SN Computer Science (2022) 3:115
https://doi.org/10.1007/s42979-021-00980-3

ORIGINAL RESEARCH

COVIDXception-Net: A Bayesian Optimization-Based Deep Learning Approach to Diagnose COVID-19 from X-Ray Images

Shifat E. Arman[1] · Sejuti Rahman[1] · Shamim Ahmed Deowan[1]

Received: 8 June 2021 / Accepted: 29 November 2021 / Published online: 30 December 2021

Abstract

COVID-19 is spreading around the world like wildfire. Chest X-rays are used as one of the primary tools for diagnosing COVID-19. However, about two-thirds of the world population do not have access to sufficient radiological services. In this work, we propose a deep learning-driven automated system, COVIDXception-Net, for diagnosing COVID-19 from chest X-rays. A primary challenge in any data-driven COVID-19 detection is the scarcity of COVID-19 data, which heavily deteriorates a deep learning model's performance. To address this issue, we incorporate a weighted-loss function that ensures the COVID-19 cases are given more importance during the training process. We also propose using Bayesian Optimization to find the best architecture for detecting COVID-19. Extensive experimentation on four publicly available COVID-19 datasets shows that our proposed model achieves an accuracy of 0.94, precision 0.95, recall 0.94, specificity 0.997, F1-score 0.94, and Matthews correlation coefficient 0.992 outperforming three widely used architectures—VGG16, MobileNetV2, and InceptionV3. It also surpasses the performance of several state-of-the-art COVID-19 detection methods. We also performed two ablation studies that show our model's accuracy degrades from 0.994 to 0.950 when a random search is used and to 0.983 when a regular loss function is employed instead of the Bayesian and weighted loss, respectively.

Keywords COVID-19 · SARS-CoV-2 · Chest X-ray · Deep learning · Bayesian optimization

Introduction

On December 31, 2019, the Chinese authorities informed the World Health Organization (WHO) about several cases of pneumonia caused by an unknown virus. This virus was later identified as a new strain of the coronavirus, now known as the Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2), causing the disease COVID-19. This virus belongs to the family of viruses called Coronaviridae [1]. Two viruses of this family, i.e., Severe Acute Respiratory Syndrome Coronavirus 1 (SARS-CoV-1) and Middle East Respiratory Syndrome (MERS) coronavirus, have previously caused epidemics. The mortality rate for these two viruses was high, with SARS 11% and MERS over 30% [2]. COVID-19 has a lower mortality rate of 5.7% [3]. However, its transmission rate is much higher. The virus spreads exponentially with new cases being identified worldwide, resulting in the WHO declaring it as a global pandemic on March 11, 2020 [4]. Six months from its first appearance, the virus has taken over the world by storm, with over 10 million confirmed cases and over 500,000 deaths. It has forced governments worldwide to shut off their borders and close various institutions to reduce the spread of the virus [5]. As a result, the world is heading towards an economic recession [6]. Nevertheless, there is a bigger problem—research suggests that about 40–45% of the patients are asymptomatic. That means they do not show any symptoms of the disease [7]. These asymptomatic patients, unaware of their infection, silently contribute to the spread of the virus. Thus, widespread testing and isolation is the only solution in this situation, emphasized by the WHO [8].

At present, Reverse Transcription Polymerase Chain Reaction (RT-PCR) is the most widely used method for testing COVID-19 patients [9]. The process involves detecting

✉ Shamim Ahmed Deowan
shamimdeowan.rme@du.ac.bd

Shifat E. Arman
shifatearman@gmail.com

Sejuti Rahman
sejuti.rahman@du.ac.bd

[1] Department of Robotics and Mechatronics Engineering, University of Dhaka, Dhaka, Bangladesh

viral Ribonucleic Acid (RNA) in a respiratory specimen collected from the patient. The RT-PCR method has several drawbacks. It provides low sensitivity to detection of COVID-19 [10]. Moreover, the result varies for the same patient at distinct points in time, even during the patient's diagnosis and treatment [11]. Hence, repeated tests are often required to get an accurate result [12]. The process is also time-consuming, and requires specific material and equipment which are not easily accessible [13, 14].

Radiological images like Chest X-Ray (CXR) and Computed Tomography (CT) can be used as an alternative to RT-PCR, as the COVID-19 primarily affects the respiratory system of the human body [15]. In COVID-19, the most common CXR and CT findings are Ground Glass Opacity (GGO) and lung consolidation [16]. CT was found helpful in early diagnosis of COVID-19, even when the patient did not show any respiratory symptoms or fever [17]. In some cases, GGO was seen in CT images of patients who were initially tested negative using RT-PCR tests. However, these patients were tested positive for COVID-19 later [18]. One experiment shows that the sensitivity of CT for detecting COVID-19 is 98% compared to RT-PCR, which is only 71% [10]. Although CT's sensitivity for detecting COVID-19 is more than RT-PCR, CT is not widely available, and it requires intense decontamination after scanning each COVID-19 patient, which disrupts the service. This is why the American College of Radiology (ACR) suggests using portable chest radiography to minimize the risk of cross-infection [19]. CXR is widely available, and the infection control issue is much less compared to CT [20].

Even though the radiological imaging techniques to detect COVID-19 give better sensitivity and can be done faster compared to RT-PCR, the system still relies heavily on radiologists for the detection of disease. Nevertheless, there is a global shortage of radiologists [21], and therefore, Artificial Intelligence (AI)-based diagnostic system can reduce the pressure on radiologists and help in faster diagnosis [22].

Deep learning is a sub-field of AI that is inspired by the brain's structure and function. It uses Artificial Neural Networks (ANN) to perform different tasks, including identifying objects from images, speech recognition, sentiment analysis, and many more [23]. It can also be applied to various types of medical images like X-ray, CT Scan, Magnetic Resonance Imaging (MRI), and Positron Emission Tomography (PET). These images can be used to segment, denoise, and classify diseases [24–28]. Convolutional Neural Network (CNN), a variant of ANN, has been used before to detect diseases from CXR [29]. It has also been applied to classify lung nodule from CT images [30].

CNN has parameters and hyperparameters. Parameters are weights and biases of a neural network that are learned during the training process and hyperparameters control the learning process of the neural network. Examples of hyperparameters are the number of layers or the number of nodes in each layer of a neural network. There are many hyperparameters of a neural network which can take any value. Selecting the right values of hyperparameters is very important as the neural network's performance is greatly affected by our choice. Nevertheless, the manual method of choosing hyperparameter is an arduous and time-consuming task. There are some automatic methods for tuning hyperparameters, e.g., random search, grid search, etc. [31]. However, the random search does not guarantee optimal results when the hyperparameter space is large, and the grid search is computationally expensive. Bayesian optimization is a more efficient method of tuning hyperparameters [32, 33]. It can give better results in fewer function evaluations compared to the grid and random search [34]. Using Bayesian optimization, it is also possible to take humans out of the loop of tuning hyperparameters [35]. In this paper, we used Bayesian optimization to tune the hyperparameters of the neural network. Our proposed network can classify X-ray images into three classes—normal, pneumonia, and COVID-19.

Data imbalance is a common problem for COVID-19 studies [36]. The amount of COVID-19 samples are much less compared to samples of other classes. To solve this issue, most of the early research on the diagnosis of COVID-19 from CXR used data augmentation [37–41] and class resampling [42–46]. In our approach, instead of using data augmentation or class resampling, we used a weighted loss function that ensures that the COVID-19 cases are given more importance while updating a neural network's parameters using the gradient of calculated loss.

The workflow of our proposed framework is presented in Fig. 1.

At first, a dataset is chosen which has three classes—normal, pneumonia, and COVID-19. Then, the data are pre-processed by performing resize and rescale operations. After pre-processing, the data were split into train, test, and validation sets. The train and validation set were used for model development and tuning. Models are built using four powerful CNN architectures—VGG16, MobileNetV2, InceptionV3, and Xception. These architectures were tuned using the Bayesian optimization technique to find the best-performing model. The best-performing model obtained from each architecture were then evaluated on the test data. To, evaluate the models, confusion matrix, accuracy, precision, recall, specificity, F1-score, Matthews correlation coefficient, and ROC curve were used. Based on the results obtained using these evaluation matrices, the best model was identified.

The overall contributions of the paper are listed below:

- We developed a novel model for diagnosis of COVID-19 called COVIDXception-Net based on Bayesian optimization of the Xception-Net architecture [47].

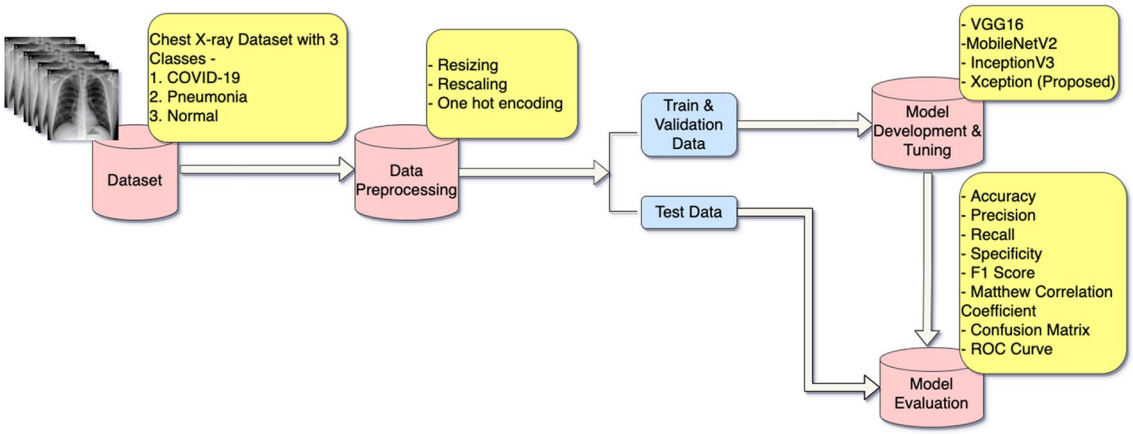


Fig. 1 Workflow of proposed framework for detection of COVID-19 using CNN

- We performed extensive experiments to determine the impact of Bayesian optimization in building our model. In addition, we demonstrated how each hyperparameter influences our model using Partial Dependence Plots.
- We proposed to use a weighted loss function to deal with the skewed distribution of samples due to the scarcity of the COVID-19 data.

The rest of the paper is organized as follows: Sect. "Related Works" discusses the related works that have been done in this field. The work methodology is discussed in Sect. "Our Approach". Section "Experimental Results" presents the results of the conducted experiments. Section "Discussion" discusses the findings from experimental results. Section "Conclusion" concludes the paper.

Related Works

The authors in [48] sought to analyze the findings obtained during the SARS and MERS outbreaks to help combat the COVID-19 pandemic. Both SARS and MERS were caused by distinct strains of the coronavirus family and produced similar respiratory symptoms. The authors have identified a substantial overlap of the imaging features of COVID-19 with that of SARS and MERS. They also found abnormalities in the early stage chest images of 85% of patients, suggesting that imaging techniques can help early diagnosis of infection.

In [49], the authors used CNN techniques to classify COVID-19 cases from CXR images. They used transfer learning to classify three classes from the CXR images: COVID-19, bacterial pneumonia, and normal. The CXR images were collected from publicly available medical repositories. The authors developed a neural network model that yielded the best accuracy of 96.78%, which suggests the feasibility of using such techniques for the diagnosis of COVID-19 cases.

The authors in [38] also explored the feasibility of using deep learning techniques for the diagnosis of COVID-19 from CXR. For this purpose, the authors created a public dataset of CXR images that combines three existing publicly available databases and images collected from various recent publications on this subject. The dataset comprises 190 cases of COVID-19, 1345 cases of viral pneumonia, and 1341 normal images. To mitigate the problem of large data requirements for CNNs, the authors used transfer learning to develop a classifier using four different pre-trained models: AlexNet, ResNet18, DenseNet201, and SqueezeNet. The authors developed both two-class and three-class classification model. The two-class classification model can classify normal and COVID-19 X-rays, whereas three-class classification model can also classify viral pneumonia X-rays along with normal and COVID-19 X-rays. Both the models were trained with and without augmentation. The authors found that CheXNet achieved best accuracy of 97.74% on three-class classification problem when the dataset was not augmented and DenseNet achieved best accuracy of 97.94% when the dataset was augmented.

In [50], the authors developed an open-source deep CNN named COVID-Net to detect COVID-19 cases from CXR images. COVID-Net classifies the CXR images into either of the three classes—no infection, non-COVID-19 infection, and COVID-19 infection. Its architecture comprises a lightweight design pattern consisting of a mix of convolution layers with diverse kernel sizes and grouping configurations. The performance of the COVID-Net was evaluated by comparing it with two popular architectures—VGG19 and ResNet50. COVID-Net achieved a test accuracy of 93.3%, which was higher than the other two. The authors also developed an open-source bench-marking dataset termed as COVIDx, which was used for training and evaluation purposes.

The COVIDx dataset consists of 13,975 CXR images from 13,870 patients, which were compiled from publicly available data repositories.

The authors in [37] did similar work to automate the diagnosis of COVID-19 using deep learning. They used the COVIDx dataset developed in [50], which we already discussed. To deal with the shortage of COVID-19 images in the dataset, they augmented the input images by vertical flipping, random rotation within 15°, and varying lighting conditions. The authors used the ResNet50 CNN architecture pre-trained on the ImageNet dataset. They trained the network in three stages using a progressive resizing technique. In each of the three stages, the images were resized, starting from the smallest size of $128 \times 128 \times 3$ to the largest size of $229 \times 229 \times 3$. It was possible to train the model in fewer epochs using this progressive resizing technique. They achieved an accuracy of 96.23% after training for only 41 epochs. The authors named their resultant CNN architecture as COVID-ResNet.

Authors in [39] used Bayes-SqueezeNet to diagnose COVID-19 from CXR images. They applied different augmentation techniques like shearing, adding noise, and increasing and decreasing brightness on the COVID-19 images to eliminate the negative effect of the imbalanced distribution of the raw dataset. The SqueezeNet architecture was pre-trained on ImageNet. They improved the SqueezeNet architecture by tuning hyperparameters using Bayesian optimization, and achieved an overall 98.3% accuracy, 98.3% correctness, 98.3% completeness, 99.1% specificity, 98.3% F1-score, and 97.4% Matthews correlation coefficient.

Most of the early studies to diagnose COVID-19 from CXR images uses CNNs. CNNs need a considerable amount of data to work efficiently, but a large amount of data is not yet available, as the COVID-19 situation is relatively new. CNNs have another disadvantage—it loses spatial information between image instances. To mitigate these problems, the authors in [51] proposed a model based on Capsule Networks named as COVID-CAPS. COVID-CAPS is capable of operating on small datasets. A small dataset comprises CXR images of COVID-19, bacterial pneumonia, viral pneumonia, and normal samples were used to train the model. Before training the model, the authors binarized the dataset's labels to COVID positive and COVID negative. Then binary classification was performed using COVID-CAPS to diagnose COVID and non-COVID cases. Using this small dataset, the model achieved an accuracy of 95.7%, sensitivity of 90%, and specificity of 95.8%.

As mentioned earlier, the primary challenge in any COVID-19 detection is the data imbalance issue due to the scarcity of COVID-19 image data. Therefore, the existing methods resort to different approaches, e.g., oversampling, under-sampling, data augmentation, and weighted loss to address the issue of skewness in the data. In oversampling, the number of images of one or more minority classes is increased by replication or selective data augmentation. In contrast, under-sampling reduces the number of images of one or more majority classes by random elimination. Data augmentation is also used to increase the number of samples and diversity by modifying the original data. Weighted loss function is used to control the learning process of an algorithm for better learning by assigning different weights to different parts of the loss function based on the number of appearances of different classes on the entire dataset.

A summary of the related works is given in Table 1 where we observe that the data imbalance issue was not addressed in [52]. The works in [37–39, 45, 50, 53] performed oversampling using different data augmentation techniques, whereas [45, 51, 53] used a weighted loss function to address the issue of data imbalance. The proposed method deals with the data imbalance issue in a way similar to the works in [51, 53, 54] and [45] using a weighted loss function. Even though

Table 1 Summary of literatures

Reference	Method	Data augmentation	Method to resolve data imbalance	Transfer learning	Hyperparameter optimization
[52]	DenseNet	No	–	Yes	No
[50]	Tailored CNN	Yes	Oversampling	Yes	No
[51]	Capsule networks	No	Weighted loss	No	No
[37]	ResNet50	Yes	Oversampling	Yes	No
[38]	Sgdm-SqueezeNet	Yes	Oversampling	Yes	No
[39]	COVIDiagnosis-Net	Yes	Oversampling	Yes	Yes
[53]	Model ensemble	Yes	Oversamping & Weighted loss	Yes	No
[45]	NASNet-Large	Yes	Oversampling & Weighted loss	Yes	No
Our approach	Bayes-XceptionNet	No	Weighted loss	Yes	Yes

'-' implies 'not addressed'

the loss function used is quite similar, there are some differences in the methodology and experimentation. Zhu et al. [54] developed a model incorporating weighted loss to classify severe and non-severe COVID-19 classification from CT scan images, which is different from ours. Punn et al. [45] also used a weighted loss function to detect COVID-19 from CXR. However, their task was slightly different, since they tried to classify CXR images into four classes—normal, COVID-19, pneumonia, and tuberculosis. Afshar et al. [51] developed a model using the Capsule network to distinguish between COVID and non-COVID images. Our task mostly resembles Goodwin et al. [53] who developed a model that classifies CXR images into one of the three classes—normal, pneumonia, and COVID-19. They used both: a weighted loss function and data augmentation. An ensemble of 14 architectures was used to make predictions. However, they did not use any algorithm for optimizing hyperparameters, whereas we used Bayesian optimization. Our model outperforms theirs in terms of accuracy and other evaluation metrics.

In a similar manner to ours, Ucar and Korkmaz [39] used Bayesian optimization to tune the hyperparameters of neural networks. However, they tuned a different set of hyperparameters, i.e., initial learning rate, momentum value and L2-regularization, whereas we tuned learning rate, number of dense layers, number of dense nodes, and activation function. The approach in [39] tuned only a single architecture, whereas we tuned four architectures and presented a quantitative comparative analysis of different models after the optimization. We obtained better results in terms of accuracy and other evaluation metrics in comparison to the method in [39].

Our Approach

We use a deep learning approach to classify CXR images into three classes—normal, pneumonia, and COVID-19. Here, we formulate the problem and describe our proposed model for solving the problem. We discuss the three important components of our solution approach namely feature extraction, learning classifier, and hyperparameter tuning using Bayesian optimization.

Problem Formulation

Assume we have a labeled training set that consists of N CXR images, $X_i = \{x_1, x_2, ..., x_n\}$ of $k = 3$ distinct classes, $\{Y_c\}_{c=1}^{k}$, that represent normal, pneumonia, and COVID-19. Given a test X-ray sample X_{test}, our goal is to assign a label $\hat{Y}_c$ by learning an end-to-end deep leaning model on the training set, $\tau = \{(X_i, Y_c) \ : \ i \in [1, N] \quad and \quad c \in [1, k]\}$.

Proposed Framework

In this paper, Bayesian optimization technique is used to optimize neural network. The overall process consists of three components—feature extraction, classification, and hyperparameter optimization. The feature extraction and classification is done by the CNN and the hyperparameter optimization is done using the Bayesian optimization technique. At first, the CNN is trained using the training data and the validation error is calculated. The validation error is used to update the Gaussian process model. After the approximation of the Gaussian process model, the expected improvement function proposes a new set of hyperparameter values. The expected improvement function decides whether to take the next set of hyperparameter values randomly or by exploiting the Gaussian process model. The new set of hyperparameter values are used to update the CNN model. Then, the CNN model is trained using the training data. The process continues for b iterations. After b iterations, the best model is picked based on the validation error and the model is evaluated on the test set. Figure 2 illustrates our proposed approach. Each of the three components is discussed in detail in the next sections.

Feature Extraction

CNN architectures have two basic parts—the feature extraction and the classification. The purpose of feature extraction layers is to extract important features from the data. In this study, feature extraction layers from four different CNN architectures: VGG16 [55], MobileNetV2 [56], InceptionV3 [57], Xception [47] are used. All these architectures are pretrained on ImageNet [58].

Classification

The purpose of the classification layers is to learn to classify the data based on the features obtained using the feature extraction layer. The classification layers consist of one or more fully connected layers. Each layer has some number of nodes. Various non-linear activation functions are used in each layer to learn complex function mapping from input to output. In this study, Bayesian optimization technique was used to learn the configuration of the classification layer.

Hyperparameter Tuning Using Bayesian Optimization

At the core of Bayesian optimization, we have the Bayes theorem. According to Bayes theorem:

$$P(H|E) \propto P(H) \times P(E|H). \tag{1}$$

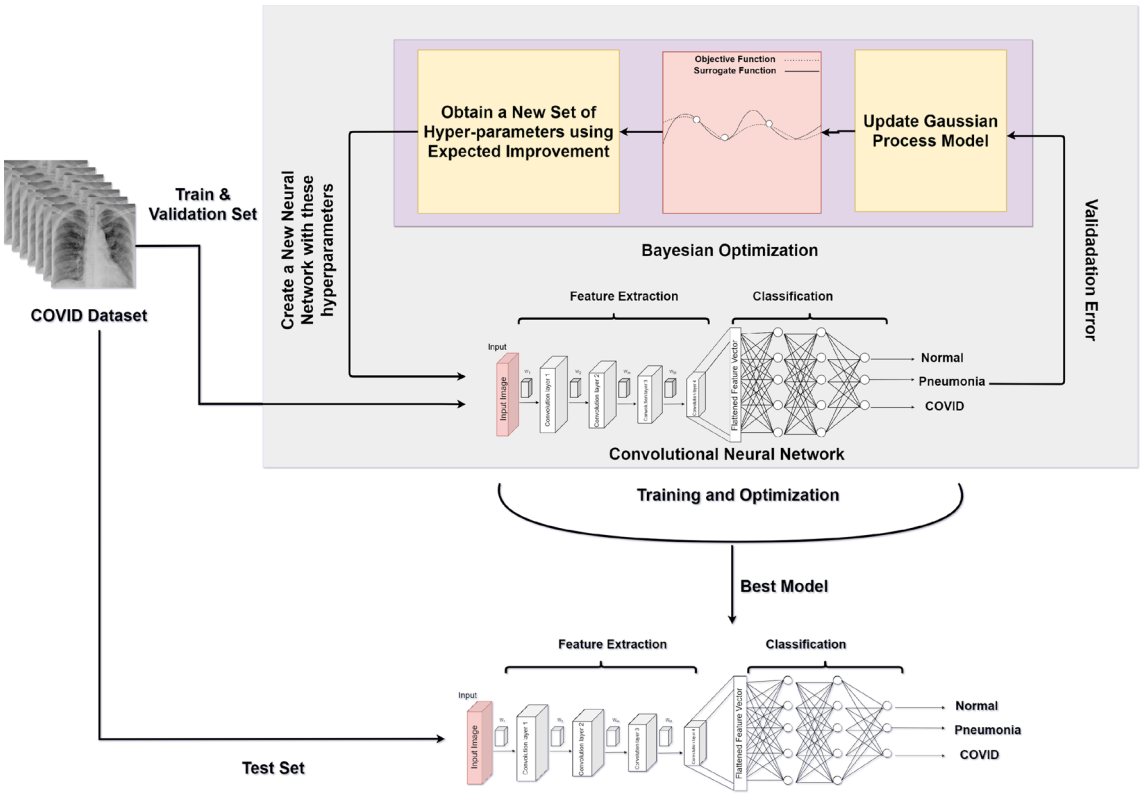


Fig. 2 Feature extraction, classification, and hyperparameter optimization

Here, H is the hypothesis. E is the evidence. $P(H)$ is the prior probability. $P(E|H)$ is the likelihood. $P(H|E)$ is the posterior probability.

The actual or true function that the Bayesian optimization technique tries to estimate is called the objective function. Bayesian optimization optimizes this function without knowing its gradient by taking sample points from the hyperparameter space. From the result of evaluating the function at these sample points, it tries to estimate the objective function. This estimation of the objective function is the surrogate function. It acts as the prior of our objective function.

$$\hat{f} \approx f. \tag{2}$$

Here, $\hat{f}$ is the surrogate function. f is the objective function.

Bayesian optimization incorporates prior belief about the objective function f using the surrogate function $\hat{f}$. It then updates the prior by drawing samples from f to get a posterior $\hat{f}$ that approximates f better. Gaussian Process (GP) is used as surrogate function in our experiments. There are two reasons for choosing GP—it is cheap to evaluate—and it approximates the objective function very well.

While choosing the sample points for evaluating the function, Bayesian optimization makes 'educated guess'. Acquisition function helps Bayesian optimization to make these educated guesses. It decides which point will have to be sampled next, based on mean and variance of the surrogate function. It also does the trade-off between exploration and exploitation. Exploration is the desire to look at an area or point where the variance of surrogate function is high, i.e., high uncertainty. Exploitation is the desire to look at an area or point where the mean of surrogate function is high, i.e., high promise.

There are different types of acquisition functions like Probability of Improvement (PI), Expected Improvement (EI), and Upper Confidence Bound (UCB). EI is used as acquisition function in our experiments.

EI can be defined by the equation:

$$EI(x) = \mathbb{E}\, max(f(x) - f(x^+), 0). \tag{3}$$

Here, x^+ represents the location of the best sample and $f(x^+)$ represents the value of the best sample so far.

Using GP model, we can evaluate expected improvement:

$$EI(x) = \begin{cases} (\mu(x) - f(x^+) - \zeta)\phi(Z) + \sigma(x)\phi(Z) & if\ \sigma(x) > 0 \\ 0 & if\ \sigma(x) = 0 \end{cases} \tag{4}$$

$$Z = \begin{cases} \frac{\mu(x) - f(x^+) - \zeta}{\sigma(x)} & if\ \sigma(x) > 0 \\ 0 & if\ \sigma(x) = 0. \end{cases} \quad (5)$$

Here, $\mu(x)$ represents the mean of GP posterior predictive at x and $\sigma(x)$ represents the standard deviation of GP posterior predictive at x. Φ is the Cumulative Distribution Function (CDF) of standard normal distribution and ϕ is the Probability Density Function (PDF) of standard normal distribution. $(\mu(x) - f(x^+) - \zeta)\phi(Z)$ maintains exploitation and $\sigma(x)\phi(Z)$ maintains exploration. The amount of exploration is controlled by ζ. If ζ is more, the amount of exploration is more. If ζ is less, the amount of exploration less.

To summarize the discussion so far: at first, an approximate function of the objective function called surrogate function is built. This function acts as the prior. Then, another function called acquisition function is used to decide whether to look at an area of high uncertainty or high promise. Based on the decision, the function is evaluated at that point and a posterior is obtained.

In this paper, the objective function is the validation loss and the sample points are values of the hyperparameters that are allowed to be tuned. The hyperparameters that we will tune are number of dense layers, number of dense nodes, learning rate, and activation function.

The algorithm of our proposed framework is shown in Algorithm 1.

Algorithm 1: Proposed Framework using Bayesian Optimization

Input: χ: Hyperparameter space,
$f(x)$: Objective function,
n: Maximum number of iterations
Output: x_{best}: Best hyperparameter configuration, y_{best}: Best objective value

```
Select an initial configuration x_0
Calculate objective value y_0 = f(x_0)
Initialize: x_best = x_0; y_best = y_0; D_0 = {x_0, y_0}
for i ← 1 to n do
    Obtain a new configuration x_i by optimizing the acquisition function g,
      x_i = arg min_{x∈χ} g(x|D_{i−1})
    Calculate objective value y_i = f(x_i)
    Augment the data D_i = D_{i−1} ∪ {x_i, y_i}
    Update the surrogate model
    if y_i < y_best then
        x_best, y_best = x_i, y_i
    end
end
```

The proposed framework takes as input a defined hyperparameter space χ, a objective function $f(x)$ which can be evaluated by training a neural network, and the maximum number of iterations n. The outputs are x_{best} and y_{best}, which are the best hyperparameter configuration and the best objective value, respectively. The first step is to select an initial configuration. Based on the initial configuration, a neural network is built and the objective value is calculated by training the neural network. After that, the variables x_{best}, y_{best} and the training set D which is used to update the surrogate model are initialized. Training D_i contains the set hyperparameter configurations and their objective values obtained until iteration i. After initialization, a loop runs for n iterations. A new hyperparameter configuration x_i is generated at each iteration by modifying an acquisition function α_i. Afterwards, objective value y_i is calculated for this hyperparameter configuration. This new configuration and objective value are then added to the training set D_i, and the surrogate model is updated. If the calculated objective y_i is less than y_{best}, both x_{best} and y_{best} are updated. After running for n iterations, the algorithm returns the best hyperparameter configuration and best objective value associated with it.

Dealing with Imbalanced Data: Weighted Loss

Since we have three classes in the problem under study, we use the categorical cross-entropy loss function given as follows:

$$L(total) = L(X, y_{normal}) + L(X, y_{pneumonia}) + L(X, y_{COVID}). \quad (6)$$

Here, $L(total)$ is the total categorical cross-entropy loss. $L(X, y_{normal})$, $L(X, y_{pneumonia})$, and $L(X, y_{COVID})$ are the cross-entropy losses for normal, pneumonia, and COVID samples, respectively.

However, since the dataset is imbalanced, we use the following weighted loss function instead that helps to ensure each class is given the same relative importance [59]:

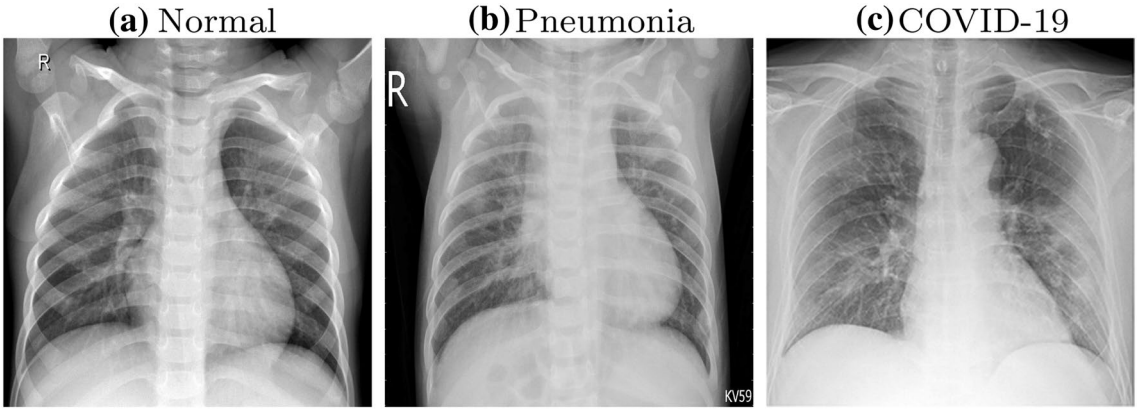


Fig. 3 Random samples of CXR from the dataset

Table 2 Class distribution of the dataset

Class	Total	Train	Validation	Test
Normal	1341	1261	20	60
Viral Pneumonia	1345	1265	20	60
COVID-19	220	140	20	60

$$L(total) = w_{normal} \times L(X, y_{normal}) + w_{pneumonia} \times L(X, y_{pneumonia}) + w_{covid} \times L(X, y_{covid}). \quad (7)$$

In Eq. 7, weight of each class is calculated using the following formula:

$$w_c = \frac{N}{k \times N_c}. \quad (8)$$

Here, N is the total number of observations in the training dataset and k is the total number of classes, whereas w_c and N_c denote the weight and total number of observations in class c, respectively.

Experimental Results

In this section, we present the experimental setup, evaluation process, results, and two ablations studies.

Dataset Description

We used the dataset compiled by Chowdhury et al. [38]. To create the dataset, normal, pneumonia and COVID-19 X-ray images from four different datasets were combined. Out of 220 COVID-19 X-ray images, 85 were collected from Italian Society of Medical and Interventional Radiology (SIRM) COVID-19 Database [60] and the rest were collected from different articles by Cohen et al. [61] and Chowdhury et al. [38]. Rest of the images of viral pneumonia and normal X-rays were collected from labeled CXR dataset on Mendeley [62]. The dataset was chosen, because it is open source and annotated by expert radiologists. Random samples of normal, viral pneumonia, and COVID-19 X-rays from the dataset are shown in Fig. 3.

The dataset was randomly split into train, test, and validation set. The distribution of data is shown in Table 2.

To train the deep learning model, the images were resized to 224 × 224 for VGG16, MobileNetV2, and InceptionV3 architecture and 299 × 299 for Xception architecture. In terms of percentage, only 7.2% of X-ray images were of COVID-19. That means the dataset is highly imbalanced. As mentioned earlier, we used the weighted loss function in Eq. 7 to address this issue.

Table 3 Training configuration

Training configuration	
Optimizer	Adam
Loss function	Weighted categorical cross-entropy
Number of epochs	250
Early stopping	Yes
Patience	5
Transfer learning	Yes
Mini-batch size	8

Implementation Details

The details of training configurations are presented in Table 3.

The experiments were performed on a laptop with Intel(R) Core(TM) i7-9750H CPU @ 2.60 GHz, NVIDIA GeForce RTX 2080 GPU, and 32 GB RAM. All the

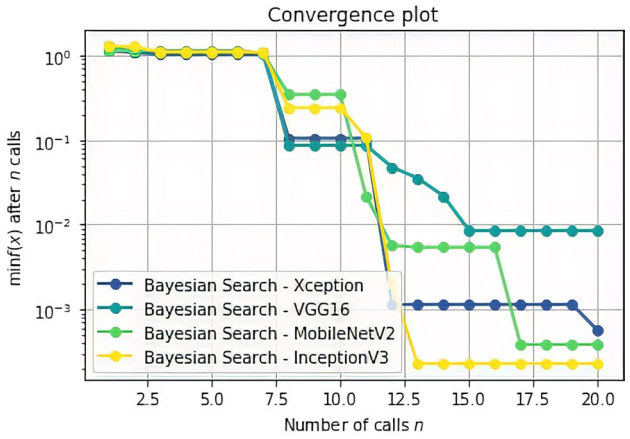


Fig. 4 Convergence plot for Bayesian search on four architectures

architectures were trained using Adam optimizer [63], and a weighted categorical cross-entropy loss function was used. The equation for the weighted loss function is given in Eq. 7. The weight for each class is calculated using Eq. 8. The number of epochs was set to 250 with patience of 5 for early stopping. The mini-batch size was set to 8. All the networks were pre-trained on ImageNet [58].

Hyperparameter Optimization of Neural Networks Using Bayesian Search

In our experiments, the Bayesian optimization technique was used to tune four hyperparameters of the neural networks. These hyperparameters were number of dense layers, number of dense nodes, learning rate, and activation function. Feature extraction layers were taken from VGG16, MobileNetV2, InceptionV3, and Xception architecture.

Figure 4 presents the convergence plot of four optimized models.

The Y-axis of the plot indicates the minimum value of the objective function $f(x)$ or the validation loss after n number of calls or trials. The X-axis represents the number of calls to the objective function. The validation losses after 20 iterations for InceptionV3, MobileNetV2, and Xception were very close. The VGG16 model achieved slightly higher validation loss.

The details of each iteration of Bayesian optimization on Xception architecture are presented in Table 4. The table presents two things—the observed value of the objective function and the hyperparameters of the model. In our experiments, the objective function was validation loss, and the model hyperparameters were learning rate, number of dense layers, number of dense nodes, and activation function. Each row of the table corresponds to the observed value of the objective function in a particular iteration and the model hyperparameters used to get that observed value. It is observed that the best or minimum value of the objective function was 0.0006 which was found

Table 4 Iteration results for Bayesian optimization on Xception architecture with model hyperparameters

Iteration no.	Objective	Learning rate	No. of Dense layers	No. of dense nodes	Activation
1	1.1405	1e-25	1	1	tanh
2	1.0885	6.62e-11	5	129	relu
3	1.0323	3.88e-10	3	45	tanh
4	1.1380	6.55e-19	3	122	tanh
5	1.1202	6.60e-16	5	51	relu
6	1.0938	1.61e-16	6	22	relu
7	1.1024	6.92e-14	5	79	relu
8	0.1052	1.04e-07	4	81	relu
9	1.0944	4.44e-23	3	29	relu
10	1.0821	2.59e-20	2	49	tanh
11	1.1208	3.61e-20	3	135	tanh
12	0.0011	3.11e-05	6	124	relu
13	0.0383	2.53e-06	3	109	relu
14	1.0984	1.85e-06	6	1	relu
15	0.0536	6.87e-04	6	98	relu
16	0.0434	2.65e-04	5	121	tanh
17	0.0383	6.34e-06	6	93	relu
18	1.1647	6.51e-02	1	150	relu
19	0.0160	7.68e-06	4	98	relu
20	**0.0006**	**6.92e-05**	**4**	**107**	**tanh**

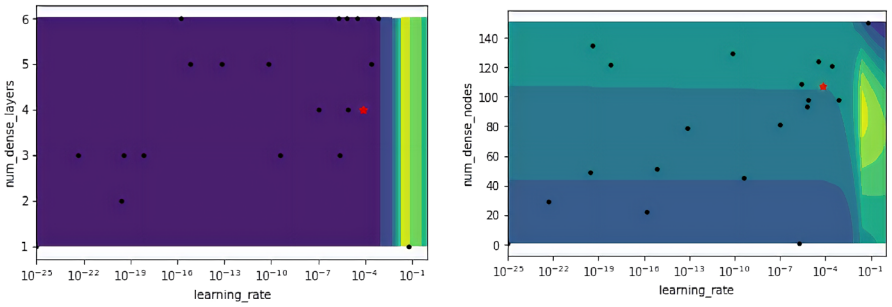
in the 20th iteration. The model hyperparameters at that iteration were learning rate $6.92e-05$, number of dense layers 4, number of dense nodes 107, and *tanh* activation.

Figure 5 presents three partial dependence plots for Bayesian optimization on Xception architecture with three hyperparameters—number of dense layers, number of dense nodes, and learning rate. Partial Dependence Plot (PDP) provides a relation between target response and target features. In this paper, the target response was the validation loss, and the target features were learning rate, number of dense layers, number of dense nodes, and activation function. A partial dependence plot can take one or more target features and express its relationship with the target response. The plot has two extreme regions:

- The yellow regions in the plots are the regions where the function value is minimum, i.e., the validation loss is minimum.
- The blue regions in the plots are the regions where the function value is maximum, i.e., the validation loss is maximum.

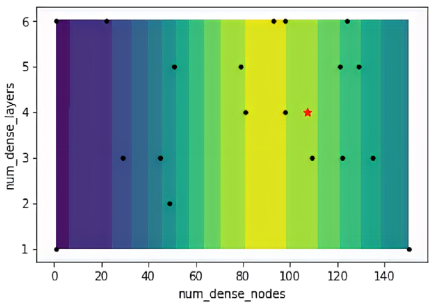
We are interested in the observations of yellow regions, as these are the regions where we get minimum validation loss. Each black dot represents a point in search space that was

Fig. 5 Partial dependence plots



(a) PDP with number of dense layers and learning rate

(b) PDP with number of dense nodes and learning rate



(c) PDP with number of dense layers and number of dense nodes

sampled. The red star shows where the best value of the hyperparameter was found.

From the PDP with the number of dense layers and the learning rate, we observe that the model performance was worse, or the validation loss was more when the learning rate was low. Performance improved with the increase of the learning rate and was dropped again after reaching a learning rate of 10^{-1}. Performance did not get much affected by the number of dense layers.

From the PDP with the number of dense nodes and the learning rate, we observe similar pattern for the learning rate. The performance was better when the learning rate was high. However, this time, the number of dense nodes had a large impact. Performance improved or validation loss decreased with the increase of the number of dense nodes. However, the impact was much less when the learning rate was low. When the learning rate was high, changing the number of dense nodes highly affected the performance of the model. Model performance was better when the learning rate was above 10^{-4}. The best value of the learning rate and the number of dense nodes were found in the red star point.

From the PDP with the number of dense layers and the number of dense nodes, we observe that the model performance was more dependent on the number of dense nodes. When the number of dense nodes was less, it did not matter how many layers we use, the performance was still bad. Model performance was better when the number of dense nodes was between 80 and 100. The best value of the number of dense layers and the number of dense nodes was found in the red star point.

Evaluation of the Optimized Models

In this subsection, we introduced different metrics for evaluating the optimized models and discussed the results of the evaluations.

Evaluation Metrics

Accuracy, precision (correctness), recall (completeness), specificity, F1-score, Matthews Correlation Coefficient (MCC), confusion matrix, and Receiver-Operating Characteristic (ROC) curve were used to evaluate the models.

Accuracy is the ratio of the number of samples correctly classified to the total number of samples.

$$\text{Accuracy} = \frac{TP_c + TN_c}{TP_c + FP_c + TN_c + FN_c}. \tag{9}$$

Here, TP_c, TN_c, FP_c, and FN_c denote the number of true-positive, true-negative, false-positive, and false-negative samples for class c, respectively.

Precision or correctness is the ratio of number of correct positive results to the number of all predicted positives.

$$Precision = \frac{TP_c}{TP_c + FP_c}. \tag{10}$$

Recall or completeness is the ratio of number of correct positive results to the number of all samples that should be identified as positives.

$$\text{Recall} = \frac{TP_c}{TP_c + FN_c}. \tag{11}$$

Specificity is the ratio of number of correct negative results to the number of all samples that should be identified as negatives.

$$\text{Specificity} = \frac{TN_c}{TN_c + FP_c}. \tag{12}$$

F1 score is the harmonic mean of precision and recall. It can take values ranging from 0 to 1. The model is perfect when the F1 score is 1 and the model is failure when the F1 score is 0.

$$\text{F1} = 2 \times \frac{Precision_c \times Recall_c}{Precision_c + Recall_c}. \tag{13}$$

Here, $Precision_c$ is the precision for class c and $Recall_c$ is the recall for class c.

Precision, recall, specificity, and F1-score are all asymmetric evaluation metrices. That means, their values change when the class labels are swapped. Matthews Correlation Coefficient (MCC) is a symmetric evaluation metrics. It does not change with the change of class label. It can take values between -1 to 1. -1 means the model misclassified all the data, 0 means the prediction of the model is random, and 1 means the model perfectly classified all the data.

$$\text{MCC} = \frac{\left(TP_c \times TN_c\right) - \left(FP_c \times FN_c\right)}{\sqrt{\left(TP_c + FP_c\right) \times \left(TP_c + FN_c\right) \times \left(TN_c + FP_c\right) \times \left(TN_c + FN_c\right)}}. \tag{14}$$

Evaluation Results

The classification performance of all four optimized models is presented in Table 5. The Xception model outperforms all the other models in terms of accuracy, precision, recall, specificity, F1-score, and Matthews correlation coefficient on the test set.

The confusion matrices for four optimized models are presented in Fig. 6.

From the confusion matrices, it is observed that none of the models managed to predict all 60 COVID-19 cases correctly. For the other two cases (normal and pneumonia), VGG16 misclassified 1, and MobileNetV2 and InceptionV3 misclassified 3, but the Xception model did not misclassify any case. The Xception model misclassified only 1 X-ray in total. The actual X-ray was of COVID-19, but the model predicted it as normal. The confusion matrices clearly demonstrate the superiority of the Xception model over the other three models.

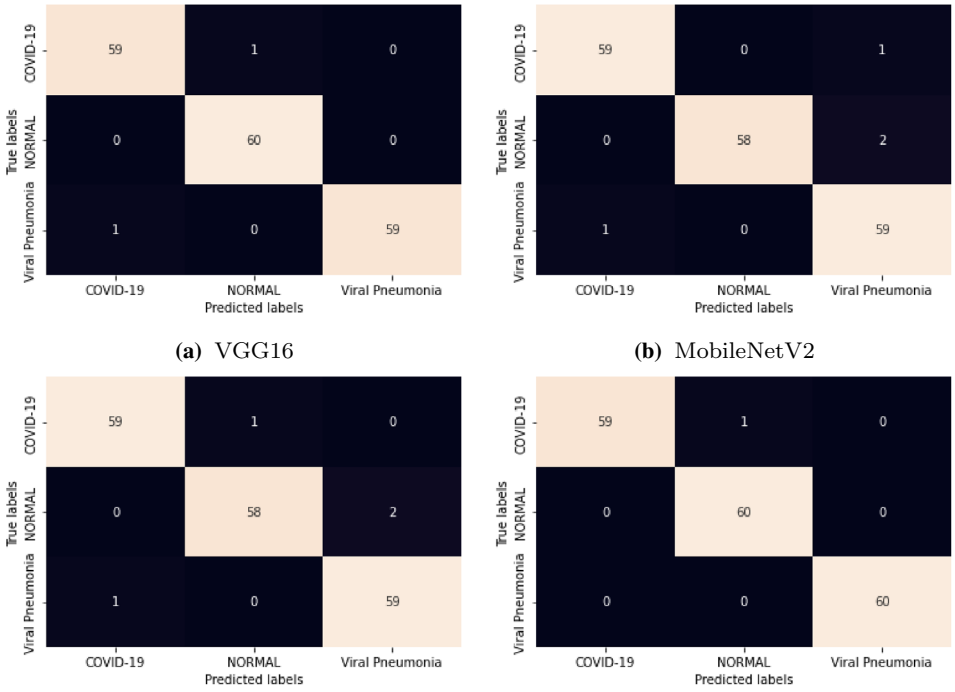
The training loss, training accuracy, validation loss, and validation accuracy of four best-performing models found by optimizing VGG16, MobileNetV2, InceptionV3, and Xception using Bayesian Optimization technique are shown in Fig. 7.

All the models were set to train for 250 epochs. However, since early stopping was used, models which did not show any improvement in validation loss for five consecutive epochs were early stopped. The final validation losses achieved by the MobileNetV2, InceptionV3, and the Xception model were very close. The validation loss for VGG16 is slightly higher compared to the other three models. At

Table 5 Classification performance of four optimized models

Model	ACC	PRE	REC	SPE	F1	MCC
VGG16	0.989	0.989	0.989	0.994	0.989	0.983
MobileNetV2	0.978	0.978	0.978	0.988	0.978	0.967
InceptionV3	0.978	0.978	0.978	0.988	0.978	0.967
Xception	**0.994**	**0.995**	**0.994**	**0.997**	**0.994**	**0.992**

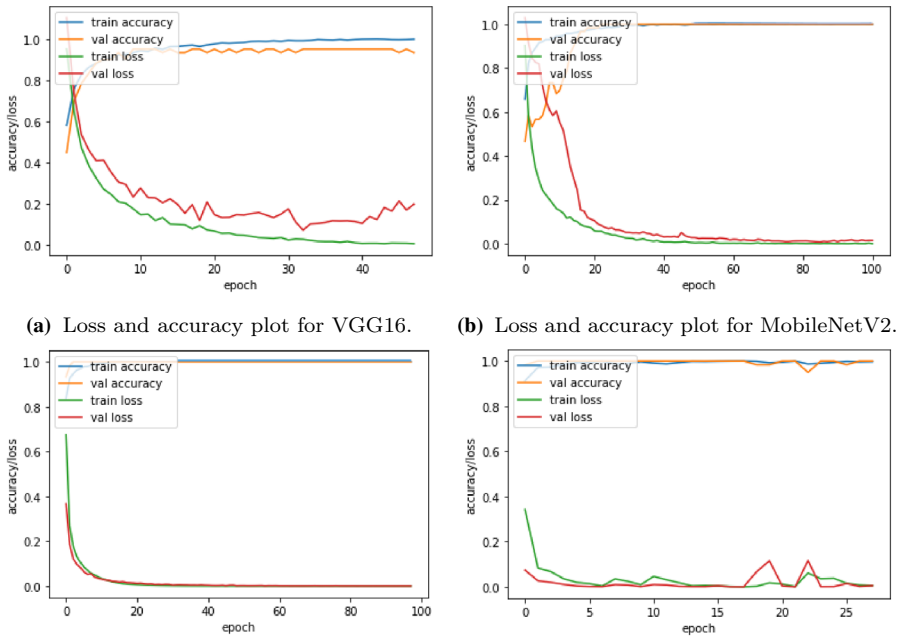
'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient



(a) VGG16 **(b)** MobileNetV2

(c) InceptionV3 **(d)** Xception

Fig. 6 Confusion matrix of four optimized models



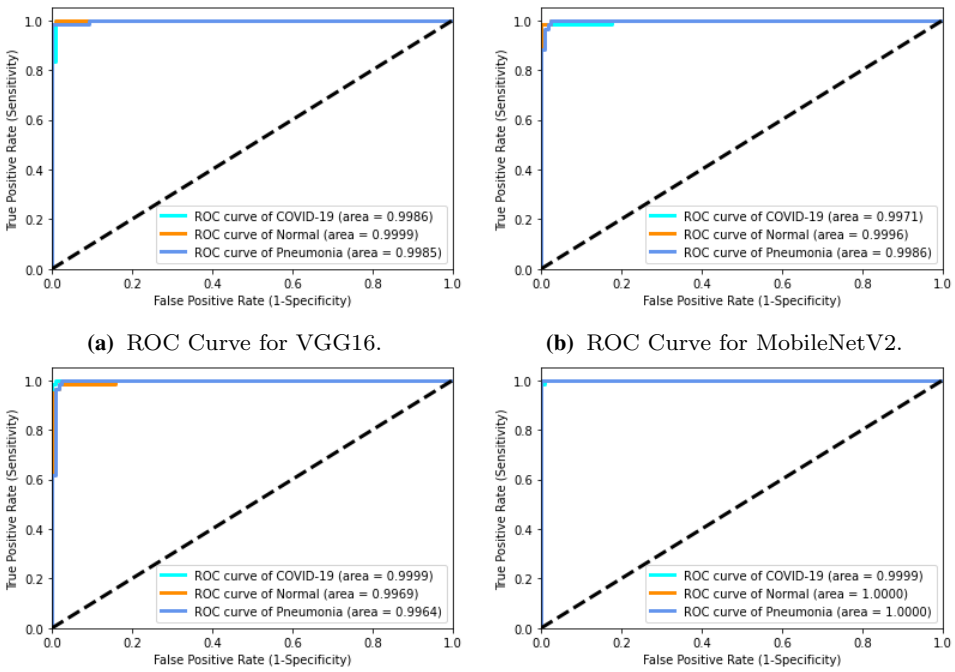
(a) Loss and accuracy plot for VGG16. **(b)** Loss and accuracy plot for MobileNetV2.

(c) Loss and accuracy plot for InceptionV3. **(d)** Loss and accuracy plot for Xception.

Fig. 7 Loss and accuracy plots of best models obtained from each architecture

some point during the training, all models achieved the highest validation accuracy of 1.0, except for the VGG16 model, which achieved a maximum validation accuracy of 0.95.

A Receiver-Operating Characteristic (ROC) curve shows the performance of the model at different thresholds. The Area Under the Curve (AUC) is used to summarize the findings of an ROC curve. It represents the ability of a model to distinguish among different classes. The ROC curves for the VGG16, MobileNetV2, InceptionV3, and Xception models are depicted in Fig. 8.



(a) ROC Curve for VGG16. **(b)** ROC Curve for MobileNetV2.

(c) ROC Curve for InceptionV3. **(d)** ROC Curve for Xception.

Fig. 8 Receiver-Operating Characteristic (ROC) Curve

From Fig. 8, it is observed that the Xception model has the highest AUC value for the normal and pneumonia class, which is 1. A large AUC value for a particular class signifies that the model is capable of discriminating that class from the other two classes very well. The AUC value of 1.0 for the normal class in the Xception model implies that the Xception model can discriminate normal samples from pneumonia and COVID-19 samples perfectly. And, the AUC value of 1.0 for pneumonia in the Xception model signifies that the Xception model can discriminate pneumonia samples from normal and COVID-19 samples perfectly. The AUC values for the normal and pneumonia classes for the VGG16, MobileNetV2, and InceptionV3 are less than 1. It means that the discrimination of each class (normal or pneumonia) from the other two classes is not perfect. In case of the COVID-19 class, the Xception and the InceptionV3 model have the highest AUC value, which is 0.9999. This means that even though the discrimination of COVID-19 class from the other two classes is not perfect for these two models, but it discriminates very well. With AUC values of 0.9986 and 0.9971, respectively, the VGG16 and the MobileNetV2 models do not discriminate as well between the COVID-19 class and the other two classes as the Xception and InceptionV3 models do.

Complexity Comparison

The comparison of the complexity of four optimized models along with two existing approaches is shown in Table 6. To compare the complexity among different models, we took into account training time, inference time, model size, number of parameters, Multiply-Accumulate (MAC) Operations, and Floating-Point Operations Per Second (FLOPS).

Training Time: MobileNetV2 took the most time to train - 5982.9s. InceptionV3 also took a similar amount of time (5815.9s). Xception took only 2255.0s which is the least.

Table 6 Comparison of complexity of different models

Model	Training time	Inference time	Model size	Number of parameters	MACs	FLOPS
VGG16	2882.3 s	26.8 ms	219.8 MB	18.31 M	30.7 G	15.4 G
MobileNetV2	5982.9 s	28.9 ms	32.1 MB	2.63 M	0.613 G	0.307 G
InceptionV3	5815.5 s	33.7 ms	335.1 MB	27.86 M	5.71 G	2.85 G
Xception	2255.0 s	26.8 ms	513.9 MB	42.8 M	16.8 G	8.41 G
CheXNet [38]	5532.1 s	40 ms	65.7 MB	28.1 M	5.7 G	2.85 G
VGG19 [49]	4907.3 s	27 ms	277.8 MB	23.14 M	39.0 G	19.5 G

Inference Time: Among the four trained models, the inference time for the VGG16 model and the Xception model is the lowest, which is 26.8 ms. MobileNetV2 and InceptionV3 took 28.9ms and 33.7ms, respectively.

Model Size: MobileNetV2 is the most lightweight model with a size of only 32.1 MB. VGG16 comes in the second with a size of 219.8 MB. InceptionV3 and Xception are 335.1 MB and 513.9 MB, respectively.

Number of Parameters: MobileNetV2 has the least number of parameters—2.63 M. The number of parameters in VGG16, InceptionV3, and Xception is 18.31 M, 27.86 M, and 42.8 M, respectively.

Multiply-Accumulate (MAC) Operations: VGG16 requires the highest MAC operations, whereas MobileNetV2 requires the least. The number of MAC operations for VGG16, MobileNetV2, InceptionV3, and Xception is 30.7 G, 0.613 G, 5.71 G, and 16.8 G, respectively.

Floating-Point Operations Per Second (FLOPS): VGG16 requires the highest FLOPS, whereas MobileNetV2 requires the least. The number of FLOPS for VGG16, MobileNetV2, InceptionV3, and Xception is 15.4 G, 0.307 G, 2.85 G, and 8.41 G, respectively

From Table 6, it is observed that, among the four models that we optimized using the Bayesian optimization technique, the Xception model is the most expensive in terms of model size and number of parameters, whereas the VGG16 model is the most expensive in terms of MACs and FLOPS. The MobileNetV2 model is the least expensive in terms of model size, number of parameters, MACs, and FLOPS. Even though MobileNetV2 is the least expensive model, from Table 5, it is observed that the MobileNetV2 is also the most underperforming model, whereas the VGG16 and the Xception models are the best-performing ones. Hence, a clear trade-off between the complexity of the models and their performance are observed. It is possible to design lightweight models but at the expense of performance. In the healthcare sector, there is no room for error. Hence, deploying a larger model which performs better is an obvious choice.

The complexity of the four optimized models were also compared with two approaches from the literature. The CheXNet [38] and the VGG19 [49] models from these literatures were trained on our dataset. The VGG19 model [49] is more expensive in terms of MACs and FLOPS compared to our models, and the inference time for the CheXNet model [38] is very high. From Table 10, it is observed that the Xception model outperforms these two models in terms of accuracy, precision, recall, specificity, F1-score, and Matthews correlation coefficient.

COVIDXception-Net: Best Model Obtained Using Bayesian Search

The Xception architecture tuned using Bayesian optimization technique performed the best. We named this architecture COVIDXception-Net. The backbone of the COVIDXception-Net architecture is the Xception network, which is used to extract features from the images. The feature extraction network is followed by dense fully connected layers to perform classification. The architecture of COVIDXception-Net is demonstrated in Fig. 9. The architecture takes as input an X-ray image of size 299×299. The input layer is followed by two convolution layers with ReLU activation function. After the first two convolution layers, three convolution layer blocks—*ConvA*, *ConvB*, and *ConvC*—are linearly stacked one after another. Instead of using regular convolution operations, these layers mostly use depthwise separable convolution operations. This is the salient feature of the Xception architecture. The advantage of using depthwise separable convolution over regular convolution is they require much fewer parameters. Due to the use of fewer parameters, they are also less prone to overfitting and are faster.

In block *ConvA*, there are two depthwise separable convolution layers. Batch normalization is used after the first layer. The output of the second depthwise separable convolution layer enters into a max-pooling layer. A skip connection with a convolution layer is also used in this block. The output obtained from the max-pooling layer and the $Conv1 \times 1$ layer is added. Three *ConvA* layers are stacked one after another. The output from the third *ConvA* block enters into the *ConvB* block.

In *ConvB* block, three depthwise separable convolution layers with batch normalization are used. The first two depthwise separable convolution layers used ReLU non-linearity, but the third one did not use any non-linearity. The output of the third depthwise separable convolution layer is added with the input of the *ConvB* block using a skip connection. Eight *ConvB* are linearly stacked one after another. The output of the eighths *ConvB* block enters into the *ConvC* block.

The input of the *ConvC* block passes through two depthwise separable convolution layers, followed by a max-pooling layer. The output from the max-pooling layer is added to the convoluted input of the *ConvC* block using a skip connection. The output obtained after performing the addition operation then passes through two depthwise separable convolution layers.

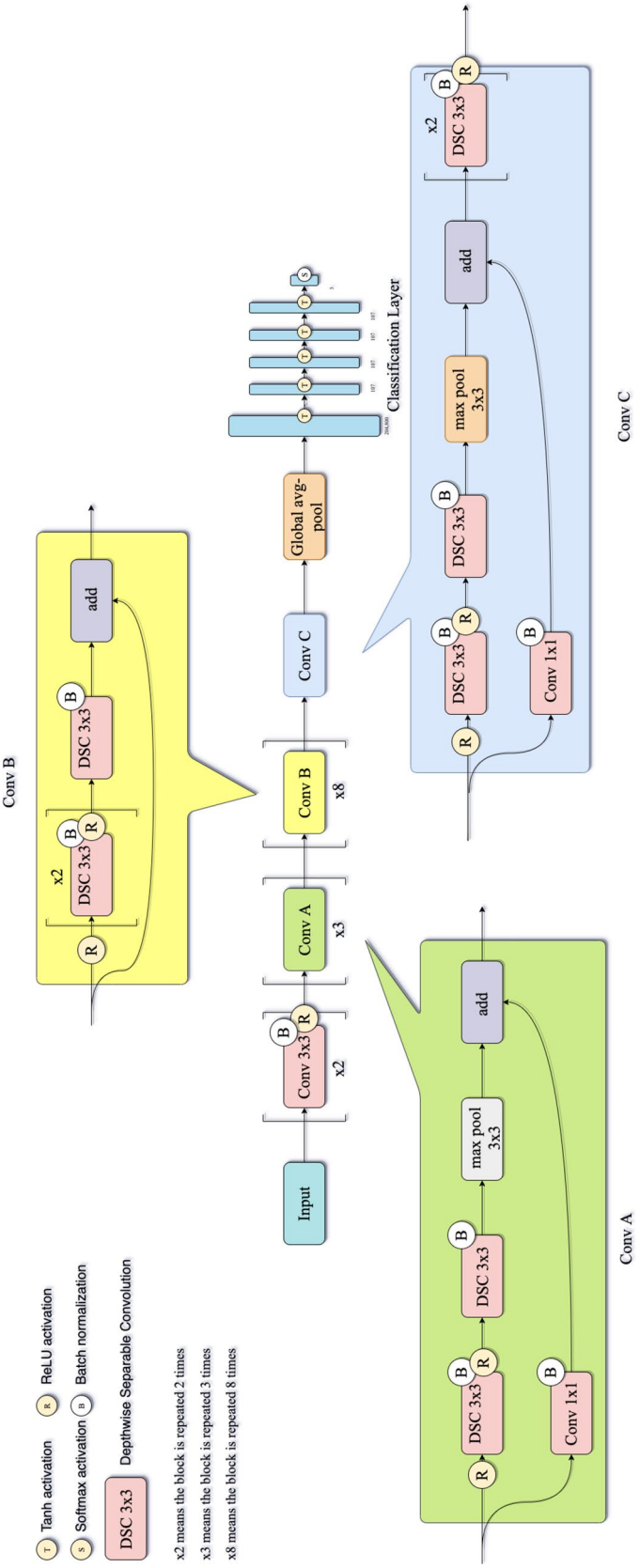


Fig. 9 Proposed COVIDXception-Net architecture

The output of *ConvC* enters into the global average pooling layer. With the global average pooling layer, the feature extraction part of the network ends. The output obtained from the global average pooling layer is flattened and enters into the classification layers of the COVIDXception-Net.

The classification layer of the COVIDXception-Net architecture consists of four dense layers. Each dense layer consists of 107 neurons with *tanh* activation function. The final layer of the COVIDXception-Net architecture consists of three neurons with softmax non-linearity representing the three classes—normal, pneumonia, and COVID-19.

The classification performance for the COVIDXception-Net is presented in Table 7.

Both normal and pneumonia classes were classified with an accuracy of 1.0. However, one COVID-19 sample was misclassified as normal. Hence, the accuracy was reduced to 0.983. Precision and recall for the COVID-19 class were 1.0 and 0.983, respectively. F1-score for pneumonia class was 1.0, which means that both precision and recall for pneumonia were 1.0. Specificity for the COVID-19 class is 1 as all the non-COVID-19 classes were predicted correctly. The COVIDXception-Net model also achieved a high overall Matthews correlation coefficient. The overall classification accuracy, precision, recall, specificity, F1-score, and Matthews correlation coefficient for the Xception model were 0.994, 0.995, 0.994, 0.997, 0.994, and 0.992, respectively.

Ablation Study

In this section, we discuss two ablation studies that we performed to evaluate the impact of Bayesian optimization and weighted loss function on our model.

Impact of Bayesian Search

To evaluate the impact of optimization using Bayesian search, we compared it with random search.

Figure 10 presents a convergence plot of Xception architecture tuned using both Bayesian and random search. The final converged validation loss for Bayesian search was 0.0006, which was lower than the final converged validation loss for random search, 0.0641791.

Table 8 presents the classification performance of Xception architecture tuned using both Bayesian and random search. Xception architecture tuned using Bayesian search outperformed the same architecture tuned using random search in terms of accuracy, precision, recall, specificity, F1-score, and Matthews correlation coefficient.

Both Fig. 10 and the Table 8 suggest that Bayesian optimization was far better compared to random guessing of hyperparameters.

Impact of Weighted Loss Function

The Xception architecture was trained without using a weighted loss function to see how it affected the performance. From Table 9, it is observed that all accuracy, precision, recall, specificity, F1-score, and Matthews correlation coefficient dropped when the weighted loss function was not used. It proves that the weighted loss function helps in the generalization of the neural network.

Qualitative Analysis

To understand the decision making process of our COVIDXception-Net model, we implemented class activation

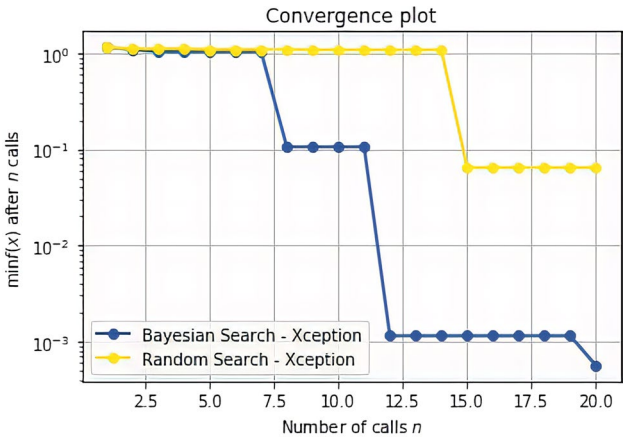


Fig. 10 Convergence plot for Bayesian and random search on Xception architecture

Table 7 Classification performance for the COVIDXception-Net model

Class	ACC	PRE	REC	SPE	F1	MCC
COVID	0.983	1.000	0.983	1	0.992	0.988
Normal	1.000	0.984	1.000	0.992	0.992	0.988
Pneumonia	1.000	1.000	1.000	1	1.000	1
Overall	0.994	0.995	0.994	0.997	0.994	0.992

'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient

Table 8 Classification performance for Bayesian and random search

Method	ACC	PRE	REC	SPE	F1	MCC
Bayesian search	**0.994**	**0.995**	**0.994**	**0.997**	**0.994**	**0.992**
Random search	0.95	0.952	0.95	0.975	0.95	0.926

'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient

Table 9 Classification performance of Xception model with and without weighted loss

Method	ACC	PRE	REC	SPE	F1	MCC
Xception without weighted loss	0.983	0.984	0.983	0.991	0.983	0.975
Xception with weighted loss	**0.994**	**0.995**	**0.994**	**0.997**	**0.994**	**0.992**

'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient

mapping using the Gradient-weighted Class Activation Mapping (Grad-CAM) [64] technique. Class activation mapping provides a way of visually understanding the rationale behind making a certain prediction. It also helps to identify whether the model is leveraging the right features to make predictions or it is using erroneous features which is leading to the right decision.

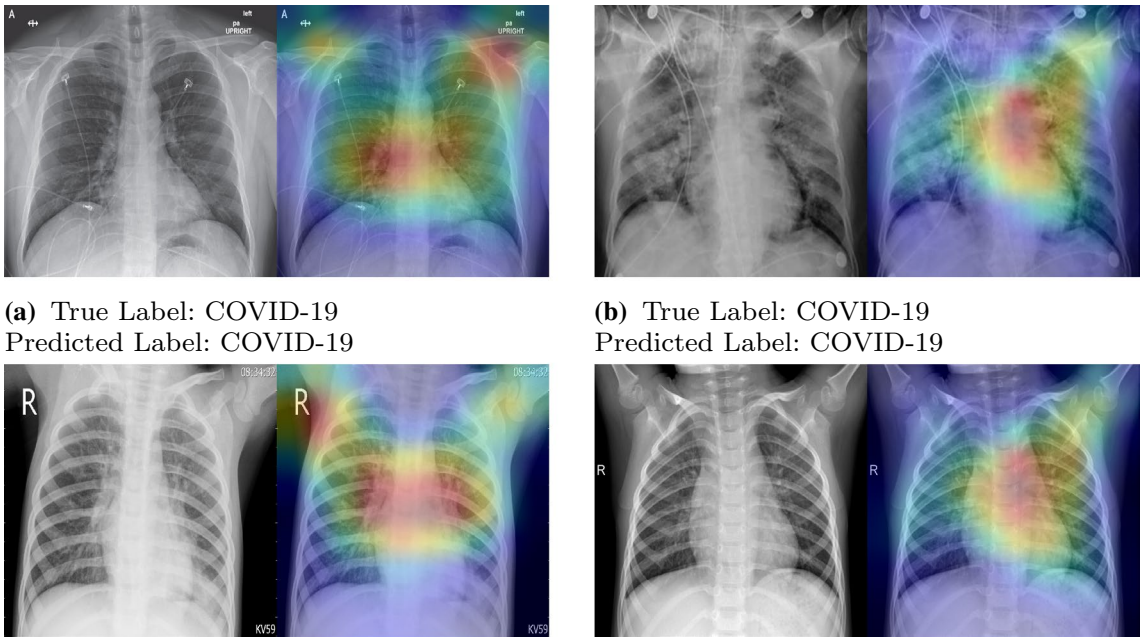
To generate a class activation map, an X-ray sample is forward propagated through the COVIDXception-Net model. Then, a heatmap is generated from the gradient information flowing into the final convolutional layer of the model for that X-ray sample. The heatmaps represent the regions responsible for making a certain prediction by the COVIDXception-Net model. The generated heatmap is then superimposed over the original X-ray image to identify the important regions for a certain prediction.

Class activation maps generated for some sample test images from the dataset along with predictions are shown in Fig. 11. The heatmaps in the figure are indicated by the red and yellow areas, with red indicating more significant and yellow indicating slightly less significant regions of the X-ray responsible for making the prediction. From Fig. 11, it is observed that the heatmaps generated from the COVIDXception-Net architecture point out the salient regions of the X-ray, i.e., the region surrounding the lungs, to make decisions rather than relying on erroneous visual indicators or imaging artifacts, etc.

Comparison with the State-of-the-Art Methods

The problem with the comparison of works related to COVID is that most studies used different datasets and the split of the dataset into train, validation, and test sets are not publicly available. For this reason, to compare with our results, we trained and tested other studies approaches using our dataset. CheXNet model of Chowdhury et al. [38] was trained and tested on our dataset, and it obtained an accuracy of 98.3% on the test set. VGG19 model of Ioannis et al. [49] was also trained and tested on our dataset, and it obtained an accuracy of 96.1%. Our proposed model COVIDXception-Net achieved an accuracy of 99.4%. The comparison is presented in Table 10.

Comparison of our model with state-of-the-art methods on similar datasets is presented in Table 11. The nature of the dataset was similar in all cases, but the distribution of data and the evaluation protocols were different. Some studies used cross-validation, whereas some split the entire dataset into train, validation, and test sets. The datasets containing the three classes had normal, pneumonia, and COVID-19 X-rays. The datasets containing the two classes had normal and COVID-19 X-rays. Li and Zhu [52] developed a DenseNet-based model to classify three classes. They used a very small dataset of 377 X-rays. They obtained an overall accuracy of 0.889 on the test set. Wang and Wong [50] used a tailored CNN. They used a large dataset of 13,962 X-ray images. The tailored CNN model obtained overall accuracy 0.923, precision 0.913, and recall 0.887. Afshar et al. [51] proposed a capsule network-based binary classification model. Their model achieved an accuracy of 0.957 and recall of 0.90. Farooq and Hafeez [37] developed an ResNet50-based model to detect four classes—normal, COVID-19, viral pneumonia, and bacterial pneumonia. Using data augmentation and transfer learning, they achieved an overall accuracy 0.962, precision 0.969 and recall 0.969 and F1-score 0.969. Chowdhury et al. [38] tried different CNN architectures, and used both transfer learning and data augmentation to improve the performance of the model. Using DenseNet architecture, they obtained an overall accuracy of 0.979, precision 0.979, recall 0.979, and F1-score 0.979. Ucar and Korkmaz [39] used Bayes-SqueezeNet. To improve



(a) True Label: COVID-19
Predicted Label: COVID-19

(b) True Label: COVID-19
Predicted Label: COVID-19

(c) True Label: Pneumonia
Predicted Label: Pneumonia

(d) True Label: Normal
Predicted Label: Normal

Fig. 11 Class activation map generated for different X-ray samples with COVIDXception-Net architecture

Table 10 Comparison between the proposed model with previous state-of-the-art methods on the same dataset

Study	Class	ACC	PRE	REC	SPE	F1	MCC
[38]	3	0.983	0.983	0.983	0.992	0.983	0.975
[49]	3	0.961	0.961	0.961	0.981	0.961	0.942
COVIDXception-Net	3	**0.994**	**0.995**	**0.994**	**0.997**	**0.994**	**0.992**

'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient

Table 11 Comparison between the proposed model with previous state-of-the-art methods on similar datasets

Study	Method	Class	ACC	PRE	REC	SPE	F1	MCC
[52]	DenseNet	3	0.889	–	–	–	–	–
[50]	Tailored CNN	3	0.923	0.913	0.887	–	0.900	–
[51]	Capsule network	2	0.957	–	0.900	0.958	–	–
[37]	ResNet50	4	0.962	0.969	0.969	–	0.969	-
[38]	DenseNet	3	0.979	0.979	0.979	0.990	0.979	–
[39]	SqueezeNet	3	0.983	0.983	0.983	0.991	0.983	0.974
[65]	ResNet50	2	0.980	**1.000**	0.960	–	0.980	–
COVIDXception-Net	Xception	3	**0.994**	0.995	**0.994**	**0.997**	**0.994**	**0.992**

'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient. '-' indicates 'not mentioned'

the performance and faster learning of the model, they used both data augmentation and transfer learning. Their Bayes-SqueezeNet model achieved 0.983 overall accuracy, 0.983 precision, 0.983 recall, 0.991 specificity, 0.983 F1-score, and 0.974 Matthews correlation coefficient. Narin et al. [65] trained a ResNet50 model that can distinguish between

normal and COVID X-ray. They used a small dataset of 100 X-ray images. Their model achieved an accuracy of 0.98, precision 1.000, recall 0.96, and F1-score 0.980. We used a dataset of 2,906 X-ray images. We did not use any augmentation on data. Our model achieved 0.994 overall accuracy, 0.995 precision, 0.994 recall, 0.997 specificity, 0.994 F1-score, and 0.992 Matthews correlation coefficient. Compared to the previous results, our proposed model COVIDXception-Net achieved the highest overall accuracy, recall, specificity, F1-score, and Matthews correlation coefficient.

Comparison of our model with state-of-the-art methods for COVID-19 class is presented in Table 12. Wang and Wong [50] achieved COVID-19 class accuracy of 0.933, precision 0.989, and recall 0.910 using their proposed model COVID-Net. Farooq and Hafeez [37] achieved COVID-19 class accuracy, precision recall, and F1-score of 1.000 using their proposed model COVID-ResNet, but their test set contained only 8 COVID-19 images which is very small to meaningfully represent the COVID-19 cases that may arise in real world. Ucar and Korkmaz [39] also achieved a COVID-19 class accuracy of 1.000, but their test set was augmented. The raw test set contained only 10 COVID-19 images which is also very small to meaningfully represent the entire population. We used 60 non-augmented COVID-19 images in our test set out of which only 1 image was misclassified as normal, and hence, we obtained COVID-19 class accuracy of 0.983. Our model also achieved a precision and specificity of 1 and overall accuracy of 0.994 which is the highest to date.

Discussion

In this study, we developed an Xception Network-based architecture referred to as the COVIDXception-Net for diagnosis of COVID-19 disease from CXR images. We applied the Bayesian optimization technique on four pre-trained architectures—VGG16, MobileNetV2, InceptionV3, and Xception. The results were compared using convergence plot, accuracy, precision, recall, specificity, F1-score, and Matthews correlation coefficient. We found that the model based on Xception architecture performed best. To tackle the class imbalance problem, a weighted loss function was used. COVIDXception-Net achieved a satisfying performance on the test set with an accuracy of 99.4%. The impact of each hyperparameter that we tuned on the COVIDXception-Net was analyzed using multiple partial dependence plots. We compared our model COVIDXception-Net with previous state-of-the-art methods, and found that our model outperforms previous methods in terms of accuracy, precision, recall, and F1-score on our dataset. When the dataset is similar in nature but not the same, our model still outperforms previous state-of-the-art methods in terms of accuracy, recall, and F1-score.

Our proposed model, COVIDXception-Net misclassified one COVID-19 CXR as normal. The misclassified case is shown in Fig. 12. The ground truth label of this CXR was COVID-19 in our dataset. We asked five separate doctors to diagnose this CXR and found that the case was a bit confusing for them as well. Two doctors said that the patient was infected with COVID-19, but the infection was at an early stage. One of the doctors said the patient was not infected with COVID-19, but he might have other lung conditions. Another doctor said the patient was in normal condition. The last doctor refused to diagnose, saying he could not diagnose without knowing the patient history. The rest of the COVID-19 chest X-rays were classified accurately by our COVIDXception-Net model.

Fig. 12 Misclassified COVID-19 case

Table 12 Comparison between proposed model with previous state-of-the-art methods for COVID-19 class

Study	Method	ACC	PRE	REC	SPE	F1	MCC	Overall Accuracy
[50]	Tailored CNN	0.933	0.989	0.910	–	–	–	0.933
[37]	ResNet50	**1.000**	**1.000**	**1.000**	–	**1.000**	-	0.962
[39]	SqueezeNet	**1.000**	0.994	**1.000**	0.997	0.997	**0.995**	0.983
COVIDXception-Net	Xception	0.983	**1.000**	0.983	**1.000**	0.992	0.988	**0.994**

'ACC' refers to accuracy. 'PRE' refers to precision or correctness. 'REC' refers to recall or completeness. 'SPE' refers to specificity or true-negative rate. 'F1' refers to F1-score. 'MCC' refers to Matthews Correlation Coefficient. '-' indicates 'not mentioned'

Two ablation studies were performed to measure the impact of Bayesian optimization and weighted loss function on our model. In the first ablation study, the impact of Bayesian optimization was measured by comparing it with random search. To compare both the techniques convergence plot, accuracy, precision, recall, and F1-score were used. It was found that the best model obtained from the Bayesian search provided an accuracy of 99.4%, whereas the best model obtained from random search provided an accuracy of 95%. Bayesian optimization significantly improved the performance of the final model also in terms of precision, recall, and F1-score. In the second ablation study, COVIDXception-Net was trained using a regular loss function instead of a weighted loss function. An accuracy of 98.3% was obtained when regular loss function was used compared to 99.4% when weighted loss function was used. From these two ablation studies, we concluded that Bayesian search can be used to efficiently tune the hyperparameters of a neural network, whereas weighted loss can be used to improve the performance of our neural network.

The complexities of the optimized models were calculated in terms of training time, inference time, model size, number of parameters, MACs, and FLOPS. Even though the COVIDXception-Net model is a bit expensive in terms of model size and number of parameters, it can perform inference much faster, and it is the best-performing model in terms of accuracy, precision, recall, specificity, F1-score, and Matthew correlation coefficient. When it comes to healthcare, making a mistake can have serious consequences. Therefore, it makes more sense to deploy the best-performing model than to deploy a lightweight model like MobileNet that is computationally cheaper. Gradient-weighted Class Activation Mapping (Grad-CAM) was used to perform qualitative analysis of the COVIDXception-Net model. By analyzing class activation maps for different X-ray samples, it was observed that the model was using important regions of the X-ray, i.e., regions surrounding the lungs to make predictions.

Conclusion

In this study, we introduced COVIDXception-Net, a deep convolutional neural network that can diagnose COVID-19 from chest X-ray images without any human intervention. We improved the Xception architecture using the Bayesian optimization technique to diagnose COVID-19 accurately. To tackle the class imbalance problem, the weighted loss function was used. The complexities of the optimized models are compared in terms of training time, inference time, model size, number of parameters, MACs, and FLOPS. COVIDXception-Net achieved a satisfying performance on the test set and outperformed state-of-the-art methods in overall accuracy, recall, and F1-score. We also performed two ablation studies to measure the impact of Bayesian optimization and weighted loss function on our model, and found that these techniques significantly improved the performance of the model. The qualitative analysis of COVIDXception-Net was done using the GRAD-CAM technique. The qualitative analysis suggest the model leveraging the salient regions from the X-ray to make predictions. A future research direction may include developing a model that can classify Chest X-rays based on mild, moderate, and severe symptoms of COVID-19 which will be helpful for the management of patients in a clinical setting.

Acknowledgements The authors would like to thank M. E. H. Chowdhury and his team for compilation of the dataset. They would also like to thank J. P. Cohen and Italian Society of Medical and Interventional Radiology (SIRM) for the COVID-19 CXR images.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Weiss SR, Leibowitz JL. Coronavirus pathogenesis. In: Advances in virus research, volume 81, Elsevier; 2011. p. 85–164.
2. Kooraki S, Hosseiny M, Myers L, Gholamrezanezhad A. Coronavirus (covid-19) outbreak: what the department of radiology should know. Journal Am College Radiol. 2020.
3. Baud D, Qi X, Nielsen-SK, Musso D, Pomar Léo, Favre Guillaume. Real estimates of mortality following covid-19 infection. Lancet Infectious Dis. 2020.
4. Cucinotta D, Vanelli M. Who declares covid-19 a pandemic. Acta bio-medica. 2020;91(1):157–60.
5. Jandrić P. Postdigital research in the time of covid-19. Postdigital Sci Educ 2020; 1–6.
6. Nuno F. Economic effects of coronavirus outbreak (covid-19) on the world economy. *Available at SSRN 3557504*, 2020.
7. Oran DP, Topol EJ. Prevalence of asymptomatic sars-cov-2 infection: a narrative review. Ann Internal Med. 2020.
8. Sheikh Aziz, Sheikh Asiyah, Sheikh Zakariya, Dhami Sangeeta, Sridhar Devi. What's the way out? potential exit strategies from the covid-19 lockdown. J Global Health. 2020;10(1).
9. Wang Wenling, Yanli Xu, Gao Ruqin, Roujian Lu, Han Kai, Guizhen Wu, Tan Wenjie. Detection of sars-cov-2 in different types of clinical specimens. Jama. 2020;323(18):1843–4.
10. Fang Y, Zhang H, Xie J, Lin M, Ying L, Pang P, Ji W. Sensitivity of chest ct for covid-19: comparison to rt-pcr. Radiology. 2020; 200432.
11. Li Y, Yao L, Li J, Chen L, Song Y, Cai Z, Yang C. Stability issues of rt-pcr testing of sars-cov-2 for hospitalized patients clinically diagnosed with covid-19. J Med Virol. 2020.
12. Feng H, Liu Y, Lv M, Zhong J. A case report of covid-19 with false negative rt-pcr test: necessity of chest ct. Jpn J Radiol. 2020:1–2.
13. Liu J, Hui Y, Zhang S. The indispensable role of chest ct in the detection of coronavirus disease 2019 (covid-19). Euro J Nucl Med Mol Imaging. 2020.

14. Xu X, Jiang X, Ma C, Du P, Li X, Lv S, Yu L, Chen Y, Su J, Lang G, et al. Deep learning system to screen coronavirus disease 2019 pneumonia. arxiv 2020. *arXiv preprint* arXiv:2002.09334, 2020.
15. Li W, Cui H, Li K, Fang Y, Li S. Chest computed tomography in children with covid-19 respiratory infection. Pediatric radiology. 2020:1–4.
16. Chung Ml, Bernheim A, Mei X, Zhang Ning, Huang Mingqian, Zeng Xianjun, Cui Jiufa, Wenjian Xu, Yang Yang, Fayad Zahi A, et al. Ct imaging features of 2019 novel coronavirus (2019-ncov). Radiology. 2020;295(1):202–7.
17. David V, Ruggiero M, Choi WS, Masri D, Flyer M, Shyknevsky I, Stein EG. Three unsuspected ct diagnoses of covid-19. Emergency Radiol. 2020:1–4.
18. Xie X, Zhong Z, Zhao W, Zheng C, Wang F, Liu J. Chest ct for typical 2019-ncov pneumonia: relationship to negative rt-pcr testing. Radiology. 2020:200343.
19. American College of Radiology et al. Acr recommendations for the use of chest radiography and computed tomography (ct) for suspected covid-19 infection. march 11, 2020, 2020.
20. Jacobi A, Chung M, Bernheim A, Eber C.. Portable chest x-ray in coronavirus disease-19 (covid-19): A pictorial review. Clinical Imaging. 2020.
21. European Society of Radiology (ESR, American College of Radiology, et al. European society of radiology (esr) and american college of radiology (acr) report of the 2015 global summit on radiological quality and safety. Insights into Imaging. 2016;7(4):481–484.
22. Kim H. Outbreak of novel coronavirus (covid-19): What is the role of radiologists?; 2020.
23. LeCun Yann, Bengio Yoshua, Hinton Geoffrey. Deep learning. Nature. 2015;521(7553):436–44.
24. Mahmud M Kaiser MS, Hussain A. Deep learning in mining biological data. *arXiv preprint* arXiv:2003.00108, 2020.
25. Mahmud Mufti, Kaiser Mohammed Shamim, Hussain Amir, Vassanelli Stefano. Applications of deep learning and reinforcement learning to biological data. IEEE Trans Neural Netw Learn Syst. 2018;29(6):2063–79.
26. Noor MBT, Zenia NZ, Kaiser MS, Mahmud M, Al Mamun S. Detecting neurodegenerative disease from mri: A brief review on a deep learning perspective. In: International Conference on Brain Informatics. Springer;2019. p. 115–125.
27. Ali Hafsa M, Kaiser MS, Mahmud M. Application of convolutional neural network in segmenting brain regions from mri data. In: International Conference on Brain Informatics. Springer; 2019. p. 136–146.
28. Shen L, Shi J, Dong Y, Ying S, Yaxin PL, Chen QZ, An H, Zhang Y. An improved deep polynomial network algorithm for transcranial sonography-based diagnosis of parkinson's disease. Cognitive Comput 2019:1–10.
29. Rajpurkar P, Irvin J, Zhu K, Yang B, Mehta H, Duan T, Ding D, Bagul A, Langlotz C, Shpanskaya K, et al. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint* arXiv:1711.05225, 2017.
30. Song QZ, Zhao L, Luo X, Dou XC. Using deep learning for classification of lung nodules on computed tomography images. J Healthcare Eng. 2017;2017.
31. Bergstra J, Bengio Y. Random search for hyper-parameter optimization. J Mach Learn Res. 2012;13(2).
32. Jia W, Chen X-Y, Zhang H, Xiong Li-Dong, Lei Hang, Deng Si-Hao. Hyperparameter optimization for machine learning models based on bayesian optimization. J Electron Sci Technol. 2019;17(1):26–40.
33. Archetti Francesco, Candelieri Antonio. Bayesian Optimization and Data Science. Springer; 2019.
34. Snoek J, Larochelle H, Adams RP. Practical bayesian optimization of machine learning algorithms. In: Adva Neural Inform Process Syst. 2012. p. 2951–2959.
35. Shahriari B, Swersky K, Wang Z, Adams RP, De Freitas Nando. Taking the human out of the loop: A review of bayesian optimization. Proc IEEE. 2015;104(1):148–75.
36. Rahman S, Sarker S, Al Miraj Mb, Nihal RA, Haque AKMN, Al Noman A. Deep learning–driven automated detection of covid-19 from radiography images: a comparative analysis. Cognitive Computation. 2021;1–30.
37. Farooq M, Hafeez A. Covid-resnet: A deep learning framework for screening of covid19 from radiographs. *arXiv preprint* arXiv:2003.14395, 2020.
38. Chowdhury MEH, Rahman T, Khandakar A, Mazhar R, Kadir MA, Mahbub Z B, Islam KR, Khan MS, Iqbal Atif, A-EN et al. Can ai help in screening viral and covid-19 pneumonia? *arXiv preprint*arXiv:2003.13145*, 2020.*
39. Ucar F, Korkmaz D. Covidiagnosis-net: Deep bayes-squeezenet based diagnostic of the coronavirus disease 2019 (covid-19) from x-ray images. Med Hypotheses. 2020;109761.
40. Rajaraman S, Antani S. Training deep learning algorithms with weakly labeled pneumonia chest x-ray data for covid-19 detection. *medRxiv*, 2020.
41. Waheed Abdul, Goyal Muskan, Gupta Deepak, Khanna Ashish, Al-Turjman Fadi, Pinheiro Plácido Rogerio. Covidgan: Data augmentation using auxiliary classifier gan for improved covid-19 detection. IEEE Access. 2020;8:91916–23.
42. Khan AI, Shah JL, Bhat MM. Coronet: A deep neural network for detection and diagnosis of covid-19 from chest x-ray images. Comput Methods Programs Biomed. 2020;196:105581.
43. Pereira RM, Bertolini D, Teixeira LO, Silla CN Jr, Costa Yandre MG. Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios. Comput Methods Programs Biomedi. 2020;194:105532.
44. Misra S, Jeon S, Lee S, Managuli Ravi, Jang In-Su, Kim Chulhong. Multi-channel transfer learning of chest x-ray images for screening of covid-19. Electronics. 2020;9(9):1388.
45. Punn Narinder Singh, Agarwal Sonali. Automated diagnosis of covid-19 with limited posteroanterior chest x-ray images using fine-tuned deep neural networks. Appl Intell. 2021;51(5):2689–702.
46. Ouyang X, Huo J, Xia L, Shan F, Liu Jun, Mo Zhanhao, Yan Fuhua, Ding Zhongxiang, Yang Qi, Song Bin, et al. Dual-sampling attention network for diagnosis of covid-19 from community acquired pneumonia. IEEE Trans Med Imaging. 2020;39(8):2595–605.
47. Chollet F. Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. p. 1251–1258.
48. Hosseiny M, Kooraki S, Gholamrezanezhad A, Reddy S, Myers Lee. Radiology perspective of coronavirus disease 2019 (covid-19): lessons from severe acute respiratory syndrome and middle east respiratory syndrome. Am J Roentgenol. 2020;214(5):1078–82.
49. Apostolopoulos I, Tzani M. Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks. Australasian physical & engineering sciences in medicine / supported by the Australasian College of Physical Scientists in Medicine and the Australasian Association of Physical Sciences in Medicine. 2020;43:03.
50. Wang L, Wong A. Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. *arXiv preprint* arXiv:2003.09871, 2020.
51. Afshar P, Heidarian S, Naderkhani F, Oikonomou A, Plataniotis KN, Mohammadi A. Covid-caps: A capsule network-based

framework for identification of covid-19 cases from x-ray images. arXiv preprint arXiv:2004.02696, 2020.
52. Li X, Zhu D. Covid-xpert: An ai powered population screening of covid-19 cases using chest radiography images. *arXiv preprint* arXiv:2004.03042, 2020.
53. Goodwin BD, Jaskolski C, Zhong C, Asmani H. Intra-model variability in covid-19 classification using chest x-ray images. *arXiv preprint* arXiv:2005.02167, 2020.
54. Zhu X, Song B, Shi F, Chen Y, Rongyao Hu, Gan Jiangzhang, Zhang Wenhai, Li Man, Wang Liye, Gao Yaozong, et al. Joint prediction and time estimation of covid-19 developing severe symptoms using chest ct scan. Med Image Anal. 2021;67:101824.
55. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint* arXiv:1409.1556, 2014.
56. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2018. p. 4510–4520.
57. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2818–2826.
58. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-FL. Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. Ieee; 2009. p. 248–255.
59. King G, Zeng L. Logistic regression in rare events data. Political Anal. 2001;9(2):137–63.
60. SIRM (2020). *COVID-19 Database*, (last accessed July 3, 2020). https://www.sirm.org/category/senza-categoria/covid-19/.
61. Cohen JP, Morrison P, Dao L. Covid-19 image data collection. arXiv:2003.11597, 2020.
62. Kermany D, Zhang K, Goldbaum MU. Labeled optical coherence tomography (oct) and chest x-ray images for classification. Mendeley data. 2018;2.
63. Kingma DP, Ba J. Adam: A method for stochastic optimization. *arXiv preprint* arXiv:1412.6980, 2014.
64. 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. 618–626.
65. Narin A, Kaya C, Pamuk Z. Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks. *arXiv preprint* arXiv:2003.10849, 2020.