



A complete framework for accurate recognition and prognosis of COVID-19 patients based on deep transfer learning and feature classification approach

Hossam Magdy Balaha¹ · Eman M. El-Gendy¹ · Mahmoud M. Saafan¹

Published online: 29 January 2022

© The Author(s), under exclusive licence to Springer Nature B.V. 2021

Abstract

The sudden appearance of COVID-19 has put the world in a serious situation. Due to the rapid spread of the virus and the increase in the number of infected patients and deaths, COVID-19 was declared a pandemic. This pandemic has its destructive effect not only on humans but also on the economy. Despite the development and availability of different vaccines for COVID-19, scientists still warn the citizens of new severe waves of the virus, and as a result, fast diagnosis of COVID-19 is a critical issue. Chest imaging proved to be a powerful tool in the early detection of COVID-19. This study introduces an entire framework for the early detection and early prognosis of COVID-19 severity in the diagnosed patients using laboratory test results. It consists of two phases (1) Early Diagnostic Phase (EDP) and (2) Early Prognostic Phase (EPP). In EDP, COVID-19 patients are diagnosed using CT chest images. In the current study, 5, 159 COVID-19 and 10, 376 normal computed tomography (CT) images of Egyptians were used as a dataset to train 7 different convolutional neural networks using transfer learning. Data augmentation normal techniques and generative adversarial networks (GANs), CycleGAN and CCGAN, were used to increase the images in the dataset to avoid overfitting issues. 28 experiments were applied and multiple performance metrics were captured. Classification with no augmentation yielded 99.61% accuracy by EfficientNetB7 architecture. By applying CycleGAN and CC-GAN Augmentation, the maximum reported accuracies were 99.57% and 99.14% by MobileNetV1 and VGG-16 architectures respectively. In EPP, the prognosis of the severity of COVID-19 in patients is early determined using laboratory test results. In this study, 25 different classification techniques were applied and from the different results, the highest accuracies were 98.70% and 97.40% reported by the Ensemble Bagged Trees and Tree (Fine, Medium, and Coarse) techniques respectively.

Keywords Computed tomography (CT) · Convolutional neural network (CNN) · COVID-19 · Data augmentation · Generative adversarial networks (GAN) · Image classification · Machine learning (ML) · Transfer learning (TL)

✉ Hossam Magdy Balaha
hossam.m.balaha@mans.edu.eg

Extended author information available on the last page of the article

1 Introduction

By the end of 2019, and specifically on 12 December 2019, a patient suffering from pneumonia came to a hospital in Wuhan, Hubei province, China. This patient was diagnosed with severe acute respiratory syndrome (SARS) (Wu 2020). The symptoms of the severe respiratory syndrome included dizziness, fever, sore throat, and cough (Singhal 2020). Wuhan became the center of that unknown virus (Wang et al. 2020), which was named COVID-19 by the World Health Organization (WHO) (Sohrabi 2020). WHO announced that the novel coronavirus resulting from the SARS coronavirus 2 (SARS-CoV-2). It was classified as a pandemic on 30 January 2020 (Xu and Li 2020).

Now, COVID-19 is the topic of the hour, with its negative effect on life and the economy. Scientists expect another wave of the virus attack that could be more severe. So, it is necessary to diagnose patients and isolate them early. Although Reverse Transcription Polymerase Chain Reaction (RT-PCR) is the standard test for confirming COVID-19 cases, this test suffers from some false positive or false negative cases (Roy 2021). Chest imaging, on the other hand, is a rapid and accurate alternative to diagnose COVID-19.

Chest images, [i.e., computed tomography (CT)] and X-rays play an important role in the early detection and diagnosis of the disease (Ozturk 2020). Chest X-ray can be used to reveal pulmonary infection with details such as location, shape, and volume, whereas CT-Scan presents a detailed image of the Alveoli (air sacs) in the lungs (Panwar 2020). Both CT and X-ray can be and have already been applied for the diagnosis of COVID-19 in several studies (Horry 2020; Chowdhury 2020; Jamshidi 2020). However, CT images are preferable in the detection of COVID-19 because they are more sensitive than X-ray images in the investigation of COVID-19 patients (Wong 2020). Chest CT images of COVID-19 patients indicates peripheral, bilateral, and basal ground-glass opacities with consolidation (Parekh et al. 2020).

Convolutional neural networks (CNN) are a class of deep learning approach that have been applied in computer vision related tasks due to their incredible ability to extract and recognize features from images automatically (Xu et al. 2014). CNN also proved to have excellent classification performance (Zhang 2018). Applications of CNNs include musculoskeletal tissue segmentation (Liu 2018), skin cancer classification (Dorj et al. 2018), lung tumor detection (Kasinathan 2019), acoustic scene classification (Han et al. 2017), Arabic handwritten recognition (Elleuch et al. 2016), face detection (Li et al. 2015) and recognition (Lawrence et al. 1997), face skin disease classification (Wu 2019), brain imaging classification (Yuan et al. 2018), breast cancer detection (Wang 2019), EEG signal classification (Xu 2019), and image dehazing (Li et al. 2018).

Constructing a CNN from scratch is a completely expensive task in terms of both time and resources. On the other hand, the resulting accuracy of such models is not very good, especially when the data available for the training process is limited. So, if there is a problem with the availability of training data, it is not preferred to build the model from the beginning. Alternatively, we can use a pre-trained CNN model and just modify its parameters. In this case, the model has already been trained, maybe on a completely different dataset, but the structure and number of layers have already been determined. All the remaining is to train, or fine-tune, the model on the new data to determine the new weights of the network (Liu 2017).

This approach is called transfer learning. It came from the fact that we “transfer” or reuse the “learning” or knowledge of already pre-trained models on other problems. In other words, we can use the experience of the pre-trained model to solve new problems.

This approach has been applied successfully in different researches (Lan et al. 2019; Cao et al. 2018; Vo et al. 2019). An important advantage of using transfer learning is the high accuracy and precision as compared to creating new models.

When dealing with text data classification, CNN is not actually the optimal classification approach. Instead, other machine learning methods can be applied for this kind of classification problem (Jianqiang and Xiaolin 2017; Van Der Walt and Eloff 2018; Cui 2019).

Machine learning (ML) is a subfield of Artificial Intelligence (AI) that is based on the fact that machines can “learn” or “train” to find the solution to a given problem. Although there are different ML algorithms, they all share the same basic principles.

First, data is collected from different sources. Second, data is formatted. Third, data is introduced to the ML classifier. Forth, the classifier chooses the most suitable features based on the input training data. Fifth, the classifier is trained on this data. Finally, the classifier becomes ready for solving the required problem.

In the current work, we propose a complete framework for the accurate diagnosis and prognosis of COVID-19 patients using CT chest images and laboratory test results. This framework allows not only diagnosing the patient with COVID-19 but also predicting the severity of the COVID-19 in patients. The working mechanism is graphically summarized in Fig. 1.

Therefore, our framework consists of two phases (1) Early Diagnostic Phase (EDP) and (2) Early Prognostic Phase (EPP). EDP is the first phase where the patient entering the hospital is diagnosed using chest CT images. EDP is divided into three steps: First, we used image preprocessing techniques in order to prepare the data and make it more applicable. Second, we used data augmentation normal techniques and generative adversarial networks (GANs), CycleGAN, and CCGAN, as a trail to increase the images in the dataset to avoid the overfitting issue. Finally, the resulting dataset is used to train 7 different pre-trained CNN architectures. They used models are EfficientNetB7, InceptionV3, ResNet-50, VGG-16, VGG-19, Xception, and MobileNetV1.

After the patient is diagnosed with positive COVID-19, it is important to predict the severity of the disease in order to know whether the patient will need a transfer to an intensive care unit (ICU) as early as possible or not. This is the goal of EPP. In this phase, we use laboratory test results extracted from patients’ records as prognostic markers of how severe pneumonia will be so that we can rescue the patient and decrease the mortality rate resulting from COVID-19.

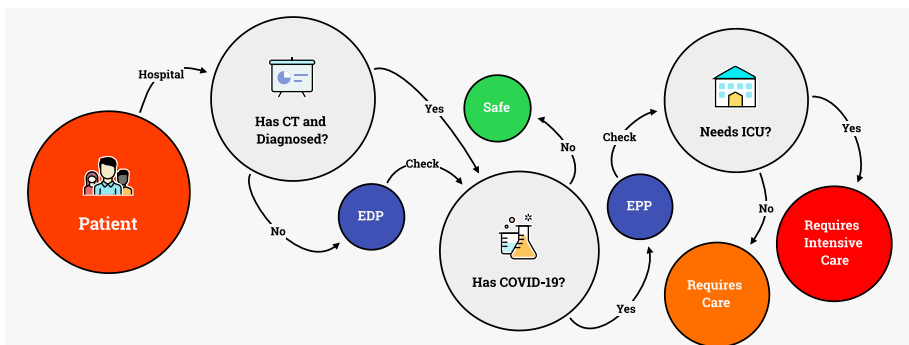


Fig. 1 Graphical summarization of the working mechanism of the suggested framework

25 different ML classification techniques are applied, namely Trees (Fine Tree, Medium Tree, and Coarse Tree); Discriminant (Linear Discriminant and Quadratic Discriminant); Regression (Logistic Regression); Naïve Bayes (Gaussian Naïve Bayes and Kernel Naïve Bayes); SVM (Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM); KNN (Fine KNN, Medium KNN, Coarse KNN, Cosine KNN, Cubic KNN, and Weighted KNN); and Ensemble (Ensemble Boosted Trees, Ensemble Bagged Trees, Ensemble Subspace Discriminant, Ensemble Subspace KNN, and Ensemble RUSBoosted Trees).

Our main purpose behind using these models is to compare the results of the different ML methods so that we can select the model with the best results.

The main contributions of the current study are:

- A complete framework for the accurate diagnosis and prognosis of COVID-19 patients using CT chest images and laboratory test results is proposed. It consists of two phases, namely (1) Early Diagnostic Phase (EDP) and (2) EPP.
- During EDP, the used images are raw data collected from Egyptian Radiology centers. During EPP, laboratory test results extracted from patients' records are used as prognostic markers of how severe pneumonia will be.
- During EDP, image pre-processing techniques and data augmentation (using normal techniques and GANs) are used to increase the images in the dataset.
- 7 different pre-trained CNN models, namely EfficientNetB7, InceptionV3, ResNet-50, VGG-16, VGG-19, Xception, and MobileNetV1 are trained using the TF approach.
- 25 different ML classification techniques were applied, namely Trees (Fine Tree, Medium Tree, and Coarse Tree); Discriminant (Linear Discriminant and Quadratic Discriminant); Regression (Logistic Regression); Naïve Bayes (Gaussian Naïve Bayes and Kernel Naïve Bayes); SVM (Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM); KNN (Fine KNN, Medium KNN, Coarse KNN, Cosine KNN, Cubic KNN, and Weighted KNN); and Ensemble (Ensemble Boosted Trees, Ensemble Bagged Trees, Ensemble Subspace Discriminant, Ensemble Subspace KNN, and Ensemble RUSBoosted Trees).
- Results of different experiments (in EDP and EPP) are reported using different performance metrics.

The rest of the paper is divided into these sections. Section II discusses the related work in applying CNN on chest images. The idea and structure of CNN are described in Section III. In Section IV, the different machine learning classification techniques are explained. Our diagnostic and prognostic framework is proposed in Section V. Results are reported and discussed in Section VI. Section VII presents the limitations of our study while Section VIII presents the conclusions and future work.

The different study sections are subsections of the current study are summarized and infograph-ed in Fig. 2. It can be used as a reference for the reader to map the manuscript content and flow.

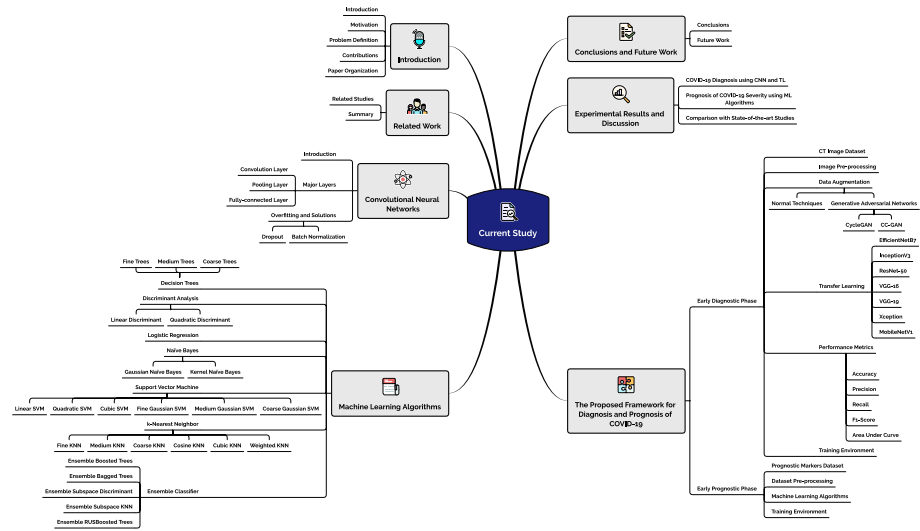


Fig. 2 The current study summarization of the different sections and subsections

2 Related work

A massive effort has been made from the start of the crisis to apply AI techniques to identify patients with COVID-19 from chest images. Apostolopoulos and Mpesiana (2020) used CNN on a dataset of 1, 427 X-ray images including both COVID-19 disease and other pneumonia diseases, obtaining an accuracy of 96.78%.

DarkCovidNet model was proposed in Ozturk (2020) as a classifier for COVID-19 and pneumonia diseases in X-rays, obtaining an accuracy of 98.08% for binary classifications. Brunese et al. (2020) proposed a three-step process that was based on binary classification using deep learning on X-ray chest images. They used a dataset of 6, 523 images and achieved an average accuracy of 97%. CoroNet, a CNN model, was proposed by Khan et al. (2020). They achieved an accuracy of 89.6%.

Ardakani et al. (2020) applied 10 CNN structures to 1, 020 CT images. The best performance was achieved by ResNet-101 with an accuracy of 99.51% and Xception with an accuracy of 99.20%. Zhang (2020) used the ResNet-18 CNN model to classify 361, 221 CT images with COVID-19, normal, and Influenza. They achieved an accuracy of 92.49%.

Hu (2020) applied a weakly supervised CNN to CT images. They achieved an accuracy of 96.2%. Cascaded deep learning classifiers applied to computer-aided design (CAD) systems were proposed in Karar et al. (2021). A dataset of 306 X-ray images was used and they achieved an accuracy of 99.9%.

Nour et al. (2020) proposed an intelligence diagnosis model CNNs. A dataset of 2, 905 chest images was used, and they achieved an accuracy of 98.97%. Waheed et al. (2020) proposed a model called CovidGAN based on Auxiliary Classifier Generative Adversarial Network. They used a dataset of 1, 124 images and achieved an accuracy of 95%.

Sakib et al. (2020) proposed a DL-CRC framework to diagnose COVID-19 from pneumonia and healthy cases. Their achieved accuracy is 94.61%. Rajaraman (2020) proposed an iteratively pruned model that combines different ensemble schemes to enhance the performance. They could achieve 99.01% accuracy.

Table 1 presents a summary of the discussed previous studies. They are just examples of the massive work concerning the application of deep learning on the detection of COVID-19 (Zhang et al. 2021a, b; Sharma 2021; Mukherjee et al. 2021; Abdulkareem et al. 2021; Le et al. 2021; Dansana et al. 2020)

3 Convolutional neural networks (CNNs)

The “ability to learn” is the main advantage of the feedforward neural networks. However, in the case of images, there are thousands of features from millions of images, which means a huge learning capacity.

A CNN is a variation of the Artificial Neural Networks (ANN) designed to deal with images. CNN is capable of extracting the mass of features in the images. Despite the huge number of features to extract, CNNs can learn only the important features using small connections and parameters (Krizhevsky et al. 2012).

However, these types of networks take a very long time for the training process due to the increased complexity in the structure of the network and the massive number of extracted features (Qin et al. 2018). A typical CNN consists of three types of layers (1) convolution, (2) pooling, and (3) fully-connected (FC) layers (Yamashita et al. 2018).

Convolution layer A convolution layer is responsible for the feature extraction. It performs mathematical operations such as convolution and passes its output through an activation function. Filters (i.e., neurons at this layer) convert the input images into output feature maps (Abdulazeem et al. 2021a). The usage of an activation function after the convolution layer is to compute the output of the neurons in this layer. The type of activation function determines the shape of the output of this layer. The most commonly used activation function in CNN is the Rectified Linear Unit (ReLU) (Balaha et al. 2021c).

Pooling layer The role of the pooling layer is to decrease the dimensionality of the feature maps. The commonly used type of pooling operation is max-pooling. In this type of pooling, patches are extracted from the input feature maps and the extreme number in every patch is produced. The output of the pooling layer is then flattened and transformed into a vector of numbers (Balaha et al. 2021d).

Fully-connected (FC) Layer This output vector is connected to fully-connected layers (i.e., dense layers). The name fully-connected came from the fact that input is connected to every output by a learnable weight. The final layer must have a number of output nodes equal to the number of classes (Bahgat et al. 2021).

The most commonly used optimization algorithm is called the backpropagation algorithm. The initial weights and biases of any layer in CNN can have a huge effect on the network performance. The most commonly used weight initializers are Glorot (Glorot and Bengio 2010), and He initializers (He et al. 2015).

The most common challenge that faces CNNs is known as overfitting. Overfitting occurs when a model learns statistical regularities of a specified training set so that by the end of the training, the model learns noise instead of correct data. This problem can reduce the performance of a CNN, especially when tested with new data.

Several solutions were proposed to eliminate this challenge. One solution is to use Dropout (Hinton et al. 2012). Dropout means selecting random weights and setting them to zero during the training. By this technique, the model becomes not susceptible to particular weight values.

Table 1 Summary of different researches on COVID-19 using Deep Learning

| Reference | Approach | Dataset size | CT or X-rays | Best accuracy |
|------------------------------------|--|--------------|--------------|---|
| Apostolopoulos and Mpesiana (2020) | Convolutional neural network | 1, 427 | X-ray | 96.78% |
| Ozturk (2020) | DarkCovidNet | 127 | X-ray | 98.08% |
| Brunese et al. (2020) | Deep Learning | 6, 523 | X-ray | 97% |
| Khan et al. (2020) | CoroNet CNN Model | 1, 300 | X-ray | 89.6% |
| Ardakani et al. (2020) | AlexNet, VGG-16, VGG-19, SqueezeNet, GoogleNet, MobileNet-V2, ResNet-18, ResNet-50, ResNet-101, and Xception | 1, 020 | CT | 99.51% in ResNet-101 and 99.20% in Xception |
| Zhang (2020) | ResNet-18 | 361, 221 | CT | 92.49% |
| Hu (2020) | Weakly Supervised CNN | 150 | CT | 96.2% |
| Karar et al. (2021) | Cascaded Deep Learning Classifiers + CAD Systems | 306 | X-ray | 99.9% |
| Nour et al. (2020) | Intelligence Diagnosis Model CNNs | 2, 905 | X-ray | 98.97% |
| Waheed et al. (2020) | CovidGAN | 1, 124 | X-ray | 95% |
| Sakib et al. (2020) | DL-CRC Framework | 33, 291 | X-ray | 94.61% |
| Rajaraman (2020) | VGG-16, VGG-19, and Inception-V3 | 16, 700 | X-ray | 99.01% |

Another solution to overfitting is to apply Batch normalization (Ioffe and Szegedy 2015). It means to adaptively normalize the input values of the next layer. This can reduce overfitting, improve the flow through the network, permit the use of high-value learning rates, and lessen the reliance on the weight initialization process.

Figure 3 is an illustrative example of the different layers of a CNN applied in the detection of COVID-19.

4 Machine learning algorithms

In this section, the used 25 ML classification algorithms are explained. We have selected these algorithms specifically because they are the most common ML classification algorithms. All of these models share the same aim (1) using the labeled training sets to train the algorithm or (2) build a model capable of recognizing and classifying unknown patterns.

4.1 Decision trees

Decision trees are ML algorithms that are used to predict results after learning the dataset. The main idea of decision trees is to split the search space recursively and make a simple model for each partition. The split process is graphically represented as a tree, hence the name decision tree (Loh 2011).

The structure of the decision tree is similar to a flowchart, in which each node represents a test, while each branch represents the result of the test, and each leaf represents a class number. The difference between Fine, Medium, and Coarse trees is the number of maximum splits available. Coarse trees have the minimum splits, while fine trees have the maximum splits.

4.2 Discriminant analysis

The discriminant analysis (DA) classifier was introduced by R. Fisher (1936). It is considered a simple but well-known classifier. Linear discriminant analysis and quadratic discriminant analysis are the main types of DA classifiers. The difference between the two types is in the decision surface. In the case of the Linear discriminant analysis classifier, and as the name indicates, the decision surface is linear. On the other hand, and for quadratic discriminant analysis classifier, the decision surface is nonlinear (Tharwat 2016).

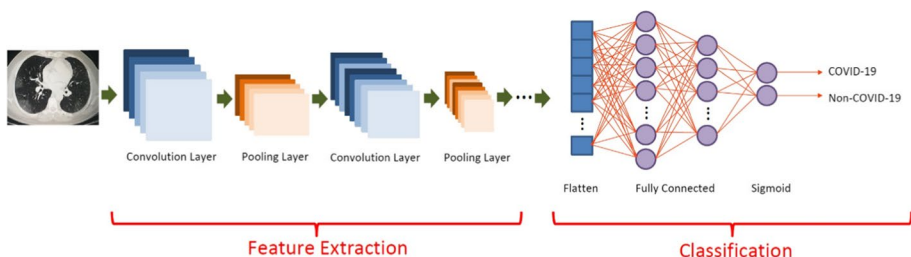


Fig. 3 COVID-19 detection via a convolutional neural network

The main drawback of the DA classifier is the singularity problem, in which the dimensions of the problem are greater than the total samples in each class. Therefore, DA becomes unable to compute the discriminant functions. Different solutions were proposed to overcome this problem, including regularized linear discriminant analysis (Friedman 1989) and the subspace method (Belhumeur et al. 1996).

For a set of m features representing a sample in the dimensional pattern space R_m , there are c discriminant functions $F = \{f_1, f_2, \dots, f_c\}$, where c is the total number of categories (i.e., classes). DA classifier uses these functions to calculate the sector of each class and hence the borders limiting the different classes.

For a class c_i at index i with region ω_i , and for the unknown pattern $X_u = \{x_1, x_2, \dots, x_l\}$, if this pattern belongs to class c_i , then the corresponding formula is shown in Eq. 1:

$$f_{c_i}(X_u) > f_{c_k}(X_u) \rightarrow (c_i, c_k = 1, 2, \dots, c), (i \neq k) \tag{1}$$

where l is the number of features (i.e., parameters) in X_u , c_k is a class at index k . A special case (shown in Eq. 2) happens when X_u lies on the border between two classes.

$$f_{c_i}(X_u) = f_{c_k}(X_u) \rightarrow (c_i, c_k = 1, 2, \dots, c), (i \neq k) \tag{2}$$

The probability of finding X_u in the class region ω_i is called the posterior probability and is calculated using Eq. 3.

$$P(\omega_i|X_u) = \frac{P(X_u|\omega_i) \times P(\omega_i)}{P(X_u)} \tag{3}$$

where $P(X_u|\omega_i)$ is the probability that X_u belongs to the class region ω_i and $P(\omega_i)$ is called the priori and is calculated using Eq. 4.

$$P(\omega_i) = \frac{n_{c_i}}{N} \tag{4}$$

where n_{c_i} is the number of samples in a class c_i and N is the total number of samples in the sample space. $P(X_u)$ is called the evidence and is calculated using Eq. 5.

$$P(X_u) = \sum_{i=1}^c \left(P(X_u|\omega_i) \times P(\omega_i) \right) \tag{5}$$

4.3 Logistic regression

Logistic Regression (LR) is a reliable statistical method in which the probability of a class depends on a set of variables (Dong et al. 2016; Tsangaratos and Ilia 2016). The logistic model is calculated using Eq. 6.

$$z = \beta_0 + \sum_{i=1}^V \left(\beta_i \times x_i \right) \tag{6}$$

where z represents a measure of dependency on variables X (or the predicted output value), x_i is a variable (or an input) at index i , β_0 is the intercept (or bias), β_i is the slope of the logistic regression model (or coefficient value) at index i , and V is the number of variables (or inputs). The probability of the variables $P(z)$ is calculated using Eq. 7.

$$P(z) = \frac{e^z}{1 + e^z} \quad (7)$$

4.4 Naïve Bayes

Naïve Bayes (NB) is a straightforward probabilistic model built upon the Bayes' theorem (Tsangaratos and Ilia 2016). This classifier, as all ML classifiers, predicts the probability of an unknown pattern belonging to a specific class. However, what differentiates this algorithm is the application of Bayes' theorem. Similar to the DA classifier, the NB classifier calculates the conditional (i.e., posterior) probability using Eq. 3.

NB assumes that every variable in the training data is independent of the other variables and has an equal contribution to the classification problem. This is a simple but insufficient assumption to face real-world problems. Due to independency of variables, $P(X_u|\omega_i)$ is calculated using Eq. 8 and $P(X_u)$ is calculated using Eq. 9.

$$P(X_u|\omega_i) = \prod_{j=1}^l P(x_j|\omega_i) \quad (8)$$

$$P(X_u) = \prod_{j=1}^l P(x_j) \quad (9)$$

Substituting by Eq. 8 and Eq. 9 in Eq. 3, the result is shown in Eq. 10.

$$P(\omega_i|X_u) = \frac{\left(\prod_{j=1}^l P(x_j|\omega_i) \right) \times P(\omega_i)}{\prod_{j=1}^l P(x_j)} \quad (10)$$

The denominator of Eq. 10 is the same for a given input pattern regardless of the class, hence it can be eliminated as shown in Eq. 11.

$$P(\omega_i|X_u) = \operatorname{argmax} \left\{ \left(\prod_{j=1}^l P(x_j|\omega_i) \right) \times P(\omega_i) \right\} \quad (11)$$

In the case of the Gaussian Naïve Bayes, the values in each category are normally distributed. The probability $P(x_j|\omega_i)$ is calculated using Eq. 12.

$$P(x_j|\omega_i) = \frac{1}{\sqrt{2 \times \pi \times \sigma}} \times e^{-\frac{(x_j - \mu)^2}{2 \times \sigma^2}} \quad (12)$$

where μ and σ are the mean and the standard deviation of x_j respectively. Kernel Naïve Bayes, on the other hand, uses kernel density estimation in case of classes with continuous distribution (Al-Khurayji and Sameh 2017; Murakami and Mizuguchi 2010). The probability $P(x_j|\omega_i)$ is calculated using Eq. 13.

$$P(x_j|\omega_i) = \frac{1}{\sqrt{n_{c_i} \times h}} \times \sum_{v=1}^{n_{c_i}} \operatorname{Kernel}(x_j, x_{vji}) \quad (13)$$

where h is a smoothing parameter optimized on the training dataset, x_{vji} is the value of the feature in the j th position of the v th input in class c_i , and $\text{Kernel}(x_j, x_{vji})$ is a Gaussian function having zero mean and variance of 1 and is calculated using Eq. 14.

$$\text{Kernel}(a, b) = \frac{1}{\sqrt{2 \times \pi}} \times e^{-\frac{(a-b)^2}{2 \times h^2}} \tag{14}$$

4.5 Support vector machine

Support vector machine (SVM) classifier was proposed by Vapnik (2013). It is a widely used ML algorithm for statistical learning problems. The idea behind SVM is to separate data into two classes so that, SVM can build a model from this data during the training process. Then, the SVM classifier becomes ready for classifying new data. SVM selects the hyperplane that maximizes the distance between the two classes measured by the closest points (Rojas-Domínguez et al. 2017; Huang et al. 2013).

For a given classification problem with training dataset: $\{(X_1, y_1), (X_2, y_2), \dots, (X_N, y_N)\}$ where y_i represents the class number and equals -1 or 1 . The hyperplane separating points belonging to class $y_i = 1$ from the points belonging to class $y_i = -1$ is chosen so that the space from the hyperplane and the nearest point is maximized. The hyperplane can be calculated using Eq. 15.

$$w^T \times X_i + b = 0 \tag{15}$$

where w is the weight vector and b is the bias.

For linearly separable data, two parallel hyperplanes can be used to separate the two categories of data. These hyperplanes are chosen so that the distance separating them is as large as possible. The gap between these two hyperplanes is called the margin.

To prevent data from falling in the margin and ensure that data is on the right side of the hyperplane, Eq. 16 is used. Eq. 17 is a simplified version of it.

$$\begin{cases} w^T \times X_i + b \geq 1, & \text{if } y_i = 1 \\ w^T \times X_i + b \leq -1, & \text{if } y_i = -1 \end{cases} \tag{16}$$

$$y_i \times (w^T \times X_i + b) \geq 1 \tag{17}$$

The SVM classifier problem works on minimizing Eq. 18.

$$\min_{a,b} \left\{ \frac{1}{2} \times ||w||^2 + C \times \sum_{i=1}^n L_{\text{hinge}}(1 - y_i \times (w^T \times X_i + b)) \right\} \tag{18}$$

where C is a parameter and L_{hinge} is the hinge loss and is calculated using Eq. 19.

$$L_{\text{hinge}}(u) = \max \{0, u\} \tag{19}$$

Different SVM strategies can be used. For linearly separable data, linear SVM can be applied. Quadratic SVM uses quadratic function for nonlinearly separable data (Dagher 2008). Cubic SVM is the case in which the kernel function is cubic (Jain et al. 2018). Fine Gaussian SVM makes fine distinctions between categories, while Medium Gaussian SVM

makes medium distinctions between categories. Coarse Gaussian SVM makes coarse distinctions between categories. The difference between the last three types is in the scale of the kernel.

4.6 k-Nearest neighbor

k-Nearest Neighbor (kNN) classifier is a simple and accurate classifier that does not build a model from the training data. Instead, training and test data enter the classifier, and the classifier computes the distance between every test sample and the whole training dataset to assign the test sample to a class based on nearest neighbors (Zhang et al. 2017).

kNN is a lazy, non-parametric classifier. Its accuracy depends on the chosen distance measure technique. Euclidean distance is the standard measurement of the closeness between two points (Deng et al. 2016).

There are different types of kNN classifiers, including Fine kNN, Medium kNN, Coarse kNN, Cosine kNN, Cubic kNN, and Weighted kNN. The difference between these types is in the variations between the different classes and the number of neighbors.

4.7 Ensemble classifier

The main idea behind the Ensemble classifier is to combine different models into one powerful Ensemble model (Scholz and Klinkenberg 2005). The motivation towards the ensemble classifier is to be able to learn more effectively than a single classifier can (Shen and Chou 2006). For instance, Ensemble Boosted Tree is a combination of AdaBoost and Decision Tree classifier. Ensemble Bagged Tree is a combination of Bagging algorithm and Decision Tree classifier.

Ensemble Subspace Discriminant is built upon the usage of random subspace ensemble method with discriminant classifier, while Ensemble Subspace kNN is built upon the usage of random subspace ensemble method with Nearest Neighbor classifier. Ensemble RUS-Boosted Trees integrates different weak tree classifiers using RUSBoost (Random Under Sampling) algorithm, adding more accuracy to tree classifiers (Zhou 2012).

5 The proposed framework for diagnosis and prognosis of COVID-19

The destructive effect caused by COVID-19 on the life of millions of people all over the world forced scientific research to try to diagnose the virus accurately and as quickly as possible. The problem increases as the number of patients increase, especially if they need to be transferred to the intensive care unit (ICU). Countries are trying to provide oxygen generators for patients with problems in breathing in order to save their lives. However, with the huge number of infected patients, this becomes a challenging task.

Our proposed framework aims to accurately diagnose COVID-19 patients using CT chest images and prognose the severity of infection in order to determine whether the patient will need ICU or not using laboratory test results. The proposed framework consists of two phases, namely (1) Early Diagnostic Phase (EDP), and (2) Early Prognostic Phase (EPP). This section gives a detailed description of the two phases of the proposed framework. The suggested framework is graphically summarized in Fig. 4.

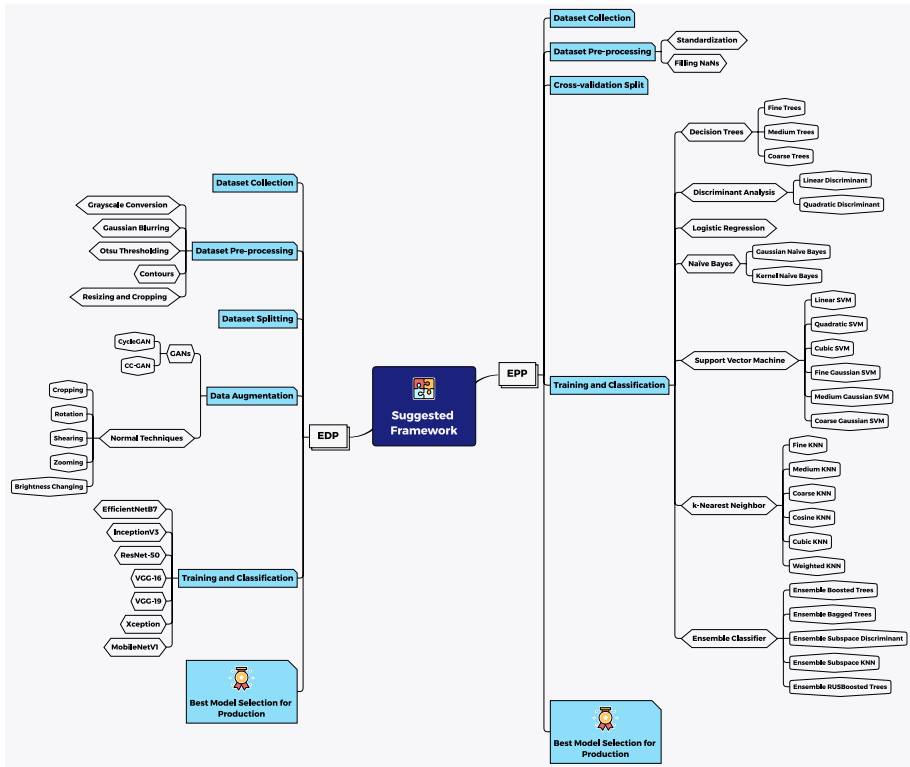


Fig. 4 Graphical summarization of the suggested framework

5.1 Early Diagnostic Phase (EDP)

The diagnosis of positive COVID-19 patients is the first step towards the treatment. The patient performs chest CT images to check whether he/she is infected or not as CT shows pulmonary ground-glass opacities, either unilateral or bilateral.

EDP is the first phase in the proposed framework. This phase verifies whether the patient is infected by COVID-19 or not. It consists of three steps, namely (1) image pre-processing, (2) data augmentation, and (3) transfer learning. These steps are explained in detail as follows.

CT image dataset In their comprehensive review, Roberts (2021) recommended using not only data from the Internet but also adding new high-quality data to avoid overfitting and to solve the bias problem. They also recommended maintaining powerful validation using external datasets in order to build robust models. Therefore, in the current study, we used a dataset of CT images including positive COVID-19 images and non-COVID-19 images of Egyptian patients collected from Egyptian Radiology centers. The dataset contains a total of 15, 535 CT images with 5, 159 images of confirmed positive COVID-19 cases and 10, 376 images of normal (i.e., non-COVID-19) cases. All images are used in the “JPG” format. A sample of the CT images with COVID-19 is shown in Fig. 5.

Image pre-processing Images obtained from the Radiology centers can’t be used directly with CNNs because:

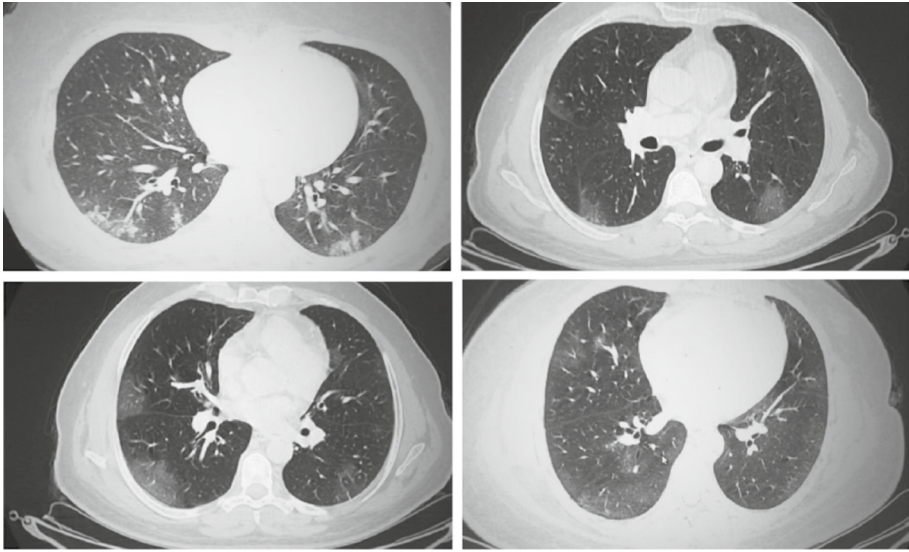


Fig. 5 CT sample images of Egyptian patients with COVID-19

- They are from different sources.
- They have different dimensions.
- They contain unnecessary details (i.e., noise).

So, the first step is to pre-process the dataset to convert it to a suitable format for CNN to detect the necessary features. The steps used for image pre-processing are shown in Fig. 6.

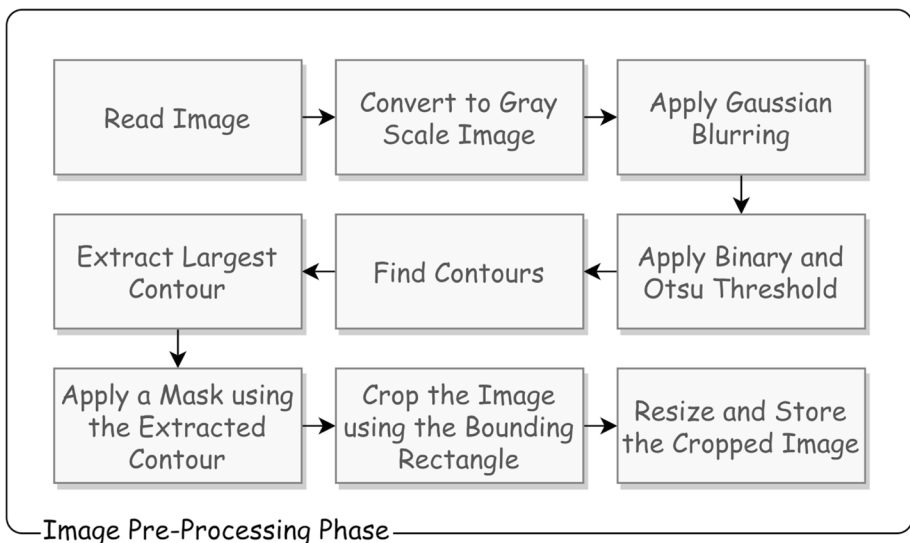


Fig. 6 Image pre-processing phase

For each image, the first step is to read the image file. The next step is to convert the image to a grayscale image (Bui et al. 2016). In the next step, Gaussian blurring is applied to eliminate the unnecessary noise (Gedraite and Hadad 2011).

After that, the Binary and Otsu threshold methods are used to separate pixels into the foreground and background pixels to generate a mask image (Yuan et al. 2016). A contour detection algorithm is then applied to separate the foreground object of interest. The largest contour is then used to create a mask to be subtracted from the original image to get the required image.

This image is then cropped and resized so that the dataset images have the same dimensions. Figure 7 shows a sample image before and after pre-processing.

Data augmentation The main problem in using CNNs in image classification is the availability of data for the training process. The less the data, the more the network becomes prone to the overfitting problem. To overcome this, data augmentation techniques are used. It is achieved by applying distortions to the training samples which results in new training data (Salamon and Bello 2017). This means that more training images can be extracted from the original dataset by augmentations (Shorten and Khoshgoftaar 2019).

The normal image augmentation techniques include brightness change, cropping, rotation, shearing, zooming, and flipping (Başaran et al. 2020). Adjusting brightness means manipulating the light of the image to make the augmented image darker or lighter. Cropping is done by taking a region from the image with specified dimensions.

Rotation generates a new image by changing the angle in the clockwise or counterclockwise directions around its center. Shearing is done by shifting one part of an image in a direction and the other part in the opposite direction.

Zooming can be either zoom in or zoom out of the image. Flipping is transforming the original image horizontally or vertically in a mirror-reversal manner. Figure 8 shows the result of different augmentation methods on a sample CT image.

A more advanced augmentation approach used in this study is the generative adversarial networks (GANs) (Goodfellow et al. 2014). GAN is a framework for training generative

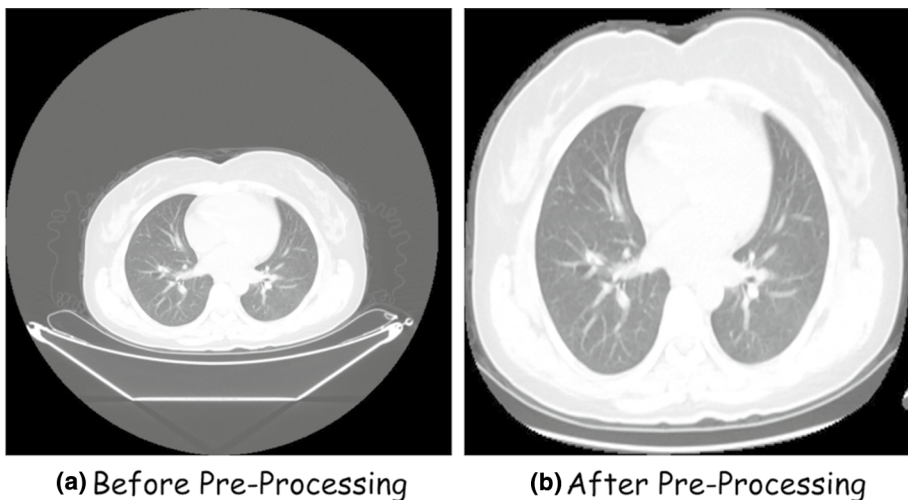


Fig. 7 A sample image before and after pre-processing

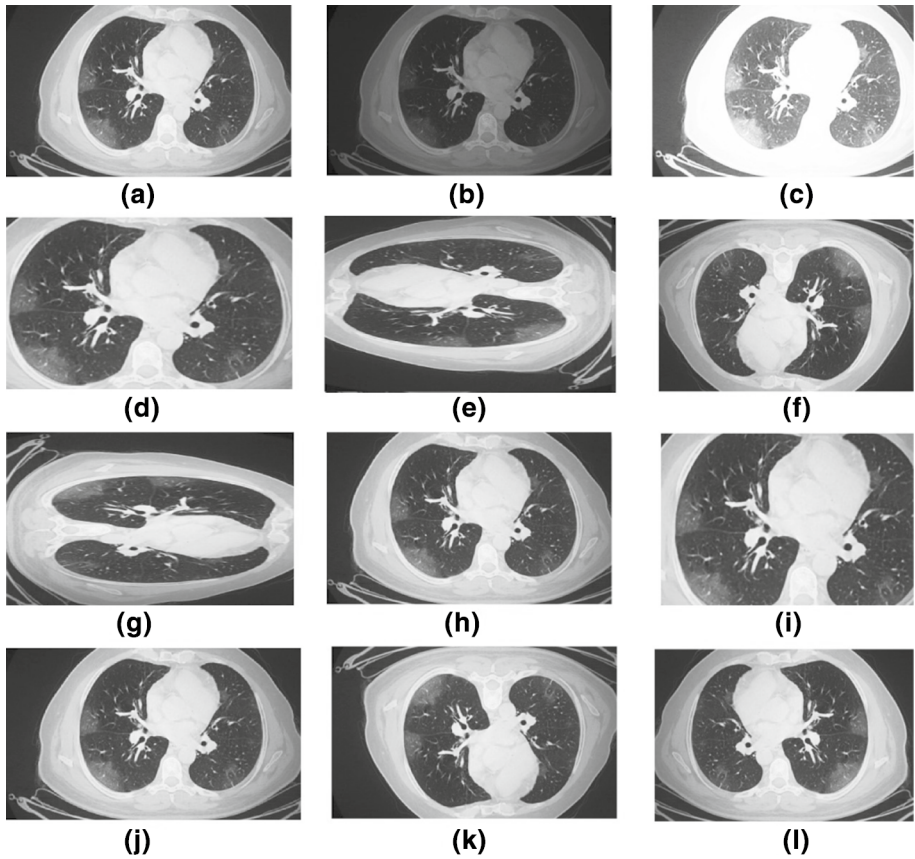


Fig. 8 Different augmentation methods: **a** original image, **b** reduce brightness, **c** increase brightness, **d** cropping, **e** rotation 90°, **f** rotation 180°, **g** rotation 270°, **h** shearing, **i** zoom in, **j** zoom out, **k** flip vertical, and **l** flip horizontal

models (Denton et al. 2015). It achieved very good results in many image generation tasks (Zhang et al. 2019). GANs have two parts (1) a generator and (2) a discriminator.

The generator is responsible for generating realistic images while the discriminator decides if the produced images can be distinguished from the real ones (Park et al. 2019). The main reason for the GANs' success is the use of adversarial loss that imposes the produced images to be indistinguishable from real photos (Salimans et al. 2016).

One type of GAN, known as CycleGAN (Zhu et al. 2017), converts an image from one domain to another while there are no paired examples (Chu et al. 2017). Another type of GAN, known as Conditional GAN (Mirza and Osindero 2014), extends GANs by adding an additional layer to both the generator and the discriminator. This layer contains some additional information such as class labels (Dai et al. 2017; Isola et al. 2017).

A modification to CGAN in Denton et al. (2016) known as context-conditional generative adversarial networks (CC-GANs), was proposed. In this method, the generator in conditional GANs is trained to complete an absent image patch. The conditions of the generator and discriminator depend on the surrounding pixels.

To explain this, the generator receives an input image with a randomly hidden patch and outputs a filled image. The discriminator receives the complete image so that it doesn't learn to distinguish cutouts over the edge of that lost patch. In this paper, we applied both CycleGAN and CC-GAN to increase the dataset.

The CycleGAN is trained on 12 epochs with a batch size of 1 while the CC-GAN is trained on 50,000 epochs with a batch size of 32. The dataset is resized to (96, 96) in the colored mode. Tables 2 and 3 show the generator and the discriminator of the CycleGAN respectively.

Tables 4 and 5 show the generator and the discriminator of the CC-GAN respectively.

The downsampling block consists of a convolutional layer, leaky ReLU activation layer, and a normalization layer. The upsampling block consists of an upsampling layer, a convolutional layer, a normalization layer, and a concatenation layer.

Figures 9 and 10 show sample results after training the CycleGAN and CC-GAN on the presented dataset respectively.

Table 2 Architecture of the used CycleGAN discriminator

| # | Architecture layer | Output shape |
|---|---------------------|---------------|
| 1 | Input Layer | (96, 96, 3) |
| 2 | Downsampling Block | (48, 48, 64) |
| 3 | Downsampling Block | (24, 24, 128) |
| 4 | Downsampling Block | (12, 12, 256) |
| 5 | Downsampling Block | (6, 6, 512) |
| 6 | Convolutional Layer | (6, 6, 1) |

Table 3 Architecture of the used CycleGAN generator

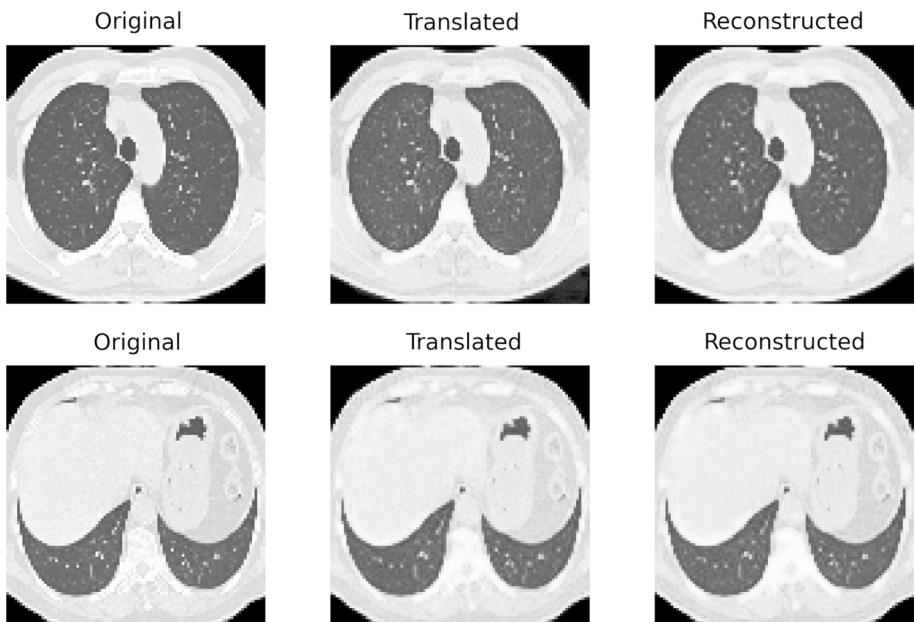
| # | Architecture layer | Output shape |
|----|---------------------|---------------|
| 1 | Input Layer | (96, 96, 3) |
| 2 | Downsampling Block | (48, 48, 32) |
| 3 | Downsampling Block | (24, 24, 64) |
| 4 | Downsampling Block | (12, 12, 128) |
| 5 | Downsampling Block | (6, 6, 256) |
| 6 | Upsampling Block | (12, 12, 256) |
| 7 | Upsampling Block | (24, 24, 128) |
| 8 | Upsampling Block | (48, 48, 64) |
| 9 | Upsampling Layer | (96, 96, 64) |
| 10 | Convolutional Layer | (96, 96, 3) |

Table 4 Architecture of the Used CC-GAN discriminator

| # | Architecture layer | Output shape |
|---|---------------------|---------------|
| 1 | Input Layer | (96, 96, 3) |
| 2 | Sequential Layer | (12, 12, 256) |
| 3 | Flatten Layer | (36864) |
| 4 | Convolutional Layer | (12, 12, 1) |
| 5 | Dense Layer | (3) |

Table 5 Architecture of the Used CC-GAN generator

| # | Architecture layer | Output shape |
|----|---------------------|---------------|
| 1 | Input Layer | (96, 96, 3) |
| 2 | Downsampling Block | (48, 48, 32) |
| 3 | Downsampling Block | (24, 24, 64) |
| 4 | Downsampling Block | (12, 12, 128) |
| 5 | Downsampling Block | (6, 6, 256) |
| 6 | Upsampling Block | (12, 12, 256) |
| 7 | Upsampling Block | (24, 24, 128) |
| 8 | Upsampling Block | (48, 48, 64) |
| 9 | Upsampling Layer | (96, 96, 64) |
| 10 | Convolutional Layer | (96, 96, 3) |

**Fig. 9** Two samples after training the CycleGAN on the presented dataset

Transfer learning (TL) Recently, transfer learning has been commonly used in deep learning problems. Utilizing the transfer learning (Wang et al. 2021), a pre-trained CNN model can be reused in a relevant application (Deepak and Ameer 2019). The goal of transfer learning is to learn from related tasks.

Studies showed that the already learned knowledge plays a great role specifically in the case of rare training data (Han et al. 2018). The CNN structures used in this study are EfficientNetB7, InceptionV3, ResNet-50, VGG-16, VGG-19, Xception, and MobileNetV1.

EfficientNetB7 (Tan and Le 2019) was introduced to overcome the MBConv mobile bottleneck. InceptionV3 (Szegedy et al. 2016) has 48 layers. It uses inception modules to lessen the number of parameters and raise the training speed. ResNet-50 (He et al. 2016)

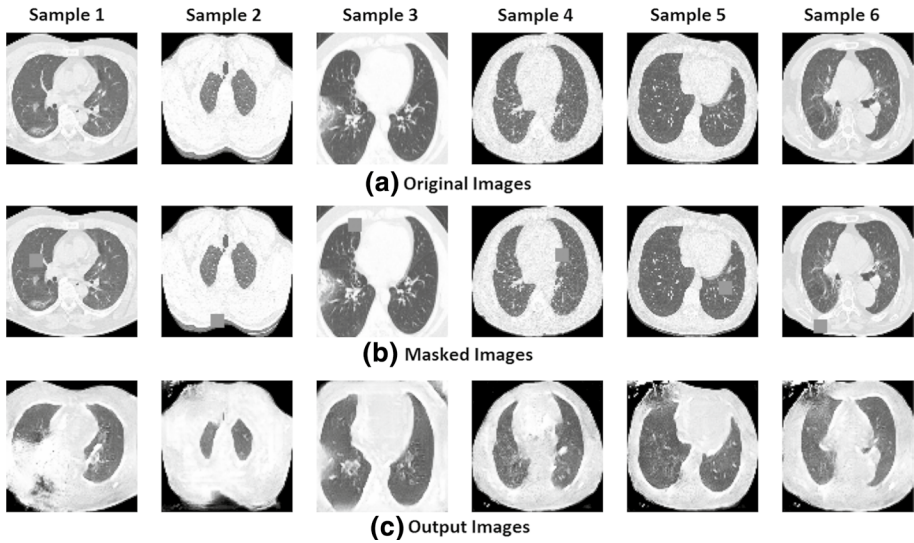


Fig. 10 Six samples after training the CC-GAN on the presented dataset

has 49 convolutional layers followed by one FC layer with 16 residual blocks. VGG-16 (Simonyan and Zisserman 2014) has 5 convolutional blocks containing 13 convolutional layers, and 3 fully-connected layers.

VGG-19 (Simonyan and Zisserman 2014) is a deeper CNN than VGG-16 because it has 5 convolutional blocks containing 16 convolutional layers and 3 fully-connected layers. Xception (Chollet 2017) modifies the inception net by replacing the inception modules with “depthwise separable convolutions”. It consists of 2 convolutional layers, depthwise separable convolution layers, 4 convolutional layers, and a fully-connected layer. MobileNetV1 (Howard et al. 2017) was intended for use on a mobile platform. It uses depthwise convolutions so that memory usage is reduced (Balaha et al. 2021b).

Performance metrics Different performance metrics were used in our study to evaluate the performance of the different pre-trained CNN architectures. The used metrics are Accuracy (Eq. 20), Precision (Eq. 21), Recall (Eq. 22), and F1-Score (Eq. 23) (Balaha et al. 2021a).

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{20}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{21}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{22}$$

$$F1_{\text{score}} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{23}$$

Table 6 A brief description of the different prognostic markers

| Marker | Description | Unit | Indication |
|------------------------|---|-----------------------|-----------------------------|
| Gender | Male or Female | – | 0: Female and 1: Male |
| Age | Age of person | Years | – |
| Admission Time | Determine the time between the disease infection and admission to the hospital | Days | – |
| INR | International normalized ratio test is a measure of the time for the blood to clot | – | – |
| PT | Prothrombin time test evaluates blood clotting | Seconds | – |
| Fibrinogen | Fibrinogen is a protein that helps stop bleeding by forming blood clots. This test determines the amount of fibrinogen in the blood | mg/L | – |
| Lung Tissue Affected | Determine the percentage of the lung tissue affected by pneumonia from Chest CT images | % | – |
| Platelet Count | Platelets are useful in stop bleeding. This test determines the average amount of platelet in the blood | $\times 10^9$ cells/L | – |
| ICU Transfer Time | Determine the time between admission to the hospital and transfer to the ICU | Days | – |
| CRP | C-reactive protein test is used to detect the inflammation | mg/L | – |
| Artificial Ventilation | Determine whether the patient needed an artificial lung ventilation after transmission to the ICU or not | – | 0: No and 1: Yes |
| RT-PCR Testing Result | Determine whether the person has COVID-19 infection or not | – | 0: Negative and 1: Positive |

Table 7 A sample of the different prognostic markers dataset

| Gender | Age | AT* | INR | PT | Fibrinogen | LTA* | LTA + Week* | Platalet* | Platalet + Week* | ICU TT* | CRP | CRP + Week* | AV* | RT-PCR |
|--------|-----|-----|------|------|------------|------|-------------|-----------|------------------|---------|-----|-------------|-----|--------|
| 1 | 57 | 5 | 1.13 | 13.8 | 1.24 | 40 | 80 | 141 | 376 | 2 | 132 | 31 | 0 | 0 |
| 0 | 45 | 10 | 1.22 | 15.9 | NA | 71 | 88 | 23 | 111 | 1 | 155 | 136 | 1 | 1 |
| 1 | 42 | 11 | 1.26 | 13.7 | 5.2 | 51 | 92 | 161 | NA | 3 | 165 | 192 | 0 | 0 |
| 1 | 75 | 7 | 1.32 | 14.3 | 5.2 | 30 | 71 | 356 | 142 | 3 | 182 | 207 | 1 | 1 |
| 0 | 57 | 11 | 1.11 | NA | 5.3 | 55 | 92 | 267 | 31 | 3 | 168 | 285 | 1 | 1 |

* NA: Not Applicable, AT: Admission Time, LTA: Lung Tissue Affected, LTA + Week: Lung Tissue Affected after a Week, Platelet: Platelet Count, Platelet + Week: Platelet Count after a Week, ICU TT: ICU Transfer Time, CRP + Week: CRP after a Week, and AV: Artificial Ventilation

By testing the network, one of four results can appear (1) TP means a true diagnosis of COVID-19 case, (2) TN means a truly non-COVID-19 case, (3) FP means that the network wrongly diagnosed COVID-19 for a healthy image, and (4) FN means the network couldn't diagnose COVID-19 in an infected image. The area under curve (AUC) is also calculated to indicate the performance.

Training environment All scripts for the EDP are written in the Python programming language. The authors used two environments for that phase (1) a Toshiba Qosmio X70-A device with Windows 10 operating system, Intel Core i7 processor, 32 GB RAM, and Nvidia GTX with 4 GB GPU graphics card, and (2) Google Colab is used as the training environment with the help of its Graphical Processing Unit (GPU). Keras (a deep learning package), NumPy, Matplotlib, OpenCV, and Pandas are the used Python packages (Balaha and Saafan 2021).

5.2 Early Prognostic Phase (EPP)

EPP begins when the patient is diagnosed as positive COVID-19. This phase is used to prognose the severity of the infection in order to predict whether the patient will need to transfer to the ICU or not as early as possible.

Different ML algorithms are applied for the classification of patients into two groups (1) group 1 needs the ICU and (2) group 2 does not need the ICU. These algorithms use laboratory test results extracted from patients' records as prognostic markers of how severe pneumonia will be so that we can rescue the patient.

Table 6 presents a brief description of the different prognostic markers (Baranovskii et al. 2020). Table 7 presents a sample of 5 random records from the dataset.

To achieve the best accuracy, we applied 25 different classification techniques, namely:

- Trees (Fine Tree, Medium Tree, and Coarse Tree).

Table 8 CNN and TL experiments' configurations summarization

| Configuration | Value |
|------------------------|---|
| Dataset Type | CT Images |
| Dataset Size | 15, 535 (5, 159 COVID and 10, 376 Non-COVID) |
| Dataset Split Ratio | 85% : 15% |
| Dataset Pre-processing | Figure 6 |
| Data Augmentation | Normal and GANs |
| Pre-trained CNN Models | EfficientNetB7, InceptionV3, ResNet-50, VGG-16, VGG-19, Xception, and MobileNetV1 |
| Pre-trained Weights | ImageNet |
| Optimizer | Adam ($\beta = 0.5$) |
| Learning Rate | 0.0002 |
| Number of Epochs | 128 |
| Batch Size | 32 |
| Performance Metrics | Accuracy, F1-score, Precision, Recall, and AUC |
| Programming Language | Python |
| Training Environment | Toshiba Qosmio X70-A and Google Colab |

Table 9 EfficientNetB7 architecture performance metrics results

| Approach | Testing subset | | | | | All dataset | | | | | | |
|-----------------------|----------------|----------|-------|--------|-----------|-------------|-------|----------|-------|--------|-----------|-------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| No augmentation | 0.022 | 99.61% | 0.998 | 99.62% | 99.62% | 0.996 | 0.007 | 99.88% | 0.999 | 99.88% | 99.88% | 0.999 |
| Normal augmentation | 0.657 | 67.78% | 0.657 | 66.89% | 66.89% | 0.669 | 0.666 | 66.96% | 0.650 | 66.97% | 66.97% | 0.670 |
| CC-GAN augmentation | 0.077 | 99.02% | 0.994 | 99.03% | 99.03% | 0.990 | 0.018 | 99.76% | 0.999 | 99.76% | 99.76% | 0.998 |
| CycleGAN augmentation | 0.042 | 99.10% | 0.999 | 99.11% | 99.11% | 0.991 | 0.015 | 99.77% | 0.999 | 99.77% | 99.77% | 0.998 |

- Discriminant (Linear Discriminant and Quadratic Discriminant).
- Regression (Logistic Regression).
- Naïve Bayes (Gaussian Naïve Bayes and Kernel Naïve Bayes).
- SVM (Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM).
- KNN (Fine KNN, Medium KNN, Coarse KNN, Cosine KNN, Cubic KNN, and Weighted KNN).
- Ensemble (Ensemble Boosted Trees, Ensemble Bagged Trees, Ensemble Subspace Discriminant, Ensemble Subspace KNN, and Ensemble RUSBoosted Trees).

Then, the algorithm with the best accuracy is chosen and the results from this model are the main lead to prognoses the infection.

Dataset pre-processing The numerical dataset is pre-processed before applying the ML classifier. The empty (i.e. null) values are filled with zeros. The dataset is applied after that to the standardization method. Standardization is a scaling method where the values are centered around the mean (i.e., the mean becomes zero) with a unit standard deviation (Gal and Rubinfeld 2019). Equation 24 shows the used standardization method.

$$X' = \frac{X - \mu}{\sigma} \quad (24)$$

where μ is the mean of the features and σ is the standard deviation of the features.

Training environment All scripts for the EPP are written in MATLAB programming language. The authors used for that phase a Toshiba Qosmio X70-A device with Windows 10 operating system, Intel Core i7 processor, 32 GB RAM, and Nvidia GTX with 4 GB GPU graphics card.

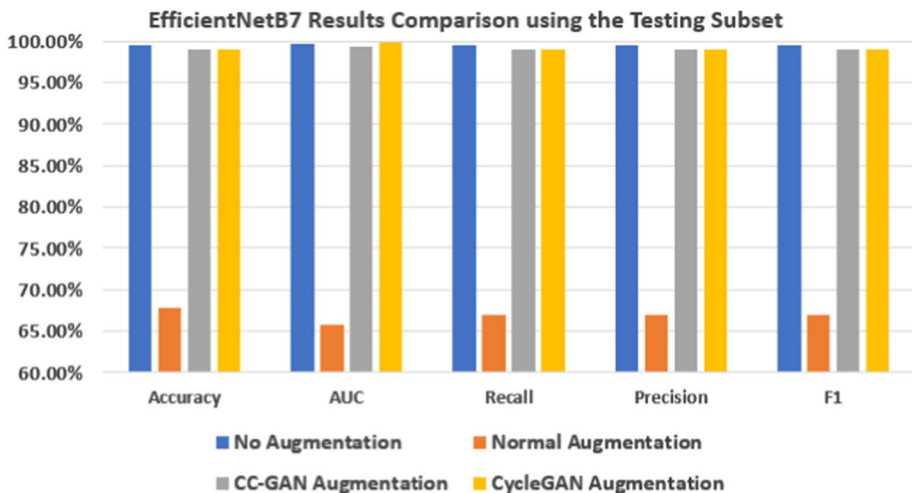


Fig. 11 EfficientNetB7 results comparison using the testing subset

Table 10 InceptionV3 architecture performance metrics results

| Approach | Testing subset | | | | | | All dataset | | | | | |
|-----------------------|-----------------|----------|--------|--------|-----------|--------|-------------|----------|--------|--------|-----------|--------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| | No Augmentation | 0.027 | 99.27% | 0.999 | 99.28% | 99.28% | 0.993 | 0.008 | 99.81% | 1.0 | 99.81% | 99.81% |
| Normal Augmentation | 0.660 | 68.38% | 0.653 | 67.48% | 67.48% | 0.675 | 0.652 | 67.57% | 0.653 | 67.58% | 67.58% | 0.676 |
| CC-GAN Augmentation | 0.038 | 98.84% | 0.999 | 98.86% | 98.86% | 0.989 | 0.010 | 99.71% | 1.0 | 99.71% | 99.71% | 0.997 |
| CycleGAN Augmentation | 0.038 | 98.93% | 0.998 | 98.94% | 98.94% | 0.989 | 0.010 | 99.75% | 1.0 | 99.75% | 99.75% | 0.997 |

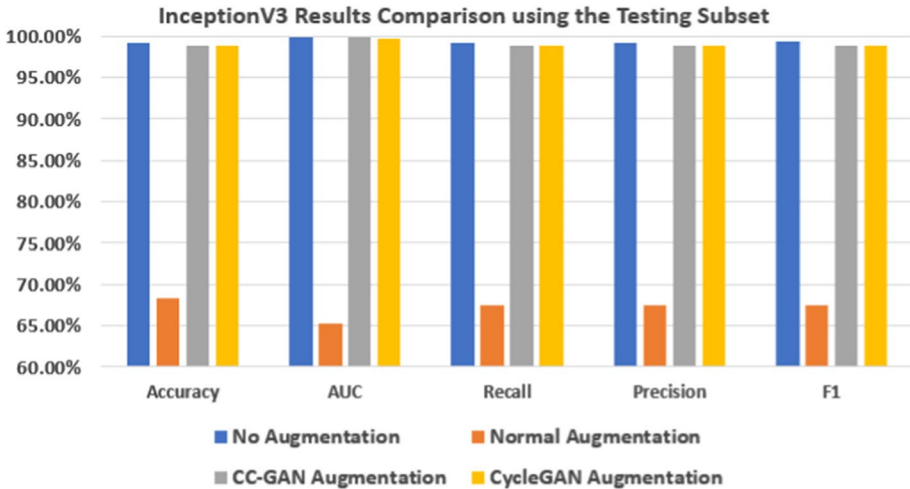


Fig. 12 InceptionV3 results comparison using the testing subset

6 Experimental results and discussion

In this section, the experiments' results of our diagnostic and prognostic framework are presented and discussed.

6.1 COVID-19 diagnosis using CNN and TL

Each transfer learning architecture used the "ImageNet" pre-trained weights. All layers are set to be trainable. A flatten layer, a 25% dropout layer, and an output dense layer followed the base transfer learning architecture. The experiment number of epochs was 128 and the batch size was 32. The user optimizer was Adam with a learning rate of 0.0002 and a beta value (β) of 0.5. The dataset is split into 85% for training and validation and 15% for testing. The training and validation portion is split internally into 85% for training and 15% for validation. Table 8 summarizes the CNN and TL experiments' configurations.

Every following subsection shows four experiments each one in a separate row. The first experiment is performed without augmentation. The second is performed with normal augmentation techniques. The third and fourth experiments are performed using CC-GAN and CycleGAN augmentation respectively.

EfficientNetB7 Table 9 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.022 and is reported by the no augmentation experiment. The highest testing accuracy value is 99.61% and is reported by the no augmentation experiment. The highest testing AUC value is 0.999 is reported by the CycleGAN augmentation experiment.

The highest testing recall value is 99.62% and is reported by the no augmentation experiment. The highest testing precision value is 99.62% and is reported by the no augmentation experiment. The highest testing F1-score value is 0.996 is reported by the CycleGAN augmentation experiment. All experiments, except the normal augmentation one, reported testing accuracies above 99%.

Table 11 ResNet50 architecture performance metrics results

| Approach | Testing subset | | | | | | All dataset | | | | | |
|-----------------------|-----------------|----------|--------|--------|-----------|--------|-------------|----------|--------|--------|-----------|--------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| | No Augmentation | 0.036 | 99.10% | 0.998 | 99.11% | 99.11% | 0.991 | 0.010 | 99.75% | 0.999 | 99.75% | 99.75% |
| Normal Augmentation | 0.655 | 66.20% | 0.663 | 65.33% | 65.33% | 0.653 | 0.653 | 66.84% | 0.667 | 66.85% | 66.85% | 0.669 |
| CC-GAN Augmentation | 0.057 | 98.97% | 0.996 | 98.99% | 98.99% | 0.990 | 0.017 | 99.71% | 0.999 | 99.71% | 99.71% | 0.997 |
| CycleGAN Augmentation | 0.023 | 99.32% | 0.998 | 99.32% | 99.32% | 0.993 | 0.010 | 99.69% | 0.999 | 99.69% | 99.69% | 0.997 |

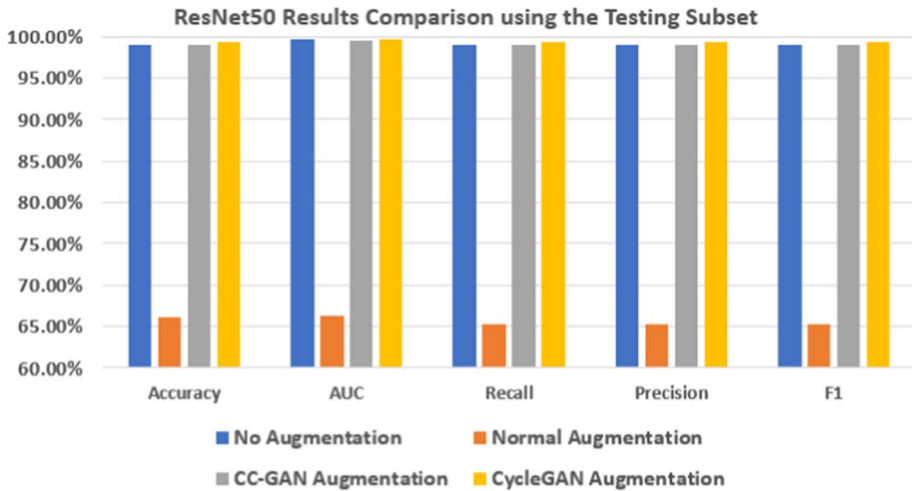


Fig. 13 ResNet50 results comparison using the testing subset

Figure 11 shows the graphical representation of the EfficientNetB7 model results comparison using the testing subset.

InceptionV3 Table 10 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.027 is reported by the no augmentation experiment. The highest testing accuracy value is 99.27% is reported by the no augmentation experiment. The highest testing AUC value is 0.999 is reported by the no augmentation CycleGAN augmentation experiments.

The highest testing recall value is 99.28% is reported by the no augmentation experiment. The highest testing precision value is 99.28% is reported by the no augmentation experiment. The highest testing F1-score value is 0.993 is reported by the no augmentation experiment. All experiments, except the normal augmentation one, reported testing accuracies above 98.8%.

Figure 12 shows the graphical representation of the InceptionV3 model results comparison using the testing subset.

ResNet50 Table 11 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.023 is reported by the CycleGAN augmentation experiment. The highest testing accuracy value is 99.32% is reported by the CycleGAN augmentation experiment. The highest testing AUC value is 0.998 is reported by the CycleGAN augmentation experiment.

The highest testing recall value is 99.32% is reported by the CycleGAN augmentation experiment. The highest testing precision value is 99.32% is reported by the CycleGAN augmentation experiment. The highest testing F1-score value is 0.993 is reported by the CycleGAN augmentation experiment. All experiments, except the normal augmentation one, reported testing accuracies above 98.9%.

Figure 13 shows the graphical representation of the ResNet50 model results comparison using the testing subset.

VGG-16 Table 12 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.048 is reported by the no augmentation experiment. The highest testing accuracy value is 99.14% is reported by the CC-GAN

Table 12 VGG-16 architecture performance metrics results

| Approach | Testing subset | | | | | All dataset | | | | | | |
|-----------------------|----------------|----------|-------|--------|-----------|-------------|-------|----------|-------|--------|-----------|-------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| No Augmentation | 0.048 | 99.10% | 0.997 | 99.11% | 99.11% | 0.991 | 0.013 | 99.78% | 0.999 | 99.78% | 99.78% | 0.998 |
| Normal Augmentation | 0.630 | 67.57% | 0.679 | 66.68% | 66.68% | 0.667 | 0.637 | 66.71% | 0.671 | 66.72% | 66.72% | 0.667 |
| CC-GAN Augmentation | 0.089 | 99.14% | 0.994 | 99.16% | 99.16% | 0.992 | 0.027 | 99.76% | 0.999 | 99.76% | 99.76% | 0.998 |
| CycleGAN Augmentation | 0.067 | 98.76% | 0.995 | 98.78% | 98.78% | 0.988 | 0.016 | 99.70% | 0.999 | 99.70% | 99.70% | 0.997 |

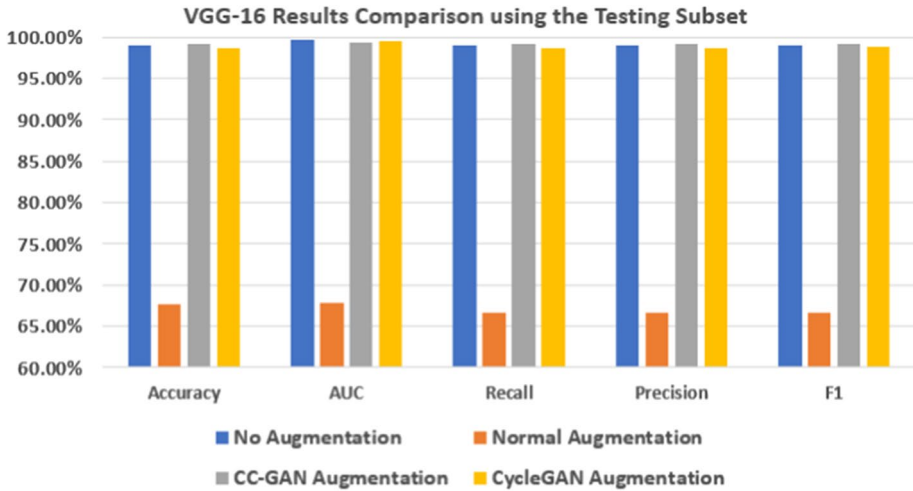


Fig. 14 VGG-16 results comparison using the testing subset

augmentation experiment. The highest testing AUC value is 0.997 is reported by the no augmentation experiment.

The highest testing recall value is 99.16% is reported by the CC-GAN augmentation experiment. The highest testing precision value is 99.16% is reported by the CC-GAN augmentation experiment. The highest testing F1-score value is 0.992 is reported by the CC-GAN augmentation experiment. All experiments, except the normal augmentation one, reported testing accuracies above 98.7%.

Figure 14 shows the graphical representation of the VGG-16 model results comparison using the testing subset.

VGG-19 Table 13 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.053 is reported by the CycleGAN augmentation experiment. The highest testing accuracy value is 99.32% is reported by the CycleGAN augmentation experiment. The highest testing AUC value is 0.996 is reported by the CycleGAN augmentation experiment.

The highest testing recall value is 99.32% is reported by the CycleGAN augmentation experiment. The highest testing precision value is 99.32% is reported by the CycleGAN augmentation experiment. The highest testing F1-score value is 0.993 is reported by the CycleGAN augmentation experiment. All experiments, except the normal augmentation one, reported testing accuracies above 98.5%.

Figure 15 shows the graphical representation of the VGG-19 model results comparison using the testing subset.

Xception Table 14 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.027 is reported by the no augmentation experiment. The highest testing accuracy value is 99.32% is reported by the no augmentation experiment. The highest testing AUC value is 0.999 is reported by the no augmentation experiment.

The highest testing recall value is 99.32% is reported by the no augmentation experiment. The highest testing precision value is 99.32% is reported by the no augmentation

Table 13 VGG-19 architecture performance metrics results

| Approach | Testing Subset | | | | | | All dataset | | | | | |
|-----------------------|-----------------|----------|--------|--------|-----------|--------|-------------|----------|--------|--------|-----------|--------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| | No Augmentation | 0.071 | 98.97% | 0.995 | 98.99% | 98.99% | 0.990 | 0.018 | 99.69% | 0.999 | 99.69% | 99.69% |
| Normal Augmentation | 0.630 | 67.57% | 0.676 | 66.68% | 66.68% | 0.667 | 0.636 | 66.71% | 0.667 | 66.72% | 66.72% | 0.667 |
| CC-GAN Augmentation | 0.072 | 98.55% | 0.995 | 98.56% | 98.56% | 0.986 | 0.022 | 99.47% | 0.999 | 99.47% | 99.47% | 0.995 |
| CycleGAN Augmentation | 0.053 | 99.32% | 0.996 | 99.32% | 99.32% | 0.993 | 0.017 | 99.79% | 0.999 | 99.79% | 99.79% | 0.998 |

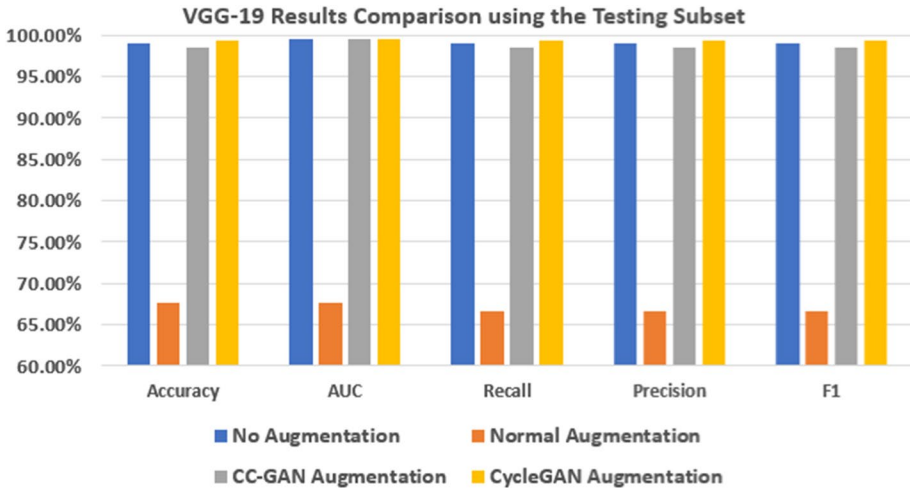


Fig. 15 VGG-19 results comparison using the testing subset

experiment. The highest testing F1-score value is 0.993 is reported by the no augmentation and CycleGAN augmentation experiments. All experiments, except the normal augmentation one, reported testing accuracies above 99.1%.

Figure 16 shows the graphical representation of the Xception model results comparison using the testing subset.

MobileNetV1 Table 15 shows the performance metrics results for the testing sub-dataset and the whole dataset. The lowest testing loss value is 0.018 is reported by the CycleGAN augmentation experiment. The highest testing accuracy value is 99.57% is reported by the CycleGAN augmentation experiment. The highest testing AUC value is 0.999 is reported by the CycleGAN augmentation experiment.

The highest testing recall value is 99.58% is reported by the CycleGAN augmentation experiment. The highest testing precision value is 99.58% is reported by the CycleGAN augmentation experiment. The highest testing F1-score value is 0.996 is reported by the no augmentation and CycleGAN augmentation experiments. All experiments, except the normal augmentation one, reported testing accuracies above 99.0%.

Figure 17 shows the graphical representation of the MobileNetV1 model results comparison using the testing subset.

From the results, the best testing accuracies are shown in Table 16. The two highest testing accuracies were 99.61% and 99.57% and were achieved by EfficientNetB7 and MobileNetV1 respectively. It is clear that using normal data augmentation methods did not report the expected results as normal methods produced images with less-feature-quality images. In other words, cropping and shearing may have abandoned the important features in the mages. So, some of the important features in the original images were lost. However, CycleGAN and CCGAN could produce images similar to the original ones in the collected dataset. Hence, the important features were preserved.

Table 14 Xception architecture performance metrics results

| Approach | Testing subset | | | | | | All dataset | | | | | |
|-----------------------|-----------------|----------|--------|--------|-----------|--------|-------------|----------|--------|--------|-----------|--------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| | No Augmentation | 0.027 | 99.32% | 0.999 | 99.32% | 99.32% | 0.993 | 0.008 | 99.81% | 1.0 | 99.81% | 99.81% |
| Normal Augmentation | 0.841 | 67.57% | 0.670 | 66.68% | 66.68% | 0.667 | 0.867 | 66.71% | 0.656 | 66.72% | 66.72% | 0.667 |
| CC-GAN Augmentation | 0.069 | 99.19% | 0.995 | 99.20% | 99.20% | 0.992 | 0.017 | 99.81% | 0.999 | 99.81% | 99.81% | 0.998 |
| CycleGAN Augmentation | 0.038 | 99.27% | 0.996 | 99.28% | 99.28% | 0.993 | 0.010 | 99.82% | 0.999 | 99.82% | 99.82% | 0.998 |

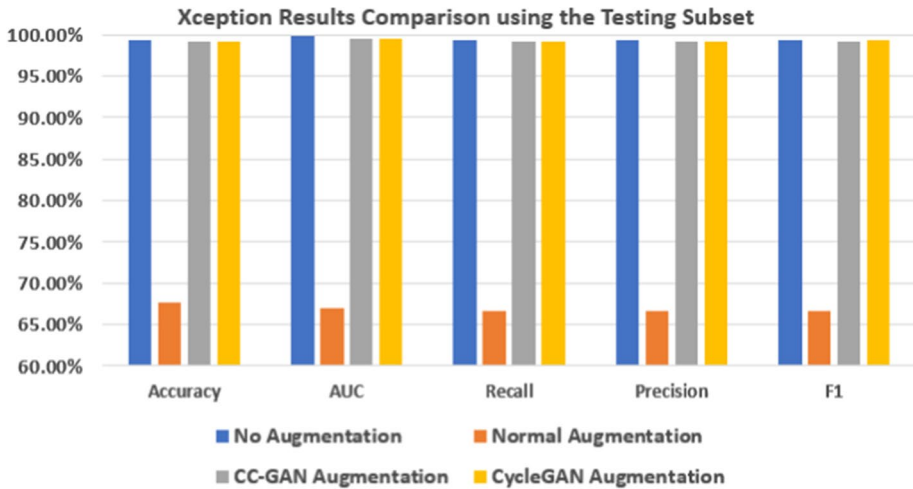


Fig. 16 Xception results comparison using the testing subset

6.2 Prognosis of COVID-19 severity using ML algorithms

Each machine learning algorithm was applied to 50 cross-validation folds to avoid the overfitting. Table 17 summarizes the ML experiments' configurations.

Table 18 shows the experiment results of the used ML algorithms with the appliance of the standardization method. The Ensemble Bagged Trees method reported the highest accuracy value (98.7%).

Table 19 shows the experiment results of the used ML algorithms without the appliance of the standardization method. The Ensemble Bagged Trees method reported the highest accuracy value (98.7%).

Figure 18 shows a graphical comparison between the results of the ML algorithms with and without standardization. It worth mentioning that (1) Quadratic Discriminant and Gaussian Naïve Bayes could not report results in both cases, (2) without standardization reported higher accuracies than standardization in six experiments, (3) standardization reported higher accuracies than without standardization in five experiments, and (4) twelve experiments reported the same results in both cases.

6.3 Comparison with state-of-the-art studies

Table 20 compares the best results of the current study using the CNN and TL approach with the discussed state-of-the-art related studies. The table reports the accuracies that each study could achieve concerning their approach and dataset.

From Table 20, the current study reports an accuracy value that is higher than 11 related studies. Figure 19 shows a graphical representation of the results in ascending order.

Table 15 MobileNetV1 architecture performance metrics results

| Approach | Testing subset | | | | | | All dataset | | | | | |
|-----------------------|-----------------|----------|--------|--------|-----------|--------|-------------|----------|--------|--------|-----------|--------|
| | Loss | Accuracy | AUC | Recall | Precision | F1 | Loss | Accuracy | AUC | Recall | Precision | F1 |
| | No Augmentation | 0.031 | 99.40% | 0.998 | 99.41% | 99.41% | 0.994 | 0.006 | 99.88% | 0.999 | 99.88% | 99.88% |
| Normal Augmentation | 0.620 | 70.05% | 0.740 | 69.13% | 69.13% | 0.691 | 0.618 | 70.19% | 0.739 | 70.20% | 70.20% | 0.702 |
| CC-GAN Augmentation | 0.050 | 99.02% | 0.996 | 99.03% | 99.03% | 0.990 | 0.015 | 99.70% | 0.999 | 99.70% | 99.70% | 0.997 |
| CycleGAN Augmentation | 0.018 | 99.57% | 0.999 | 99.58% | 99.58% | 0.996 | 0.008 | 99.83% | 0.999 | 99.83% | 99.83% | 0.998 |

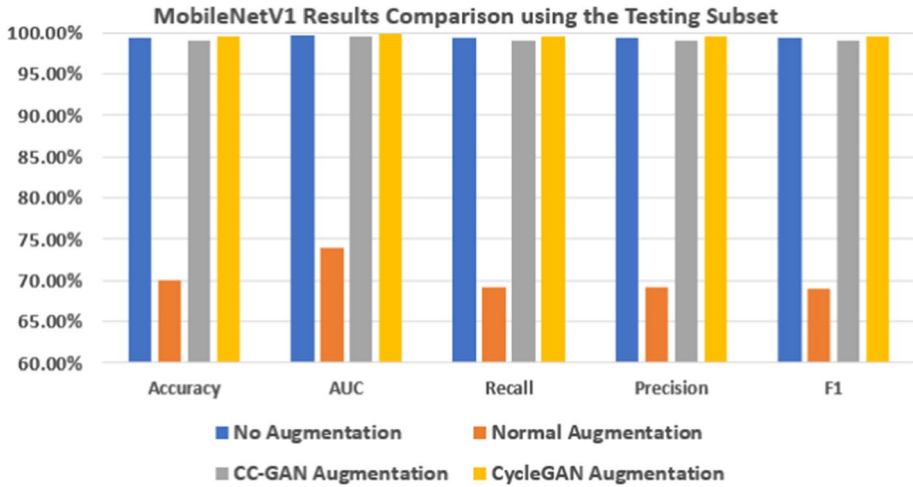


Fig. 17 MobileNetV1 results comparison using the testing subset

Table 16 Highest testing accuracies from the different experiments

| Architecture | Approach (experiment) | Testing accuracy (%) |
|----------------|-----------------------|----------------------|
| EfficientNetB7 | No Augmentation | 99.61 |
| InceptionV3 | No Augmentation | 99.27 |
| ResNet50 | CycleGAN Augmentation | 99.32 |
| VGG-16 | CC-GAN Augmentation | 99.14 |
| VGG-19 | CycleGAN Augmentation | 99.32 |
| Xception | No Augmentation | 99.32 |
| MobileNetV1 | CycleGAN Augmentation | 99.57 |

7 Limitations

Despite the promising results of the current study, there are still some limitations. First, the unavailability of images is a challenging task. Second, the dataset is collected from different centers, and therefore, there may be some vendors’ differences, such as image quality and encoding formats, that may cause small errors after that. Third, normal data augmentation methods results should be improved. However, the results obtained from our study are promising and the proposed framework’s concepts can be applied in hospitals and COVID-caring centers to make more reliable systems.

Table 17 ML experiments' configurations summarization

| Configuration | Value |
|------------------------|--|
| Dataset Type | Numeric |
| Dataset Size | 231 |
| Dataset Pre-processing | Standardization Method |
| ML Models | Trees (Fine Tree, Medium Tree, and Coarse Tree); Discriminant (Linear Discriminant and Quadratic Discriminant); Regression (Logistic Regression); Naïve Bayes (Gaussian Naïve Bayes and Kernel Naïve Bayes); SVM (Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM, and Coarse Gaussian SVM); KNN (Fine KNN, Medium KNN, Coarse KNN, Cosine KNN, Cubic KNN, and Weighted KNN); and Ensemble (Ensemble Boosted Trees, Ensemble Bagged Trees, Ensemble Subspace Discriminant, Ensemble Subspace KNN, and Ensemble RUS-Boosted Trees) |
| Number of Features | 15 |
| Cross-Validation Folds | 50 |
| Programming Language | MATLAB |
| Performance Metrics | Accuracy |
| Training Environment | Toshiba Qosmio X70-A |

8 Conclusions and future work

Despite the availability of different vaccines, the COVID-19 nightmare is still threatening the lives of millions of people all over the world. Not only fast diagnosis of COVID-19 but also prognosis of the severity of the infection is very important, especially in the lack of oxygen crisis. The deficiency of ICU can result in numerous deaths because of breathing problems. Therefore, a massive effort is done to try to overcome the pandemic. In this work, we proposed a diagnostic and prognostic framework for the COVID-19 patients. For the diagnosis of COVID-19, we applied the transfer learning approach and used 7 different pre-trained CNN structures to classify COVID-19 from CT chest images. The patients' data were collected from the Egyptian Radiology centers. Therefore, data needed pre-processing to be suitable for deep learning. Different data augmentation methods were applied to increase the dataset image diversity to avoid overfitting. 28 experiments were applied using the 7 pre-trained CNN architectures and performance metrics were captured. The two highest testing accuracies were 99.61% and 99.57% and were reported by EfficientNetB7 and MobileNetV1 architectures respectively. For the prognosis of the severity of COVID-19 in positive cases, 25 different machine learning algorithms were applied on numerical test results. Among the different algorithms, the best accuracy value was 98.70% and was reported by the Ensemble Bagged Trees ML method. In future work, we plan to apply the Internet of Things (IoT) as an improvement to our framework. The usage of IoT can facilitate the diagnosis and add more flexibility in resource management.

Table 18 ML algorithms experiment results with standardization

| ML algorithm | Accuracy |
|--------------------------------|----------|
| Fine Tree | 96.5% |
| Medium Tree | 96.5% |
| Coarse Tree | 97.0% |
| Linear Discriminant | 94.8% |
| Quadratic Discriminant | NA |
| Logistic Regression | 95.7% |
| Gaussian Naïve Bayes | NA |
| Kernel Naïve Bayes | 90.0% |
| Linear SVM | 95.2% |
| Quadratic SVM | 95.7% |
| Cubic SVM | 94.8% |
| Fine Gaussian SVM | 56.7% |
| Medium Gaussian SVM | 94.8% |
| Coarse Gaussian SVM | 94.8% |
| Fine KNN | 90.0% |
| Medium KNN | 88.3% |
| Coarse KNN | 68.0% |
| Cosine KNN | 92.6% |
| Cubic KNN | 87.0% |
| Weighted KNN | 90.5% |
| Ensemble Boosted Trees | 56.7% |
| Ensemble Bagged Trees | 98.7% |
| Ensemble Subspace Discriminant | 94.4% |
| Ensemble Subspace KNN | 94.8% |
| Ensemble RUSBoosted Trees | 84.4% |

Table 19 ML algorithms experiment results without standardization

| ML algorithm | Accuracy |
|--------------------------------|----------|
| Fine Tree | 97.4% |
| Medium Tree | 97.4% |
| Coarse Tree | 97.4% |
| Linear Discriminant | 94.8% |
| Quadratic Discriminant | NA |
| Logistic Regression | 96.1% |
| Gaussian Naïve Bayes | NA |
| Kernel Naïve Bayes | 92.2% |
| Linear SVM | 94.8% |
| Quadratic SVM | 95.7% |
| Cubic SVM | 94.8% |
| Fine Gaussian SVM | 56.7% |
| Medium Gaussian SVM | 94.8% |
| Coarse Gaussian SVM | 94.8% |
| Fine KNN | 90.0% |
| Medium KNN | 88.3% |
| Coarse KNN | 68.0% |
| Cosine KNN | 91.8% |
| Cubic KNN | 86.6% |
| Weighted KNN | 90.5% |
| Ensemble Boosted Trees | 56.7% |
| Ensemble Bagged Trees | 98.7% |
| Ensemble Subspace Discriminant | 95.2% |
| Ensemble Subspace KNN | 87.4% |
| Ensemble RUSBoosted Trees | 79.7% |

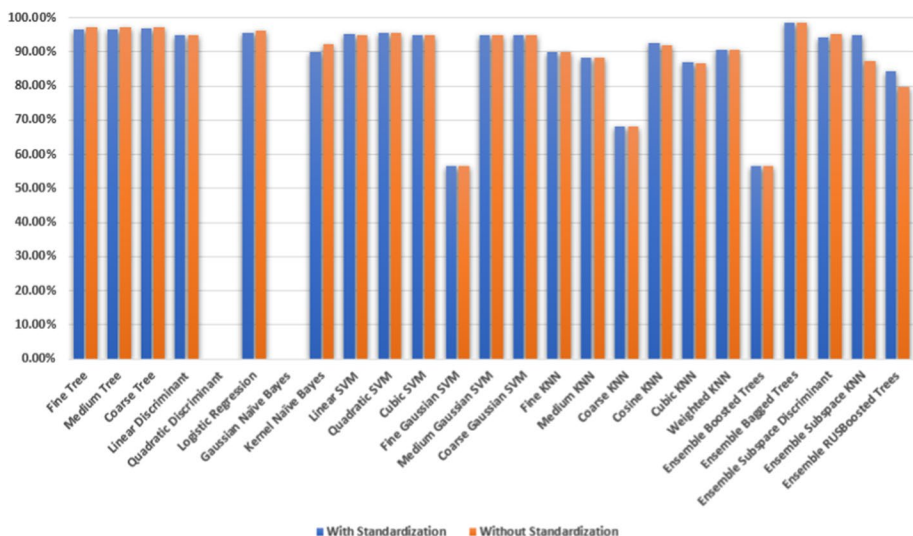


Fig. 18 Graphical comparison between the ML algorithms

Table 20 Comparison with state-of-the-art studies: tabular representation

| Reference | Accuracy |
|------------------------------------|----------|
| Apostolopoulos and Mpesiana (2020) | 96.78% |
| Ozturk (2020) | 98.08% |
| Brunese et al. (2020) | 97% |
| Khan et al. (2020) | 89.6% |
| Ardakani et al. (2020) | 99.51% |
| Zhang (2020) | 92.49% |
| Hu (2020) | 96.2% |
| Karar et al. (2021) | 99.9% |
| Nour et al. (2020) | 98.97% |
| Waheed et al. (2020) | 95% |
| Sakib et al. (2020) | 94.61% |
| Rajaraman (2020) | 99.01% |
| Current Study | 99.61% |

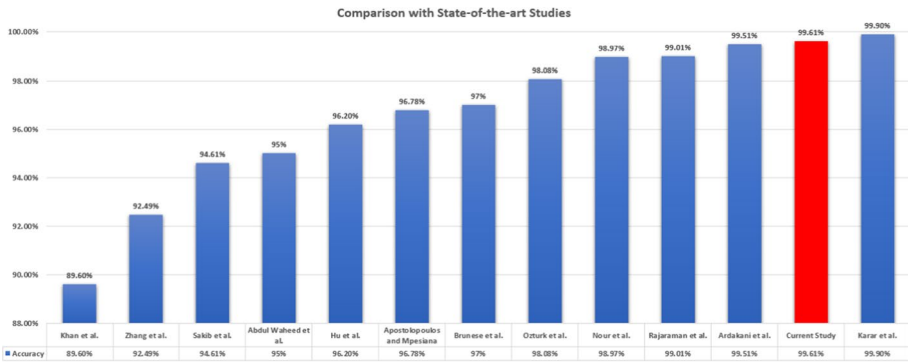


Fig. 19 Comparison with state-of-the-art studies: graphical representation

Appendices

Table of Symbols

Table 21 shows the used symbols in the current study.

Table 21 Table of symbols

| Symbol | Meaning |
|-------------------|--|
| m | The number of features |
| R_m | The dimensional pattern space |
| c | The number of categories (i.e. classes) |
| c_i | A class at index i |
| ω_i | A class region at index i |
| X_u | An unknown pattern |
| f | A set of discriminant functions |
| f_{c_i} | A discriminant function at c_i |
| c_k | A class at index k |
| $P(\omega_i X_u)$ | The posterior probability of finding X_u in the class region ω_i |
| $P(X_u \omega_i)$ | The probability that X_u belongs to the class region ω_i |
| $P(\omega_i)$ | The priori of a class region ω_i |
| $P(X_u)$ | The evidence of X_u |
| n_{c_i} | The number of samples in a class c_i |
| N | The total number of samples in the sample space |
| z | A measure of dependency on variables or the predicted output value |
| X | The variables or inputs |
| i | An index |
| k | An index |
| β_0 | The intercept or bias |
| β_i | The slope of the logistic regression model or the coefficient value at index i |
| V | The number of variables or inputs |
| $P(z)$ | The probability of the variables |
| l | The number of features (i.e., parameters) in X_u |
| j | An index |
| μ | The mean value |
| σ | The standard deviation value |
| h | A smoothing parameter optimized on the training dataset |
| x_{vji} | The value of the feature in the j th position of the v th input in class c_i |
| y_i | The class number |
| w | The weight vector |
| b | The bias |
| C | A parameter |
| L_{hinge} | The hinge loss |
| TP | True positive |
| TN | True negative |
| FP | False positive |
| FN | False negative |

References




- Abdulazeem Y, Balaha HM, Bahgat WM, Badawy M (2021a) Human action recognition based on transfer learning approach. *IEEE Access* 9:82058–82069
- Abdulkareem KH et al (2021b) Realizing an effective covid-19 diagnosis system based on machine learning and iot in smart hospital environment. *IEEE Internet Things J.* <https://doi.org/10.1109/JIOT.2021.3050775>
- Al-Khurayji R, Sameh A (2017) An effective arabic text classification approach based on kernel naive bayes classifier. *Int J Artif Intell Appl.* <https://doi.org/10.1007/s00521-021-06542-1>
- Apostolopoulos ID, Mpesiana TA (2020) Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks. *Phys Eng Sci Med* 43(2):635–640
- Ardakani AA, Kanafi AR, Acharya UR, Khadem N, Mohammadi A (2020) Application of deep learning technique to manage covid-19 in routine clinical practice using CT images: results of 10 convolutional neural networks. *Comput Biol Med* 121:103795
- Balaha HM, Saafan MM (2021) Automatic exam correction framework (AECF) for the MCQS, essays, and equations matching. *IEEE Access* 9:32368–32389
- Bahgat WM, Balaha HM, AbdulAzeem Y, Badawy MM (2021) An optimized transfer learning-based approach for automatic diagnosis of covid-19 from chest X-ray images. *PeerJ Comput Sci* 7:e555
- Balaha HM, Balaha MH, Ali HA (2021a) Hybrid covid-19 segmentation and recognition framework (HMB-HCF) using deep learning and genetic algorithms. *Artif Intell Med* 119:102156
- Balaha HM et al (2021b) Recognizing arabic handwritten characters using deep learning and genetic algorithms. *Multimedia Tools Appl* 80, 32473–32509
- Balaha HM, Ali HA, Badawy M (2021c) Automatic recognition of handwritten arabic characters: a comprehensive review. *Neural Comput Appl* 33(7):3011–3034
- Balaha HM, Ali HA, Saraya M, Badawy M (2021d) A new arabic handwritten character recognition deep learning system (ahcr-dls). *Neural Comput Appl* 33(11):6325–6367
- Baranovskii DS, et al. (2020) Prolonged prothrombin time as an early prognostic indicator of severe acute respiratory distress syndrome in patients with COVID-19 related pneumonia. *Curr Med Res Opin* 37(1):21–25
- Başaran E, Cömert Z, Çelik Y (2020) Convolutional neural network approach for automatic tympanic membrane detection and classification. *Biomed Signal Process Control* 56:101734
- Belhumeur PN, Hespanha JP, Kriegman DJ (1996) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. In: *European conference on computer vision*. Springer, New York, pp 43–58
- Brunese L, Mercaldo F, Reginelli A, Santone A (2020) Explainable deep learning for pulmonary disease and coronavirus COVID-19 detection from X-rays. *Comput Methods Programs Biomed* 196:105608
- Bui HM, Lech M, Cheng E, Neville K, Burnett IS (2016) Using grayscale images for object recognition with convolutional-recursive neural network. In: *2016 IEEE sixth international conference on communications and electronics (ICCE)*. IEEE, Ha Long, pp 321–325
- Cao P, Zhang S, Tang J (2018) Preprocessing-free gear fault diagnosis using small datasets with deep convolutional neural network-based transfer learning. *IEEE Access* 6:26241–26253
- Chollet F (2017) Xception: deep learning with depthwise separable convolutions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 1251–1258
- Chowdhury ME et al (2020) Can AI help in screening viral and COVID-19 pneumonia? *IEEE Access* 8:132665–132676
- Chu C, Zhmoginov A, Sandler M (2017) Cyclegan, a master of steganography. *arXiv preprint.* [arXiv:1712.02950](https://arxiv.org/abs/1712.02950)
- Cui M et al (2019) Regular expression based medical text classification using constructive heuristic approach. *IEEE Access* 7:147892–147904
- Dagher I (2008) Quadratic kernel-free non-linear support vector machine. *J Glob Optim* 41(1):15–30
- Dai B, Fidler S, Urtasun R, Lin D (2017) Towards diverse and natural image descriptions via a conditional GAN. In: *Proceedings of the IEEE international conference on computer vision*, pp 2970–2979
- Dansana D et al (2020) Early diagnosis of covid-19-affected patients based on X-ray and computed tomography images using deep learning algorithm. *Soft Comput.* <https://doi.org/10.1007/s00500-020-05275-y>
- Deepak S, Ameer P (2019) Brain tumor classification using deep cnn features via transfer learning. *Comput Biol Med* 111:103345
- Deng Z, Zhu X, Cheng D, Zong M, Zhang S (2016) Efficient KNN classification algorithm for big data. *Neurocomputing* 195:143–148
- Denton EL, Chintala S, Fergus R et al (2015) Deep generative image models using a Laplacian pyramid of adversarial networks. *arXiv preprint* [arXiv:1506.05751](https://arxiv.org/abs/1506.05751)

- Denton E, Gross S, Fergus R (2016) Semi-supervised learning with context-conditional generative adversarial networks. arXiv preprint [arXiv:1611.06430](https://arxiv.org/abs/1611.06430)
- Dong L, Wesseloo J, Potvin Y, Li X (2016) Discrimination of mine seismic events and blasts using the Fisher classifier, naive Bayesian classifier and logistic regression. *Rock Mech Rock Eng* 49(1):183–211
- Dorj UO, Lee KK, Choi JY, Lee M (2018) The skin cancer classification using deep convolutional neural network. *Multimedia Tools Appl* 77(8):9909–9924
- Elleuch M, Maalej R, Kherallah M (2016) A new design based-SVM of the cnn classifier architecture with dropout for offline arabic handwritten recognition. *Procedia Comput Sci* 80:1712–1723
- Fisher RA (1936) The use of multiple measurements in taxonomic problems. *Ann Eugenics* 7(2):179–188
- Friedman JH (1989) Regularized discriminant analysis. *J Am Stat Assoc* 84(405):165–175
- Gal MS, Rubinfeld DL (2019) Data standardization. *NYUL Rev* 94:737
- Gedraite ES, Hadad M (2011) Investigation on the effect of a gaussian blur in image filtering and segmentation. In: *Proceedings ELMAR-2011*. IEEE, pp 393–396
- Glorot X, Bengio Y (2010) 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
- Goodfellow IJ et al (2014) Generative adversarial networks. arXiv preprint. [arXiv:1406.2661](https://arxiv.org/abs/1406.2661)
- Han Y, Park J, Lee K (2017) Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification. In: *The detection and classification of acoustic scenes and events (DCASE)*, pp 1–5
- Han D, Liu Q, Fan W (2018) A new image classification method using cnn transfer learning and web data augmentation. *Expert Syst Appl* 95:43–56
- He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: *Proceedings of the IEEE international conference on computer vision*, pp 1026–1034
- He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 770–778
- Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR (2012) Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint. [arXiv:1207.0580](https://arxiv.org/abs/1207.0580)
- Horry MJ et al (2020) Covid-19 detection through transfer learning using multimodal imaging data. *IEEE Access* 8:149808–149824
- Howard AG et al (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint. [arXiv:1704.04861](https://arxiv.org/abs/1704.04861)
- Hu S et al (2020) Weakly supervised deep learning for covid-19 infection detection and classification from ct images. *IEEE Access* 8:118869–118883
- Huang X, Shi L, Suykens JA (2013) Support vector machine classifier with pinball loss. *IEEE Trans Pattern Anal Mach Intell* 36(5):984–997
- Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arxiv preprint. [arXiv:1502.03167](https://arxiv.org/abs/1502.03167)
- Isola P, Zhu JY, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 1125–1134
- Jain U et al (2018) Cubic svm classifier based feature extraction and emotion detection from speech signals. In: *2018 International conference on sensor networks and signal processing (SNSP)*. IEEE, pp 386–391
- Jamshidi M et al (2020) Artificial intelligence and covid-19: deep learning approaches for diagnosis and treatment. *IEEE Access* 8:109581–109595
- Jianqiang Z, Xiaolin G (2017) Comparison research on text pre-processing methods on twitter sentiment analysis. *IEEE Access* 5:2870–2879
- Karar ME, Hemdan EED, Shouman MA (2021) Cascaded deep learning classifiers for computer-aided diagnosis of COVID-19 and pneumonia diseases in X-ray scans. *Complex & Intelligent Systems* 7(1):235–247
- Kasinathan G et al (2019) Automated 3-D lung tumor detection and classification by an active contour model and cnn classifier. *Expert Syst Appl* 134:112–119
- Khan AI, Shah JL, Bhat MM (2020) Coronet: A deep neural network for detection and diagnosis of COVID-19 from chest X-ray images. *Comput Methods Prog Biomed* 196:105581
- Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. *Adv Neural Inf Process Syst* 25:1097–1105
- Lan L, Ye C, Wang C, Zhou S (2019) Deep convolutional neural networks for wce abnormality detection: CNN architecture, region proposal and transfer learning. *IEEE Access* 7:30017–30032

- Lawrence S, Giles CL, Tsoi AC, Back AD (1997) Face recognition: a convolutional neural-network approach. *IEEE Trans Neural Netw* 8(1):98–113
- Le DN et al (2021) Iot enabled depthwise separable convolution neural network with deep support vector machine for COVID-19 diagnosis and classification. *Int J Mach Learn Cybernet* 12:3235–3248
- Li H, Lin Z, Shen X, Brandt J, Hua G (2015) A convolutional neural network cascade for face detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 5325–5334
- Li J, Li G, Fan H (2018) Image dehazing using residual-based deep CNN. *IEEE Access* 6:26831–26842
- Liu W et al (2017) A survey of deep neural network architectures and their applications. *Neurocomputing* 234:11–26
- Liu F et al (2018) Deep convolutional neural network and 3d deformable approach for tissue segmentation in musculoskeletal magnetic resonance imaging. *Magn Reson Med* 79(4):2379–2391
- Loh WY (2011) *Classification and regression trees*. Wiley Interdiscip Rev Data Mining Knowl Discov 1(1):14–23
- Mirza M, Osindero S (2014) Conditional generative adversarial nets. *arXiv preprint*. [arXiv:1411.1784](https://arxiv.org/abs/1411.1784)
- Mukherjee R et al (2021) Iot-cloud based healthcare model for COVID-19 detection: an enhanced k-nearest neighbour classifier based approach. *Computing*. <https://doi.org/10.1007/s00607-021-00951-9>
- Murakami Y, Mizuguchi K (2010) Applying the naïve Bayes classifier with kernel density estimation to the prediction of protein–protein interaction sites. *Bioinformatics* 26(15):1841–1848
- Nour M, Cömert Z, Polat K (2020) A novel medical diagnosis model for COVID-19 infection detection based on deep features and bayesian optimization. *Appl Soft Comput* 97:106580
- Ozturk T et al (2020) Automated detection of COVID-19 cases using deep neural networks with X-ray images. *Comput Biol Med* 121:103792
- Panwar H et al (2020) A deep learning and grad-cam based color visualization approach for fast detection of COVID-19 cases using chest X-ray and CT-scan images. *Chaos Solitons Fractals* 140:110190
- Parekh M, Donuru A, Balasubramanya R, Kapur S (2020) Review of the chest CT differential diagnosis of ground-glass opacities in the covid era. *Radiology* 297(3):E289–E302
- Park T, Liu MY, Wang TC, Zhu JY (2019) Semantic image synthesis with spatially-adaptive normalization. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 2337–2346
- Qin Z, Yu F, Liu C, Chen X (2018) How convolutional neural network see the world—a survey of convolutional neural network visualization methods. *arXiv preprint*. [arXiv:1804.11191](https://arxiv.org/abs/1804.11191)
- Rajaraman S et al (2020) Iteratively pruned deep learning ensembles for COVID-19 detection in chest X-rays. *IEEE Access* 8:115041–115050
- Roberts M et al (2021) Common pitfalls and recommendations for using machine learning to detect and prognosticate for COVID-19 using chest radiographs and CT scans. *Nat Mach Intell* 3(3):199–217
- Rojas-Domínguez A, Padierna LC, Valadez JMC, Puga-Soberanes HJ, Fraire HJ (2017) Optimal hyperparameter tuning of SVM classifiers with application to medical diagnosis. *IEEE Access* 6:7164–7176
- Roy S (2021) Physicians’ dilemma of false-positive RT-PCR for COVID-19: a case report. *SN Comp Clin Med* 3(1):255–258
- Sakib S, Tazrin T, Fouda MM, Fadlullah ZM, Guizani M (2020) DI-CRC: deep learning-based chest radiograph classification for COVID-19 detection: a novel approach. *IEEE Access* 8:171575–171589
- Salamon J, Bello JP (2017) Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Process Lett* 24(3):279–283
- Salimans T et al (2016) Improved techniques for training gans. *arXiv preprint*. [arXiv:1606.03498](https://arxiv.org/abs/1606.03498)
- Scholz M, Klinkenberg R (2005) An ensemble classifier for drifting concepts. In: *Proceedings of the second international workshop on knowledge discovery in data streams*, Porto, Portugal, vol 6, pp 53–64
- Sharma N et al (2021) A smart ontology-based iot framework for remote patient monitoring. *Biomed Signal Process Control* 68:102717
- Shen HB, Chou KC (2006) Ensemble classifier for protein fold pattern recognition. *Bioinformatics* 22(14):1717–1722
- Shorten C, Khoshgoftaar TM (2019) A survey on image data augmentation for deep learning. *J Big Data* 6(1):1–48
- Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. *arXiv preprint*. [arXiv:1409.1556](https://arxiv.org/abs/1409.1556)
- Singhal T (2020) A review of coronavirus disease-2019 (COVID-19). *Indian J Pediatr* 87(4):281–286
- Sohrabi C et al (2020) World health organization declares global emergency: a review of the 2019 novel coronavirus (COVID-19). *Int J Surg* 76:71–76
- Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 2818–2826

- Tan M, Le Q (2019) Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning (PMLR), pp 6105–6114
- Tharwat A (2016) Linear vs. quadratic discriminant analysis classifier: a tutorial. *Int J Appl Pattern Recogn* 3(2):145–180
- Tsangaratos P, Iliia I (2016) Comparison of a logistic regression and naïve bayes classifier in landslide susceptibility assessments: the influence of models complexity and training dataset size. *Catena* 145:164–179
- Van Der Walt E, Eloff J (2018) Using machine learning to detect fake identities: Bots vs humans. *IEEE Access* 6:6540–6549
- Vapnik V (2013) *The nature of statistical learning theory*. Springer, Berlin
- Vo AH, Vo MT, Le T et al (2019) A novel framework for trash classification using deep transfer learning. *IEEE Access* 7:178631–178639
- Waheed A et al (2020) Covidgan: data augmentation using auxiliary classifier gan for improved COVID-19 detection. *IEEE Access* 8:91916–91923
- Wang Z et al (2019) Breast cancer detection using extreme learning machine based on feature fusion with CNN deep features. *IEEE Access* 7:105146–105158
- Wang C, Horby PW, Hayden FG, Gao GF (2020) A novel coronavirus outbreak of global health concern. *Lancet* 395(10223):470–473
- Wang SH, Nayak DR, Guttery DS, Zhang X, Zhang YD (2021) Covid-19 classification by CSHNet with deep fusion using transfer learning and discriminant correlation analysis. *Inf Fusion* 68:131–148
- Wong HYF et al (2020) Frequency and distribution of chest radiographic findings in COVID-19 positive patients. *Radiology* 296(2):E72–E78
- Wu Z et al (2019) Studies on different CNN algorithms for face skin disease classification based on clinical images. *IEEE Access* 7:66505–66511
- Wu F et al (2020) A new coronavirus associated with human respiratory disease in China. *Nature* 579(7798):265–269
- Xu S, Li Y (2020) Beware of the second wave of COVID-19. *Lancet* 395(10233):1321–1322
- Xu L, Ren JS, Liu C, Jia J (2014) Deep convolutional neural network for image deconvolution. *Adv Neural Inf Process Syst* 27:1790–1798
- Xu G et al (2019) A deep transfer convolutional neural network framework for EEG signal classification. *IEEE Access* 7:112767–112776
- Yamashita R, Nishio M, Do RKG, Togashi K (2018) Convolutional neural networks: an overview and application in radiology. *Insights Imaging* 9(4):611–629
- Yuan X, Martínez JF, Eckert M, López-Santidrián L (2016) An improved otsu threshold segmentation method for underwater simultaneous localization and mapping-based navigation. *Sensors* 16(7):1148
- Yuan L, Wei X, Shen H, Zeng LL, Hu D (2018) Multi-center brain imaging classification using a novel 3d cnn approach. *IEEE Access* 6:49925–49934
- Zhang S, Li X, Zong M, Zhu X, Wang R (2017) Efficient KNN classification with different numbers of nearest neighbors. *IEEE Trans Neural Netw Learn Syst* 29(5):1774–1785
- Zhang C et al (2018) A hybrid MLP-CNN classifier for very fine resolution remotely sensed image classification. *ISPRS J Photogram Remote Sens* 140:133–144
- Zhang H, Goodfellow I, Metaxas D, Odena A (2019) Self-attention generative adversarial networks. In: International conference on machine learning (PMLR), pp 7354–7363
- Zhang K et al (2020) Clinically applicable AI system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography. *Cell* 181(6):1423–1433
- Zhang YD, Khan MA, Zhu Z, Wang SH (2021a) Pseudo zernike moment and deep stacked sparse autoencoder for COVID-19 diagnosis. *CMC Comput Mater Continua* 69(3):3145–3162
- Zhang YD, Zhang Z, Zhang X, Wang SH (2021b) Midcan: a multiple input deep convolutional attention network for COVID-19 diagnosis based on chest CT and chest X-ray. *Pattern Recogn Lett* 150:8–16
- Zhou ZH (2012) *Ensemble methods: foundations and algorithms*. CRC Press, Boca Raton
- Zhu JY, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp 2223–2232

Authors and Affiliations

Hossam Magdy Balaha¹  · Eman M. El-Gendy¹  · Mahmoud M. Saafan¹ 

Eman M. El-Gendy
eman_elgendy@mans.edu.eg

Mahmoud M. Saafan
saafan2007@mans.edu.eg

¹ Computers and Control Systems Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt