Randomly initialized convolutional neural network for the recognition of COVID-19 using X-ray images

Safa Ben Atitallah1 | Maha Driss1,2 | Wadii Boulila1,3 | Henda Ben Ghézala1

1RIADI Laboratory, University of Manouba, Tunisia
2Security Engineering Lab, Prince Sultan University, Saudi Arabia
3Robotics and Internet-of-Things Lab, Prince Sultan University, Saudi Arabia

Correspondence
Wadii Boulila, RIADI Laboratory, University of Manouba, Tunisia.
Email: wadii.boulila@riadi.rnu.tn

Abstract
By the start of 2020, the novel coronavirus (COVID-19) had been declared a worldwide pandemic, and because of its infectiousness and severity, several strands of research have focused on combatting its ongoing spread. One potential solution to detecting COVID-19 rapidly and effectively is by analyzing chest X-ray images using Deep Learning (DL) models. Convolutional Neural Networks (CNNs) have been presented as particularly efficient techniques for early diagnosis, but most still include limitations. In this study, we propose a novel randomly initialized CNN (RND-CNN) architecture for the recognition of COVID-19. This network consists of a set of differently-sized hidden layers all created from scratch. The performance of this RND-CNN is evaluated using two public datasets: the COVIDx and the enhanced COVID-19 datasets. Each of these datasets consists of medical images (X-rays) in one of three different classes: chests with COVID-19, with pneumonia, or in a normal state. The proposed RND-CNN model yields encouraging results for its accuracy in detecting COVID-19 results, achieving 94% accuracy for the COVIDx dataset and 99% accuracy on the enhanced COVID-19 dataset.

KEYWORDS
COVID-19, deep learning, random initialized CNN, recognition

1 | INTRODUCTION

The novel coronavirus (COVID-19) appeared in Wuhan, China at the end of 2019, and by March 11, 2020, the World Health Organization (WHO)* had categorized this virus as a pandemic. Since then, COVID-19 has spread rapidly across the world, and to date has resulted in almost three million deaths. As demonstrated in Figure 1, the number of confirmed cases increases day by day and had reached about 725 275 cases worldwide on May 13, 2021. In the United States alone, the number of deaths caused by COVID-19 surpassed 575 000 cases by May 4, 2021. Figure 2 illustrates the total number of COVID-19 deaths in the most impacted countries worldwide.

Patients infected with COVID-19 exhibit some symptoms in common to those infected with the general flu. These include fever, cough, headaches, and loss of taste or smell. However, COVID-19 often also comes with more serious symptoms, including difficulty or shortness of breath, pain in the chest, and the inability to speak or move normally. Although COVID-19 causes only mild illness for most, it can be fatal for others, especially for older people and those with preexisting medical conditions, such as heart problems and high blood pressure. About 15% of COVID-19 cases progress to grave diseases, and 5% become critical cases. Between the infectiousness of this disease and its rapid transmission, the early diagnosis of COVID-positive patients is crucial for avoiding further spread and minimizing the cases of critical illness.
To this end, many types of interdisciplinary research have focused on finding ways of combatting COVID-19, often through its prevention or treatment. In addition, although, detecting those who have been infected in a quick and automatic way could be an effective solution, particularly in the process of isolating and treating patients as a means of preventing further spread. In this context, there has been an increased interest in developing computer-aided diagnosis systems using artificial intelligence (AI) techniques. For instance, well-trained deep learning (DL) models can focus on specific details that are not noticeable to the human eye and can also utilize existing forms of medical data. More specifically, X-ray images constitute a promising avenue of exploration, and they also already play a vital role in the early diagnosis and treatment of COVID-19. Likewise, these particular medical images tend to be readily accessible for disease diagnosis and are already widely used in health centers worldwide, making them an even more desirable avenue for DL models. The use of X-rays can also help circumvent existing difficulties. For instance, due to the limited number of radiologists present in many hospitals, fast and simple AI models can present an effective solution for early diagnosis of COVID-19 by eliminating multiple problems already observed with RT-PCR test kits, such as the cost and wait times associated with getting back test results.

This is not an entirely novel idea, either. Innovations in computer-aided vision, including DL techniques, have made many improvements in the larger domain of healthcare. In particular, CNNs have demonstrated high performance in medical imaging, offering high accuracy of diagnosis through their ability to extract multiple levels of features from medical images. To date, CNNs have been employed in the recognition of several different diseases and medical conditions, such as screening mammographic images for signs of cancer and other issues.

Likewise, radiology images are already being used to detect COVID-19 in rapid and automatic ways. To date,
several studies have been conducted on COVID-19 recognition from chest X-ray images that employed different architectures of CNN with transfer learning. For instance, Wang et al. have proposed a COVID-Net that can detect relevant abnormalities in chest X-ray images from a sample population of patients with COVID-19, those with pneumonia infections, and those with no illnesses or health conditions. Their COVID-Net achieved an overall accuracy of 92.4% using the COVIDx dataset, but it was pre-trained with the ImageNet dataset and initialized with the obtained weights. These researchers are also the ones who created and curated the COVIDx dataset used both in the current study and many others that have worked on designing DL models for the automatic detection of COVID-19. For example, Karim et al. designed a deep neural network (DNN) called DeepCovidExplainer for the detection of COVID-19 using chest X-ray images. These researchers used transfer learning based on a combination of DenseNet, ResNet, and VGG architectures to create their model snapshots. The results they achieved demonstrated that their approach could automatically detect COVID-19 positive patients with 94.6% accuracy of precision, 94.3% accuracy of recall, and 94.6% accuracy of F1-score. Elsewhere, Luz et al. also designed a low computational model for the automatic detection of COVID-19 patients using chest X-ray images. These researchers used the EfficientNet family of DNN, in addition to the hierarchical classifier. To test and evaluate the results of this model, the COVIDx dataset version 1 and its 13,569 X-ray images were used. The experimental results that Luz et al. achieved demonstrate that this model also performed effectively, accomplishing 93.9% accuracy and 90% sensitivity.

As these examples demonstrate, though, previous work regarding COVID-19 detection using DL models has focused mainly on pre-trained networks and transfer learning. Few researchers have addressed the issue of designing a CNN from scratch and tuning the right parameters to improve its performance. One of the few exceptions to this is Irmak, who proposed two novel CNN architectures for COVID-19 detection whose parameters are automatically tuned by the Grid Search algorithm. The first CNN is developed for binary classification: that is, to determine whether or not a patient is infected with COVID-19 using the chest X-ray images. The second CNN here then classifies the chest X-ray images into one of three classes, which include “COVID-19,” “pneumonia,” and “normal” images. Although these models’ performance is interesting, Irmak fails to account for the data augmentation techniques that may help to improve overall performance. Likewise, in Ref. [16] researchers proposed a new CNN-based approach for the classification of COVID-19 severity. This CNN divides and categorizes COVID-19 patients into one of four severity groups: mild, moderate, severe, and critical. This study employs also grid search optimization to select the CNN parameters. This experimental results demonstrate the effectiveness of the proposed CNN model, which achieves an accuracy of 95.5%.

In this study, we propose a randomly initialized convolutional neural network (RND-CNN) as a means of classifying chest X-ray images and identifying the patient’s condition as one of three classes: pneumonia, COVID-19, or normal state. Randomized Neural Networks (RNN) are defined as neural networks with multi-layered architecture, where the connections between these layers are untrained before the initialization. Recent works show that RNNs tend to demonstrate acceptable performance for feature extraction and classification correlated with the results of pre-trained models.

The following objectives outline our specific goals in this work:

1. Propose a classification CNN model with randomly initialized connections between layers, for the automatic recognition of COVID-19;
2. Study the impact of data preprocessing and balancing as means of enhancing the performance of this proposed model;
3. Apply different techniques of data augmentation on the dataset samples, such as rotation, flips, and scaling;
4. Testing the proposed method using different datasets, including the COVIDx dataset (15,000+ chest X-ray images) and the enhanced COVID-19 dataset (1000+ enhanced images);
5. Compare the performance of the proposed model with other models that use different weight initialization techniques as well as with methods proposed in previous works.

To achieve these objectives, this article is structured as follows. Section 2 presents an overview of the basic concepts of CNN and initialization methods, while Section 3 provides details about the architecture of our proposed RND-CNN model. In Section 4, we describe the datasets utilized and how we have prepared the data from each one to be used in training our own model. Section 5 discusses the experiments carried out and the results obtained. Section 6 features comparisons made between our proposed model and other existing models using different weight initialization techniques. Finally, Section 7 offers our concluding remarks and thoughts on directions for future work.
2 | BACKGROUND AND BASIC CONCEPTS

In this section, we present foundational knowledge related to CNNs and weight initialization methods.

2.1 Architecture of convolutional neural network

A wide range of healthcare services have been improved by the rapid development of DL techniques. Different applications have been proposed and developed for various healthcare services, including elder care and disease prediction. CNN, for instance, has been presented as a deep neural network (DNN) that consists of a sequence of layers, wherein different filter operations are performed. This type of DL model is suitable for processing images and/or videos. In addition, because CNN employs supervised learning, it can be classified as a discriminative DL architecture.

This network utilizes a set of layers that begins with an input layer, then includes a set of hidden layers, and finally ends with an output layer. It also consists of a sequence of convolutional and pooling layers, followed by a fully connected layer. In each of these convolutional layers, several filters are embedded in order to produce different analyses on the input data, which in turn creates feature maps.

Thus, three essential layers build up the CNN network: the convolutional layer, the pooling layer, and the fully connected layer. A brief definition of these layers is as follows:

- **Convolutional layer**: The CNN model begins with this layer. Here, a filter is applied over the input image so that it is converted into a matrix. A feature map is the output of this convolutional layer, including the learned features. Briefly, a filter is presented as a simple matrix with a predefined size smaller than the input data as well as randomly chosen values. This filter goes through the input data from left to right, and then goes down with a step size specified by the stride until it covers the completely input matrix. Equation (1) represents the convolutional layer process:

\[
y_j = f \left( \sum_i K_{ij} \times x_i + b_j \right)
\]

where, \(x\) is the input image, \(y_j\) is the jth convolutional layer output, \(f\) is the activation function, \(K_{ij}\) presents the convolutional kernel multiplied by the ith input \(x_i\), and \(b_j\) denotes the bias. We illustrate convolutional operations in Figure 3.

- **Pooling layer**: This layer is applied to lower the complexity of the overall CNN model. The pooling layer captures the most relevant parts of the feature maps generated by applying an average or max-pooling operation. In turn, the pooling operation applies a kernel, or a predefined window, to the feature map. This kernel is responsible for gathering the average or maximum value of the matrix elements according to the method being used, so that it slides across the whole feature map with a predefined stride. Figure 4 depicts the pooling layer as it uses the max-pooling operation. In the illustrated example, four slide positions are performed on the feature map, as presented in Figure 4 with these four different colors. The resultant pooling values demonstrate how the complexity of the model computations will be reduced.

- **Fully connected layer**: This layer performs the last step in this network, reconnecting the processed portions in order to obtain the full image. The two-dimensional array is converted into a single list. Then, using

![FIGURE 3 Illustration of the operations completed within the convolutional layer](image-url)
different activation functions (Sigmoid, Tanh, and ReLu), this layer is converted to probability values, which indicate the probability that the processed image will be in a specific class. Based on the highest probability value, the output layer assigns the image to its output class.

Generally, the convolutional and pooling layers are stacked together at the head of the architecture. The fully connected layers, however, are stacked with one another at the end of the network architecture.

### 2.2 How do CNNs work?

CNNs have demonstrated effective performance in many different areas, including image classification and recognition. One of the greatest advantages of CNNs is their ability to extract and learn hidden features from big datasets and raw data. To achieve this, CNN proceeds through a set of steps, which can be explained as follows.26,27

Once the training phase starts, the input layer assigns the weights to the input data to be passed to the next layer. According to the type of initializer used, the weights are determined as having either constant values or random ones. The following layers get the input weights, perform the filter operations, and determine the output that is passed on to the next layer as an input. Then in the last layer, the final output is defined.

Within the training process, a loss function is used to examine the performance of this prediction. This function calculates the error rate by examining and comparing both predicted and actual results. Various loss functions are also designed for different purposes. For example, the binary cross-entropy function is used to deal with classification problems that distinguish between only two classes, while categorical cross-entropy is used for classification problems with more than two classes and mean-squared error is utilized for regression problems. Different optimization algorithms, such as Adam and Stochastic Gradient Descent, can be utilized in order to check the neuron’s weights. These algorithms examine the gradient of the loss function and then attempt to change and update the network weights or learning rates to minimize these losses. This set of steps is repeated throughout the training phase until the weights become balanced for each layer’s neuron and the error rate value falls. Then once this training has finished, the model is ready for use.

Tuning the right parameters of a CNN model will help improve its performance. These parameters include the internal values of the model configuration, which are estimated from data such as the weights between neural network layers.

### 2.3 Weight initialization

In order to understand the importance of weight initialization, it is first crucial to understand the neurons; that is, the units that make up each layer of the CNN. These neurons take the input data and perform calculations upon it in order to achieve a weighted summation, and then produce an output through an activation function.28 Every neuron consists of weights and a bias. In the first layer, the weights are initialized and assigned according to input size, while the bias is optimized throughout the training process. The structure of a neuron is depicted in Figure 5.

Weight initializers are meant to regulate the initial weight values of various neural network layers.29 The process of weight initialization is intended to keep the layer activation outputs from falling into the common problem of gradients vanishing and exploding. In particular, the vanishing gradients occur due to back-propagation during the training phase.30 Propagating a feedback signal from the output loss to the earlier layers may affect it to the point of signals becoming weak or getting lost, which in turn makes the network untrainable. Therefore, a careful weight setting is required in order to achieve better results and higher performance.

There are three categories of initialization methods.31 The first category includes constant methods, which employ the same set of weights for the network
connections’ initialization, such as the zero and one initializers. However, when using these initialization methods, the equations of the learning algorithm often become incapable of changing or updating the network weights, which often leads to the model becoming locked. For all iterations, then, all layers should have the same weights and perform the same calculations. The second category of initialization methods presents the distribution options, wherein either the Gaussian or the uniform distribution is used and input matrix numbers are assigned with random values. However, assigning the appropriate parameters for the network—including both the mean and the standard deviation of the distribution—may be done incorrectly, which will affect the performance of the model during training and may also lead to the problem of vanishing gradients. Finally, the third category includes the random initialization based on previous knowledge. To initialize layer weights here, heuristics are used in addition to the nonlinear activation functions. “Heuristics” of course describes the approach of solving a problem without using a method that ensures an optimal solution. Using this type of randomization, normal distribution variance is assigned based on the number of inputs. Heuristics largely mitigate the issue of exploding or vanishing gradients, but they cannot prevent this issue entirely. Table 1 compares the different types of neural networks initialization.

Randomized initialization methods with previous knowledge serve as a good starting point for weight initialization. The main advantages of this means include its abilities in the following areas:

- initializing the layer weights randomly but more intelligently;
- reducing the chances of gradients vanishing and/or exploding;
- helping avoid slow speeds of convergence;
- mitigating the oscillating of minima.

3 | PROPOSED METHOD

We now turn to detailing the architectural design of our proposed RND-CNN, as well as the techniques used for data preprocessing. Our motivation here is to develop a deep CNN capable of automatically learning the features of COVID-19 and recognizing instances of it across two different datasets.

The proposed RND-CNN consists of an input layer and four hidden blocks for feature learning and extraction, followed by two fully connected layers and a SoftMax layer for case classification (classes: COVID-19/ Pneumonia/Normal). Figure 6 presents the proposed RND-CNN architecture.

### Table 1: Comparison among the different types of initialization

| Initializer type                  | Examples            | Neuron computations                  | Learned features                  | Performance                              |
|----------------------------------|---------------------|-------------------------------------|----------------------------------|------------------------------------------|
| Constant methods                 | Zero initializer    | Same calculations result             | Same learned features             | Low performance                          |
|                                   | One initializer     |                                     |                                  | Bad accuracy                             |
|                                   |                     |                                     |                                  | Symmetry hidden layers                   |
| Distribution methods             | Random normal initializer | Set weights randomly that may be low or high | Different features are learned    | Prone to overfitting                     |
|                                   | Random uniform initializer |                                     |                                  | Higher or lower initialization value leads to slower optimization |
| Randomly with based knowledge methods | Xavier initializer | Use heuristics                      | Various features are learned and extracted | Break the symmetry                      |
|                                   | He initializer      | Calculate weights smartly with randomization |                                  | Prevent the signal from exploding to a high value or vanishing to zero |
3.1 Random initialization: Xavier initializer

The Xavier initializer is a form of randomly based knowledge initialization. To solve the problem of selecting correct parameters, this initializer is used to automatically determine the scale of initialization according to the number of input and output neurons.32 This method helps keep the signal within a reasonable range of values between layers.

Let $y$ denote the output of a layer; then, its value is computed according to Equation 2:

$$ y = w_1 i_1 + w_2 i_2 + \ldots + w_j i_j + b $$

where, $i$ presents the input image matrix, $w$ is the weight, and $b$ is a bias.

Using the Xavier initializer, the weights are initialized in such a way that the variance of the input and output remains the same. The values of these weights are calculated using Equation 3.

$$ \text{Var}[W] = \frac{2}{n_i + n_{i+1}} $$

where, $n$ signifies the number of layer neurons.

Thus, the network weights are initialized in such a way that the neuron activation functions are neither
too small nor too large within a reasonable range. The values of the weights that connect two successive layers are usually initialized within the following range:

\[
\left[ -\frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}, \frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}} \right]
\]

\[\text{(4)}\]

### 3.2 | Network architecture

In this section, we now introduce the design of the proposed RND-CNN. As previously mentioned, this network consists of an input layer, four hidden blocks, and a classification layer. The input images of this network are sized as (150, 150, 3), while the output can be one of three different classes: COVID-19, pneumonia, or normal.

This network model consists of four different-sized hidden blocks, each of which consists of a set of convolutional and pooling layers. As we go deeper within the model architecture, the number of convolutional layers also increases. Consequently, this difference in layer blocks gives the model considerable ability to cover different features through its set of convolutions. The model also comes with a significant reduction in the number of parameters and is capable of achieving high performance within a reasonable execution time, particularly as compared with other existing CNNs.

Before the model training, we had to assign its initial parameters. We adopted the Xavier initializer to define the weights of the network. This method is proposed by Glorot and is based on the assumption that the activation function is linear.

The model proposed here consists of 10 convolutional layers and 4 pooling layers. For each convolutional layer, filters of \((3 \times 3)\) size are applied with padding, and every pooling layer implements a max-pooling window of \((2 \times 2)\) size. In the following, “Conv2D,” “MaxPool2D,” and “FC” refer to the convolution, the pooling, and the fully connected layers, respectively. Table 2 illustrates the architecture of our proposed CNN and defines the learning parameters used.

### 4 | DATASETS

In order to validate the approach, we propose here, we used two different datasets to evaluate the performance of this model. A description of these datasets and how we preprocessed them is provided in the following subsections.

### 4.1 | COVIDx dataset

In this work, we used the COVIDx dataset recently created and published by COVID-Net researchers.\(^\text{11}\) COVIDx is an open-access benchmark dataset that is continuously updated and enriched with the addition of more images from different sources.\(^\text{33}\) The version of the

| Layer     | Description | Values                  |
|-----------|-------------|-------------------------|
| Input layer | Images input layer | Input shape = \((150, 150, 3)\) |
| Hidden block 1 | 1st Conv2D | 16 feature maps |
|           |           | Kernel size = \((3,3)\) |
|           |           | Kernel initialization = Xavier initializer |
|           | MaxPool2D  | Pool size = \((2,2)\) |
|           |           | Strides = \((2,2)\) |
| Hidden block 2 | 2nd Conv2D | 32 feature maps |
|           |           | Kernel size = \((3,3)\) |
|           |           | Kernel initialization = Xavier initializer |
|           | MaxPool2D  | Pool size = \((2,2)\) |
|           |           | Strides = \((2,2)\) |
| Hidden block 3 | 4th Conv2D | 64 feature maps |
|           |           | Kernel size = \((3,3)\) |
|           |           | Kernel initialization = Xavier initializer |
|           | MaxPool2D  | Pool size = \((2,2)\) |
|           |           | Strides = \((2,2)\) |
| Hidden block 4 | 7th Conv2D | 128 feature maps |
|           |           | Kernel size = \((3,3)\) |
|           |           | Kernel initialization = Xavier initializer |
|           | MaxPool2D  | Pool size = \((2,2)\) |
|           |           | Strides = \((2,2)\) |
| Classification layer | 2 FC layers | 1st layer units = 10,368 |
|           |           | 2nd layer units = 4,096 |
|           | SoftMax    | 3 classes |

\[\text{TABLE 2} \text{ The proposed randomly initialized CNN and its learning parameters}\]
dataset that we used consists of more than 15,000 chest X-ray images, created as both a combination and a modification of five open access data repositories. It includes three different types of images, namely: (a) Normal (i.e., no infection or condition), (b) pneumonia, and (c) COVID-19. The dataset also consists of two image folders: one for training and one for testing. The distribution of these images is depicted in Table 3.

Figure 7 presents some samples of patient chest X-ray images from each of the three classes: COVID-19, normal, and pneumonia, respectively.

The COVIDx dataset takes its data from five different repositories. As a result, the images in this dataset are of all different sizes and shapes. These differences affect the effectiveness of any classification attempted, so in order to enhance the performance of our classification approach, image preprocessing is first applied to all images across the dataset. For starters, all input images for the proposed method are resized to a standard size, which is defined by height: 150 and width: 150.

It is also worth noting that the COVIDx dataset is imbalanced. In other words, the number of COVID-19 X-ray images is much fewer than the images available for the other two classes (pneumonia and normal). However, it is important to work with balanced data in order to achieve better results.\(^\text{34}\) Imbalanced datasets return inaccurate results because they bias the model toward the predictions of the majority class. Different techniques have been proposed to handle this problem, including random oversampling, random undersampling, Synthetic Minority Oversampling Technique (SMOTE), and reweighting of classes.\(^\text{35}\) Each method has its own benefits and disadvantages. For instance, undersampling consists of randomly sampling from the majority class and reducing its number of instances to be equilibrated with the other classes, while oversampling treats the minority class by replicating its samples to be balanced with other classes. SMOTE is a type of oversampling that generates new instances from the samples of the minority class. However, SMOTE is not effective for high-dimensional data and can actually lead to the model being overfitted. Here, the class reweight method is used in order to consider the asymmetry of cost errors within the model training directly.

Due to the severe imbalance between the three different classes in this COVIDx dataset, resampling methods are not suitable for our problem. As previously stated, our goal is to detect COVID-19 from X-ray images, but also as mentioned above, we did not have as many samples of those images to work with. Because of the high dimensionality of the used dataset, we choose to apply the class reweight approach as a balancing method, which penalized the model if a positive sample was misclassified.\(^\text{36}\)

To achieve this, we calculated the weight for each class and assigned these weights to the classifier model. The heaviest weight is applied for the COVID-19 class, which allows the model to pay more attention to the COVID-19 samples:

- Weight for COVID-19 class: 9.57
- Weight for Normal class: 0.57
- Weight for Pneumonia class: 0.83

| Classes      | # of images in the training dataset | # of images in the testing dataset |
|--------------|-------------------------------------|-----------------------------------|
| Normal       | 7966                                | 885                               |
| Pneumonia    | 5459                                | 594                               |
| COVID-19     | 471                                 | 100                               |
| Total        | 13,896                              | 1,579                             |

**Figure 7** Samples of chest X-ray images from the COVIDx dataset and illustrating different classes.
The weight of each class is computed using Equation (5):

\[ W_i = \frac{n}{k n_i} \]  

where, \( W_i \) represents the weight for the \( i \)th class, \( n \) is the total number of samples, \( k \) is the number of classes, and \( n_i \) is the number of samples in class \( i \).

### 4.2 Enhanced COVID-19 dataset

The second dataset used to test our model was merged and enhanced by Canayas. The enhanced COVID-19 dataset consists of more than 1000 images and includes three balanced classes: COVID-19, pneumonia, and normal chest X-ray images. The images of the dataset are gathered from two different sources: the first part contains 145 images of labeled COVID-19 X-ray images available on GitHub, while the second is collected by Chowdhury et al., publicly available on Kaggle, and contains 219 X-ray images of chests with COVID-19 infections. The distribution of chest X-ray images in the enhanced COVID-19 dataset is depicted in Table 4.

In Ref. [37] the author made changes to this dataset by applying a contrast enhancement on each image from the original dataset. Using the Image Contrast Enhancement Algorithm (ICEA), the best contrast was applied on the dataset images and the noise was eliminated. Figure 8 plots select samples of X-ray images from this enhanced dataset.

### 4.3 Data augmentation

While CNNs offer several benefits and can make great strides in solving important problems, these networks rely heavily on big data in order to learn properly. Unfortunately, many different use cases, especially in healthcare, do not have the types of big data needed for these purposes.

Data augmentation is the approach of modifying some of the data provided to the learning model and thus creating more training data in order to avoid overfitting. Overfitting occurs when a model learns a function with high variance, which means that it models the training data well but does not perform properly with new data, leading to poor generalization. With data augmentation, more images are generated through different random transformations applied to the existing dataset images. As the size of input data increases, this helps to improve the training model’s generalization abilities.

In this work, we choose to apply six augmentation strategies for data augmentation and transformations, including scaling, horizontal flip, random rotation (10 degrees), zoom, intensity shift, and lighting conditions. The images of the datasets are only flipped horizontally; we do not apply vertical flips since these do not reflect the images in their normal form. Thus, data augmentation was employed to enlarge the training dataset, while valid and test data were not augmented. Figure 9 illustrates some samples of X-ray images resulting from the data augmentation process.

### 5 EXPERIMENTS

In this section, we present the experimental setup, workflow, and parameters. Then, we describe the metrics used for the evaluation of the model’s performance. After that, we discuss the results obtained and examine the impact of data augmentation and balancing on the model’s performance.

#### 5.1 Experimental setup

In this study, the Jupyter notebook was used to encode the whole process in Python 3.8. Both the Keras library and the TensorFlow backend were also used to program neural networks. Keras is a high-level library that works atop the TensorFlow and Theano frameworks and is suitable for convolutional networks as it delivers high performance when conducting multiple experiments. TensorFlow is a flexible DL framework developed using C++, and it helps to run experiments with low latency and high performance. In addition to these Keras and TensorFlow, OpenCV was used for data loading and preprocessing, while Sci-Kit Learn was used to generate the classification reports.

For faster computation, we also used the Nvidia GeForce MX 250 GPU with CUDA and cuDNN library. cuDNN is a GPU-accelerated library designed to optimize different DL frameworks. The infrastructure used to conduct our experiments was a PC with the following

| Classes   | # of images for training | # of images for testing |
|-----------|--------------------------|-------------------------|
| Normal    | 314                      | 50                      |
| Pneumonia | 314                      | 50                      |
| COVID-19  | 314                      | 50                      |
| Total     | 942                      | 150                     |
5.2 Experimentation workflow overview

The experimentation workflow consisted of a set of steps that included: (a) data preprocessing, (b) data augmentation, (c) model training, (d) model evaluation using the validation data, (e) final evaluation of the model with the best weights using the test data, and (f) calculation of the performance metrics. These steps are detailed and illustrated in Figure 10.

5.3 Parameters tuning

The model configuration was set up as follows:

1. **Initializer**: before starting the training process, we initialize the layers with proper weights in order to ensure accurate functions. In this context, we have used the Xavier initializer with a learning rate value of \((1e^{-3})\). This initializer is capable of determining
the scale of initialization randomly according to the input and output nodes number.

2. **Optimizer**: the Adam optimizer,\(^45\) which was proposed recently as an extension to the stochastic gradient descent,\(^46\) is used for its ability to achieve high performance in a short time.

3. **Loss function**: categorical cross-entropy function was used to measure the network’s performance on the training data.

4. **Activator**: rectified linear unit (ReLu)\(^47\) has been used as an activation function. ReLu is known as a faster training function than other options, such as Sigmoid, Tanh, and so on.

### 5.4 Performance metrics

Precision, accuracy, sensitivity, specificity, loss, and F1-score measures were used to evaluate the results achieved by the model in this work. These terms are defined as follows:

- **Precision**: is used to assess the number of classes classified accurately or truly by the model, and this describes the model's ability to not classify a negative sample as positive. Therefore, a high precision indicates that the errors in classification are low.

- **Accuracy**: is the ratio of the number of correct predictions to the total number of input samples. High accuracy requires high precision.

- **Sensitivity**: is used to assess the overall number of correctly predicted labels according to the total number of labels predicted. This factor describes the model's ability to classify samples correctly.

- **Specificity**: is used to measure the model's ability in the recognition of negative samples.

- **Loss**: is used to calculate the error value and determine how well the model treats the data. A lower value for loss indicates that the model is making fewer errors.

- **F1-score**: is computed using precision and recall in order to achieve a balanced average result.

These measures are computed according to the following Equations 6 to 10:

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{6}
\]

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}
\]

\[
\text{Sensitivity/recall} = \frac{TP}{TP + FN} \tag{8}
\]

\[
\text{Specificity} = \frac{TN}{TN + FP} \tag{9}
\]

\[
\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}
\]

where, True Positive (TP): the predicted case is positive (pneumonia/COVID-19) and the result is true; True Negative (TN): the predicted case is negative (normal) and the result is true; False Positive (FP): the predicted case is positive (pneumonia/COVID-19) and the result is false; and False Negative (FN): the predicted case is negative (normal) and the result is false.

### 6 RESULTS

In this section, we present the experimental results obtained by applying our proposed RND-CNN on the COVIDx and enhanced COVID-19 datasets. We also demonstrate the effectiveness of this proposed model by showing the impact of data augmentation and balancing through several performance measures.
6.1 | Experiments using COVIDx dataset

In this subsection, we illustrate the details and results obtained by implementing the proposed approach described in Section 3 on the COVIDx dataset. For model building, we need to begin with three sets of data: training, validation, and testing. The training data are used to train the model, while the validation data are utilized for evaluating it during the training process. Once the model completes training, then the test data are employed to test its performance.

Following this distribution, we randomly split the training folder of the COVIDx dataset into 80% for training and 20% for validation. Figure 11 represents the distribution of images of each class into our training, validation, and test sets.

The proposed RND-CNN was trained using the COVIDx dataset over 100 epochs. Figure 12 visualizes the accuracy and loss achieved by this RND-CNN during the training and validation phases. During the training, the model achieved 95% accuracy, while the value of loss continued to decrease until it had reached its minimum by the end of the training phase.

After that, the overall performance of our proposed RND-CNN was tested using over 1500 new chest X-ray images. According to Table 5, our proposed model achieved an accuracy of 95% in training, 92% in validation, and 94% in testing.

Figure 13 shows that we obtained a high area under the ROC curve for the COVID-19, pneumonia, and normal classes alike, which demonstrates that our approach achieved high performance.

Examples of features extracted from chest X-ray images across the first, second, and last convolution layers are presented in Figure 14. Additional interesting observations are discovered in feature maps as we dive into layers. In the first convolutional layer, the edges of each image are detected and most of its information is scanned. Going deeper into the CNN, the filters focus further on specific features.

6.2 | Experiments using the enhanced COVID-19 dataset

In order to examine the effectiveness of our proposed RND-CNN, we implemented it using another dataset collected and enhanced for the purpose of COVID-19 detection. Unlike the COVIDx dataset, this enhanced COVID-19 dataset is balanced and consists of the same number of images in each class with a contrast enhancement. Figure 15 illustrates the distribution of the

![Figure 11](image1.png)

**FIGURE 11** Distribution of COVIDx's images of each class into training, validation, and test sets

![Figure 12](image2.png)

**FIGURE 12** Accuracy and loss achieved during the training and validation phases of the RND-CNN model using the COVIDx dataset

| Proposed model | Dataset | Training accuracy | Validation accuracy | Testing accuracy |
|----------------|---------|-------------------|---------------------|-----------------|
| RND-CNN        | COVIDx  | 95%               | 92%                 | 94%             |
enhanced dataset images of each class into training, validation, and test sets.

Our RND-CNN was trained using the enhanced COVID-19 dataset over 100 epochs. Figure 16 visualizes the accuracy and loss achieved by the RND-CNN during the training and validation phases. During the training, the model achieved an accuracy of 98% and a very small value of loss of only 0.0822.

Using the test data, we evaluated the overall performance of the newly-trained model. According to Table 6, our proposed model achieved an accuracy of 99% in training, 98% in validation, and 99% in testing. Figure 17 plots the ROC curves for COVID-19, pneumonia, and normal classes. High results of AUC are achieved, with 99% for COVID-19, 100% for normal, and 98% for pneumonia.

6.3 Impact of data augmentation

Our results reveal that using different data augmentation techniques on dataset images has a significant impact on the efficiency of the model, especially for imbalanced datasets. To examine the results obtained, we trained the same proposed CNN architecture without augmentation for the dataset images, and when we did, the results showed a significant drop in accuracy. The results
6.4 Impact of data balancing

To deal with the imbalanced distribution of data across the COVIDx dataset, we employed the class reweight method to re-balance the whole dataset. We then checked the model’s performance both before and after making this change. Balancing the dataset was particularly important in order to ensure better results for COVID-19 recognition. As depicted in Table 8, the accuracy of the model when using the data balancing is higher than its accuracy without data balancing. It is also noticeable that the performance achieved with the second dataset (i.e., the enhanced COVID-19 dataset) was much better than that achieved with the first dataset (i.e., the COVIDx dataset) due to its balance in the number of images in each class.

From the results obtained, we conclude that correcting the dataset’s imbalance is a very important step to consider before starting the model’s training.

7 COMPARISONS AND DISCUSSION

In order to validate the performance of our proposed RND-CNN model, we compared the results we obtained with the COVIDx dataset with the results obtained by employing other models and different types of weight initialization.

7.1 Comparison with other deep neural networks

As explained previously, a zero initializer will result in poor model performance. Both to prove this fact and for the sake of comparison, we trained the same CNN
architecture using the zero initializer for layers’ weights. We refer to this architecture as CNN-0. The results of performance metrics were with the same values in each training epoch, ending with an accuracy of 56%. Typically, with the backpropagation algorithm, the weights of layers are updated in each iteration. When the initial weight value of the first layer is 0, the operation of multiplying it by any value in the backpropagation delta does not change the weight. Therefore, the weights of each layer are still with the same value for each iteration without being optimized. All neurons in every layer network perform the same calculation, giving the same output. Thus, we also employed the random uniform initializer to train the proposed CNN architecture, creating an architecture we refer to as RU-CNN. We achieved 90% accuracy using the RU-CNN network, which was acceptable but still lower than the accuracy of the RND-CNN we are proposing. Accordingly, we can conclude that the choice of the Xavier initializer will help provide better results.

Besides changing the network’s weight initialization, we also examine the results of two other DL models that use transfer learning. Transfer learning is the approach of learning based on previous knowledge and then transferring that knowledge thus gained to address new problems. The networks developed using this approach are based on VGG16 and Xception pre-trained models. VGG16 is a deep CNN proposed by the Visual Geometry Group at Oxford University. VGG16 consists of 16 layers and as a network, has demonstrated strong generalization on many large benchmarking datasets for different tasks. Meanwhile, Xception is a CNN composed of 71 layers.

We loaded pre-trained versions of these two models, which had been performed on the ImageNet dataset that includes more than one million images. For sake of comparison, the same parameter values were used across the developed models: optimizer: Adam, learning rate: $10^{-4}$, and loss function: categorical-cross-entropy.

Table 9 shows that our proposed model outperforms all the other DL models in terms of accuracy, precision, sensitivity, specificity, and F1-score. It also achieves the minimum loss rate we have seen, as compared with the other DL models considered.

These results indicate that the architecture of a DL network and the choice of its parameters will have a direct impact on its performance. In addition, the choice of the right method of initialization will help obtain better results for the tasks of classification and recognition. As we can see with the results illustrated in Table 9, using the randomized method of initialization provides better results for all performance metrics as compared with both the constant and distributed methods. In addition, compared with the high-performing VGG16 and Xception models, the obtained results demonstrate the excellent effectiveness of our proposed RND-CNN. These indicate that the proposed architecture with its different sets of layers could extract several features, although with random weights.

### 7.2 Comparison with existing works

Significant research has been conducted seeking ways of combatting the novel coronavirus, or COVID-19. In the field of machine learning and deep learning, though, most of this work thus far has been based on transfer learning approaches. In our work, though, instead of using trained weights we created a DL model from scratch for the detection of COVID-19 cases using chest X-ray images. With the proposed model, we obtained excellent results that demonstrate its effectiveness.
recognition of COVID-19. This comparison demonstrates that our proposed approach produces excellent results for both the COVIDx and enhanced COVID-19 datasets. It is also worth noting that we have obtained similar results as the models presented in Ref. [37] but the later was only applied on one small dataset, which is the enhanced COVID-19 dataset. Also Ref. [13], provides a higher F1-score when using COVIDx dataset compared with our model, but it needs to conduct more exhaustive experiments to measure additional performance metrics such as accuracy, precision, sensitivity, and specificity. Given these considerations, then, we are confident in noting that our model offers some of the highest performance currently available.

### 7.3 Summary

In this study, we developed a novel CNN model to classify chest X-ray images as a means of detecting COVID-19 cases. The model was tested using two different datasets, a large dataset with a high imbalance of classes (the COVIDx dataset) and a small dataset with balanced classes and enhanced images (the enhanced COVID-19 dataset).

Following several experiments, the results we achieved demonstrate the excellent performance of the proposed model for both datasets. However, better results are reached using the enhanced COVID-19 dataset. We also observe that the enhancement of contrast for chest X-ray images helped the model to learn more features, therefore increasing its accuracy and ability to detect the presence of COVID-19. In addition, using a dataset that has balanced classes helps to achieve better outcomes than an unbalanced dataset, even while correcting the imbalance.

Besides this, we demonstrate that using different techniques of data augmentation for the training images helps to enhance the final model's predictions. The experiments that did not apply data augmentation achieved a significantly reduced classification accuracy as compared with the experiments that adopt augmentation.

Results show that a randomly initialized CNN (RND-CNN) can be used for analyzing chest X-ray images and
can also reach higher accuracy rates than using pre-trained networks.

8 | CONCLUSION AND FUTURE WORK

In this article, an efficient and low-computational approach is proposed to detect COVID-19 patients from chest X-ray images. This approach is based on a novel randomly initialized CNN architecture, or RND-CNN. This proposed architecture is used to classify images into one of three classes: normal, pneumonia, and COVID-19. We have used two datasets for the evaluation of this model: a large dataset with a high imbalance of classes (the COVIDx dataset) and a small dataset with balanced classes and enhanced images (the enhanced COVID-19 dataset). We analyzed the performance of our model through six performance metrics: precision, accuracy, sensitivity, specificity, loss, and F1-score. The conducted experiments recorded insightful results for both the COVIDx and enhanced COVID-19 datasets. Based on the obtained results, we demonstrated the high rates of recognition achieved by our RND-CNN model compared with other models and other types of weight initialization.

Possible extensions of our work include applying the RND-CNN model to analyze different types of images such as CT and MRI images. Future work could also expand its ability to classify images according to additional labels, such as Pneumothorax, Emphysema, and Fibrosis, among others.

ACKNOWLEDGMENT

The authors would like to thank Prince Sultan University for their support.

CONFLICT OF INTERESTS

The authors declare that they have no conflict of interest.

DATA AVAILABILITY STATEMENT

Data will be available upon request to the corresponding author.

ORCID

Wadii Boulila https://orcid.org/0000-0003-2133-0757

ENDNOTES

* https://www.who.int/
† MH-CovidNet/enhancement.part1.rar at master · mcanayaz/MH-CovidNet · GitHub
‡ GitHub-ieee8023/covid-chestxray-dataset: We are building an open database of COVID-19 cases with chest X-ray or CT images.

REFERENCES

1. Statista. COVID-19 New Cases Worldwide by Day. https://www.statista.com/statistics/1103046/new-coronavirus-covid19-cases-number-worldwide-by-day; 2021.
2. Statista COVID-19 Cases and Deaths Statistics by Country. https://www.statista.com/statistics/1105264/coronavirus-covid-19-cases-most-affected-countries-worldwide; 2021.
3. Hajjaji Y, Bouilla W, Farah IR, Romdhani I, Hussain A. Big data and IoT-based applications in smart environments: a systematic review. Comput Sci Rev. 2020;39:100318.
4. Al-Sarem M, Bouilla W, Al-Harby M, Qadir J, Alsaeedi A. Deep learning-based rumor detection on microblogging platforms: a systematic review. IEEE Access. 2017;7:152788-152812.
5. Faust O, Hagiwara Y, Hong TJ, Lih S, Acharya R. Deep learning for healthcare applications based on physiological signals: a review. Comput Methods Programs Biomed. 2018;161:1-13.
6. Wang J, Ding H, Bidgoli FA, et al. Detecting cardiovascular disease from mammograms with deep learning. IEEE Trans Med Imaging. 2017;36:1172-1181.
7. Ur Rehman M, Shafique A, Khalid S, Driss M, Rubaiee S. Future forecasting of COVID-19: a supervised learning approach. Sensors. 2021;21(10):3322.
8. Pereira R, Bertolini D, Teixeira L, Silla CN Jr, Costa YM. COVID-19 identification in chest X-ray images on flat and hierarchical classification scenarios. Comput Methods Programs Biomed. 2020;194:105532.
9. Ozturk T, Talo M, Yildirim E, Baloglu UB, Yildirim O, Acharya UR. Automated detection of COVID-19 cases using deep neural networks with X-ray images. Comput Biol Med. 2020;121:103792.
10. Apostolopoulos ID, Mpesiana TA. Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks. Phys Eng Sci Med. 2020;43:635-640.
11. Wang L, Lin ZQ, Dong W, Socher R, Li L-J, Fei-Fei L. COVID-19 disease severity assessment using CNN and efficient deep learning model for COVID-19 patterns detection. IEEE Access. 2020;8:248-255.
12. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. Paper presented at: 2009 IEEE Conference on Computer Vision and Pattern Recognition; pp. 248-255; 2009.
13. Karim MR, Rehbolz-Schuhmann D, Decker S, Cochez M, Beyan O. Deepcovidexplainer: Explainable covid-19 predictions based on chest X-ray images. arXiv Preprint. 2020;1-10.
14. Luz E, Silva P, Silva R, Silva L, CoRR GM. Towards an effective and efficient deep learning model for COVID-19 patterns detection in X-ray images. Res. Biomed. Eng. 2020.
15. Irmak E. Implementation of convolutional neural network approach for COVID-19 disease detection. Physiol Genomics. 2020;52:590-601.
16. Irmak E. COVID-19 Disease Severity Assessment Using CNN Model. IET Image Processing; 2021.
17. Gallicchio C, Scardapane S. Deep randomized neural networks. In: Oneto L, Navarin N, Sperduti A, Anguita D, eds. Recent Trends in Learning from Data. Springer; 2020:43-68.
18. Ulyanov D, Vedaldi A, Lempitsky V. Deep image prior. Paper presented at: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; pp. 9446-9454; 2018.
19. Pons J, Serra X. Randomly weighted CNNs for (music) audio classification. Paper presented at: ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); pp. 336-340, 2019.
20. Ravi D, Wong C, Deligianni F, et al. Deep learning for health informatics. *IEEE J Biomed Health Inform.* 2016;21:4-21.
21. Shah SA, Yang X, Abbasi QH. Cognitive healthcare system and its application in pill-rolling assessment. *Int J Numer Model: Electron Netw Devices Fields.* 2019;32(6):2632.
22. Shah SA, Ahmad J, Masood F, et al. Privacy-preserving wandering behavior sensing in dementia patients using modified logistic and dynamic Newton Leipnik maps. *IEEE Sens J.* 2020; 21:3669-3679.
23. Shah SA, Ahmad J, Tahir A, et al. Privacy-preserving non-wearable occupancy monitoring system exploiting Wi-fi imaging for next-generation body centric communication. *Micromachines.* 2020;11:379.
24. Bouilla W, Sellami M, Driss M, Al-Sarem M, Safaei M, Ghaleb FA. RS-DCNN: a novel distributed convolutional-neural-networks based-approach for big remote-sensing image classification. *Comput Electron Agric.* 2021;182:106014.
25. Bouilla W, Ghandorh H, Khan MA, Ahmed F, Ahmad J. A novel CNN-LSTM-based approach to predict urban expansion. *Ecol. Inform.* 2021;64:101325.
26. Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks. *Pattern Recognit.* 2018;77:354-377.
27. Ben Atitallah S, Driss M, Bouilla W, Ben Ghézala H. Leveraging deep learning and IoT big data analytics to support the smart cities development: review and future directions. *Comput Sci Rev.* 2020;35:100303.
28. Mohammadi M, Al-Fuqaha A, Sorour S, Guizani M. Deep learning for IoT big data and streaming analytics: a survey. *IEEE Commun Surv Tutor.* 2018;20(4):2923-2960.
29. Li H, Kréék M, Perin G. A comparison of weight initializers in deep learning-based side-channel analysis. Paper presented at: International Conference on Applied Cryptography and Network Security; pp. 126-143; 2020.
30. Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int J Uncertain Fuzziness Knowledge-Based Syst.* 1998;6(2):107-116.
31. Sun Y, Xue B, Zhang M, Yen GG. Evolving deep convolutional neural networks for image classification. *IEEE Trans Evol Comput.* 2020;24(2):394-407.
32. Glorot X, Yoshua B. Understanding the difficulty of training deep feedforward neural networks. Paper presented at: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings; pp. 249-256; 2010.
33. Wang L. COVIDx Dataset, 2020 [Online]. https://github.com/lindawangg/COVID-Net/blob/master/docs/COVIDx.md
34. Johnson JM, Khoshgoftaar TM. Survey on deep learning with class imbalance. *J Big Data.* 2019;6(27):1-54.
35. Leeve JL, Khoshgoftaar TM, Bauder RA, Seliya N. A survey on addressing high-class imbalance in big data. *J Big Data.* 2018; 5(1):1-30.
36. TensorFlow, Classification on imbalanced data [online]. https://www.tensorflow.org/tutorials/structured_data/imbalanced_data; 2020.
37. Canayaz M. MH-COVIDNet: diagnosis of COVID-19 using deep neural networks and meta-heuristic-based feature selection on X-ray images. *Biomed Signal Process Control.* 2021;64:102257.
38. Chowdhury ME, Rahman T, Khandakar A, et al. Can AI help in screening viral and COVID-19 pneumonia? *IEEE Access.* 2020;8:132665-132676.
39. Ying Z, Li G, Ren Y, Wang R, Wang W. A new image contrast enhancement algorithm using exposure fusion framework. Paper presented at: International Conference on Computer Analysis of Images and Patterns; pp. 36-46; 2017.
40. Mikolajczyk A, Grochowski M. Data augmentation for improving deep learning in image classification problem. Paper presented at: 2018 International Interdisciplinary PhD Workshop (IIPhDW); pp. 117-122; 2018.
41. Hawkins DM. The problem of overfitting. *J Chem Inf Comput Sci.* 2004;44(1):1-12.
42. Shorten C, Khoshgoftaar TM. A survey on image data augmentation for deep learning. *J Big Data.* 2019;6(60):1-48.
43. Francois C. *Deep Learning with Python.* Vol 361. Manning; 2018.
44. TensorFlow. https://www.tensorflow.org; 2020.
45. Kingma DP, Rezende DJ, Mohamed S, Welling M. Semi-supervised learning with deep generative models. *Adv Neural Inf Process Syst.* 2014;2:3581-3589.
46. Bottou L. Stochastic gradient descent tricks. In: Montavon G, Orr GB, Müller K-R, eds. *Neural Networks: Tricks of the Trade;* Springer: Berlin, Heidelberg; 2012:421-436.
47. Dahl GE, Sainath TN, Hinton GE. Improving deep neural networks for LVCSR using rectified linear units and dropout. Paper presented at: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing; pp. 8609-8613; 2013.
48. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. *arXiv Preprint.* 2014.
49. Chollet F. Xception: deep learning with depthwise separable convolutions. Paper presented at: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; pp. 1251-1258; 2017.

**How to cite this article:** Ben Atitallah S, Driss M, Bouilla W, Ben Ghézala H. Randomly initialized convolutional neural network for the recognition of COVID-19 using X-ray images. *Int J Imaging Syst Technol.* 2022;32(1):55-73. doi: 10.1002/ima.22654