Intelligence-Based Medicine 5 (2021) 100034

Contents lists available at ScienceDirect

Intelligence-Based Medicine

journal homepage: www.journals.elsevier.com/intelligence-based-medicine

Optimal hyperparameter selection of deep learning models for COVID-19 chest X-ray classification

Adeyinka P. Adedigba [a,*], Steve A. Adeshina [b], Oluwatomisin E. Aina [b], Abiodun M. Aibinu [a]

[a] *Department of Mechatronics Engineering, Federal University of Technology, Minna, Nigeria*
[b] *Department of Computer Engineering, Nile University of Nigeria, Abuja, Nigeria*

A R T I C L E I N F O

Keywords:
Chest X-ray
Computer-aided diagnosis
COVID-19
Cyclical learning rate
Deep convolutional neural network (CNN)
Discriminative fine-tuning
Hyperparameter optimisation
Memory and computation efficient
Mixed-precision training
Overfitting

A B S T R A C T

The first and most critical response to curbing the spread of the novel coronavirus disease (COVID-19) is to deploy effective techniques to test potentially infected patients, isolate them and commence immediate treatment. However, several test kits currently in use are slow and in a shortage of supply. This paper presents techniques for diagnosing COVID-19 from chest X-ray (CXR) and address problems associated with training deep models with less voluminous datasets and class imbalance as obtained in most available CXR datasets on COVID-19. We used the discriminative fine-tuning approach, which dynamically assigns different learning rates to each layer of the network. The learning rate is set using the cyclical learning rate policy that changes per iteration. This flexibility ensured rapid convergence and avoided being stuck in saddle point plateau. In addition, we addressed the high computational demand of deep models by implementing our algorithm using the memory- and computational-efficient mixed-precision training. Despite the availability of scanty datasets, our model achieved high performance and generalisation. A Validation accuracy of 96.83%, sensitivity and specificity of 96.26% and 95.54% were obtained, respectively. When tested on an entirely new dataset, the model achieves 97% accuracy without further training. Lastly, we presented a visual interpretation of the model's output to prove that the model can aid radiologists in rapidly screening for the symptoms of COVID-19.

1. Introduction

The first and most critical response to curbing the spread of the novel coronavirus disease (COVID-19) is to deploy effective techniques to test potentially infected patients, isolate them and commence immediate treatment [1]. Most current COVID-19 test kits can be categorised into two types – the molecular tests and serological tests. The molecular tests or nucleic acid tests involve the swab collection of tissue samples from a patient's nose or mouth. From these samples, the specific genetic signature of the virus that causes COVID-19 (i.e. the severe acute respiratory syndrome coronavirus 2, termed SARS-CoV-2) is checked for by using reverse transcriptase-polymerase chain reaction (RT-PCR) procedure [2]. On the other hand, the serological test involves checking the blood samples of potentially infected persons for traces of specific antibodies [3].

One major drawback with these screening techniques is the shortage of supply. For instance, with a population of over 200 million people and over 1.7 million households, Nigeria is unable to perform up to 100,000 tests due to the limited supply of these test kits.[1] Another drawback is the complicated manual process of conducting these tests for which there is a shortage of expertise in many developing countries, thus requiring the need to train new medical personnel amidst the growing pandemic. Other drawbacks of these tests include laborious processes and the long delay in outputting results, as some test kits take hours to yield results [1].

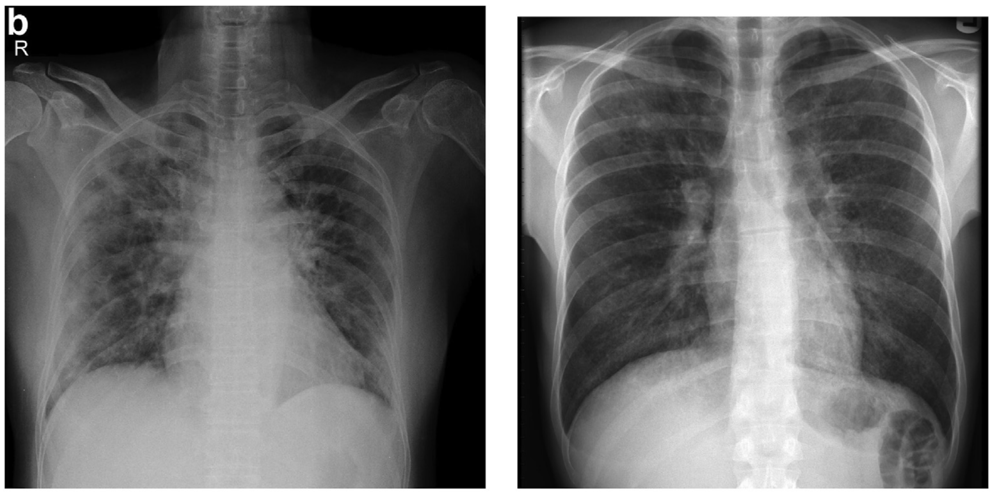
An alternative screening technique, which is capable of rapidly detecting COVID-19 is chest radiography – chest X-ray (CXR) and computed tomography (CT) imaging (see Fig. 1 for the chest x-ray of a COVID-19 and another Pneumonia patient).

X-ray imaging has been the de-facto approach for detecting lung inflammations, enlarged lymph nodes, pneumonia, and other breathing-related problems. When SARS-CoV-2 infects a person, it begins by affecting the epithelial cells that line the lungs. In this case, CXR can be used to analyse the patient's lungs for features of COVID-19 infection.

* Corresponding author.
E-mail addresses: adeyinka.adedigba@futminna.edu.ng (A.P. Adedigba), steve.adeshina@nileuniversity.edu.ng (S.A. Adeshina), oluwatomisin.aina@nileuniversity.edu.ng (O.E. Aina), abiodun.aibinu@futminna.edu.ng (A.M. Aibinu).

[1] Source: https://covid19.ncdc.gov.ng/accessed on 5th June 2020.

https://doi.org/10.1016/j.ibmed.2021.100034
Received 2 July 2020; Received in revised form 5 March 2021; Accepted 8 April 2021



(a) Chest X-ray of COVID-19 Patient (b) Chest X-ray of Pneumonia Patient

Fig. 1. Example of Chest X-ray COVID-19 and Pneumonia infected Patients: both diseases are associated with breathing difficulty.

Additionally, CXR can be used to study disease progression and post disease effect on the lungs. By using x-ray scans, the radiologist is expected to carry out a visual inspection of the scan to recognise indicators associated with the viral infection. In this regard, some earlier studies have identified abnormalities in chest radiography, which are linked to abnormalities caused by the COVID-19 infection [4].

The following advantages of CXR imaging for the detection of COVID19 were identified in Ref. [1]: firstly, CXR is considered the standard healthcare equipment; thus, it is readily available and accessible in many hospitals. Secondly, portable CXR systems are equally available; consequently, imaging can be performed in isolation rooms. Lastly, in isolation and test centres with many patients, CXR systems can allow for the rapid triaging of patients.

The questions that immediately follow are: how rapidly can a radiologist recognise indicators from a chest radiograph associated with the viral infection? What level of expertise will reduce false-negative results? Moreover, how many of these expert radiologists are available in low- or middle-income countries? These countries typically do not have the required amount of expertise needed to keep up with the demands of the pandemic coupled with the fact that COVID-19 indicators in chest radiographs are quite subtle and interpretation by expertise may be prone to false-negatives [5]. In parallel, deep learning systems can be designed to provide a rapid interpretation of radiographic images, identify regions of attention, and pass these results to the radiologist for further verification purposes. This computer-aided diagnostic (CAD) system can then be made available on the internet, a mobile device, or incorporated into a country's healthcare management portal.

Deep learning is a powerful, cutting-edge technology, which has found wide-spread application in medical diagnosis such as in the detection of breast cancers from mammograms, histopathology, breast ultrasound, etc. Deep learning can also be used to check for lung infections such as pneumonia, SARS and tuberculosis from x-ray scans; similarly, brain tumour has been detected from magnetic resonance imaging (MRI) scans [6]. For high performance, deep learning algorithms need to be trained on large datasets for long hours and require with tremendous computational power demands.

Based on the current 2020 global pandemic caused by COVID-19, the following problems are identified: (1) the performance of deep learning models depends on the availability of large datasets, which are limited at the moment as expected in any sudden disease outbreak, such as in the present experience, (2) deep learning models need high computational power requirements, which are unevenly distributed particularly in developing countries, thus limiting the capacity of researchers and research works from such regions, and (3) deep learning models are often treated as black-boxes, and critical medical decisions must be subjected to rigorous scrutiny and analysis. Similarly, deep learning models must be transparent by exhaustively analysing the inference process to gain wide acceptance.

Consequently, this article poses the following contributions in line with the problems mentioned above:

1. Insufficient data to train deep CNNs: We present a data-efficient method for training deep CNN to make judicious use of the scarcely available public datasets.
2. Deep CNNs require high computational resources: We present an algorithm capable of enhancing rapid convergence based on memory-efficient mixed-precision training techniques.
3. Deep CNN is treated as black-boxes: We implement a technique that improves the reliability of our model's inference statistics by providing visual clues coupled with its inference to aid the screening process.

The rest of this article is organised as follows: a review of related work is presented in Section 2, a data-efficient discriminative fine-tuning of deep CNN and mixed-precision training are formally introduced in Section 3. The dataset used and experimental setup are presented in Section 4, whereas the discussion of the result is presented in Section 5.

2. Literature review

One technique used to train deep learning models in problem domains with scanty datasets is the transfer learning technique. This technique can be efficient in training deep learning models to detect features of COVID-19 infection from chest x-rays since the availability of scans of positive cases are limited. Transfer learning is an example of domain adaptation techniques, where knowledge acquired from one domain (source domain) is transferred to a target domain which usually contains less training instances compared to the source domain. In transfer learning, knowledge transfer is facilitated by exploring domain-invariant structures that underline distribution discrepancies in two domains. Specifically, this is realised by retraining layers of previously trained deep neural networks (base models) with training data from the target domain [7]. The most common source domain used in many computer vision applications is the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), which comprises hundreds of millions of training instances. For a comprehensive review on transfer learning, readers may find [8] interesting.

Further, ILSVRC serves as a benchmark for testing advanced classification models [9]. The base model used would include models that perform appreciably well on ILSVRC, such as AlexNet, VGG, ResNet, DenseNet, among others. In this regard, Zeiler and Fergus [10] showed that regardless of dataset domain, deep CNNs can learn similar features in their early layers, thus, owing to the presence of these features in images, they are referred to as *general features* in Ref. [11]. Additionally, deep CNN can disentangle underlying features in an image and hierarchically group them according to their related invariance features such as edges, curves, and colour blobs [10,11].

Using this technique, authors in Ref. [12] trained three deep learning models with 100 images (50-50 positive and negative training examples were used, respectively). They reported accuracies of 100%, as well as appreciable specificity and sensitivity results, respectively. Their results present a case of too few datasets, wherein the model overfit during the training process. Similarly, it was noted that models trained using transfer learning could still overfit when the training dataset is too small [11]. Overfitting is a phenomenon where a model performs exceptionally well on training examples, but poorly on test and validation sets, thus, such models should not be deployed for critical real-world applications. In Ref. [13], the authors trained five deep learning models with 1427 images, 224 of which were COVID-19 positive. They achieved a high accuracy performance of 98.75% using VGG-19, while MobileNet v2 achieved a sensitivity of 99.10%. Their method suffers from the class imbalance problem, where one class presents more training examples than the other. Another instance of class imbalance can be found in Ref. [14], where ResNet50 was trained using 5941 images from which only 68 images were COVID-19 positive.

Nevertheless, authors in Ref. [1] achieved 83.5% accuracy and 87.1% sensitivity while training on a class imbalanced dataset. In Ref. [1], they used a generative synthesis technique to design a model tailored towards the detection of COVID-19. The generative synthesis is a machine-driven design exploration strategy. It is based on an intricate interplay between a generator-inquisitor pair that works in tandem to garner insights and learn to generate deep neural network architectures. The resulting network architecture called COVIDNET was firstly trained on ImageNet dataset and then retrained on 13800 chest x-rays (out of which only 183 were COVID-19 positive). Similarly [15], proposed a two-stage approach for classification of COVID-19 patient from CXR. In the first stage, lungs and heart contours were extracted from the CXR patches; this was fed to the second stage where classification was performed. Their method achieved 88.9% accuracy and 96.4% sensitivity.

It should be noted that models trained with imbalanced data are usually biased toward the class with larger training examples. Furthermore, it can be argued that the models presented in Refs. [12–14] suffered from overfitting owing to the scanty volume of datasets; thus, testing and validation of the model were limited. When these models are deployed in real-world applications, they may perform poorly. At this point, we should ask how we can overcome the class imbalance problem? In the following subsection, we provided a brief review of methods for overcoming class imbalance as found in the literature.

2.1. Methods for overcoming class imbalance

Development of a CAD system for diagnosis of new diseases (such as COVID-19) is one of the areas that suffers from class imbalance. Class imbalance reflects the prevalence of the disease in the study population, where we see that there are a lot more examples of negative cases than positive cases.

Class imbalance is a challenge to deep learning model and results in over-classification of majority group. Consequently, the minority group is often misclassified as belonging to the majority group. In binary classification, this misclassification could not be noticed from the model's accuracy but by observing the sensitivity and specificity of the model. Popular techniques for handling class imbalance include the following: random minority oversampling, random majority undersampling, and cost-sensitive learning [16].

Minority Oversampling: In its basic form, this technique randomly samples data from minority class and duplicate them in the dataset. It has been shown that this technique does not prevent overfitting [17]. Hence, several variations to this basic idea have been reported, such as cluster-based oversampling [18] which first clusters the dataset then over-sample each cluster separately. DataBoost-IM [19] first identifies difficult samples with boosting preprocessing and then generates synthetic images from these samples.

Majority Undersampling: In contrast to minority oversampling, samples from the majority class are randomly removed until a balance is attained [20]. Intuitively, this method is preferred to oversampling and often results in superior performance [21]. However, a good number of samples with relevant learnable information are discarded. When the dataset is too small (such as in Refs. [12,13]), this is not a desirable method.

Cost-Sensitive Learning: The majority class in a class-imbalanced dataset contributes more to the loss than the minority class. Consequently, the model learns more from the majority class than from the minority class. In cost-sensitive learning, the solution is to modify the loss function to weigh the majority class differently from the minority class. The weight is selected to amplify the contribution of the minority class to the loss; it eventually forces the model to learn from this class and the majority class. Techniques used for selecting the weight of minority/majority class include re-balancing, base rate by Bayesian learning, base rate by decision tree growing, decision tree pruning etc. [22].

2.2. Method for overcoming overfitting

Class imbalance affects the learning process and training accuracy, while overfitting affects our trust in the model. A model that overfits fail when deployed in a real-world application. How can we circumvent overfitting to achieve a good generalisation? One method is to train with large sets of training examples, which is limited and often unavailable at the moment for COVID-19 infections. Other approaches suggested in literature include data augmentation [23] and regularisation methods [24]. Data augmentation describes methods of increasing the size and diversity of datasets used to train machine learning models to achieve better generalisation [23]. These methods include random cropping, random intensity shift, horizontal and vertical flips, random rotation, centre-cropping, zooming, and random patches. In applying augmentation techniques, authors in Ref. [25] trained a deep learning model with 1531 images out of which only 100 images were COVID-19 positive. Nevertheless, they achieved an accuracy of 96% and 70.65% sensitivity. This result demonstrates that their model was less sensitive to COVID-19 symptomatic features, which stems from data imbalance on model performance.

On the other hand, Regularisation, refers to techniques that make slight modifications to the learning algorithm such that the model generalises better. Which, in turn, improves the model's performance on the unseen dataset. Regularisation can be achieved by optimal selection of hyperparameters such as learning rates, weight decay, batch-size, and dropout [26]. As shown in Refs. [24,27], various forms of regularisation (i.e. choice of hyperparameters) must be balanced per dataset and architecture to obtain good generalisations and to achieve faster training processes.

2.3. Memory-efficient deep learning approaches

In addition to preventing overfitting in deep learning model used for classification of COVID-19, it is desirable to design a memory- and computation-efficient models which can run on low computational resources such as mobile phone or embedded system such as raspberry pi. Approaches such as quantisation, pruning and mixed-precision training are considered herewith.

Model Pruning: this introduces sparsity into deep CNN weight connection by removing some redundant parameters in the network. Intuitively, deep CNN's have huge parameter; some of these parameters contribute little or nothing to the accuracy of the model, thus removing these redundant parameter does not affect model accuracy rather it reduces the model's memory demands [28]. Pruning approaches include weight pruning, neuron pruning, filter pruning and layer pruning [29, 30]. A combination of pruning and quantisation is presented in Ref. [31].

Quantisation: training deep neural network involves an iterative process with the following primary operations at each mini-batch: the forward propagation, the backward propagation, weight gradient computation, and loss optimisation. These operations are carried out using IEEE 754 single-precision floating-point numbers (FP32). The memory requirement of a deep CNN can be significantly reduced by reducing the number of bits used in these operations. Quantisation aims at reducing the number of bits used to represent the weights and activations of deep learning models. The idea of training neural networks with binary weights was proposed in Ref. [32]. In Ref. [33], weights and activations are quantised using 2, 4 and 6 bits; however, the gradient was estimated using FP32. Notwithstanding, quantisation leads to loss of accuracy due to the limited precision used for storing the network's weights and activations.

Mixed-precision training: Although quantisation uses a reduced number of bits to store weights and activations, the loss is calculated in FP32. In mixed-precision training, all tensors and arithmetic computation for both forward and backward passes used IEEE 754 half-precision floating-point numbers (FP16). More on this in section 3.3.

2.4. Model interpretation

Interpreting deep learning models is pertinent to its wide acceptance, especially in the medical field [34]. It provides a mechanism for assessing the trustworthiness of a model and enhances human-machine interaction. We present a brief review visualisation and model interpretation as found in the literature herewith.

Visualising deep learning models has been drawing research attention following the work in Ref. [10], which provides clues to what a deep CNN learns in each layer of the network. Authors in Refs. [35,36] extended this by developing methods for visualising CNN prediction by highlighting 'important pixels' that contribute to the model's prediction. Rather than highlighting pixels [37], proposed class activation mapping (CAM) for identifying discriminative regions used for interpreting a restricted class of classification task. However [34], provided a mechanism for interpreting the existing state-of-the-art deep models without altering their architecture.

In summary, COVID-19 symptoms can be detected from chest x-ray despite scantily available datasets by using transfer learning. Too few training datasets may cause the model to overfit; besides, data imbalance biases the model toward the class with larger training data. To prevent the model from overfitting, data augmentation and regularisation techniques are used as suggested in the literature. In the next section, a method that performs automatic and efficient hyperparameter tuning is presented. This method will be implemented using a computational and memory-efficient technique.

3. Methodology

In this section, we present a data- and computational-efficient method of fine-tuning deep learning models that results in improved accuracy and better generalisation. We begin by presenting a layer-wise fine-tuning process optimised for faster loss convergence. The whole process was implemented using memory and computational-efficient mixed-precision training technique.

3.1. Discriminative fine-tuning approach to transfer learning

Transfer learning is the reuse of a previously trained model on a new problem. It is realised by retraining layers of the base model with training data from the new problem domain. The process of retraining the base model involves two steps: modifying the base model's architecture to suite the classification or regression problem at hand, and re-training the model. Architecture modification is realised by replacing the output layer of the model with a new layer that outputs the desired number of outputs for the regression problem or desired number of classes in the multiclass or binary classification problem. Subsequently, the weight of the base model is loaded. The model can then be re-trained in the following modes: (i) as a feature extractor, where only the newly added output layer is trained, while other layers retain their default weights [38]; (ii) in a gradual unfreezing mode, where several of the last layers of the base model are retrained while leaving other layers frozen [39]; and (iii) a fine-tuning mode, where the entire base network is retrained as well as the newly added output layer [40]. A method for optimal fine-tuning is presented here.

The goal of the learning (fine-tuning) process is to minimise the objective function [41] given by:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(F(x_i,\theta), y_i) + \lambda\Omega(\theta) \tag{1}$$

where $F(x_i,\theta)$ is the network output (prediction) given input, x_i and network parameters, θ. y_i is the training label, N is the number of training samples in the dataset. $\Omega(\theta)$ is the regularisation function that penalises the weight gradient from becoming too large or too small and λ controls this penalty's strength. $L(\cdot)$ is the loss function that measures the deviation of the network's predictions from the ground-truth labels.

Using the stochastic gradient descent (SGD) method [41] as an optimiser, the parameter update is obtained as:

$$\theta_{t+1} = \theta_t - \alpha\frac{dJ(\theta_t)}{d\theta_t} \tag{2}$$

where α is the learning rate. Equation (2) uses a single learning rate to adjust network parameters throughout the training episode. It has been shown in Ref. [11] that different layers of deep CNN often capture different features, which range from *general* to *task-specific* features. Furthermore, since each layer learns different features, it follows that each layer of the network has its local objective, which enables it to learn the right feature. In transfer learning, the base model has been trained to recognise the general features learnt by earlier layers of the network because they are common in images, and are transferable to new tasks. Hence, the network parameters in these layers need not be rigorously updated compared to other layers. To take advantage of this, we break (1) and (2) into layers. This shows that each layer has its local objective function $J(\theta^l)$ updated by the parameters in the layer θ^l. Hence, the modified (1) and (2) for each layer is given by:

$$J\left(\theta^l\right) = \frac{1}{N}\sum_{i=1}^{N} L\left(F\left(x_i,\theta^l\right), y_i\right) + \frac{1}{2}\lambda\Omega\left(\theta^l\right) \tag{3}$$

and

$$\theta^l_{t+1} = \theta^l_t - \alpha^l\frac{dJ\left(\theta^l_t\right)}{d\theta^l_t}. \tag{4}$$

Further, since each layer minimises its objective function, it follows that each layer's parameter update can be tuned with different learning rates as shown in (4). The earlier layers are tuned with small learning rates while other layers (mostly, the last layer) are updated using larger learning rates to speedup network convergence (see result section for evidence of this significant speedup in network convergence). Equation (4) mathematically represents the concept of discriminative fine-tuning (DFT), where θ_t^l defines network parameters at layer l; these parameters are updated using layer-specific learning rate α^l.

It should be noted that DFT works well with all optimisers (not just SGD) and it does not affect the backpropagation of error. However, whereas backpropagation uses a constant learning rate for all layers of the network, DFT supplies dynamic learning rate to the backpropagation for parameter update. Consequently, all the desired qualities of optimiser of choice and the backpropagation algorithm are preserved.

In the following section, we discuss how this layer-specific learning rate is assigned to speedup network convergence without being stuck in saddle points.

3.2. Cyclical momentum and learning rate in discriminative fine-tuning

The optimal selection of learning rate is critical in DFT to facilitate the learning process. It should be noted that other hyperparameters such as momentum, dropout rate, weight decay and batch-size also affect the convergence of the loss function. Wrong selection of these hyperparameters (either too high or too low) can cause the algorithm to diverge significantly or progress slowly.

Methods of optimal selection of hyperparameters reported in the literature include grid search, random search [42], Bayesian optimisation in different forms [43,44], orthogonal array tuning [45] and cyclical learning rate (CLR) [24].

In this paper, we extended the concept of CLR introduced in Ref. [24] to select the learning rate and momentum. CLR begins with a learning rate ($\alpha_{\min}$) and rapidly increases its value using equation (5) to $\alpha_{\max}$ for some iterations. When it reaches $\alpha_{\max}$, it then gradually reduces the learning rate again using (5) to $\alpha_{\min}$ [24]. This approach follows from Ref. [46] wherein it was observed that high dimensional loss functions have negligible local minimal but are more likely to suffer from saddle points. Saddle points have insignificant gradients, which make learning to progress slowly and make it challenging to reach the global optimum. By rapidly traversing the saddle point plateau with high learning rate, the network converges faster. The learning rate can be obtained as.

$$\alpha_t = \alpha_{min} + \frac{\alpha_{max} - \alpha_{min}}{\alpha_{min}} t \tag{5}$$

where$t \in \left[1, \frac{N}{\text{batch size}}\right]$ is the iteration number in an epoch. Our objective is to vary the learning rate for each layer, whereas CLR varies the learning rate for each iteration. We combine these ideas in equation (6) and show parameter update with a layer-specific learning rate that changes per iteration t to accelerate training and avoid saddle points.

$$\theta_{t+1}^l = \theta_t^l - \alpha_{t+1}^l \frac{dJ(\theta_t^l)}{d\theta_t^l} \tag{6}$$

Then, in addition to the learning rate, the momentum parameter also contributes to the rapid convergence of the training algorithm. Hence, with further modifications, using SGD with momentum, we extend this idea to include layer-specific momentum as well as a momentum parameter that changes per iteration as follows:

$$v_{t+1}^l = m_{t+1}^l v_t^l - \alpha_{t+1}^l \frac{dJ(\theta_t^l)}{d\theta_t^l} \tag{7}$$

$$\theta_{t+1}^l = \theta_t^l + v_t^l \tag{8}$$

where v_{t+1}^l is the velocity of the moving average gradient, m_{t+1}^l is the momentum andα_{t+1}^l is the learning rate of the current iteration t+1 in layer l.

To summarise, DFT is a fine-tuning technique that selects different learning rates to update the parameters in each layer of the network. To speed up convergence and avoid being stuck in saddle point plateau, each iteration in the training epoch is selected to quickly increase the learning rate and momentum to take quicker strides from the plateau thus avoiding divergence of loss. The pseudocode of the DFT process is presented in Algorithm 1. Next, we show how this optimisation can be performed faster and with less memory usage.

3.3. Mixed-precision training

Training deep neural networks require high processing power and large memory capacity. This high demand results from the use of 32-bit IEEE.

Algorithm 1. Discriminative Fine-tuning Algorithms.

```
procedure DFT
    Input: ($\alpha_{min}$ : minimum learning rate,
            $\alpha_{max}$ : maximum learning rate,
            $m_{min}$: minimum momentum,
            $m_{max}$: maximum momentum,
            $\mathcal{N}$ : size of dataset, batch size)
    Output: ($\theta$: Network parameters)

    $t \leftarrow \frac{\mathcal{N}}{batch\ size}$
    $pct$ ← random number between 0.5 and 1
    // pct determines how rapidly the learning rate increases or reduces

while $t \leq pct \times t$ do:
    $\alpha_t \leftarrow \alpha_{min} + (\frac{\alpha_{max}-\alpha_{min}}{\alpha_{max}})t$    // Increase the learning rate by iteration
    $m_t \leftarrow m_{min} + (\frac{m_{max}-m_{min}}{m_{min}})t$    // Increase the momentum by iteration
    for $l$ in each layer do:
        $\alpha_t^l \leftarrow \alpha_{min} + (\frac{\alpha_t-\alpha_{min}}{\alpha_{min}})l$    // Increasing the learning rate per layer
        $m_t^l \leftarrow m_{min} + (\frac{m_t-m_{min}}{m_{min}})l$    // Increasing the momentum per layer
        $v_t^l \leftarrow m_t^l v_t^l - \alpha_t^l \frac{dJ(\theta_t^l)}{\theta_t^l}$
        $\theta_t^l \leftarrow \theta_t^l + v_t^l$    // Update the layer parameters
    end for
    $t \leftarrow t+1$
end while

while $pct \times t < t < t_{max}$ do:
    $\alpha_t \leftarrow \alpha_{max} - (\frac{\alpha_{max}-\alpha_{min}}{\alpha_{max}})t$    // Reduce the learning rate by iteration
    $m_t \leftarrow m_{max} - (\frac{m_{max}-m_{min}}{m_{min}})t$    // Reduce the momentum by iteration
    for $l$ in each layer do:
        $\alpha_t^l \leftarrow \alpha_{min} + (\frac{\alpha_t-\alpha_{min}}{\alpha_{min}})l$
        $m_t^l \leftarrow m_{min} + (\frac{m_t-m_{min}}{m_{min}})l$
        $v_t^l \leftarrow m_t^l v_t^l - \alpha_t^l \frac{dJ(\theta_t^l)}{\theta_t^l}$
        $\theta_t^l \leftarrow \theta_t^l + v_t^l$
    end for
    $t \leftarrow t+1$
end while
```

754 single-precision floating-point (FP32) which is the mainstay for deep learning training. Graphics processing units (GPUs) of different memory capacities have been used to speed up training time, which results in higher power consumption rates and incurs additional monetary cost. Recently, the use of 16-bit IEEE 754 half-precision floating-point (FP16) for training deep neural networks has been a topic of interest. Half-precision floating-point computation can attain 2 to 8 times speedup of training compared to single-precision [47].

The half-precision training reduces the number of mantissas from 23 in single-precision to 10, thus resulting in the loss of accuracy of the model. To address this problem, mixed-precision training (MPT) is proposed. According to Ref. [48], mixed-precision training involves three stages: (1) a master-copy of weights and weight-updates are maintained in FP32 to retain the accuracy of the network, (2) In each iteration, an FP16 copy of the weight is used for forward- and backward-propagation to store parameters as well as activations, thus, reducing by half the memory required during training, (3) In avoiding memory overflow and accurate representation of gradient values, the loss is log-scaled with small magnitude.

Recall that neural network training is an iterative process with the following primary operations at each mini-batch: the forward propagation, the backward propagation, weight gradient computation, and loss optimisation. Therefore, instead of computing these operations using traditional single-precision, mixed-precision training is implemented for the fine-tuning process (see Fig. 2 for the modified implementation of mixed-precision training).

3.4. CNN model architecture

Two deep learning models were fine-tuned to test our hypotheses – i.e. the selection of learning rates using the modified CLR technique to train the DFT with mixed-precision prevents overfitting, improves convergence, and speeds up training time. ResNet [49] and DenseNet [50] were adopted due to their large number of parameters and improved performance in ILSVRC.

Resnet152 is a 152-layer deep network that surpasses human-level performance in the 2015 ILSVRC with 3.57% top-5 error rate. Deeper networks like this have been shown to perform substantially better than shallower counterparts [51]. However, deeper networks are more prone to vanishing gradient problems, making them difficult to train [49]. This problem was addressed by the implementation of *Residual Block* in ResNet (see Fig. 3). The Residual block modelled in equation (9) creates a connection between the output of a convolutional layer and the earlier input to the layer using identity mapping [49]. Thus, the activation of a Residual block is given as:

$$a_l = H(a_{l-1}) + a_{l-1}, \tag{9}$$

where $\mathbf{a}_l$ is the activation of layer l, $H(\cdot)$ is a nonlinear convolutional transformation of the layer and $\mathbf{a}_{l-1}$ is the activation of previous layer $l-1$. The skip connection of (9) enables more layers to be stacked on each other resulting in a remarkably deep network.

DenseNet169 [50] is a 169-layer network with 14 million parameters, which can easily overfit on small data. This model is deeper; hence, it achieves higher performance than the ResNet152 on ImageNet dataset due to its *dense block*. The dense block implements a connection that allows a layer to be connected to all layers before it within the network

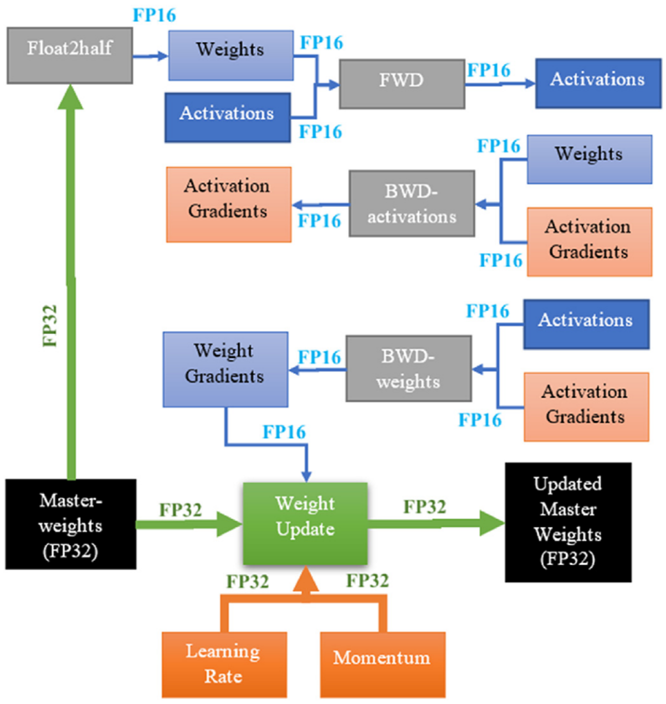


Fig. 2. Summary of Mixed-precision training implementation for one layer. Arrows show the precision used to implement each operation. Both FP16 and FP32 are used for weight update.

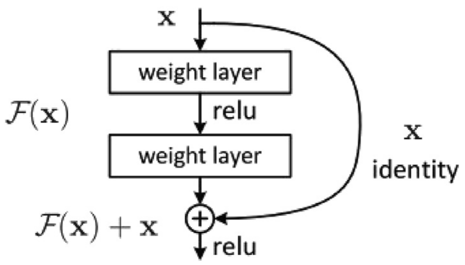

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 3. Residual Block [49].

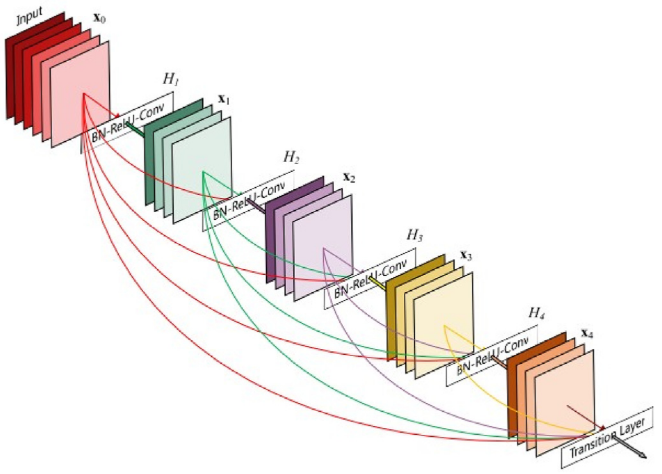


Fig. 4. Connection of a typical densenet [50].

[50] (see Fig. 4). That is, layer l receives feature activations from all its preceding $l-1$ layers as follows:

$$a_l = T([a_0, a_1, \ldots a_{(l-1)}]), \tag{10}$$

where **a** is the activation of the lth layer, $[a_0, a_1, \ldots a_{(l-1)}]$ is a concatenation of all the previous layer activations, which can be seen as a form of collective information gathered by the network up to that layer $l-1$. $T(\cdot)$ is a nonlinear transformation function that maps the concatenated activation to the activation of layer l.

Hence, these deep networks with huge parameters are the candidate choice to observe the performance of our fine-tuning algorithm and the mixed-precision training.

In this section, fine-tuning deep learning models using few computational resources without loss of accuracy and generalisation have been presented. This method leverages on DFT trained using mixed-precision training. Also, we presented the deep learning models which will be used to evaluate the method. In the next section, an experimental setup to test the efficiency of this method is presented.

4. Experimental setup

In this section, we discuss the experiment constructed to test the DFT presented in section 3. First, we present the COVID-19 CXR dataset, then we present data augmentation techniques implemented. The section ends with an overview of the experiments.

4.1. Dataset

Two CXR datasets from two different sources were used in this work. For clarity, we denote them as D_A, dataset A and D_B, dataset B. D_A was used for training and validation while D_B was used for testing the model.

The D_A was primarily obtained from the GitHub repository of Dr

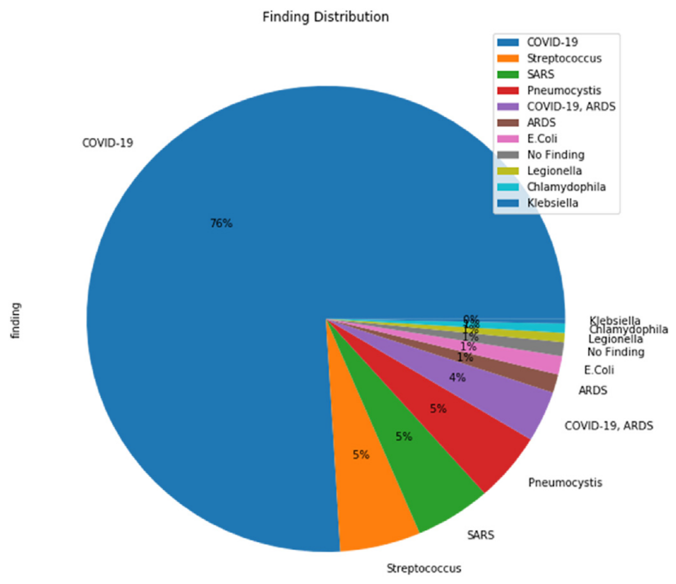


Fig. 5. Distribution of cases found in the dataset [52].

Joseph Cohen [52]. As at the time we accessed it, the repository contained chest x-ray of 339 patients, out of which 258 were tested positive for COVID-19 (see Fig. 5 for dataset distribution).

In this work, we focused on a binary classification of COVID-19 cases from chest x-ray images; however, it was observed that the data is mostly biased toward positive cases. Also, we observed that data imbalance contributes largely to the poor performance of the deep learning model developed thus far (see Section 2). Hence, we complemented the negative cases by randomly selecting chest x-ray of pneumonia from Kaggle,[2] and Tuberculosis chest x-ray from Montgomery County X-ray[3] set to obtain a balanced number of training examples from both positive and negative classes. Overall, D_A dataset contains 516 chest x-rays of balanced classes – 258 in each class. 70% of this dataset was then used for training, while the remaining 30% was used for validation.

On the other hand, D_B was obtained from COVID-19 radiography database on Kaggle.[4] The dataset contains 219 COVID-19 positive images with 1341 normal and 1345 viral pneumonia images [53]. For our use-case, 100 images were randomly selected from COVID-19 and normal images, respectively; a total of 200 images to form a balanced test set. Hence, D_B contains 200 test images that were used to show how well the model generalizes to an unseen dataset and proves that the model is not biased to any class.

4.2. *Data augmentation*

As noted in section 2, when the deep model is trained with too few data it suffers from overfitting, despite transfer learning. To guide against overfitting, data augmentation and regularisation techniques were employed. Regularisation, which involves the use of optimal hyper-parameters, has been designed in section 3.

However, in our previous work [23], we showed that improved performance could be achieved with carefully augmented datasets. Hence, the augmentation techniques developed in Ref. [23] was combined with regularisation techniques discussed in section 3. Data augmentation can be realised at run-time, during training, or at the pre-processing level before training commences. Data augmentation during training is implemented in most popular deep learning frameworks such as PyTorch, Tensorflow, Theano etc.

Data augmentation at the pre-processing level is implemented in Ref. [23]. The advantage of this includes the following: it supports visual inspection, editing and cleaning of the newly added augmented images; it limits the number of call-backs during training, thus, speeding up the training process; finally, it allows further augmentation to be carried out during training if it is so desired.

The parameters for the augmentation pre-processes are presented in Table 1 and samples of the augmented images are shown in Fig. 6. After dividing the dataset from the GitHub repository of Dr Joseph Cohen into training and validation set. The augmentation transformations were applied on each set and the resulting images were saved to disk for visual inspection and cleaning of inappropriate transformed images as well as repeated images. Meanwhile, no data augmentation was performed on the dataset from Kaggle which was used for testing the model.

4.3. *Overview of the experiments*

Discriminative fine-tuning starts by selecting a learning rate for each layer of the network using equation (5). This learning rate was used for CLR as well as layer-wise learning rates. This learning rate choice was accomplished by running a single epoch trial experiment using different learning rates and observing how the loss function increases or decreases during this epoch. Fig. 7 presents the result of this trial experiment obtained for ResNet; a similar graph is obtained for DenseNet. The learning rate selected is within the range where the slope of the loss function reduces sharply. From Fig. 7, this range is taken from $1e^{-3}$ to $1e^{-1}$; hence $\alpha_{\max}$ is $1e^{-1}$ while $\alpha_{\min}$ is $1e^{-3}$. A similar experiment was conducted for DenseNet and the $\alpha_{\max}$ was selected to be $1e^{-2}$, $\alpha_{\min}$ was $1e^{-4}$. These learning rates ($\alpha_{\min}$ and $\alpha_{\max}$) were plunged into equation (5) to vary the learning rate for each layer of the network, which was then used for parameter update of equation (6). The momentum was chosen in the range 0.8–0.99 for ResNet and 0.79 to 0.9 for DenseNet.

The learning rates ($\alpha_{\min}$ and $\alpha_{\max}$) and the momentum ($m_{\min}$ and $m_{\max}$) serves as input to the Algorithm 1 for the DFT experiment; this experiment was performed for ResNet as well as DenseNet with Adam optimiser and a constant L2 weight-decay of 0.01. The results of these experiments were presented in the next section.

5. Results and discussions

The results obtained from our experiments are presented and discussed in this section. MPT was shown to reduce computational demands of the algorithm, allowing the use of larger batch-size and higher resolution. In addition, DFT was shown to speed-up the training time, so that the best accuracy was achieved within 20 epochs. Using the validation and test results, we proved that the model generalises to unseen data. Then validation accuracy was compared to those reported in the literature. Furthermore, reliability and usability tests were conducted to prove that the model is readily useable to aid radiologists in triaging COVID-19 patients.

The augmentation experiment was carried out using Python and OpenCV library while the CNN training and benchmarking was done using PyTorch deep learning framework on a Lenovo Y520 computer with Nvidia GTX 1050Ti 4 GB GPU memory. Implementing mixed-precision allowed us to use a larger batch-size (64) and higher image

Table 1
Table of parameters for the augmentation pre-processes.

Data Augmentation	Parameter	Value(s)
Rotation	Rotation angle	± [5,10,15]
Gaussian Blurring	Kernel size	3
Random Zoom	Scale	1.3
Random Lighting	Intensity	1.4
Random Warp (Affine)	Magnitude	0.4

[2] https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia.
[3] http://openi.nlm.nih.gov/imgs/collections/NLM-MontgomeryCXRSet.zip.
[4] https://www.kaggle.com/tawsifurrahman/covid19-radiography-database.

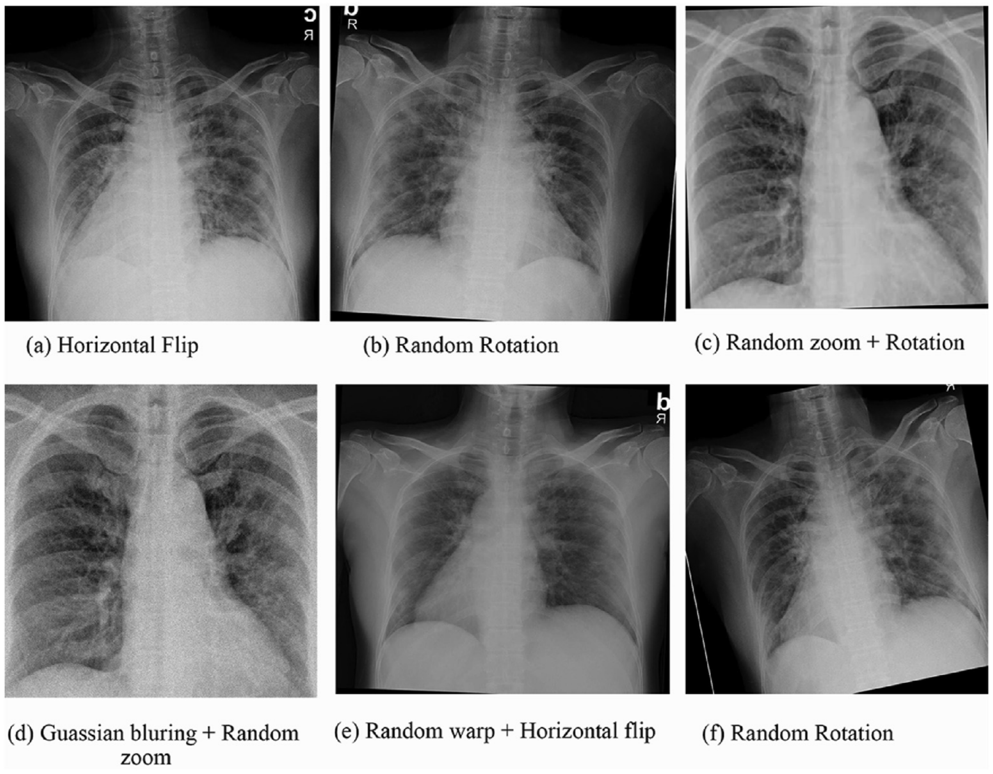

(a) Horizontal Flip
(b) Random Rotation
(c) Random zoom + Rotation
(d) Guassian bluring + Random zoom
(e) Random warp + Horizontal flip
(f) Random Rotation

Fig. 6. Samples of augmented images.

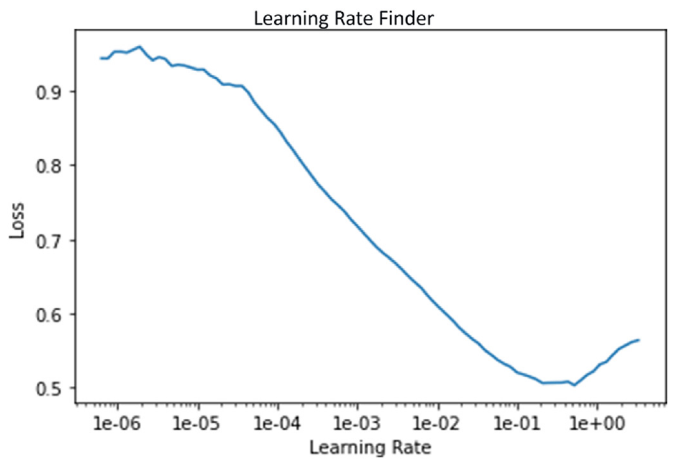


Fig. 7. Finding the optimum learning rate that best optimises the loss function. The Graph shows the variation of training loss with learning rate.

resolution (400×400) against using single-precision training (with batch-size of 8 and resolution of 224×224). DFT and mixed-precision training resulted in a speed-up of network convergence; hence, we ran the model for just 20 epochs, and the validation results of the models with and without data augmentation are presented in Table 2.

From the results in Table 2, it can be observed that both ResNet and DenseNet were not overfitting the small data owing to the use of the DFT technique. These results were improved by implementing augmentation techniques discussed in section 4.2 and the highest validation accuracy of 96.83% was recorded. In addition, we benchmarked our result against the traditional transfer learning method, and the result is presented in Table 3. The traditional method was trained using similar training hyperparameters with the DFT method except a constant learning rate and momentum of $1e^{-3}$ and 0.99 respectively for ResNet. The learning rate and momentum were set to $1e^{-4}$ and 0.9 respectively for DenseNet. Table 3 shows that it took longer training epochs and time to achieve the best accuracy from each model compared to our method. While it took approximately 30 min to complete one epoch in the traditional transfer learning method and about 100 epochs to attain the best result, it took approximately 14 min to complete one epoch and 20 epochs to attain best result using DFT. Furthermore, these results compared well to those reported in the literature – see Table 4. The model correctly classifies 196 out of 200 images on the test dataset, which accounts for 97% test accuracy. This test result is as good as the benchmark training accuracy reported in Ref. [53] without training our model on this dataset.

Why did the model not overfit our small dataset? Recall, regularisation refers to techniques that make slight modifications to the learning algorithm to enhance better generalisation of the model to unseen data. Also, regularisation can be seen as different knobs to fine-tune deep CNN to facilitate the effective learning process. A model's generalisation reflects its training process; an effective learning process guarantees good performance on unseen data.

Table 2
Validation results of discriminative fine-tuning.

Accuracy%	
DenseNet	94.17
ResNet	94.17
ResNet + augmentation	95.43
DenseNet + augmentation	96.83

Table 3
Performance Comparison of DFT with traditional Transfer learning method.

Model	Accuracy (%)	No of Epoch	Time (Hr)
ResNet	92.28	104	52
DenseNet	92.85	98	49
ResNet + DFT	95.43	20	5
DenseNet + DFT	96.83	20	4.67

Table 4
Performance comparison of our result with those reported.

Reference	No of COVID-19	Accuracy (%)	Sensitivity	Specificity
[1]	28	83.5	87.1	**97.0**
[25]	70	–	90.0	87.84
[15]	180	88.9	–	96.4
[54]	6	90 .0	83.3	80.0
Our Model	**256**	**96.83**	**96.26**	**95.54**

DFT technique provides flexibility and dynamic method of assigning network's hyperparameters which hitherto are constrained throughout the learning process. In the DFT technique, we selected different learning rate for each layer of the network to ensure optimal fine-tuning of the layer's parameter. This learning rate, as well as the momentum, are increased or reduced per iteration to ensure that the loss is not stuck in saddle point, which would hamper gradient flow, hence the learning process. This dynamic configuration positively enhances the learning algorithm and facilitate rapid convergence of the loss function, thus better generalisation to unseen data is observed.

Most regularisation techniques either penalise network parameters to limit the capacity of the model, constrain optimisation from diverging, enforce sparse representation of the model's activation or enforce early stopping. On the other hand, DFT is compatible and can be used along with other regularisation techniques such as parameter norm penalties (L1 or L2), drop out, early stopping and data augmentation. Furthermore, many of these techniques impose severe constrain on model architecture or parameters; DFT allows parameters to be more flexible and dynamic. All these make DFT a better regularisation technique than others; therefore, it can prevent deep CNN from overfitting as seen from the small dataset utilised in this work.

5.1. *Reliability analysis*

In developing a CAD system, it is desirable to show the reliability of the model in detecting or screening the disease for which it was designed. The model reliability is defined in terms of its sensitivity and specificity metrics, which are mathematically expressed as:

$$Sensitivity = \frac{TP}{TP + FN} \tag{11}$$

$$Specificity = \frac{TN}{TP + FP} \tag{12}$$

where *TP* (true positive) is the number of cases of COVID-19 that were correctly classified, *FN* (false negative) is the number of cases that were falsely reported as COVID-19, *TN* (true negative) is the number of cases that was correctly flagged as COVID-19 negative. In contrast, *FP* (false positive) is the number of cases reported as COVID-19 negative, whereas the patient was positive with the virus. The confusion matrix of DenseNet with data augmentation is shown in Fig. 8. Also, Table 5 shows the sensitivity and specificity of each model, with and without data augmentation.

The high sensitivity (also known as the true positive rate) is a measure of how often the model correctly classifies a positive COVID-19 case as positive. It means that the model will not confuse a positive case for a negative case.

This high sensitivity is essential; a positive person can infect many healthy individuals; hence, all infected persons must be correctly identified. On the other hand, high specificity measures how often the model correctly identifies negative COVID-19 cases. High specificity is equally essential so as not to create unnecessary fear in the population. As shown in Fig. 8, DenseNet correctly identifies 103 COVID-19 cases out of 108 and 107 negative cases out of 111. Hence, we can say the model is very reliable in identifying COVID-19 cases as well as negative patients.

Finally, it has been shown that a model that confidently predicts (with

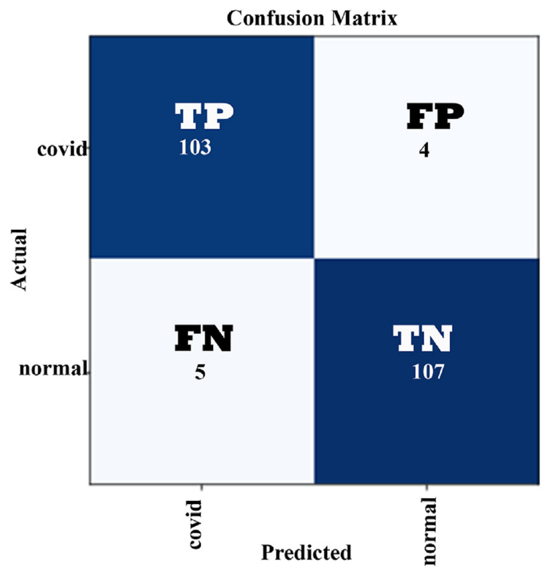


Fig. 8. DenseNet's Confusion Matrix showing the TP, FN, TN and FP.

Table 5
Reliability result of the models.

	Sensitivity (%)	Specificity (%)
ResNet	95.65	75.44
DenseNet	95.65	92.98
ResNet + augmentation	94.39	93.75
DenseNet + augmentation	96.26	95.54

high probability) a wrong class label should not be deployed in real-world applications [55]. Thus, we observed the wrongly classified labels by the.

Model, as shown in Fig. 9. The figure shows the model's predicted class, the actual ground-truth class, the loss for wrongly classifying the image, and the model's prediction probability of the actual class for each image displayed. From the figure, a high loss means the model confidently predicts a wrong class, whereas lower loss means the model's prediction is quite close to the actual ground-truth prediction. In binary classification, model prediction probability is $\hat{y}$ and the probability of belonging to the other class is $1 - \hat{y}$ (this is the probability referred to in Fig. 9). A low probability with high loss means the model wrongly classifies the image with high confidence (since $1 - \hat{y}$ is high). In contrast, a high probability in the figure means the model wrongly classifies the image with lower confidence. It should be further noted that the images in Fig. 9 are arranged in order of most confused images – i.e., the top left image is the most confused (i.e., images the model find difficult to classify). Fig. 9 shows that the loss associated with wrongly classified images are relatively small (0.76), which means they are not predicted with high confidence. Additionally, it can be verified from the figure that the model is not biased towards a particular class.

5.2. *Usability analysis*

We explored the question of how clinically useful this model will be. Many deep learning models have failed to achieve broad acceptance because they are seen as a black box, and critical decisions such as medical diagnosis must be subjected to rigorous scrutiny [56]. Validation accuracy, sensitivity, and specificity describe the reliability of the model, this section deals with usability analysis.

Although the models presented here achieved high validation accuracy, specificity, and sensitivity; these results may not guarantee the usability of these models in real-world applications. This is because these

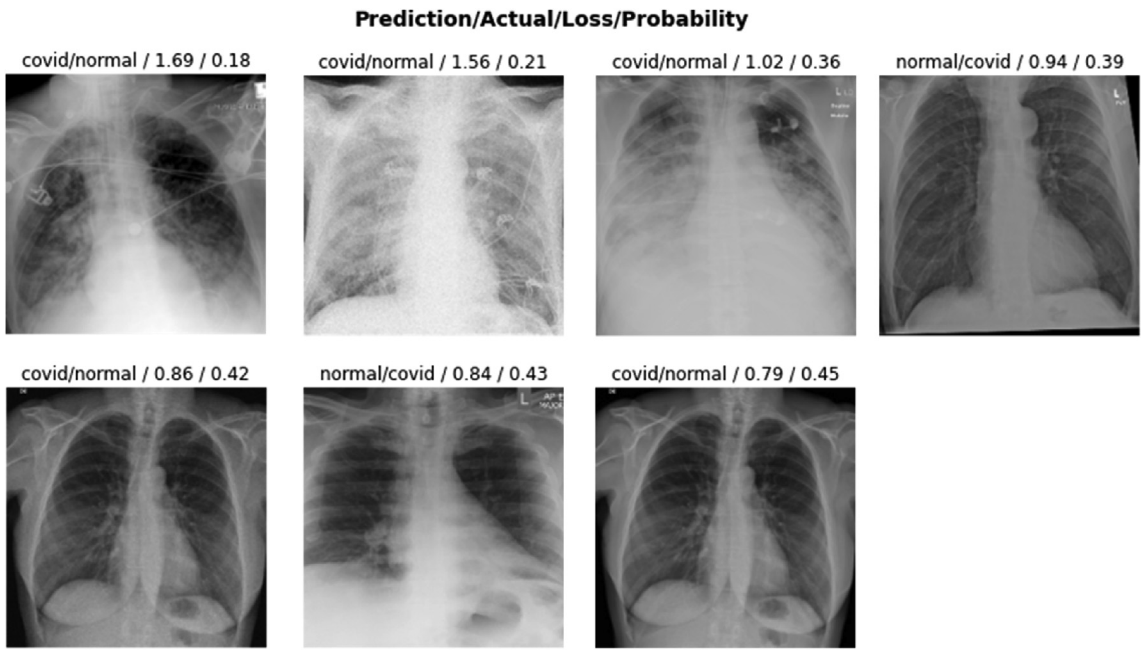

Prediction/Actual/Loss/Probability
covid/normal / 1.69 / 0.18
covid/normal / 1.56 / 0.21
covid/normal / 1.02 / 0.36
normal/covid / 0.94 / 0.39
covid/normal / 0.86 / 0.42
normal/covid / 0.84 / 0.43
covid/normal / 0.79 / 0.45

Fig. 9. Model's top confused images. Each Image has the predicted class by the model, actual class it belonged to, the loss for wrongly classifying the image, and the model's prediction probability of the actual class (i.e. the probability when the output is the actual class). It should be noted that the images are arranged in the order of most confused – with the top left image as the most confusing image.

models are designed to aid radiologists and clinicians in screening chest x-rays scans. Therefore, merely presenting the result as positives or negatives without intuitive justification will not foster the radiologists' trust, limiting widespread acceptance. The radiologists will see a model's performance in terms of its ability to give visual clues to screening chest x-ray, not often in terms of accuracy, specificity, or sensitivity. This is in line with [57], wherein it was noted that the success of deep learning models does not solely rely on its accuracy. Hence, for successful deployment, results should be interpretable in the context of the radiologist (users) – by providing a visual clue to the diagnosis of screening results.

We visualised the result of the model using the gradient-weighted class activation map (Grad-CAM) [34]. Grad-CAM uses gradients that flow to the final convolutional layer of the network to produce a form of a localisation heat map showing important neurons responsible for the model decision. This means, by observing the Grad-CAM output, the radiologist can intuitively understand why the model makes a particular decision. Further, Grad-CAM can be used to select a superior model in terms of usability, not just reliability.

As earlier stated, the SARS-CoV-2 virus affects the epithelial cells that line the lungs; further clinical studies show the affected regions of the lungs are the right middle zone, right lower zone and left upper zone [58]. In addition, the presence of opacity in the trachea is suggested as a sign of dry cough in COVID-19 patients and could be used to monitor the recovery progress of a patient [4]. The output activation map of our model visualised through Grad-CAM technique in Fig. 10. The upper right image highlights the ground glass opacity in the lower region of the right lung, similar to clinical findings in Ref. [59] as shown by the arrows. Similarly, the lower right image highlights the ground glass opacity in the lower right lungs as well as trachea region in agreement with the clinical findings reported in Ref. [60]. Hence, this model can aid the radiologist in diagnosing COVID-19 by accurately predicting an infected patient. In addition, this model can provide useful insights about lungs changes which can then be helpful during the treatment process. Thus, we can conclude that the model is as useable as it is reliable.

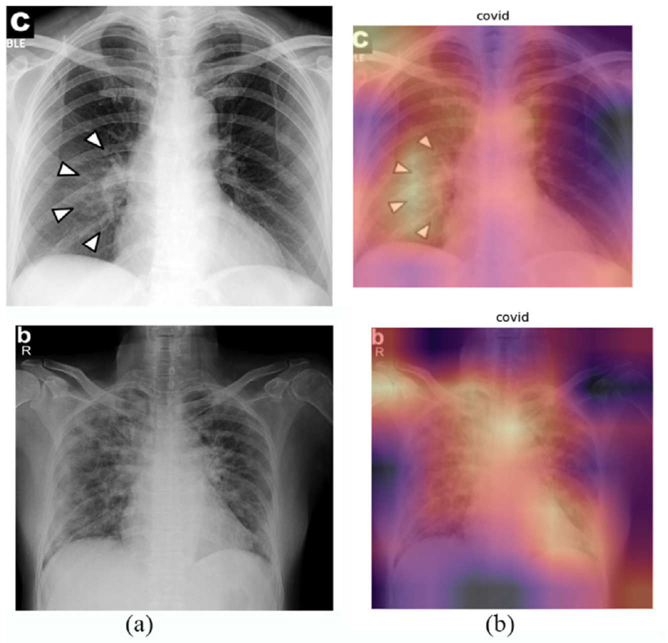


Fig. 10. Visualisation of model detection output using Gradient Activation Map.

6. Conclusion

An automatic method for optimal hyperparameter selection has been presented in this article. We have investigated discriminative fine-tuning, which dynamically assigns different learning rate to each layer of the network to ensure optimal fine-tuning of the layer's parameter. The learning rate, and momentum keep changing in each iteration to overcome being stuck in saddle point, which would hamper gradient flow and, consequently, the learning process. This dynamic tuning of hyper-parameter ensured rapid convergence of the loss function. Furthermore, DFT serves as a regulariser for the network, combined with other regularisation techniques such as weight decay, drop out and data augmentation, it produces better generalisation to unseen data. We also proposed using mixed-precision training, which makes efficient use of memory and guarantees faster computations. Our propositions serve a dual purpose of regularisation and faster convergence; hence, the models presented in this article converge within 20 training epochs. Our method achieves good performance without overfitting the data; besides, this was realised within a few training epochs. ResNet and DenseNet achieved the highest accuracy of 95.43% and 96.83%, respectively, on the validation set, 97% accuracy on the test set. The sensitivity and specificity for ResNet on validation set are 94.39% and 93.75%, respectively; similarly, DenseNet achieved 96.26% and 95.54% sensitivity and specificity, respectively.

In addition to correctly classifying CXRs and visualising its activation map, our model also identified potential challenging areas in the CXR, which corresponds to those reported in the literature. Therefore, we present this model as a COVID-19 computer-aided diagnostic tool that can also monitor a patient's lung changes during the treatment process. We have also reduced the computational burden of deep learning models via the mixed-precision training. Hence, this model can run on a low specification computer commonly found in the hospitals. This work will further be extended by designing a Graphical User Interface that will accept an incoming CXR and classify such as appropriate while providing visual clues to the Clinician. The entire system will be integrated into the hospital management system at the Nizamiye Hospital (Nile University of Nigeria's Teaching Hospital) and subjected to live testing while comparing results with manual diagnosis by Radiologists and Clinicians at the Medical centre. The authors believe that a successful testing will aid the diagnosis of COVID-19 in developing countries, especially with the second wave of the disease. Furthermore, the model can be re-trained to classify other lung infection and any other Severe Acute Respiratory Syndrome (SARS) mutations.

Declaration of competing interest

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.

Acknowledgement

The authors will like to acknowledge Professor Dilli Doge (MD and Provost) and Professor Senol Dane both of College of Health Sciences, Nile University of Nigeria, for their clinical perspectives. We also acknowledge Dr Adeiza James Onumanyi, of the Federal University of Technology, Minna for his insightful contribution to this work.

References

[1] Wang L, Wong A. COVID-Net: a tailored deep convolutional neural network design for detection of COVID-19 cases from chest radiography images. 2020. arXiv:2003.09871. URL, http://arxiv.org/abs/2003.09871.
[2] Zhu N, Zhang D, Wang W, Li X, Yang B, Song J, Zhao X, Huang B, Shi W, Lu R, Niu P, Zhan F, Ma X, Wang D, Xu W, Wu G, Gao GF, Tan W. A novel coronavirus from patients with pneumonia in China, 2019. N Engl J Med 2020;382(8):727–33. https://doi.org/10.1056/NEJMoa2001017.
[3] Bendavid E, Mulaney B, Sood N, Shah S, Ling E, BromleyDulfano R, Lai C, Weissberg Z, Saavedra R, Tedrow J, et al. Covid19 antibody seroprevalence in santa clara county, California. medRxiv; 2020.
[4] Huang C, Wang Y, Li X, Ren L, Zhao J, Hu Y, Zhang L, Fan G, Xu J, Gu X, Cheng Z, Yu T, Xia J, Wei Y, Wu W, Xie X, Yin W, Li H, Liu M, Xiao Y, Gao H, Guo L, Xie J, Wang G, Jiang R, Gao Z, Jin Q, Wang J, Cao B. Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China. Lancet 2020;395(10223):497–506. https://doi.org/10.1016/S0140-6736(20)30183-5.
[5] Ghoshal B, Tucker A. Estimating uncertainty and interpretability in deep learning for coronavirus (COVID-19) detection. 2020. 1– 14arXiv:2003.10769. URL, http://arxiv.org/abs/2003.10769.
[6] Adeshina SA, Adedigba AP, Adeniyi AA, Aibinu AM. Breast cancer histopathology image classification with deep convolutional neural networks. In: 2018 14th international conference on electronics computer and computation (ICECCO). IEEE; 2018. p. 206–12.
[7] Han S, Pool J, Tran J, Dally W. Learning both weights and connections for efficient neural network. In: Advances in neural information processing systems; 2015. p. 1135–43.
[8] Tan C, Sun F, Kong T, Zhang W, Yang C, Liu C. A survey on deep transfer learning. In: International conference on artificial neural networks. Springer; 2018. p. 270–9.
[9] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, et al. Imagenet large scale visual recognition challenge. Int J Comput Vis 2015;115(3):211–52.
[10] Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer; 2014. p. 818–33.
[11] Yosinski J, Clune J, Bengio Y, Lipson H. How transferable are features in deep neural networks?. In: Advances in neural information processing systems; 2014. p. 3320–8.
[12] Narin A, Kaya C, Pamuk Z. Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks. 2020. p. 10849. arXiv preprint arXiv:2003.
[13] Apostolopoulos ID, Mpesiana TA. Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks. Physical and Engineering Sciences in Medicine 2020:0123456789. https://doi.org/10.1007/s13246-020-00865-4. 1–6. arXiv:2003.11617, doi:10.1007/s13246-020-00865-4. URL.
[14] Farooq M, Hafeez A. COVID-ResNet: a deep learning framework for screening of COVID19 from radiographs. 2020. arXiv:2003.14395. URL, http://arxiv.org/abs/2003.14395.
[15] Oh Y, Park S, Ye JC. Deep learning covid-19 features on cxr using limited training data sets. IEEE Transactions on Medical Imaging; 2020.
[16] Buda M, Maki A, Mazurowski MA. A systematic study of the class imbalance problem in convolutional neural networks. Neural Network 2018;106:249–59.
[17] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. Smote: synthetic minority over-sampling technique. J Artif Intell Res 2002;16:321–57.
[18] Jo T, Japkowicz N. Class imbalances versus small disjuncts. ACM Sigkdd Explorations Newsletter 2004;6(1):40–9.
[19] Guo H, Viktor HL. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. ACM Sigkdd Explorations Newsletter 2004;6(1):30–9.
[20] Haixiang G, Yijing L, Shang J, Mingyun G, Yuanyue H, Bing G. Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl 2017;73:220–39.
[21] Drummond C, Holte RC, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on learning from imbalanced datasets II. vol. 11. Citeseer; 2003. p. 1–8.
[22] Elkan C. The foundations of cost-sensitive learning. In: International joint conference on artificial intelligence. vol. 17. Lawrence Erlbaum Associates Ltd; 2001. p. 973–8.
[23] Adedigba AP, Adeshina SA, Aibinu AM. Deep learning-based mammogram classification using small dataset. In: 15th international conference on electronics, computer and computation (ICECCO). IEEE; 2019. p. 1–6. 2019.
[24] Smith LN. Cyclical learning rates for training neural networks. IEEE winter conference on applications of computer vision. WACV), IEEE; 2017. p. 464–72. 2017.
[25] Zhang J, Xie Y, Li Y, Shen C, Xia Y. COVID-19 screening on chest X-ray images using deep learning based anomaly detection. 2020. arXiv:2003.12338. URL, http://arxiv.org/abs/2003.12338.
[26] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15(1):1929–58.
[27] Li H, Chaudhari P, Yang H, Lam M, Ravichandran A, Bhotika R, Soatto S. Rethinking the hyperparameters for fine-tuning. 2020. p. 11770. arXiv preprint arXiv:2002.
[28] Choudhary T, Mishra V, Goswami A, Sarangapani J. A comprehensive survey on model compression and acceleration. Artif Intell Rev 2020:1–43.
[29] Chen S, Zhao Q. Shallowing deep networks: layer-wise pruning based on feature representations. IEEE Trans Pattern Anal Mach Intell 2018;41(12):3048–56.
[30] Han S, Mao H, Dally WJ. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding. 2015. arXiv preprint arXiv:1510.00149.
[31] Zhu M, Gupta S. To prune, or not to prune: exploring the efficacy of pruning for model compression. 2017. arXiv preprint arXiv:1710.01878.
[32] Courbariaux M, Bengio Y, David J-P. Binaryconnect: training deep neural networks with binary weights during propagations. In: Advances in neural information processing systems; 2015. p. 3123–31.
[33] Hubara I, Courbariaux M, Soudry D, El-Yaniv R, Bengio Y. Quantized neural networks: training neural networks with low precision weights and activations. J Mach Learn Res 2017;18(1):6869–98.
[34] Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-cam: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision; 2017. p. 618–26.
[35] Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: visualising image classification models and saliency maps. 2013. p. 6034. arXiv preprint arXiv:1312.
[36] Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M. Striving for simplicity: the all convolutional net. 2014. arXiv preprint arXiv:1412.6806.

[37] Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A. Learning deep features for discriminative localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2921–9.

[38] Sharif Razavian A, Azizpour H, Sullivan J, Carlsson S. Cnn features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops; 2014. p. 806–13.

[39] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 3431–40.

[40] Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T. Decaf: a deep convolutional activation feature for generic visual recognition. In: International conference on machine learning; 2014. p. 647–55.

[41] Bottou L. Large-scale machine learning with stochastic gradient descent. Proceedings of COMPSTAT'2010. Springer; 2010. p. 177–86.

[42] Bergstra J, Bengio Y. Random search for hyper-parameter optimization. J Mach Learn Res 2012;13(Feb):281–305.

[43] Swersky K, Snoek J, Adams RP. Multi-task bayesian optimization. In: Advances in neural information processing systems; 2013. p. 2004–12.

[44] Bertrand H, Ardon R, Perrot M, Bloch I. Hyperparameter optimization of deep neural networks: combining hyperband with bayesian model selection. In: Conf'erence sur l'Apprentissage Automatique; 2017.

[45] Zhang X, Chen X, Yao L, Ge C, Dong M. Deep neural network hyperparameter optimization with orthogonal array tuning. In: International conference on neural information processing. Springer; 2019. p. 287–95.

[46] Dauphin YN, Pascanu R, Gulcehre C, Cho K, Ganguli S, Bengio Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In: Advances in neural information processing systems; 2014. p. 2933–41.

[47] Das D, Mellempudi N, Mudigere D, Kalamkar D, Avancha S, Banerjee K, Sridharan S, Vaidyanathan K, Kaul B, Georganas E, et al. Mixed precision training of convolutional neural networks using integer operations. 2018, 00930. arXiv preprint arXiv:1802.

[48] Micikevicius P, Narang S, Alben J, Diamos G, Elsen E, Garcia D, Ginsburg B, Houston M, Kuchaiev O, Venkatesh G, et al. Mixed precision training. 2017. arXiv preprint arXiv:1710.03740.

[49] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 770–8.

[50] Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. p. 4700–8.

[51] Goodfellow IJ, Bulatov Y, Ibarz J, Arnoud S, Shet V. Multi-digit number recognition from street view imagery using deep convolutional neural networks. 2013. arXiv preprint arXiv:1312.6082.

[52] Cohen JP, Morrison P, Dao L. COVID-19 image data collection. 2020. arXiv: 2003.11597. URL, http://arxiv.org/abs/2003.11597.

[53] Chowdhury ME, Rahman T, Khandakar A, Mazhar R, Kadir MA, Mahbub ZB, Islam KR, Khan MS, Iqbal A, Al-Emadi N, et al. Can ai help in screening viral and covid-19 pneumonia?. 2020. p. 13145. arXiv preprint arXiv:2003.

[54] Hemdan EE-D, Shouman MA, Karar ME, COVIDX-Net. A framework of deep learning classifiers to diagnose COVID-19 in XRay images. 2020. arXiv:2003.11055. URL, http://arxiv.org/abs/2003.11055.

[55] Thulasidasan S, Chennupati G, Bilmes J, Bhattacharya T, Michalak S. On mixup training: improved calibration and predictive uncertainty for deep neural networks (NeurIPS). 2019. p. 1–15. arXiv:1905.11001. URL, http://arxiv.org/abs/1905.11001.

[56] Ribeiro MT, Singh S, Guestrin C. Why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining; 2016. p. 1135–44.

[57] Shah NH, Milstein A, Bagley SC. Making machine learning models clinically useful. Jama 2019;322(14):1351–2.

[58] Rousan LA, Elobeid E, Karrar M, Khader Y. Chest x-ray findings and temporal lung changes in patients with covid-19 pneumonia. BMC Pulm Med 2020;20(1):1–9.

[59] Phan LT, Nguyen TV, Luong QC, Nguyen TV, Nguyen HT, Le HQ, Nguyen TT, Cao TM, Pham QD. Importation and human-to-human transmission of a novel coronavirus in vietnam. N Engl J Med 2020;382(9):872–4.

[60] Chen N, Zhou M, Dong X, Qu J, Gong F, Han Y, Qiu Y, Wang J, Liu Y, Wei Y, et al. Epidemiological and clinical characteristics of 99 cases of 2019 novel coronavirus pneumonia in wuhan, China: a descriptive study. Lancet 2020;395(10223):507–13.