological relationships between ribbons and regions." Journal of Visual Languages & Computing vol. 26, pp. 66-81, 2015.

[59] J. Allen, Time and time again: them any way store present time, Int. J. Intell.Syst. vol. 6, pp. 341–355, 1991.

[60] M.Egenhofer, J.Herring, "A Mathematical Framework for the Definition of Topological Relationships, "in: Proceedings of the 4th International Symposiumon Spatial Data Handling, pp.803–813, 1990.

[61] Clementini, Eliseo, and Paolino Di Felice. "A comparison of methods for representing topological relationships." Information sciences-applications vol. 3, no. 3, no. 149-178, 1995.

[62] Clementini, Eliseo, Jayant Sharma, and Max J. Egenhofer. "Modelling topological spatial relations: Strategies for query processing." Computers & graphics vol. 18, no. 6, pp. 815-822, 1994.

[63] Yang, Liping, Diane Oyen, and Brendt Wohlberg. "Image classification using topological features automatically extracted from graph representation of images." In Proceedings of the 15th International Workshop on Mining and Learning with Graphs (MLG), vol. 1, no. 3, p. 7, 2019.

[64] Love, Ephy, Benjamin Filippenko, Vasileios Maroulas, and Gunnar E. Carlsson. "Topological Convolutional Neural Networks." In NeurIPS 2020 Workshop on Topological Data Analysis and Beyond. 2020.

[65] Jegou, H., Douze, M., Schmid, C., Perez, P. (2010, June). "Aggregating local descriptors into a compact image representation," In Computer Vision and Pattern Recognition (CVPR), IEEE Conference on IEEE, pp. 3304- 3311, 2010.

[66] Sitaula, Chiranjibi, Yong Xiang, Anish Basnet, Sunil Aryal, and Xuequan Lu. "Hdf: Hybrid deep features for scene image representation." International Joint Conference on Neural Networks (IJCNN), IEEE, pp. 1-8, 2020.

[67] Babenko, Artem, and Victor Lempitsky. "Aggregating deep convolutional features for image retrieval." arXiv preprint arXiv, pp. 1510.07493 , 2015.

[68] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. “From generic to specific deep representations for visual recognition.” CoRR, abs, pp. 1406.5774, 2014.

[69] A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. Visual instance retrieval with deep convolutional networks. CoRR, abs, pp.1412.6574, 2014.

[70] Hamreras, Safa, Rafaela Benítez-Rochel, Bachir Boucheham, Miguel A. Molina-Cabello, and Ezequiel López-Rubio. “Content based image retrieval by convolutional neural networks.” In International Work-Conference on the Interplay Between Natural and Artificial Computation, Springer, Cham, pp. 277-286, 2019.

[71] Singh, Anshuman Vikram. “Content-based image retrieval using deep learning,” Rochester Institute of Technology, 2015.

[72] Deng, Li, and Dong Yu. "Deep learning: methods and applications." Foundations and trends® in signal processing vol. 7, no. 3–4 , pp. 197-387, 2014.

[73] Indolia, Sakshi, Anil Kumar Goswami, Surya Prakesh Mishra, and Pooja Asopa. “Conceptual understanding of convolutional neural network-a deep learning approach.” Procedia computer science vol. 132, pp. 679-688, 2018.

[74] Yasar, Huseyin, and Murat Ceylan. “A new deep learning pipeline to detect Covid-19 on chest X-ray images using local binary pattern, dual

tree complex wavelet transform and convolutional neural networks.” Applied Intelligence vol. 51, no. 5, pp. 2740-2763, 2021.

[75] Wu, Jianxin. “Introduction to convolutional neural networks,” National Key Lab for Novel Software Technology. Nanjing University. China vol.5, no. 23, pp. 495, 2017.

[76] Shanmugamani, Rajalingappaa, “Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras,” Packt Publishing Ltd, 2018.

[77] Sagar, Md Motiur Rahman, and Martin Dyrba. “Learning Shape Features and Abstractions in 3D Convolutional Neural Networks for Detecting Alzheimer's Disease.” arXiv preprint arXiv, pp. 2009.05023, 2020.

[78] Kuchera, Michelle P., Raghuram Ramanujan, Jack Z. Taylor, Ryan R. Strauss, Daniel Bazin, Joshua Bradt, and Ruiming Chen. “Machine learning methods for track classification in the AT-TPC,” Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment vol. 940, pp. 156-167, 2019.

[79] Zan, Tao, Hui Wang, Min Wang, Zhihao Liu, and Xiangsheng Gao. “Application of multi-dimension input convolutional neural network in fault diagnosis of rolling bearings.” Applied Sciences vol. 9, no. 13, pp. 2690, 2019.

[80] Gholamalinezhad, Hossein, and Hossein Khosravi. “Pooling methods in deep neural networks, a review,” arXiv preprint arXiv, pp. 2009.07485, 2020.

[81] Ioffe, Sergey, and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” In International conference on machine learning, PMLR, pp. 448-456, 2015.

[82] Suriya, M., V. Chandran, and M. G. Sumithra. “Enhanced deep convolutional neural network for malarial parasite classification,” International Journal of Computers and Applications pp. 1-10, 2019.

[83] Carson, Jason M., Neeraj Kavan Chakshu, Igor Sazonov, and Perumal Nithiarasu. “Artificial intelligence approaches to predict coronary stenosis severity using non-invasive fractional flow reserve,” Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine vol. 234, no. 11, pp. 1337-1350, 2020.

[84] Dewa, Chandra Kusuma. “Suitable CNN weight initialization and activation function for Javanese vowels classification,” Procedia computer science vol. 144 , pp.124-132, 2018.

[85] Es-Sabery, Fatima, Abdellatif Hair, Junaid Qadir, Beatriz Sainz-De-Abajo, Begoña García-Zapirain, and Isabel De La Torre-Díez. “Sentence-level classification using parallel fuzzy deep learning classifier.” IEEE Access vol. 9, pp. 17943-17985, 2021.

[86] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. “Gradient-based learning applied to document recognition.” Proceedings of the IEEE vol. 86, no. 11, pp. 2278-2324, 1998.

[87] Perin, Guilherme, and Stjepan Picek. “On the influence of optimizers in deep learning-based side-channel analysis,” In International Conference on Selected Areas in Cryptography, Springer, Cham, pp. 615-636, 2020.

[88] Keren, Gil. “Neural Network Supervision: Notes on Loss Functions, Labels and Confidence Estimation.” PhD diss, Universität Passau, 2020.

[89] Gowdra, Nidhi, Roopak Sinha, Stephen MacDonell, and WeiQi Yan. “Maximum Categorical Cross Entropy (MCCE): A noise-robust alternative loss function to mitigate racial bias in Convolutional Neural

Networks (CNNs) by reducing overfitting," 2020.

[90] Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks, In Proceedings of the thirteenth international conference on artificial intelligence and statistics," JMLR Workshop and Conference Proceedings, pp. 249-256, 2010.

[91] Thoma, Martin. "Analysis and optimization of convolutional neural network architectures," arXiv preprint arXiv, pp. 1707.09725, 2017.

[92] Zhang, W.; Peng, G.; Li, C.; Chen, Y.; Zhang, Z. "A New Deep Learning Model for Fault Diagnosis with Good Anti-Noise and Domain Adaptation Ability on Raw Vibration Signals," Sensors, vol.17, pp. 425, 2017.

[93] Levent, E.; Turker, I.; Serkan, K. "A Generic Intelligent Bearing Fault Diagnosis System Using Compact Adaptive 1D CNN Classifier. J. Signal Process," Syst. Vol. 91, pp. 179–189, 2018.

[94] Peng, D.; Liu, Z.; Wang, H.; Qin, Y.; Jia, L. "A Novel Deeper One-Dimensional CNN With Residual Learning for Fault Diagnosis of Wheelset Bearings in High-Speed Trains," IEEE Access vol. 7, pp. 10278–10293, 2019.

[95] Abuadbba, Sharif, Kyuyeon Kim, Minki Kim, Chandra Thapa, Seyit A. Camtepe, Yansong Gao, Hyoungshick Kim, and Surya Nepal. "Can we use split learning on 1d cnn models for privacy preserving training," In Proceedings of the 15th ACM Asia Conference on Computer and Communications Security, pp. 305-318, 2020.

[96] Chaerun Nisa, Elsa, and Yean-Der Kuan. "Comparative Assessment to Predict and Forecast Water-Cooled Chiller Power Consumption Using Machine Learning and Deep Learning Algorithms," Sustainability vol.13, no. 2, pp. 744, 2021.

[97] Alkhawlani, Mohammed, Mohammed Elmogy, and Hazem El Bakry. "Text-based, content-based, and semantic-based image retrievals:

a survey," Int. J. Comput. Inf. Technol vol. 4, no. 01, pp. 58-66, 2015.

[98] Marshall, A. Malcom, and S. Gunasekaran. "A survey on image retrieval methods." CIET-ECE DEPT , 2014.

[99] N. Shanmugapriya and R. Nallusamy, "A new content based image retrieval system using GMM and relevance feedback," Journal of Computer Science vol. 10, no. 2, pp. 330-340, 2013.

[100] da Silva Torres, Ricardo, and Alexandre X. Falcao. "Content-based image retrieval: theory and applications," RITA vol. 13, no. 2, pp. 161-185, 2006.

[101] Kokare, Manesh, B. N. Chatterji, and P. K. Biswas. "A survey on current content based image retrieval methods," IETE Journal of Research vol.48, no. 3-4, pp. 261-271, 2002.

[102] Hameed, Ibtihaal M., Sadiq H. Abdulhussain, and Basheera M. Mahmmod. "Content-based image retrieval: A review of recent trends," Cogent Engineering vol. 8, no. 1, pp. 1927469, 2021.

[103] Al-Jubouri, Hanan Ahmed. "Content-based image retrieval: Survey," J. Eng. Sustainable Dev. Vol. 23, pp. 42-63, 2019.

[104] Kuo, Chien-Hao, Yang-Ho Chou, and Pao-Chi Chang. "Using deep convolutional neural networks for image retrieval,"Electronic Imaging 2016, no. 2, pp. 1-6, 2016.

[105] Hu, Rui, Stefan Ruger, Dawei Song, Haiming Liu, and Zi Huang. "Dissimilarity measures for content-based image retrieval," In 2008 IEEE International Conference on Multimedia and Expo, IEEE, pp. 1365-1368, 2008.

[106] Márquez-de-Silva, Sergio, Edgardo Felipe-Riverón, and Luis Pastor Sánchez Fernández. "A simple and effective method of color image quantization." Iberoamerican Congress on Pattern Recognition. Springer, Berlin, Heidelberg, 2008.

[107] Mantina, Mohammed S., and Allen R. Stubberud. "Basics of

sampling and quantization." Handbook of networked and embedded control systems. Birkhäuser Boston, PP. 45-69, 2005.

[108] Cho, Yong-Hyun. "A Performance Improvement of GLCM Based on Nonuniform Quantization Method." Journal of the Korean Institute of Intelligent Systems Vol. 25, no.2, PP.133-138. 2015.

[109] Mehmood, Anam, et al. "A non-uniform quantization scheme for visualization of CT images." Mathematical Biosciences and Engineering Vol. 18. No. 4, PP. 4311-4326, 2021.

[110] A. Othman, T.S.M.T. Wook, S.M. Arif, "Quantization selection of colour histogram bins to categorize the colour appearance of landscape paintings for image retrieval," Int. J. Adv. Sci. Eng. Inf. Technol, vol. 6, pp.930-936, 2016.

[111] Heckbert, Paul. "Color image quantization for frame buffer display." ACM Siggraph Computer Graphics vol. 16, No. 3, pp. 297-307, 1982.

[112] Palus H. "On Color Image Quantization by the K-Means Algorithm." InWorkshop Farbbildverarbeitung ,pp. 58-65, 2004.

[113] S. P. Lloyd, "Least Squares Quantization in PCM," IEEE Trans. Information Theory, vol. 28, no. 2, pp. 129-137, March 1982.

[114] Singh, Gursharn, and Anand Mittal. "Various image enhancement techniques-a critical review." International Journal of Innovation and Scientific Research vol. 10, no. 2, pp. 267-274, 2014.

[115] Image Enhancement Techniques: A Study, Indian Journal of Science and Technology DOI, vol. 8, no. 22, 2015.

[116] Kotkar, Vijay A., and Sanjay S. Gharde. "Review of various image contrast enhancement techniques." International journal of innovative research in Science, Engineering and Technology vol. 2, no. 7 , 2013.

[117] Renjie He, Sheng Luo, Zhanrong Jing and Yangyu Fan "Adjustable Weighting Image Contrast Enhancement Algorithm and Its

Implementation", 6th IEEE Conference on Industrial Electronics and Applications, pp.1750-1754, 2011.

[118] Cheng, Heng-Da, and Huijuan Xu. "A novel fuzzy logic approach to contrast enhancement." Pattern Recognition Vol. 33, no. 5, pp. 809-819, 2000.

[119] Rajput, Seema, and S. R. Suralkar. "Comparative study of image enhancement techniques." International Journal of Computer Science and Mobile Computing Vol. 2, no. 1, pp. 11-21, 2013.

[120] Garg, Priyanka, and Trisha Jain. "A comparative study on histogram equalization and cumulative histogram equalization." International Journal of New Technology and Research vol. 3, no. 9, pp. 263242, 2017.

[121] Chen, Soong-Der, and Abd Rahman Ramli. "Minimum mean brightness error bi-histogram equalization in contrast enhancement." Consumer Electronics, IEEE Transactions on vol. 49, no. 4, pp.1310-1319, 2003.

[122] Pisano, Etta D., Shuquan Zong, Bradley M. Hemminger, Marla DeLuca, R. Eugene Johnston, Keith Muller, M. Patricia Braeuning, and Stephen M. Pizer. "Contrast limited adaptive histogram equalization image processing to improve the detection of simulated spiculations in dense mammograms." Journal of Digital Imaging vol. 11, no. 4, pp.193-200, 1998.

[123] Boudraa, Omar, Walid Khaled Hidouci, and Dominique Michelucci. "Degraded historical documents images binarization using a combination of enhanced techniques." preprint arXiv:1901.09425 ,2019.

[124] Reza, Ali M. "Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement." Journal of VLSI signal processing systems for signal, image and video technology vol. 38, no. 1, pp. 35-44, 2004.

[125] Van der Maaten, Laurens; Postma, Eric; van den Herik, Jaap, "Dimensionality Reduction: A Comparative " , J Mach Learn Res, vol. 10, no. 13, pp. 66–71, 2009.

[126] Uddin, Md Palash, Md Al Mamun, and Md Ali Hossain. "PCA-based feature reduction for hyperspectral remote sensing image classification." IETE Technical Review vol. 38, no. 4, pp. 377-396, 2021.

[127] Al-Omairi, Lamyaa J., Jemal Abawajy, Morshed U. Chowdhury, and Tahsien Al-Quraishi. "An empirical analysis of graph-based linear dimensionality reduction techniques." Concurrency and Computation: Practice and Experience vol. 33, no. 5, pp. e5990, 2021.

[128] Zhang, Yu, and Yuan Jiang. "Multimodal Linear Discriminant Analysis via Structural Sparsity." In IJCAI, pp. 3448-3454, 2017.

[129] Cao, L. J., Kok Seng Chua, W. K. Chong, H. P. Lee, and Q. M. Gu. "A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine." Neurocomputing vol. 55, no. 1-2, pp. 321-336, 2003.

[130] Leskovec, J., A. Rajaraman, J. D. Ullman, J. Leskovec, A. Rajaraman, and J. D. Ullman, "Dimensionality reduction," Mining of Massive Datasets,pp. 415-447, 2014.

[131] Cox, M., Cox, T., "Multidimensional Scaling," In: Handbook of Data Visualization, Springer Handbooks Comp. Statistics. Springer, Berlin, Heidelberg, pp. 315-347, 2008.

[132] Jingyan Wang, Yongping Li, E. Marchiori, Chao Wang, "Iterated Large-Margin Discriminant Analysis for feature Dimensionality Reduction in medical image retrieval," in: International Journal of Biomedical Engineering and Technology, vol. 7, Issue: 2, pp. 116-34, 2011.

[133] B. Ghojogh, M.N. Samad, S.A. Mashhadi, T. Kapoor, W. Ali, F. Karray, M. Crowley, "Feature selection and feature extraction in pattern

analysis," A literature review, preprint arXiv, pp. 1905.02845, 2019.

[134] Anowar, Farzana, Samira Sadaoui, and Bassant Selim. "Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne)." Computer Science Review vol. 40, pp. 100378, 2021.

[135] Hore, Alain, and Djemel Ziou. "Image quality metrics: PSNR vs. SSIM," In 2010 20th international conference on pattern recognition IEEE, pp. 2366-2369, 2010.

[136] Garg, Meenakshi, and Gaurav Dhiman. "A novel content-based image retrieval approach for classification using GLCM features and texture fused LBP variants," Neural Computing and Applications vol. 33, no. 4, pp.1311-1328, 2021.

[137] A. Amyar, R. Modzelewski, H. Li, S. Ruan, "Multi-task deep learning based CT imaging analysis for COVID-19 pneumonia: Classification and segmentation," Comput. Biol. Med, vol. 126, pp. 1-10, 2020.

[138] Kukanov, Ivan, Ville Hautamäki, Sabato Marco Siniscalchi, and Kehuang Li. "Deep learning with maximal Figure-of-merit cost to advance multi-label speech attribute detection," In 2016 IEEE Spoken Language Technology Workshop (SLT), IEEE, pp. 489-495, 2016.

[139] Lai, Tuan, Trung Bui, and Sheng Li. "A review on deep learning techniques applied to answer selection," In Proceedings of the 27th international conference on computational linguistics, pp. 2132-2144, 2018.

[140] Jonathan H. Chan, Puttipong Thammachart, Vasin Virasak, Phubeth Rodklang, DLAI3 Hackathon: COVID-19 chest Xray challenge, version 1.0, 2020. https://www.kaggle.com/c/dlai3/data.

الخلاصة

في الآونة الأخيرة ، أصبح استخدام المعلومات الطبية في المستشفيات ضرورة لا غنى عنها ويتزايد بسرعة مع التزايد الكبير في عدد البيانات التصوير التشخيصية. لذا ظهرت الحاجة إلى تطوير نظم معلومات يساعد في الحصول على هذه المعلومات وجمعها وإدارتها ومعالجتها في مجاميع للبيانات المرتبة. يعد استرجاع الصور القائم على المحتوى (CBIR) أحد الأنظمة المهمة التي تم تصميمها للمساعدة في إدارة عملية استرداد الصور المماثلة للصورة المراد البحث عنها. اقترحت هذه الأطروحة نظام CBIR تلقائي يعتمد على العديد من الصفات المستخلصة التي يتم تتدرج من صفات منخفضة المستوى بما في ذلك الملمس والشكل والعلاقات الطوبولوجية وصولا إلى الصفات عالية المستوى المستخلصة باستخدام طرق التعلم العميق .ويشمل النظام المقترح ثلاث مراحل.

المرحلة الأولى هي استخلاص ميزة الملمس التي تتكون من العديد من الخطوات. الخطوة الأولى هي اقتراح طريقة تكميم غير منتظمة لتحويل الصورة الرمادية إلى ثلاثة إصدارات من الصورة مع مستويات رمادية 16 و 32 و 64. تعتمد هذه الطريقة على الاختيار الأمثل لقيم العتبة مع مراعاة افضلية الـ PSNRبين الصورة الناتجة والصورة الاصلية. الطريقة تعتمد على إحدى خوارزميات برمجة الطماع وهي استراتيجية فرق تسد. تتضمن الخطوة الثانية إنشاء العديد من مصفوفات GLCMs للصور الثلاث المحددة كميا باستخدام ثلاث مسافات 1 و 2 و 3 ولاتجاهات سياق مختلفة. يتم تسطيح مصفوفات GLCMsالناتجة وتسلسلها ثم تقليصها باستخدام طريقة PCA للحصول على ثلاثة متجهات بأطوال مختلفة من المعلومات الإحصائية. تتضمن الخطوة الثالثة إدخال متجهات GLCMs الثلاثة إلى نموذج المقترح 1D CNN المتعدد المدخلات الذي يحتوي على ثلاثة خطوط أنابيب ذات معماريات مختلفة للحصول أخيرا على متجه واحد من الصفات لتمثيل الصورة.

المرحلة الثانية هي استخلاص صفات الشكل والعلاقات الطوبولوجية. تتضمن هذه المرحلة اقتراح طبقة الالتفاف تستخدم توزيعا عشوائيا لبعض الأوزان المقيدة (المعاملات القابلة للتدريب) في الفلتر. تقع بشكل عشوائي في مواقع محددة مع مراعاة نسبة مئوية المحددة من قبل المستخدم. يتم تقديم هذه الطبقة لاستخلاص نوع خاص من الصفات مع مراعاة الشكل المحلي لصورة فرعية (نافذة) والعلاقات الطبولوجية بين بيكسلات المجموعة. يتم تنفيذ هذه الطبقة باستخدام نموذج CNN مقترح يستخدم طبقات الالتفاف المقترحة بالإضافة إلى طبقات CNN التقليدية.

المرحلة الثالثة هي عملية استرجاع الصور. تتضمن هذه المرحلة استخلاص صفات الصور لمجموعة البيانات باستخدام المرحلتين السابقتين في مرحلة عدم الاتصال. بعد ذلك، تسلسل هذه

الصفات في متجه واحد لكل صورة في مجموعة البيانات وتخزينها في قاعدة بيانات واحدة. أيضا ، في المرحلة عبر الإنترنت، استخلاص صفات صورة الاستعلام باستخدام نفس المرحلتين السابقتين. بعد ذلك ، يتم إجراء مطابقة التشابه بين الصفات في قاعدة البيانات وصفات الاستعلام لاسترجاع الصور المرتبة الاكثر مشابهة للاستعلام.

في النتائج التجريبية ، تتفوق طريقة التكميم المقترحة على الطرق التقليدية الأخرى من حيث مقاييس PSNR و SSIM لمستويات رمادية مختلفة. يتم تنفيذ وتقييم نظام CBIR المقترح باستخدام ثلاثة أنواع من مجموعات البيانات الطبية المخصصة لمرض COVID-19 لكل منها على حدة. يتم تنفيذ نماذج 1D CNN و 2D CNN المقترحة باستخدام مجموعات البيانات الثلاث هذه. تثبت النماذج كفاءة الصفات المستخلصة وتتفوق على DNNs المدربة مسبقا والطرق المطبقة حديثا. تحقق نماذج CNN المقترحة دقة تبلغ حوالي 98٪ و 89٪ و 93٪ لمجموعات البيانات الثلاث ، على التوالي. علاوة على ذلك ، يحصل نظام CBIR المقترح على نتائج فعالة ومقبولة ويحقق لأفضل 10 نتائج مسترجعة 99٪ و 94٪ و 93٪ من حيث مقياس mAP لمجموعات البيانات الثلاث ، على التوالي.

جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جـامعـــة بـابـــل

كلية تكنولوجيا المعلومات

قسم البرمجيات

استرجاع الصور المعتمد على المحتوى القائم على النسيج والشكل باستخدام الشبكة العصبية العميقة

أطروحة مقدمة
الى مجلس كلية تكنولوجيا المعلومات - جامعة بابل وهي جزء من متطلبات نيل درجة الدكتوراه فلسفة في تكنولوجيا المعلومات / برمجيات

من قبل

ايلاف علي عبود حسن

بإشراف

أ . د . توفيق عبد الخالق عباس عبد الرضا

1444هـ **2022م**