http://communications.science.ankara.edu.tr

Commun.Fac.Sci.Univ.Ank.Ser. A2-A3
Volume 65, Number 2, Pages 79–99 (2023)
DOI:10.33769/aupse.1227857
ISSN 1303–6009 E-ISSN 2618–6462

Research Article; Received: January 08, 2023; Accepted: April 20, 2023

ML BASED PREDICTION OF COVID-19 DIAGNOSIS USING STATISTICAL TESTS

Sifa OZSARI[1], Fatıma Zehra ORTAK[2], Mehmet Serdar GUZEL[1], Mukerrem Bahar BASKIR[3], Gazi Erkan BOSTANCI[1]

[1]Department of Computer Engineering, Ankara University, Ankara, TÜRKİYE
[2]MSc at Big Data and Business Analytics, Istanbul Technical University, İstanbul, TÜRKİYE
[3]Department of Statistics, Bartın University, Bartın, TÜRKİYE

Abstract. The first case of the novel Coronavirus disease (COVID-19), which is a respiratory disease, was seen in Wuhan city of China, in December 2019. From there, it spread to many countries and significantly affected human life. Deep learning, which is a very popular method today, is also widely used in the field of healthcare. In this study, it was aimed to determine the most suitable Deep Learning (DL) model for diagnosis of COVID-19. A popular public data set, which consists of 2482 scans was employed to select the best DL model. The success of the models was evaluated by using different performance evaluation metrics such as accuracy, sensitivity, specificity, precision, F1 score, kappa and AUC. According to the experimental results, it has been observed that DenseNet models, AdaGrad and NADAM optimizers are effective and successful. Also, whether there are statistically significant differences in each performance measure/score of the architectures by the optimizers was observed with statistical tests.

1. INTRODUCTION

The Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) virus, which causes the novel Coronavirus disease (COVID - 19), belongs to the family of coronaviruses, which are large enveloped, positive single-stranded RNA viruses that can infect humans and animals [1]. This disease spread rapidly around the world and have had a serious impact on the health and life of many people [2]. COVID-19, which emerged in November 2019 and defined as an epidemic by the World Health Organization (WHO), is very contagious. The lack of vaccine when it first appeared

Keywords. COVID-19, deep learning, CT images, statistical analysis.
ozsaris@ankara.edu.tr-Corresponding author; 0000-0002-0531-4645
fatmazehraortak@gmail.com; 0000-0002-6420-9116
mguzel@ankara.edu.tr; 0000-0002-3408-0083
baharbaskir@yahoo.com; 0000-0002-1107-0659
ebostanci@ankara.edu.tr; 0000-0001-8547-7569.

is one of the main reasons why the virus is dangerous. Therefore, it is very important to detect the disease quickly and isolate the infected person immediately in order to prevent the spread of the disease [3]. Reverse Transcription Polymerase Chain Reaction (RT-PCR) is the gold standard to diagnose COVID-19 [3–6]. It is performed by detecting the RNA virus that causes disease from sputum or nasopharyngeal swab [3]. However, in addition to the limited number of materials, a certain period of time must pass for the results to get. Chest imaging methods like Computed Tomography (CT) or X-ray are an effective techniques and could be used for diagnosis [2, 3, 5, 7–9]. While X-ray shows visual signs associated with COVID-19 [10], CT images have a high sensitivity for diagnosing COVID-19 [4].

In recent years, Artificial Intelligence (AI) has a major and fast growth in solving the complex subjects in some fields including engineering, medicine, economy. Specifically, Deep Learning (DL) a major area of AI have become very popular in medical applications. Previously, most things were done manually by doctors. With DL, this time-consuming process has started to improve [11]. This has attracted great interest in the proposal and development of deep learning-based studies for the diagnosis of COVID-19 using both CT and X-ray samples such as [12–18].

One of the studies performed on CT images is [15] by Ardakani et al. In their study, they used ten Convolutional Neural Network (CNN) (AlexNet [19], VGG-16, VGG-19 [20], SqueezeNet [21], GoogleNet [22], MobileNetV2 [23], Residual Neural Network (ResNet)-18, ResNet-50, ResNet-101 [24], and Xception [25]) to diagnose COVID-19. 1020 CT slices were used. The number of COVID-19 patients was 108 (laboratory proven) and the number of patients without COVID-19 was 86. The non-COVID-19 group included those with other atypical and viral pneumonia diseases. In all networks, Stochastic Gradient Descent with Momentum (SGDM) was used for the optimizer, 0.01 for the initial learning rate and 5 for the validation frequency. 80% of the data set was employed for training and 20% was considered for validation. The training and validation data set is the same for all networks. The data set was shuffled at each epoch. When the training process did not change remarkably, the training process was stopped. It is noted that among the networks, Xception and ResNet-101 and provided the best performance. Another study is [14]. In this study, authors constructed a multi-view deep learning fusion model, based on the modification of ResNet-50 architecture. They aimed to differentiate the COVID-19 patient with using computed tomography images. Chest CT images of 495 patients were obtained from different hospitals located in China. The data sets were randomly divided into the training set (395 cases), the validation set (50 cases) and the test set (50 cases). For the training set, 294 cases were diagnosed as COVID-19 and 101 were diagnosed as pneumonia. In validation and test sets, 37 cases were diagnosed as COVID-19 and 13 other was pneumonia. RMSprop optimizer with a learning rate of 1×10^{-5} and batch-size of 4 was used to update the parameters of network during training phase. In the study carried out by Singh et al. [17], a CNN model was employed to classify whether the patients

as infected or not. They used the chest CT images. Hyperparameters of CNN, which were kernel size, kernel type, number of epochs, learning rate, padding, stride, hidden layer, activation functions, momentum and batch size, were regulated by using Multi Objective Differential Evolution (MODE) algorithm. Jaiswal et al. [26] proposed a Dense Convolutional Network (DenseNet)-201 based [27] deep transfer learning model to classify patients as COVID-19 infected or not based on chest CT images. They utilized the proposed model to extract feature, followed by appropriate classifiers. A data set consisting of 2492 CT scans available on kaggle was used for experiments. Also, they augmented the data for obtaining higher accuracy. In the study conducted by Wang et al. [28], a weakly-supervised deep learning framework was developed for both COVID-19 classification and lesion localization problems. They segmented the lung area using a pre-trained U-Net [29], then in order to predict the possibility of COVID-19, the segmented 3D lung area was fed into a 3D deep neural network. 499 and 131 3D CT volumes were used for training and testing, respectively. In training of the network, they used Adaptive Moment Estimation (ADAM) [30] optimizer with a constant $1e-5$ learning rate. Epoch size was taken as 100. Another relevant study using deep learning and CT images is [31]. Chen et al. built their study on UNet++ [32] and used Resnet-50 as the base of UNet++. ResNet-50 is pre-trained on the ImageNet dataset. All pre-training parameters of ResNet-50 were transmitted to UNet++. 46,096 anonymous images were used for model creation and validation. Ying et al. [33] designed a pre-trained ResNet-50 model with the addition of Feature Pyramid Network (FPN) to extract the top-K details from CT images. The data set consists of chest CT scans of 88 patients diagnosed with COVID-19, 86 healthy people and 101 patients infected with bacterial pneumonia. The model is capable of both determining the most important part of the images and interpreting the outputs of the neural network using FPN and attention modules (to learn the importance of every detail). In the study of Gozes et al. [34], they first extracted the relevant lung area using segmentation. For this, they trained U-Nnet architecture using 6,150 CT slices of cases with lung abnormalities. Then, they utilized Resnet-50-2D with fine-tuned parameters to detect coronavirus-related abnormalities. He et al. [35] aimed to develop deep learning methods that can give high diagnostic accuracy rate even the training CT samples are limited. They presented an Self-Trans approach. In order to reduce the over-fitting risk, contrastive self-supervised learning was combined with transfer learning for learning robust and unbiased feature representations. Besides, they published a public data set containing hundreds of COVID-19 positive CT scans.

This study, on the other hand, claims to obtain the best model by using different Deep Learning (DL) architectures with varying optimizers, tested on a public

CT data set [36]. While DenseNet-121, DenseNet-201, DenseNet-169 [27], MobileNetV2 [23], VGG16, VGG19 [20], U-Net [29] and ResNet-50 [24] were determined as architectures, different optimizers, involving Stochastic Gradient Descent (SGD), Adaptive Gradient Algorithm (AdaGrad) [37], ADAM, RMSProp and Nesterov-Accelerated Adaptive Moment estimation (NADAM) [38] were integrated into the models. In addition, statistical tests were used to examine whether there were statistically significant differences in each performance measure/score of the architectures by the optimizers.

The structure of the rest of this paper is as follows. In Section 2, the data set, architectures and optimizers used in the study are presented. In section 3, model parameters are detailed regarding models. Experimental results, success of the models and statistical tests are also discussed in this section, whereas, Section 4 concludes the study.

2. Material and Method

In this section, information about the data set and models used in the study is detailed. Subsections involve Data set and Architecture.

2.1. **Data set.**

In this study, a benchmark public data set (available in Kaggle) consisting of 2482 CT scans of patients [36] is employed. The data set contains 1252 SARS-CoV-2 infected CT scans and 1230 CT scans non-infected by SARS-CoV-2. Patients who are not infected by COVID-19 have other pulmonary diseases. The data was collected from hospitals of Sao Paulo, Brazil. Figure 1 illustrates sample CT scans from this data set. 2234 of these CT scans were used for training, whereas the remaining 248 scans were employed for testing.

2.2. **Architectures.**

2.2.1. *DenseNet.*

DenseNet is one of the leading DL architecture, connecting each layer to following layers in a feed-forward approach, was proposed in [27]. With this architecture, it is aimed to deepen deep learning networks, to be more accurate, as well as to make them more efficient to train by using shorter connections between layers. There are several important advantages of DenseNets, which are detailed as follows [27]:

- They reduce the vanishing-gradient problem.
- They strengthen feature propagation.
- They encourage feature reuse.
- They quite decrease the number of parameters.

In addition, they showed that DenseNets scales to hundreds of layers without optimization difficulties.

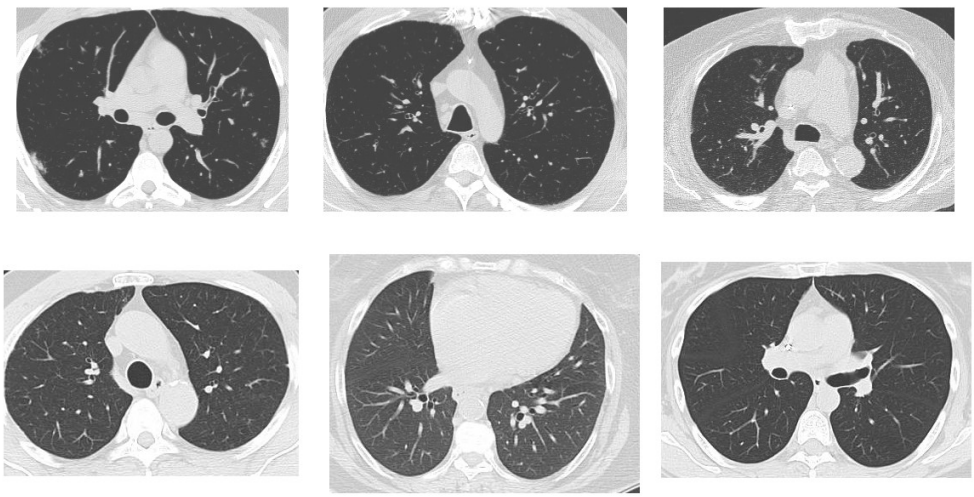


FIGURE 1. Sample CT scans [36].

2.2.2. *MobileNetV2.* MobileNetV2, introduced by [23], is a neural network architecture specifically developed for resource constrained environments, such as mobile platforms. MobileNetV2 architecture involves a novel layer, providing low-dimensional compressed representation for input data. Firstly, this representation is extended to high dimension and filtered with a "lightweight depth-wise convolution". Afterwards, features are projected back to a low-dimensional representation by using a linear convolution filter. [23]. Although MobileNetV2 uses deep separable convolution, the point where it differs is that it has a bottleneck residual block rather than a deep separable convolution block.

2.2.3. *VGG.* VGG16 is a convolutional neural network model, achieved to win the first and second place in the localization and classification tracks respectively in "The ImageNet Large Scale Visual Recognition Challenge 2014". It was proposed by [20]. During the training, the input of ConvNet is a fixed size 224 × 224 RGB image. The image passes through a convolutional layer stacks, in which filters (3×3) with a very small receptive are used. 1×1 convolution filters, which can be viewed as a linear transformation of the input channels (followed by nonlinearity), are used in one of the configurations. The padding is 1 pixel for 3×3 convolution layers and spatial pooling is performed with five max-pooling layers. Five max-pooling layers follow some convolution layers, but max-pooling (which is carried out over a 2×2 pixel window, with stride 2) does not follow all convolution layers. Three Fully-Connected (FC) layers follow a convolutional layers stack. In different

architectures, this convolutional layers stack may have different depth. The softmax, the classifier layer, is the final layer. In all networks, the configuration of fully connected layers is defined as identical can be seen in [20].

2.2.4. *U-Net.* U-Net [29] is a convolutional neural network developed for obtaining better segmentation performance in case of having limited amount of biomedical image data. It was proposed by Ronnenberger et al. [29]. In their study, they offered a network and training scheme based on the excessive use of data augmentation to employ existing annotated samples more effectively. It is an important change of the architecture they present to have a large number of feature channels in the upsampling section. These feature channels provide the network to pass context information to higher layers [29].

2.2.5. *ResNet.* ResNet architecture is one of the most popular deep neural networks available in many varieties with different number of layers. It was the winner of ILSVRC & COCO 2015 competitions on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation. ResNet was introduced by [24]. He et al. showed that the accuracy becomes saturated with increasing network depth, adding more layers to the network results in higher training error. In addition, they showed that this was not caused by overfitting contrary to popular belief. Due to the problem of vanishing/exploding gradients, training of deep networks is difficult. An identity shortcut connection that skips one or more layers was defined in ResNet. Thus, the number of layers can be increased without the problem of vanishing gradients.

2.3. **Optimizer.** Optimizer is used to reduce the loss, which is the difference between actual value and predicted value. The choice of these algorithm or method is very important. In this study, five different optimizers, which are SGD, AdaGrad, RMSProp, ADAM and NADAM, are used.

Stochastic gradient descent is a popular iterative method used to optimize an objective function. In SGD, which is a variant of Gradient Descent (GD), random samples are selected from training data in each iteration to update the parameter during optimization. Equation 1 is used for parameter updating:

$$\theta = \theta - \eta . \bigtriangledown_\theta J(\theta; x, y) \tag{1}$$

where, θ is a parameter, η is the learning rate, $\bigtriangledown$ is the gradient and J is objective function. SGD performs one update at a time.

AdaGrad [37], which adapts the learning rate to the parameters, is an algorithm for gradient-based optimization [39]. Each parameter in AdaGrad has its own learning rate, so it eliminates manual adjustment of the learning rate. It decreases the learning rate of parameters proportionally to previous updates of parameters. AdaGrad makes large updates for infrequent parameters, while smaller updates for frequent parameters. The disadvantage of AdaGrad is that the system stops learning after a certain point due to the reduction of the learning rate.

RMSprop[1], which was proposed by Geoff Hinton, is an unpublished adaptive learning rate method. It is one of the algorithms developed due to the need to solve the decreasing learning rates problem in AdaGrad. RMSprop keeps the moving average of the squared gradients and divides the gradient by the root of this average. The update rule is as follows [39]:

$$\begin{aligned} E[g^2]_t &= \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2 \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \end{aligned} \tag{2}$$

where, $E[g]$ is the running average. While Hinton recommends setting γ (moving average parameter) to 0.9, a good default value for η (learning rate) is 0.001.

The ADAM [30], which is an adaptive learning rate optimization algorithm, was introduced by Kingma and Ba in [30] study. They designed ADAM to combine the advantages of two popular methods, AdaGrad and RMSProp. Some of the advantages that ADAM has are as follows [30]:

- ADAM's step size is approximately limited by the step size hyper parameter.
- A fixted objective is not required.
- It works with sparse gradients.
- A form of step size annealing is naturally.
- The magnitudes of the parameter updates does not change with the rescaling of the gradient.

In the ADAM algorithm, adaptive learning rates are calculated for each parameter. It stores an exponentially decreasing average of past square of gradients, as in Adadelta and RMSprop. It also maintains an exponentially decreasing average of past gradients. ADAM update rule is given in the following equation [39]:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{\upsilon}_t} + \epsilon} \hat{m}_t \tag{3}$$

where, $\hat{m}_t$ and $\hat{\upsilon}_t$ are first moment vector and the second moment vector respectively.

Finally, NADAM [38], which utilize ADAM optimizer with Nesterov Accelerated Gradient (NAG), is a variation of ADAM. NADAM can be used on noisy gradients as well as gradients with high curvatures. Equation 4 shows the equation of NADAM update rule.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{\upsilon}_t} + \epsilon} (\beta_1 \hat{m}_t + \frac{(1-\beta_1) g_t}{1-\beta_1^t}) \tag{4}$$

3. The Experimental Section and Discussion

In this section, parameter values used in the study are explained and experiments have been carried out. Success of the models on data set is observed.

[1]http://www.cs.toronto.edu/ tijmen/csc321/slides/lecture_slides_lec6.pdf

3.1. **Parameter Settings.** In artificial neural networks, activation function generates the output of a node. Since it has an significant effect on success of neural network, the choice of activation function is very important for design of a neural network. In this study, while Rectified Linear Unit (ReLU) [40] is used in the hidden layer, sigmoid is used in the output layer. The batch-size is a hyperparameter that corresponds to the number of training examples to propagated through the network. Learning rate, that has an effect on updating weights of model at each iteration, is one of the important hyperparameters for deep neural networks. Iteration number, batch-size and learning rate values were set as follows:

- Iteration number: 50, 100
- Batch-size: 128
- Learning rate: Learning rate was determined using ReduceLROnPlateau. Started with 0.01, the minimum can be 0.00001.

3.2. **Experimental Results.** Models were trained separately on the training data set with each optimizer and iteration value. After each training, the models were tested. Accuracy, sensitivity (recall), specificity, precision, F1 score, kappa and Area Under the ROC Curve (AUC) were used to evaluate the performance of architectures. In equations 5, 6, 7, 8 and 9, the formulas for "accuracy", "sensitivity", "specificity", "precision" and "F1 score" are given respectively.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{5}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{6}$$

$$Specificity = \frac{TN}{TN + FP} \tag{7}$$

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

$$F1\ score = 2 * \frac{precision * recall}{precision + recall} \tag{9}$$

For equations 5, 6, 7 and 8, TP corresponds to true positives, TN true negatives, FN false negatives and FP false positives. Table 1 indicates the performances of networks. In the table, iteration is abbreviated as iter, accuracy as acc, sensitivity as sens and specificity as spec, precision as prec and F1 score as F1.

TABLE 1. Performances of networks.

Networks	Optimizer	Iter	Acc	Sens	Spec	Prec	F1	Kappa	AUC
DenseNet-121	ADAM	50	0.94	0.94	0.95	0.94	0.94	0.89	0.94
DenseNet-121	AdaGrad	50	0.93	0.99	0.88	0.85	0.93	0.85	0.92
DenseNet-121	SGD	50	0.92	0.99	0.87	0.84	0.92	0.83	0.91
DenseNet-121	NADAM	50	0.90	0.90	0.90	0.89	0.90	0.79	0.89
DenseNet-121	RMSprop	50	0.92	0.96	0.89	0.87	0.92	0.83	0.91
DenseNet-121	ADAM	100	0.94	0.93	0.96	0.95	0.94	0.88	0.94
DenseNet-121	AdaGrad	100	0.91	0.99	0.85	0.81	0.91	0.81	0.90
DenseNet-121	SGD	100	0.91	0.99	0.85	0.81	0.91	0.81	0.90
DenseNet-121	NADAM	100	0.94	0.98	0.90	0.89	0.94	0.87	0.93
DenseNet-121	RMSprop	100	0.88	0.97	0.83	0.78	0.89	0.76	0.87
DenseNet-201	ADAM	50	0.93	0.96	0.90	0.89	0.93	0.86	0.92
DenseNet-201	AdaGrad	50	0.87	0.99	0.81	0.74	0.89	0.74	0.87
DenseNet-201	SGD	50	0.91	0.99	0.85	0.81	0.91	0.80	0.81
DenseNet-201	NADAM	50	0.94	0.93	0.95	0.94	0.94	0.87	0.93
DenseNet-201	RMSprop	50	0.89	0.97	0.83	0.78	0.90	0.77	0.88
DenseNet-201	ADAM	100	0.93	0.95	0.92	0.91	0.93	0.87	0.93
DenseNet-201	AdaGrad	100	0.91	0.99	0.86	0.83	0.92	0.82	0.91
DenseNet-201	SGD	100	0.93	0.97	0.89	0.87	0.93	0.85	0.92
DenseNet-201	NADAM	100	0.94	0.93	0.95	0.94	0.94	0.87	0.94
DenseNet-201	RMSprop	100	0.94	0.95	0.93	0.93	0.94	0.88	0.94
DenseNet-169	ADAM	50	0.93	0.96	0.91	0.89	0.93	0.86	0.93
DenseNet-169	AdaGrad	50	0.96	0.97	0.94	0.94	0.96	0.91	0.95
DenseNet-169	SGD	50	0.89	0.98	0.83	0.78	0.90	0.78	0.88
DenseNet-169	NADAM	50	0.88	0.82	0.94	0.94	0.88	0.76	0.88
DenseNet-169	RMSprop	50	0.79	0.99	0.71	0.57	0.83	0.58	0.78
DenseNet-169	ADAM	100	0.90	0.88	0.92	0.92	0.90	0.80	0.90
DenseNet-169	AdaGrad	100	0.93	0.98	0.88	0.86	0.93	0.85	0.92
DenseNet-169	SGD	100	0.86	0.99	0.79	0.72	0.88	0.73	0.85
DenseNet-169	NADAM	100	0.91	0.88	0.93	0.93	0.90	0.81	0.90
DenseNet-169	RMSprop	100	0.92	0.93	0.91	0.89	0.92	0.83	0.91

MobileNetV2	ADAM	50	0.76	0.98	0.69	0.52	0.81	0.52	0.75
MobileNetV2	AdaGrad	50	0.74	0.89	0.69	0.57	0.78	0.47	0.73
MobileNetV2	SGD	50	0.72	0.98	0.65	0.42	0.78	0.42	0.70
MobileNetV2	NADAM	50	0.91	0.95	0.88	0.86	0.91	0.82	0.90
MobileNetV2	RMSprop	50	0.87	0.93	0.82	0.78	0.88	0.73	0.86
MobileNetV2	ADAM	100	0.90	0.96	0.86	0.84	0.91	0.80	0.89
MobileNetV2	AdaGrad	100	0.72	0.71	0.74	0.73	0.72	0.44	0.72
MobileNetV2	SGD	100	0.90	0.97	0.75	0.64	0.84	0.63	0.81
MobileNetV2	NADAM	100	0.88	0.94	0.84	0.81	0.89	0.76	0.88
MobileNetV2	RMSprop	100	0.86	0.94	0.82	0.77	0.87	0.72	0.86
VGG16	ADAM	50	0.91	0.98	0.85	0.82	0.91	0.81	0.90
VGG16	AdaGrad	50	0.81	0.93	0.75	0.67	0.83	0.61	0.80
VGG16	SGD	50	0.86	0.92	0.82	0.78	0.86	0.70	0.85
VGG16	NADAM	50	0.89	0.97	0.85	0.81	0.91	0.79	0.89
VGG16	RMSprop	50	0.92	0.96	0.88	0.86	0.92	0.83	0.91
VGG16	ADAM	100	0.89	0.96	0.84	0.80	0.90	0.78	0.88
VGG16	AdaGrad	100	0.86	0.92	0.82	0.78	0.86	0.70	0.85
VGG16	SGD	100	0.87	0.93	0.84	0.80	0.88	0.74	0.87
VGG16	NADAM	100	0.92	0.96	0.90	0.89	0.93	0.85	0.92
VGG16	RMSprop	100	0.91	0.95	0.89	0.87	0.92	0.83	0.91
VGG19	ADAM	50	0.86	0.94	0.82	0.77	0.87	0.72	0.85
VGG19	AdaGrad	50	0.78	0.90	0.73	0.63	0.81	0.55	0.77
VGG19	SGD	50	0.73	0.80	0.71	0.65	0.76	0.46	0.73
VGG19	NADAM	50	0.89	0.93	0.87	0.84	0.90	0.79	0.89
VGG19	RMSprop	50	0.87	0.93	0.84	0.80	0.88	0.74	0.87
VGG19	ADAM	100	0.89	0.96	0.86	0.83	0.90	0.79	0.89
VGG19	AdaGrad	100	0.81	0.91	0.76	0.69	0.83	0.61	0.80
VGG19	SGD	100	0.73	0.83	0.71	0.63	0.76	0.47	0.73
VGG19	NADAM	100	0.89	0.93	0.87	0.85	0.90	0.79	0.89
VGG19	RMSprop	100	0.88	0.94	0.84	0.81	0.89	0.76	0.88
U-Net	ADAM	50	0.80	0.98	0.73	0.61	0.84	0.60	0.79
U-Net	AdaGrad	50	0.66	0.98	0.60	0.31	0.75	0.30	0.64
U-Net	SGD	50	0.64	0.99	0.59	0.26	0.74	0.26	0.63

U-Net	NADAM	50	0.82	0.97	0.75	0.64	0.84	0.63	0.81
U-Net	RMSprop	50	0.83	0.98	0.75	0.65	0.85	0.64	0.82
U-Net	ADAM	100	0.78	0.98	0.71	0.56	0.82	0.55	0.77
U-Net	AdaGrad	100	0.80	0.98	0.72	0.59	0.83	0.59	0.79
U-Net	SGD	100	0.68	0.98	0.62	0.36	0.76	0.35	0.67
U-Net	NADAM	100	0.80	0.98	0.73	0.61	0.84	0.60	0.79
U-Net	RMSprop	100	0.75	0.98	0.68	0.50	0.80	0.49	0.74
ResNet-50	ADAM	50	0.67	0.98	0.61	0.33	0.75	0.32	0.66
ResNet-50	AdaGrad	50	0.60	0.98	0.57	0.33	0.72	0.18	0.58
ResNet-50	SGD	50	0.50	0.98	0.60	0.33	0.66	0.12	0.52
ResNet-50	NADAM	50	0.61	0.98	0.57	0.20	0.72	0.19	0.59
ResNet-50	RMSprop	50	0.69	0.95	0.64	0.42	0.76	0.38	0.68
ResNet-50	ADAM	100	0.68	0.96	0.63	0.37	0.76	0.35	0.67
ResNet-50	AdaGrad	100	0.61	0.98	0.57	0.33	0.72	0.20	0.60
ResNet-50	SGD	100	0.52	0.99	0.52	0.35	0.68	0.15	0.50
ResNet-50	NADAM	100	0.69	0.95	0.63	0.41	0.76	0.37	0.68
ResNet-50	RMSprop	100	0.64	0.98	0.59	0.26	0.74	0.26	0.62

When the Table 1 is examined, DenseNet-169 have the highest accuracy rate with 96%. For this rate, the optimizer is AdaGrad and the number of iterations is 50. When the maximum values produced by other architectures are examined, NADAM was used in 8 experiments, RMSprop in 4 experiments and ADAM optimizer in 3 experiments. 7 of these results were obtained from 50 iterations and 8 of them were obtained from 100 iterations. ResNet-50 yielded the lowest accuracy value of 50% with SGD optimizer at 50 iterations. In addition, the highest accuracy rate of 69% of ResNet-50 is considerably lower than other architectures. DenseNet's are very successful with results of 94% and above.

When the AUC values are examined, with a value of 0.95, DenseNet-169 have the best result in 50 iterations and AdaGrad optimizer. Considering the best values of other architectures, NADAM was used in 6 results, ADAM and RMSprop optimizer was used in 3 results. 5 of them belong to 50 iterations and 7 of them to 100 iterations. ResNet-50 gave the worst AUC value of 0.5 at 100 iterations with SGD optimizer. The AUC value of 0.68 of ResNet-50 is lower than the best results yielded by other architectures. According to all the AUC values, DenseNets are more efficient.

When the results obtained with the kappa evaluation method are analyzed, it is seen that DenseNet-169 is the most successful architecture with AdaGrad optimizer and 50 iterations. This kappa value is 0.91. ResNet-50 is the lowest rate network

with 0.12. At this value, SGD was used as optimizer and the number of iterations was 50.

In precision values, the best result belongs to DenseNet-121 with 0.95. This value was obtained by using 100 iterations and ADAM opitmizer. The worst result is 0.20 and it was produced by ResNet-50. It is seen that NADAM was used as optimizer and the number of iterations was 50. When all precision values are examined, it is seen that NADAM was used as optimizer and the number of iterations was 50 in most of the best results produced by the models. Again, it is clear that DenseNets are more successful in these results.

According to the specificity results, DenseNet-121 is the best architecture, while ResNet-50 is the most unsuccessful network. DenseNet-121 yielded 0.96 in 100 iterations with ADAM optimizer. In the worst result, SGD optimizer was used and the number of iterations was 100. It is seen that NADAM was used as optimizer and the number of iterations was 50 in most of the best values that the architectures had.

The most effective architectures for sensitivity values are DenseNets, U-Net and ResNet-50 with the value of 0.99. DenseNet-121 gave this value as a result of the experiments using AdaGrad and SGD optimizer in 50 and 100 iterations, respectively. DenseNet-201 yielded 0.99 sensitivity value with AdaGrad optimizer at 50 and 100 iterations and with SGD optimezer at 50 iterations. In DenseNet-169, RMSprop was used in 50 iterations and SGD optimizer in 100 iterations. SGD optimizer was used in ResNet-50 and U-Net networks and the iteration numbers were 100 and 50 respectively.

Finally, considering the F1 scores, the best result is 0.96, the worst result is 0.66. In the 0.96 value produced by DenseNet-169 architecture, AdaGrad optimizer was used and the number of iterations was 50. 0.66 belongs to ResNet-50 network with SGD optimizer 50 iterations. It is seen that DenseNet models are superior in F1 score values too.

When the evaluation is made considering all the results in the table, it is seen that DenseNets are the most successful architecture while ResNet-50 is a less effective network. Although AdaGrad was the optimizer for the majority of the highest results, the NADAM optimizer in general also produced effective results. The success of 50 iterations shows that effective results can be obtained with a small number of iterations. Figure 2 and 3 show Grad-CAM [41] visualization of 4 and 5 images, respectively, classified as COVID-19 using DenseNet169 50 iterations and AdaGrad optimizer. Figure 4 shows the normalized confusion matrix and ROC curve of DenseNet169 with 50 iterations and AdaGrad optimizer.

3.3. **Statistical Significance in Each Performance Evaluation.** In this study, each architecture was evaluated by the well-known performance measures. These performance measures were calculated for five optimizers with 50 and 100 iterations. One of the important concerns is whether there were statistically significant

TABLE 2. Comparisons of the optimizers by the architecture-performances ($iter = 50$).

	Assumption		Comparing the optimizers	
Performance Score (PS)	Normality	Variance-homogeneity	Hypothesis testing ($p-value$)	Pairwise comparisons
PS^1 : Acc-score	✗ ($p-value < 0.005$)	✓ ($p-value = 0.508$)	No difference/Similar effects ($p-value = 0.052$)	—
PS^2 : Sens-score	✗ ($p-values < 0.005$)	✓ ($p-value = 0.649$)	No difference/Similar effects ($p-value = 0.545$)	—
PS^3 : Spec-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.890$)	Significant difference/Dissimilar effects ($p-value = 0.011$)	Significant differences: •NADAM-AdaGrad ($95\%CI = (0.64, 17.86)$, $p-value = 0.031$) •SGD-NADAM ($95\%CI = (-18.48, -1.27)$, $p-value = 0.018$)
PS^4 : Prec-score	✗ ($p-value < 0.005$)	✓ ($p-value = 0.873$)	Significant difference/Dissimilar effects ($p-value = 0.030$)	Significant differences: •ADAM-SGD ($95\%CI = (5.0, 22.5)$, $p-value = 0.022$) •NADAM-SGD ($95\%CI = (1.5, 30.0)$, $p-value = 0.049$)
PS^5 : F1-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.636$)	Significant difference/Dissimilar effects ($p-value = 0.014$)	Significant difference: •SGD-NADAM ($95\%CI = (-11.49, -0.26)$, $p-value = 0.037$)
PS^6 : Kappa-score	✗ ($p-value < 0.005$)	✓ ($p-value = 0.350$)	No difference/Similar effects ($p-value = 0.052$)	—
PS^7 : AUC-score	✗ ($p-value < 0.005$)	✓ ($p-value = 0.449$)	Significant difference/Dissimilar effects ($p-value = 0.015$)	Significant differences: •AdaGrad-ADAM ($95\%CI = (-10.0, -2.0)$, $p-value = 0.030$) •ADAM-SGD ($95\%CI = (5.0, 13.5)$, $p-value = 0.014$) •NADAM-SGD ($95\%CI = (2.0, 17.0)$, $p-value = 0.035$)

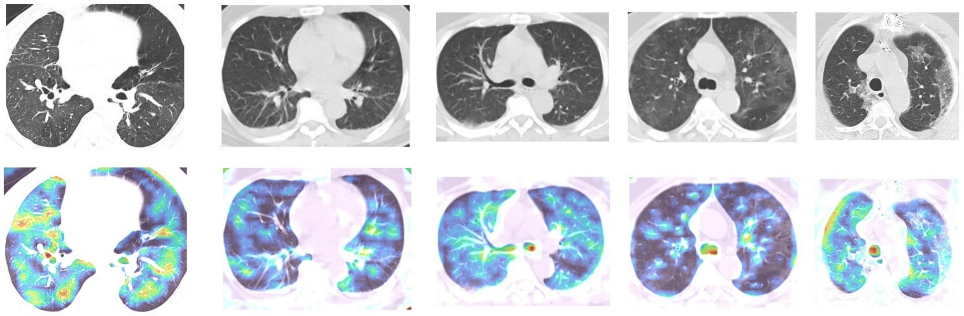


FIGURE 2. Grad-CAM [41] visualization. First row is original images with COVID-19; second row is Grad-CAM visualizations.

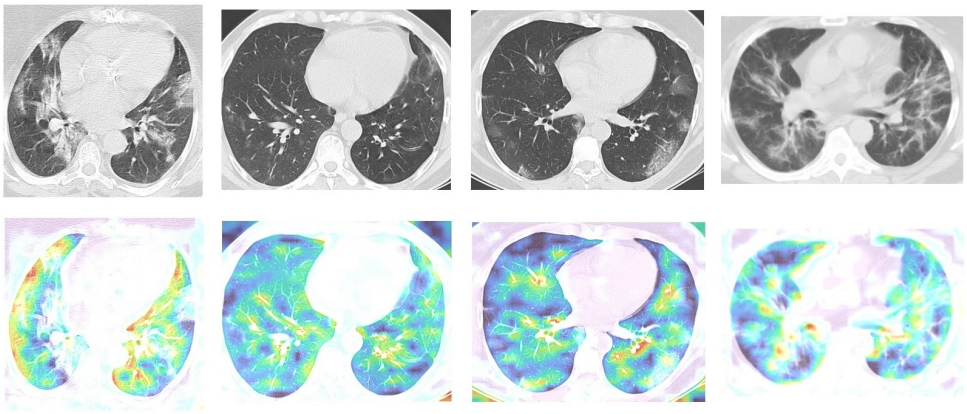


FIGURE 3. Grad-CAM [41] visualization: First row is original images with COVID-19; second row is Grad-CAM visualizations.

differences in each performance measure/score of the architectures by the optimizers. Performance-based data was organized and prepared as in Figure 5. Normality test was performed by using Shapiro-Wilk test, where α-level equals 0.01. Variance-homogeneity was investigated using Bartlett test for normally-distributed data and Levene test for non-normally distributed data. Comparisons of k-paired samples ($k > 2$) were analyzed by using two-way variance analysis where normality and variance-homogeneity assumptions were provided. Turkey test was conducted to perform pairwise-comparisons. Besides, Friedman test was used when normality and/or variance-homogeneity assumptions were not provided. Besides of comparing grand and estimated medians, pairwise-comparisons of k-paired samples were

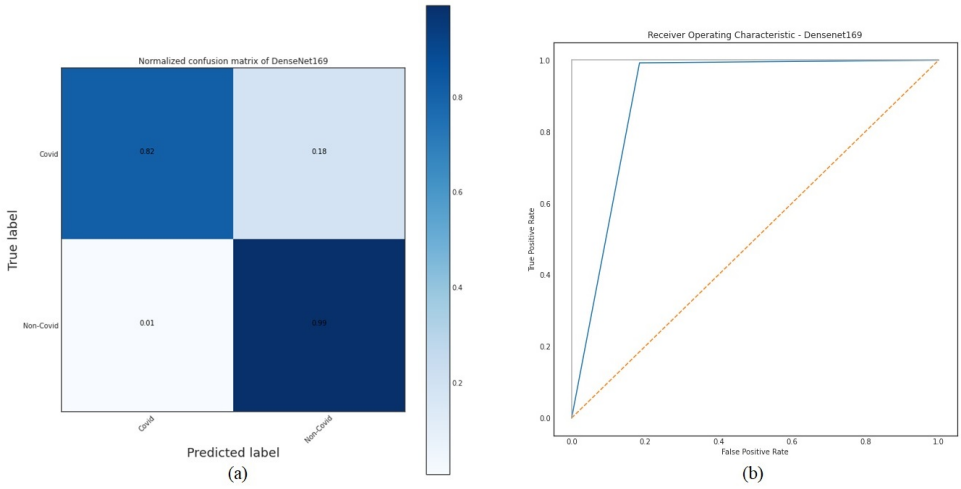


FIGURE 4. Normalized confusion matrix and ROC curve: The left image is normalized confusion matrix; the right one is ROC curve

performed by using Wilcoxon sign-rank tests. Significance level was selected as 0.05 to compare the optimizers. The lists of assumption-results and statistical inferences were summarized as in Table 2 and Table 3, where iteration number equals 50 and 100 respectively.

TABLE 3. Comparisons of the optimizers by the architecture-performances ($iter = 100$).

	Assumption		Comparison of k-paired samples ($p-value$)	Pairwise-comparisons
Performance Score (PS)	Normality	Variance-homogeneity		
PS^1 : Acc-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.655$)	Significant difference/Dissimilar effects ($p-value = 0.011$)	Significant differences: •SGD-ADAM ($95\%CI = (-12.57, -0.18)$, $p-value = 0.042$) •SGD-NADAM ($95\%CI = (-13.32, -0.93)$, $p-value = 0.018$)
PS^2 : Sens-score	✗ ($p-values < 0.005$)	✓ ($p-value = 0.565$)	No difference/Similar effects ($p-value = 0.636$)	—

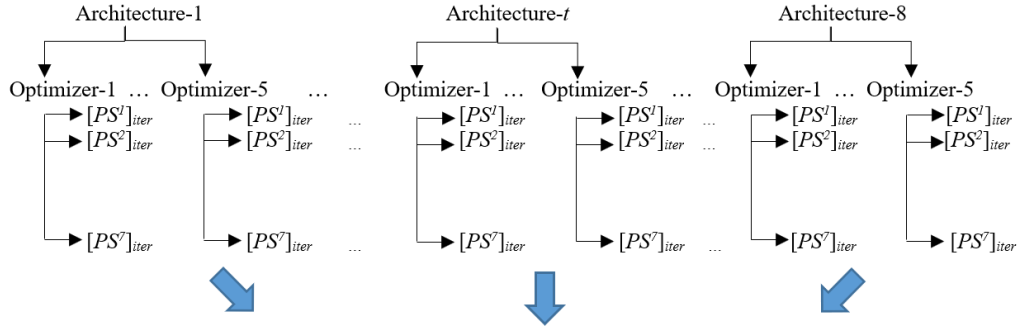
PS^3 : Spec-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.988$)	Significant difference/Dissimilar effects ($p-value = 0.000$)	Significant differences: •ADAM-AdaGrad ($95\%CI = (1.81, 10.69)$, $p-value = 0.003$) •NADAM-AdaGrad: ($95\%CI = (2.43, 11.32)$, $p-value = 0.001$) •SGD-ADAM: ($95\%CI = (-13.57, -4.68)$, $p-value = 0.000$) •SGD-NADAM: ($95\%CI = (-14.19, -5.31)$, $p-value = 0.000$) •SGD-RMSprob: ($95\%CI = (-10.94, -2.06)$, $p-value = 0.002$)
PS^4 : Prec-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.964$)	Significant difference/Dissimilar effects ($p-value = 0.000$)	Significant differences: •NADAM-AdaGrad: ($95\%CI = (1.29, 16.46)$, $p-value = 0.016$) •SGD-ADAM: ($95\%CI = (-20.08, -4.92)$, $p-value = 0.000$) •SGD-NADAM: ($95\%CI = (-21.96, -6.79)$, $p-value = 0.000$) •SGD-RMSprob: ($95\%CI = (-15.46, -0.29)$, $p-value = 0.039$)
PS^5 : F1-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.814$)	Significant difference/Dissimilar effects ($p-value = 0.005$)	Significant difference: •SGD-ADAM ($95\%CI = (-10.14, -0.36)$, $p-value = 0.031$) •SGD-NADAM ($95\%CI = (-10.64, -0.86)$, $p-value = 0.015$)

PS^6 : Kappa-score	✓ ($p-values > 0.01$)	✓ ($p-value4 = 0.898$)	Significant difference/Dissimilar effects ($p-value = 0.001$)	Significant difference: •NADAM-AdaGrad ($95\%CI = (0.85, 21.65)$, $p-value = 0.029$) •SGD-ADAM ($95\%CI = (-24.03, -3.22)$, $p-value = 0.006$) •SGD-NADAM ($95\%CI = (-25.28, -4.47)$, $p-value = 0.002$)
PS^7 : AUC-score	✓ ($p-values > 0.01$)	✓ ($p-value = 0.745$)	Significant difference/Dissimilar effects ($p-value = 0.001$)	•SGD-ADAM ($95\%CI = (-13.30, -2.20)$, $p-value = 0.003$) •SGD-NADAM ($95\%CI = (-14.05, -2.95)$, $p-value = 0.001$) •SGD-RMSprob ($95\%CI = (-11.55, -0.45)$, $p-value = 0.029$)

4. Conclusion

In this study, the success of optimizers in diagnosing disease from COVID-19 CT images using different optimizers in different architectures was examined. In addition, the number of iterations was set at two different values, 50 and 100. DenseNet-169, DenseNet-121, DenseNet-201, MobileNetV2, U-Net, ResNet-50, VGG16 and VGG19 were used as models. The efficiency of ADAM, AdaGrad, SGD, NADAM and RMSprop optimizers was observed. Accuracy, sensitivity, specificity, precision, F1 score, kappa and AUC were used as evaluation metrics. According to the results, DenseNets were quite successful, while ResNet-50 was the less effective architecture. While NADAM is the superior optimizer for the majority of architectures' own best results, the majority of the top values in evaluation metrics include AdaGrad optimizer. Considering that the number of images in the data set used in the study is insufficient, it should be noted that the models yield very good results.

The differences in architecture-performances can be effected by the selected optimizers. Thus, the optimizer-effects were analyzed for each performance metric of the architectures. As the results of statistical inferences, there were statistically significant differences in 4 out of 7 architecture-performance metrics and 6 out of 7 architecture-performance metrics by the optimizers, when iteration-numbers were 50 and 100. According to pairwise comparisons, it has been seen that these



	Architecture-1	….	Architecture-8
Optimizer-1	$\left[PS_{11}^{j}\right]_{iter}$	…	$\left[PS_{18}^{j}\right]_{iter}$
.	.	…	.
.	.	…	.
.	.	…	.
Optimizer-5	$\left[PS_{51}^{j}\right]_{iter}$	…	$\left[PS_{58}^{j}\right]_{iter}$

**where j=1,…,7 and iter∈{50,100}* PS^1*: Acc-score,* PS^2*: Sens-score,* PS^3*: Spec-score,* PS^4*: Prec-score,* PS^5*: F1-score,* PS^6*: Kappa-score,* PS^7*: AUC-score.*

FIGURE 5. Data-preparation for statistical inference.

differences were mostly occurred by NADAM-optimizer. Compared to the performances of AdaGrad, NADAM has the best specification-performances for both 50 ($p-value=0.005$) and 100 ($p-value=0.000$) iterations. Besides, NADAM has better precision- ($p-value=0.000$) and kappa- ($p-value=0.014$) performances than AdaGrad, when iteration-number was 100.

Author Contribution Statements The authors contributed equally to this work.

Declaration of Competing Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

REFERENCES

[1] Velavan, T. P., Meyer, C. G., The COVID-19 epidemic, *TM & IH*, 25 (3) (2020), 278, https://doi.org/10.1111/tmi.13383.

[2] Minaee, S., Kafieh, R., Sonka, M., Yazdani, S., Soufi, G. J., Deep-covid: Predicting covid-19 from chest X-ray images using deep transfer learning, *Med. Image Anal.*, 65 (2020), 101794, https://doi.org/10.1016/j.media.2020.101794.

[3] Amyar, A., Modzelewski, R., Li, H., Ruan, S., Multi-task deep learning based CT imaging analysis for COVID-19 pneumonia: Classification and segmentation, *Comput. Biol. Med.*, 126 (2020), 104037, https://doi.org/10.1016/j.compbiomed.2020.104037.

[4] Ai, T., Yang, Z., Hou, H., Zhan, C., Chen, C., Lv, W., Tao, O., Sun, Z., Xia, L. Correlation of chest CT and RT-PCR testing for coronavirus disease 2019 (COVID-19) in China: a report of 1014 cases, *Radiology*, 296 (2020), E32-E40, https://doi.org/10.1148/radiol.2020200642.

[5] Islam, M. M., Karray, F., Alhajj, R., Zeng, J., A review on deep learning techniques for the diagnosis of novel coronavirus (covid-19), *IEEE Access*, 9 (2021), 30551-30572, https://doi.org/10.1109/ACCESS.2021.3058537.

[6] Xu, X., Jiang, X., Ma, C., Du, P., Li, X., Lv, S., Yu, L., Ni, O., Chen, Y., Su, J., et al., A deep learning system to screen novel coronavirus disease 2019 pneumonia, *Engineering*, 6 (10) (2020), 1122-1129, https://doi.org/10.1016/j.eng.2020.04.010.

[7] Wang, S., Kang, B., Ma, J., Zeng, X., Xiao, M., Guo, J., Cai, M., Yang, J., Li, Y., Meng, X., et al., A deep learning algorithm using CT images to screen for Corona Virus Disease (COVID-19), *Eur. Radiol.*, (2021), 1-9, https://doi.org/10.1007/s00330-021-07715-1.

[8] Kanne, J. P., Chest CT findings in 2019 novel coronavirus (2019-nCoV) infections from Wuhan, China: key points for the radiologist, *Radiological Society of North America*, (2020), https://doi.org/10.1148/radiol.2020200241.

[9] Rubin, G. D., Ryerson, C. J., Haramati, L. B., Sverzellati, N., Kanne, J. P., Raoof, S., Schluger, N. W., Volpi, A., Yim, J. J., Martin, I. B. K., et al., The role of chest imaging in patient management during the COVID-19 pandemic: a multinational consensus statement from the Fleischner Society, *Radiology*, 296 (11) (2020), 172-180, https://doi.org/10.1016/j.chest.2020.04.003.

[10] Kanne, J. P., Little, B. P., Chung, J. H., Elicker, B. M., Ketai, L. H., Essentials for radiologists on COVID-19: an update—radiology scientific expert panel, *Radiological Society of North America*, (2020), https://doi.org/10.1148/radiol.2020200527.

[11] Bhattacharya, S., Maddikunta, P. K. R., Pham, Q. V., Gadekallu, T. R., Chowdhary, C. L., Alazab, M., Piran Md. J., et al., Deep learning and medical image processing for coronavirus (COVID-19) pandemic: A survey, *Sustain. Cities Soc.*, 65 (2021), 102589, https://doi.org/10.1016/j.scs.2020.102589.

[12] Shan, F., Gao, Y., Wang, J., Shi, W., Shi, N., Han, M., Xue, Z., Shen, D., Shi, Y., Lung infection quantification of COVID-19 in CT images with deep learning, *arXiv preprint arXiv:2003.04655*, (2020), https://doi.org/10.1002/mp.14609.

[13] Huang, L., Han, R., Ai, T., Yu, P., Kang, H., Tao, Q., Xia, L., Serial quantitative chest CT assessment of COVID-19: a deep learning approach, *Radiology: Cardiothoracic Imaging*, 2 (2) (2020), e200075, https://doi.org/10.1148/ryct.2020200075.

[14] Wu, X., Hui, H., Niu, M., Li, L., Wang, L., He, B., Yang, X., Li, L., Li, H., Tian, J., and others, Deep learning-based multi-view fusion model for screening 2019 novel coronavirus pneumonia: A multicentre study, *Eur. J. Radiol.*, 128 (2020), 109041, https://doi.org/10.1016/j.ejrad.2020.109041.

[15] Ardakani, A. A., Kanafi, A. R., Acharya, U. R., Khadem, N., Mohammadi, A., 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 (2020), 103795, https://doi.org/10.1016/j.compbiomed.2020.103795.

[16] Apostolopoulos, I. D., Mpesiana, T. A., Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks, *Phys. Eng. Sci. Med.*, 43 (2) (2020), 635-640, https://doi.org/10.1007/s13246-020-00865-4.

[17] Singh, D., Kumar, V., Kaur, M., and others, Classification of COVID-19 patients from chest CT images using multi-objective differential evolution–based convolutional neural networks, *Eur. J. Clin. Microbiol. Infect. Dis.*, 39 (7) (2020), 1379-1389, https://doi.org/10.1007/s10096-020-03901-z.

[18] Khan, A. I., Shah, J. L., Bhat, M. M., CoroNet: A deep neural network for detection and diagnosis of COVID-19 from chest X-ray images, *Comput. Meth. Prog. Bio.*, 196 (2020), 105581, https://doi.org/10.1016/j.cmpb.2020.105581.

[19] Krizhevsky, A., Sutskever, I., Hinton, G. E., Imagenet classification with deep convolutional neural networks, *Adv Neural Inf Process Syst*, 25 (2012), 1097-1105, https://doi.org/10.1145/3065386.

[20] Simonyan, K., Zisserman, A., Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556*, (2014), https://doi.org/10.48550/arXiv.1409.1556.

[21] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., Keutzer, K., SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and $<$ 0.5 MB model size, *arXiv preprint arXiv:1602.07360*, (2016), https://doi.org/10.48550/arXiv.1602.07360.

[22] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., Going deeper with convolutions, *CVPR*, (2015), 1-9, https://doi.org/10.48550/arXiv.1409.4842.

[23] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L. C., Mobilenetv2: Inverted residuals and linear bottlenecks, *CVPR*, (2018), 4510-4520, https://doi.org/10.48550/arXiv.1801.04381.

[24] He, K., Zhang, X., Ren, S., Sun, J., Deep residual learning for image recognition, *CVPR*, (2016), 770-778, https://doi.org/10.48550/arXiv.1512.03385.

[25] Chollet, F., Xception: Deep learning with depthwise separable convolutions, *CVPR*, (2017), 1251-1258, https://doi.org/10.48550/arXiv.1610.02357.

[26] Jaiswal, A., Gianchandani, N., Singh D., Kumar, V., Kaur, M., Classification of the COVID-19 infected patients using DenseNet201 based deep transfer learning, *J. Biomol. Struct. Dyn.*, (2017), 4700-4708, https://doi.org/10.1080/07391102.2020.1788642.

[27] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K. Q., Densely connected convolutional networks, *CVPR*, (2020), 1-8, https://doi.org/10.48550/arXiv.1608.06993.

[28] Wang, X., Deng, X., Fu, Q., Zhou, Q., Feng, J., Ma, H., Liu, W., Zheng, C., A weakly-supervised framework for COVID-19 classification and lesion localization from chest CT, *IEEE Trans. Med. Imaging.*, 39 (8) (2020), 2615-2625, https://doi.org/10.1109/TMI.2020.2995965.

[29] Ronneberger, O., Fischer, P., Brox, T., U-net: Convolutional networks for biomedical image segmentation, *MICCAI*, (2015), 234-241, https://doi.org/10.48550/arXiv.1505.04597.

[30] Kingma, D. P., Ba, J., Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980*, (2014), https://doi.org/10.48550/arXiv.1412.6980.

[31] Chen, J., Wu, L., Zhang, J., Zhang, L., Gong, D., Zhao, Y., Chen, Q., Huang, S., Yang, M., Yang, X., et al., Deep learning-based model for detecting 2019 novel coronavirus pneumonia on high-resolution computed tomography, *Sci. Rep.*, 10 (1) (2020), 1-11, https://doi.org/10.1038/s41598-020-76282-0.

[32] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., Liang, J., Unet++: A nested u-net architecture for medical image segmentation, *DLMIA and ML-CDS*, (2018), 3-11, https://doi.org/10.48550/arXiv.1807.10165.

[33] Song, Y., Zheng, S., Li, L., Zhang, X., Zhang, X., Huang, Z., Chen, J., Wang, R., Zhao, H., Zha, Y., et al.,Deep learning enables accurate diagnosis of novel coronavirus (COVID-19) with CT images, *TCBB*, (2021), https://doi.org/10.1109/TCBB.2021.3065361.

[34] Gozes, O., Frid-Adar, M., Greenspan, H., Browning P. D., Zhang, H., Ji, W., Bernheim, A., Siegel, E., Rapid ai development cycle for the coronavirus (covid-19) pandemic: Initial results for automated detection & patient monitoring using deep learning ct image analysis, *arXiv preprint arXiv:2003.05037*, (2020), https://doi.org/10.48550/arXiv.2003.05037.

[35] He, X., Yang, X., Zhang, S., Zhao, J., Zhang, Y., Xing, E., Xie, P., Sample-efficient deep learning for COVID-19 diagnosis based on CT scans, *medrxiv*, (2020), https://doi.org/10.1101/2020.04.13.20063941.

[36] Soares, E., Angelov, P., Biaso, S., Froes, M. H., Abe, D. K., SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification, *medRxiv*, (2020) https://doi.org/10.1101/2020.04.24.20078584.

[37] Duchi, J., Hazan, E., Singer, Y., Adaptive subgradient methods for online learning and stochastic optimization, *JMLR*, 12 (7) (2011).
[38] Dozat, T., Incorporating nesterov momentum into adam, *ICLR*, (2016), 1-4.
[39] Rude, S., An overview of gradient descent optimization algorithms, *arXiv preprint arXiv:1609.04747*, (2016), https://doi.org/10.48550/arXiv.1609.04747.
[40] Nair, V., Hinton, G. E., Rectified linear units improve restricted boltzmann machines, *ICML*, (2010), 807-814.
[41] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D., Grad-cam: Visual explanations from deep networks via gradient-based localization, *Proc. IEEE Int. Conf. Comput. Vis.*, (2017), 618-626 https://doi.org/10.48550/arXiv.1610.02391.