Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Biomedical Signal Processing and Control 78 (2022) 103848

Contents lists available at ScienceDirect

Biomedical Signal Processing and Control

journal homepage: www.elsevier.com/locate/bspc

CoviXNet: A novel and efficient deep learning model for detection of COVID-19 using chest X-Ray images

Gaurav Srivastava, Aninditaa Chauhan[1], Mahesh Jangid *, Sandeep Chaurasia

Department of Computer Science and Engineering, Manipal University Jaipur, 303007, Rajasthan, India

ARTICLE INFO

Keywords:
COVID-19
Chest X-Ray images
Deep transfer learning
Convolutional neural networks
CoviXNet

ABSTRACT

The Coronavirus (COVID-19) pandemic has created havoc on humanity by causing millions of deaths and adverse physical and mental health effects. To prepare humankind for the fast and efficient detection of the virus and its variants shortly, COVID-19 detection using Artificial Intelligence and Computer-Aided Diagnosis has been the subject of several studies. To detect COVID-19, there are numerous publicly accessible datasets of Chest X-rays that the researchers have combined to solve the problem of inadequate data. The cause for concern here is that in combining two or more datasets, some of the images might be duplicates, so a curated dataset has been used in this study, taken from an author's paper. This dataset consists of 1281 COVID-19, 3270 Normal X-rays, and 1656 viral-pneumonia infected Chest X-ray images. Dataset has been pre-processed and divided carefully to ensure that there are no duplicate images. A comparative study on many traditional pre-trained models was performed, analyzing top-performing models. Fine-tuned InceptionV3, Modified EfficientNet B0&B1 produced an accuracy of 99.78% on binary classification, i.e., covid-19 infected and normal Chest X-ray image. ResNetV2 had a classification accuracy of 97.90% for 3-class classification i.e., covid-19 infected, normal, and pneumonia. Furthermore, a trailblazing custom CNN-based model, CoviXNet, has been proposed consisting of 15 layers that take efficiency into account. The proposed model CoviXNet exhibited a 10-fold accuracy of 99.47% on binary classification and 96.61% on 3-class. CoviXNet has shown phenomenal performance with exceptional accuracy and minimum computational cost. We anticipate that this comparative study, along with the proposed model CoviXNet, can assist medical centers with the efficient real-life detection of Coronavirus.

1. Introduction

The Covid-19 pandemic has impacted individuals all over the world, with more than 350 million cases infected and more than 5 million fatalities to far [1]. This also creates a tremendous burden on the medical systems when there is a sudden influx of infected patients, making basic healthcare facilities inaccessible to the needy. With its new variants and waves taking over different nations causing wreckage every few months, we need a guaranteed and effective method for quick testing to detect the Coronavirus presence in the lungs. Efficient testing will help identify the infected people rapidly and allow for easy tracking and isolation of these persons to avoid more damage.

In the initial days, testing for Covid-19 was done using RT-PCR and antibody test only. These have various disadvantages, like the shortage of testing kits in times of need. RT-PCR has shown false negatives in many cases. An antibody test can be done only after a certain period of already being infected and is not a safe option [2]. CT scans and X-rays are being used instead for this process [3]. The rays emitted in CT scans are extremely harmful to the health and much more dangerous than those of X-rays [4]. This is why Chest X-rays, in comparison, are a lot more reliable, safe, and efficient.

Deep learning is a subset of machine learning that exclusively works with neural networks (inspired by the working and functioning of neurons in our brain) [5,6]. Deep learning approaches have gained popularity in recent years as a consequence of their capacity to learn without assistance and develop highly efficient solutions that have never been seen before [7]. As a result, deep learning is being used in various industries and areas of academics.

In medical imaging especially, much relevant research has been done already in predicting pneumonia using Chest X-ray (CXR); also in analyzing CT scans and MRI images [8–10]. As a result, deep learning is the most sought-after research area for detecting Covid-19 in chest

* Corresponding author.
E-mail addresses: mailto.gaurav2001@gmail.com (G. Srivastava), aninditaa2001@gmail.com (A. Chauhan), mahesh_seelak@yahoo.co.in (M. Jangid), sandeep.chaurasia@jaipur.manipal.edu (S. Chaurasia).
[1] The author Aninditaa Chauhan has contributed equally to the first author.

https://doi.org/10.1016/j.bspc.2022.103848
Received 24 February 2022; Received in revised form 19 May 2022; Accepted 27 May 2022
Available online 8 June 2022

X-rays. The testing of Covid-19 by X-rays images does not require any new machinery and is efficient, accurate, and feasible.

Viral Pneumonia has also shown symptoms and effects similar to cases of Covid-19 infected lungs. Due to this, there is often some parallelism between Viral Pneumonia infected lung images and Coronavirus infected lung images. We have considered this and included CXR images of Viral Pneumonia in the dataset to accurately predict the presence of Coronavirus and distinguish it from normal and Pneumonia infected Chest X-ray images.

As of now, the complex patterns of Chest X-ray images can be understood and analyzed only by specialty radiologists [11]. Unfortunately, these are extremely few, especially in underdeveloped cities, as compared to the need for testing of Coronavirus, which is humongous [12]. This is why we conducted this research: to assist physicians and general radiologists in achieving efficient and accurate prediction of Coronavirus with minimum requirements.

CNNs (Convolutional Neural Networks) have shown to be extremely successful in computer vision and are frequently used in medical imaging [13,14]. This is because the CNN models have proved to be decisive when it comes to the prediction of output by analyzing and mapping the images to the correct output [15]. The proposed algorithm aims to accurately anticipate variations in complicated patterns of chest X-ray images between Covid-19 infected, Viral Pneumonia infected, and healthy individuals that are indistinguishable to the naked eye.

Image preprocessing also plays a significant role in boosting a classifier's performance. Improving the image quality of the input images in the training set can assist the machine learning algorithms in increasing their accuracy. Image quality has been proven to be improved by several metaheuristic-based optimization techniques [16,17]. In recent years several types of research have been ongoing in multi-modal medical image fusion through meta-heuristic-based optimization [18–20] as it enhances the quality of fused images while preserving critical information from the input images.

Significant challenges faced in COVID-19 detection are, firstly, finding a dataset that is best suitable for the work. The data should be abundant and labeled to allow for accurate classification of Covid-19 from the Chest X-ray images. In building a diagnostic method that is state of the art in terms of accuracy and efficiency. This is vital for the real-life implementation of this method as the model would not be feasible if it did not diagnose with appreciable accuracy or required heavy computational power to function. In this regard, a curated dataset with a sufficient number of adequately labeled image data has been used in this paper. Furthermore, a lightweight CNN architecture CoviXNet has been developed to balance out the issue of accuracy and efficiency.

The need for computationally efficient methods is essential in the field of biomedical imaging for producing viable results. However, deep learning techniques are generally computationally heavy, making their application in the real world impractical. The pre-trained models are trained on a large-scale ImageNet dataset [21] which contains around 14 million images and has 1000 classes. Using transfer learning with these models produces fruitful results, even on different datasets. But since we are classifying only between 3 classes, Normal, Viral Pneumonia, and Covid-19, the need for complex models with a large number of parameters is not necessary.

In this regard, we propose CoviXNet, a deep learning neural network architecture, as an alternative efficient screening model for identifying COVID-19 by evaluating patients' Chest X-rays and looking for visual markers in the chest radiography imaging of COVID-19 patients. The proposed 15-layer CNN architecture shows accuracy compared to the pre-trained networks and uses 30 times fewer parameters. With the method proposed, we aim to address this issue and assist radiologists, medical staff, and patients with accurate and feasible detection of Covid-19.

Our paper's main contributions are stated below:

1. Deep features of input Chest X-ray images were extracted using various traditional fine-tuned models, pre-trained on the ImageNet dataset, and a comparative study was conducted to analyze the best performing models.
2. Both 2-class and 3-class classification was done on top-performing models to test model robustness.
3. A novel 15-layer CNN architecture—CoviXNet was constructed taking both efficiency and accuracy into account after experimentation and analysis of pre-trained models. In addition, 10-fold cross-validation was also performed by splitting the dataset two times to properly evaluate the CoviXNet model's robustness.
4. A curated dataset was used in this paper in which the Sait et al. [22] removed all the duplicate images. Then, the dataset is carefully pre-processed and partitioned into training, testing, and cross-validation sets to prevent data leakage. In a pre-processing step, different image sizes were selected in accordance with different models considering both accuracy and efficiency. After partitioning, only training data is augmented, and the model has been validated and tested on the rest of the data.

The rest of the study's contents may be summarized as follows. Section 2 is dedicated to analyzing the previous work of various scholars done in the detection of COVID-19. Section 3 deals with the various Materials and Methods used and also entails dataset and proposed algorithms. Section 3.4 dives deep into the different deep learning architectures analyzed and used for the study. Section 3.5 discloses the model proposed in this research. Finally, Section 4 disclose and cover experimentation and the results found from the previous experiments.

2. Related works

Ever since the Covid-19 pandemic has taken over the globe, there has been significant work done for the efficient detection of Coronavirus by various medical experts, researchers, and scientists. The application of deep learning techniques for the quick detection of the Coronavirus using Chest X-rays has been widely researched. Extensive research on these works has been taken on.

Jain et al. [23] used 6432 chest X-rays image data from their Kaggle repository to compare and report the accuracy of InceptionV3, Xception, and ResNeXt models. The authors claimed the Xception model gave the highest accuracy of 97.97%. The author used LeakyReLU instead of relu as an activation function and claimed it as a novel approach. Finally, the authors emphasized that the high accuracy found may be the reason for concern because it might be the consequence of overfitting. The authors suggested considering large datasets in the future to validate their suggested model. Basu et al. [24] proposed Domain Extension Transfer Learning (DETL). The author stated that training a CNN from scratch requires significant expertise for architecture to work properly and requires huge data for training. After performing a 5-fold Cross-Validation, the claimed accuracy was 82.98% for Alexnet, 90.13% for VGGNet, and 85.98% for ResNet. The author also used Gradient Class Activation Map (Grad-CAM) idea to see if a model paid more attention during image categorization. The nCOVnet model, which is based on transfer learning, was given by Panwar et al. [25]. The input layer forms the initial layer for the nCOVnet architecture, followed by 18 layers of convolutional, then ReLU, and max-pooling layers from the pre-trained VGG16 model. Then in the 2nd algorithm, the author added 5 more custom layers as head layers. The training accuracy claimed was 93%–97%, and the testing accuracy was 97.62% when tested on Covid-19 positive patients. The dataset utilized contains 192 covid positive patients' X-ray scans for 337 images. The author also took care of the possibility of data leakage, so they manually split the dataset into training and testing sets.

Ismael et al. [26] used three deep learning CNN approaches for identifying Covid-19 in chest X-rays. Deep feature extraction, transfer

learning, fine-tuning approaches, and end-to-end training of a developed CNN model were among them. Deep feature extraction and fine-tuning were done using pre-trained deep CNN models such as VGG16, VGG19, ResNet18, ResNet50, and ResNet101. The deep characteristics are classified using an Support Vector Machine (SVM) classifier with several kernel functions. A dataset of 180 Covid-19 and 200 normal chest X-ray images were used in the investigation. The deep features produced from the ResNet50 model had the highest accuracy score of all the findings, at 94.7%. The fine-tuned ResNet50 model was 92.6% accurate, whereas the built CNN model's end-to-end training yielded a 91.6% result. Several local texture descriptors and SVM classifications had been used to compare performance, finding that the deep approaches were more efficient. Luz et al. [27] used CNN architecture to detect Covid-19 by CXR. B3-X, from the family of proposed models based on EfficientNet, has an accuracy—93.9%, COVID-19 sensitivity—96.8%, positivity prediction of 100 percent (no false negatives) while requiring 5 to 30 times lesser parameters. A dataset consisting of 13,569 X-ray images of patients classed as healthy, COVID-19 pneumonia, and non-COVID-19 pneumonia were used to train the recommended approaches and the other 5 competing architectures. A hierarchical technique was used, as well as a cross-dataset analysis. The cross-dataset examination revealed that even the most advanced models lack generalization capacity. As can be seen, only a small number of images for Covid-19 positive patients was included in this study. The authors deployed the recommended models on a broad and varied dataset to evaluate their performance.

Further, Hussain et al. [28] proposed a 22-layer CNN-based model named CoroDet and used both Chest X-rays and CT Scan Dataset for 2, 3, and 4 classes, i.e., COVID, Normal, non-COVID bacterial pneumonia and non-COVID viral pneumonia. With the CoroDet Model on X-ray Dataset, they could reach an accuracy of 99.1%, 94.2%, and 91.2% for 2,3 and 4 class classification, respectively. Karakanis et al. [29] proposed two lightweight models, one for binary classification and another one for 3-class classification, and they have compared it to the state of art ResNet8 Pre-trained model. They have also used Conditional Generative Adversarial Networks (cGANs) to generate synthetic images since there was a shortage of datasets. With the proposed binary class model, they achieved an accuracy of 96.5% and 94.3% with the multiclass model. Using cGANs, their accuracy improved to 98.7% and 98.3%, respectively, for binary and multiclass Models. Khan et al. [30] presented a CNN-based model called Coronet, which has been based on Xception. Using X-ray images, the model achieved a binary classification accuracy of 99% (COVID-19, Normal), 95% for 3-class classification, and 89.6% for four classes (COVID-19, Normal, Pneumonia-bacterial, and Pneumonia-viral). Ozturk et al. [31] developed a 17-layer CNN-based model called DarkCovidNet (modified Darknet model), which obtained an accuracy of 98.08% for binary classification (COVID vs. No-Findings) and an accuracy of 87.02% for three-class classification (COVID vs. No-Findings vs. Pneumonia). DarkCovidNet was built using the Darknet-19 classification model (the backbone of YOLOv2) as a starting point. Hammoudi et al. [32] suggested Tailored deep learning algorithms to detect pneumonia infected patients from chest X-rays. The author said that viral pneumonia cases discovered during COVID-19 had a high likelihood of becoming COVID-19 infections. DenseNet169 achieved the highest accuracy of 95.72%, outperforming other models such as ResNet34 (90.54%) and ResNet50 (90.54%) (93.92%).

Further, in some recent studies, Ayalew et al. [33] propose a method, DCCNet, for quick detection of Covid-19 using Chest X-ray images. This is a hybrid CNN and HOG (Histogram of Oriented Gradients) based approach using an SVM classifier which produces a testing accuracy of 99.67% on only binary classification. Before classification, object detection was done using YOLO to verify if the lung images were of a human being. The dataset used consisted of 2500 images divided into the classes—Normal and Covid-19 infected CXR images. The results were compared to those obtained by implementation on AlexNet; an improvement of over 6% was observed. Khan et al. [34] propose two deep learning frameworks, Deep Hybrid Learning (DHL) and Deep Boosted Hybrid Learning (DBHL), to detect Covid-19. The dataset utilized in this work consisted of 3224 Covid-19 and 3224 Normal Chest X-ray images for binary classification. The proposed DHL framework uses COVID-RENet 1&2 models to extract deep features and individually passed through an SVM classifier for Covid-19 detection. In the COVID-RENet models, the edge and region-based operations are applied to extract region and boundary features. In the proposed DBHL framework, COVID-RENet models are fine-tuned by concatenating the feature spaces. The authors claim that the proposed methods significantly reduce the number of false negatives and false positives compared to previous work done. Muhammad et al. [35] introduce a feature augmentation mechanism using reconstruction independent component analysis (RICA) to deal with the lack of sufficient labeled data available. A CNN-BiLSTM method was proposed for the detection of Covid-19. The CNN assists with the high-level features extracted, while the augmentation mechanism provides the most relevant low-dimensional features. BiLSTM is then used to classify the information processed. Experiments on this proposed technique were conducted on three Covid-19 datasets, proving excellent performance compared to previous methods used. PCA and t-SNE plots were used for the visualization of the results.

The thorough and beneficial work done so far has one common denominator, it has been implemented on datasets that were inadequate in size due to the unavailability of appropriate data. This gives way for error in real-life implementation, even if the performance achieved was excellent. In other cases, if the dataset was large enough, the accuracy and efficiency of the model were not correctly balanced. We have analyzed all these parameters and introduced a well-constructed architecture with phenomenal and efficient performance on an extremely well-curated and processed dataset. The primary motivation for proposing a new architecture—CoviXNet is that some give excellent results but are computationally costly. And, if it is very efficient, then it does not perform very well on 3-class classification, where we have to classify COVID-19 when both pneumonia infected and normal Chest X-ray images are present. The proposed architecture is explained in detail in further sections.

3. Materials and methods

In the proposed method, Chest X-ray images are used for COVID-19 detection, as represented in Fig. 1. The input X-ray images are scaled at first to comply with CNN models. The three deep learning strategies explored were deep feature extraction utilizing pre-trained networks, fine-tuning the pre-trained CNN models, and end-to-end CNN model training. Various pre-trained networks were employed for deep feature extraction. The deep features were trained using the softmax classifier [36] for classification purpose. In addition, a 15-layer CNN-based model was developed and trained end-to-end. In Section 3.5, this proposed model, CoviXnet, is further illustrated.

3.1. Dataset formation

The dataset used in our work is collected from a paper titled "Curated Dataset for COVID-19 Posterior-Anterior Chest Radiography Images (X-rays)" [22]. Since there were many CXR Datasets available publicly for COVID-19 Detection, Sait et al. [22] have combined 15 different publicly available datasets of CXR [37–53]. The authors have curated this combined dataset of Covid-19 CXR images, consisting of 4558 COVID-19 X-rays, 5768 bacterial pneumonia X-rays, 4497 Viral pneumonia X-rays, and 5403 Normal X-rays. In these 15 publicly available repositories, there were many duplicate images found, which can impact the training process. The proper evaluation of models cannot be done properly if there is a leakage in training and testing data. Data leakage has been explained further in Section 3.1.1.

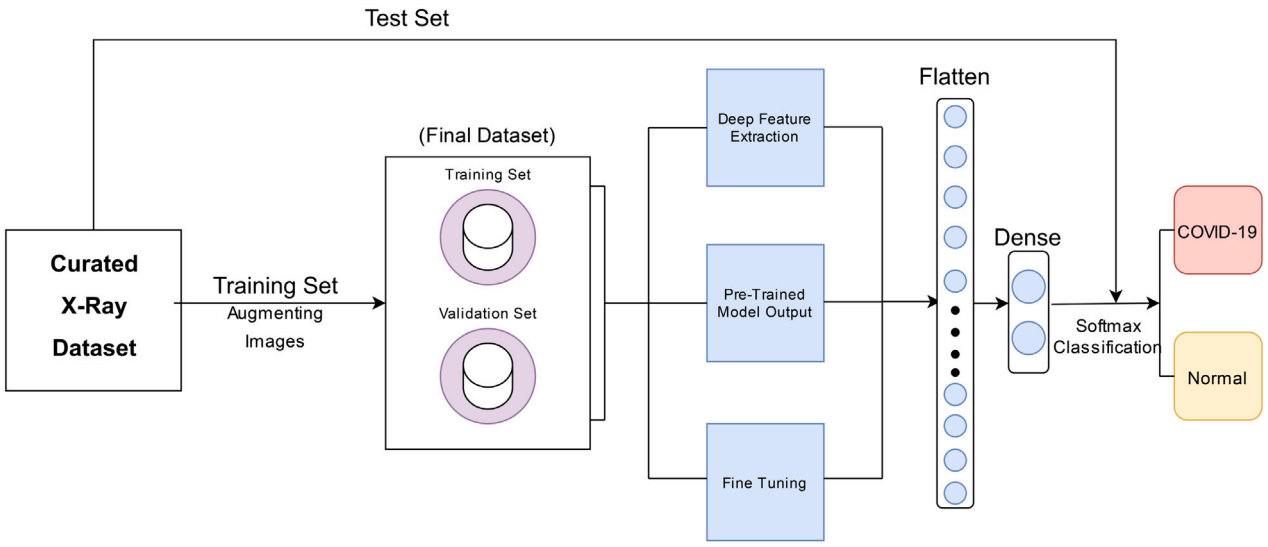

Fig. 1. Graphical Abstract of the Proposed work.

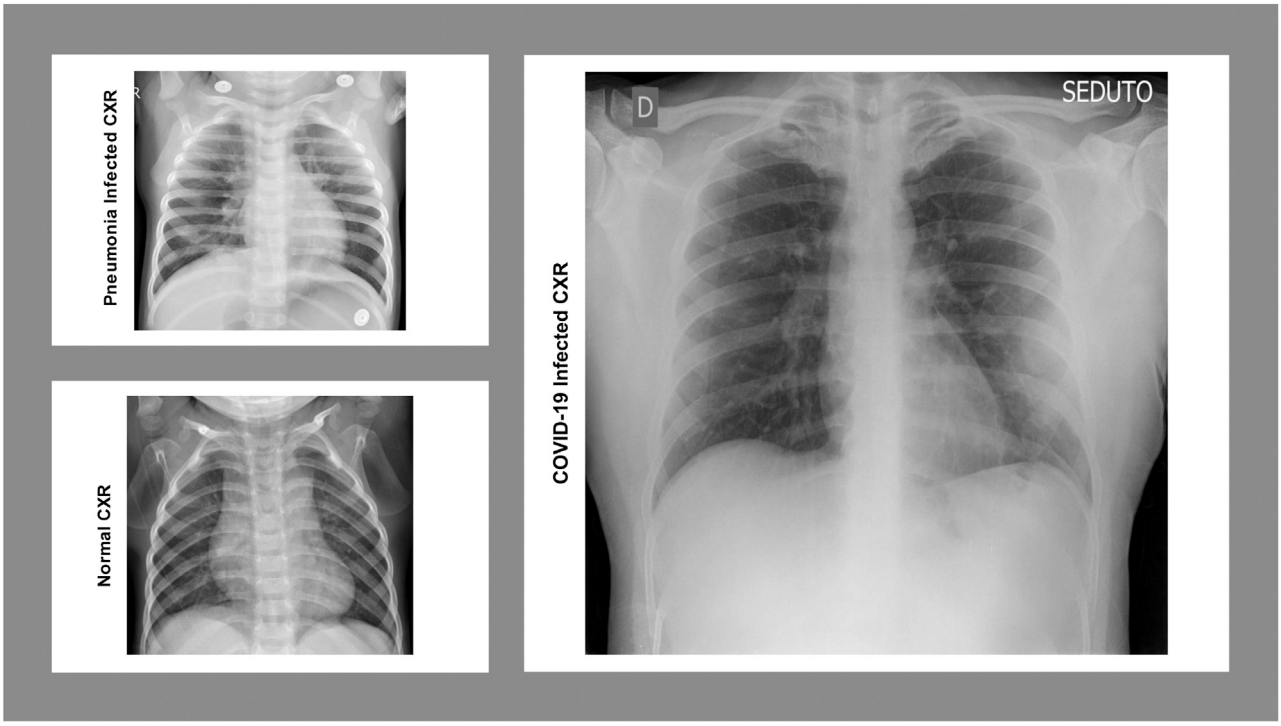

Fig. 2. Visualization of Normal, Covid-19 infected and Pneumonia infected Chest X-ray.

Sait et al. [22] removed those duplicate images by the use of Inception V3 architecture followed by cosine similarity distances based on unsupervised learning algorithms. After the cleaning process, the final curated dataset was prepared, containing 1281 COVID-19 X-rays, 3001 bacterial pneumonia X-rays, 1656 viral pneumonia X-rays, and 3270 Normal X-rays.

Since our work primarily uses 2-class and 3-class classification, i.e., COVID-19 Infected, Pneumonia Viral, and Normal X-ray images, we have used 1281 Covid-19 CXR, 1656 viral-pneumonia CXR, and 3270 Normal CXR images as our final dataset. The dataset used in our research has been visualized in Fig. 2.

3.1.1. Data leakage

Data Leakage is a widespread problem that occurs while training a deep learning model [54]. While training a model, it may achieve an accuracy of 99%, but when tested on real-world images, it fails to classify them accurately. The most common cause of this is having duplicate images in the dataset or leakage of data while pre-processing.

In some publicly accessible data sets, the authors enlarge the database by augmenting it. This dataset contains many repeated images. If we divide these images into training, validation, and testing sets, the images may then repeat in the testing and validation sets. Suppose an image is further augmented into six images, three of which stay in the training set while the others are split between the validation and testing sets; accuracy might be deceptive. The model can easily classify those images as it has already seen them in the training set. But in this scenario, the model will fail when tested on real-world images.

To tackle this issue and ensure that there is no data leakage during training and evaluation of our models, we have used a Curated

Dataset [22] in which Sait et al. have confirmed that there are no duplicate CXR images in the dataset. Further, we have carefully divided the dataset into training, cross-validation, and final testing/model evaluation data. We first split the data and augmented only training data to ensure no data leakage.

3.1.2. Dataset preprocessing and augmentation

We transformed all X-ray images into a common size of 256 × 256 pixels as a preprocessing step before training because the dataset's X-ray images are not homogeneous and come in various sizes. Images were RGB reordered, and the final input for the proposed model was delivered as a 256 × 256 × 3 image. For other models like EfficientNet, and VGG16, the images are resized to 224 × 224 also. This was done according to the model requirements. For our proposed Model CoviXNet, we have resized the images to 64 × 64 taking both accuracy and efficiency into account. After applying RGB ordering, the final input was 64 × 64 × 3. This resizing has been done after performing experiments and achieving the best results. The images have also been scaled, i.e., all the pixels from 0 to 255 are normalized.

In the case of most real-life problems, there is hardly ever an abundance of data, especially for medical imaging. To overcome the shortage, we should expand the training dataset by data augmentation [55], in such a manner that important information is not lost. Data augmentation functions as a regularizer and aids in the management of data overfitting. It can increase the robustness of the model by creating more training data and exposing the model to diverse versions of data.

In the used dataset [22], even though it contains an adequate amount of images, we cannot directly say that it is sufficient for training a deep learning model, particularly when it comes to end-to-end CNN training. As a result, we performed our experiments on both the original and augmented datasets. In the augmented dataset, we have applied horizontal flip and scaling the dataset to best fit the models trained. We augmented our dataset with a sheer range of 20 and a zoom range of also 20. The images were also flipped horizontally to increase the heterogeneity of the dataset remarkably.

3.2. Deep learning

Techniques for deep learning are extensively used in processing and recognizing images from datasets efficiently [5]. We can use these equations to summarize how deep learning methodologies work.

First and foremost we compute z using the inputs x_i as shown in Eq. (1),

$$z = \sum_i w_i \star x_i + b \tag{1}$$

where, w_i are the weights, b is the bias

Secondly we compute a, which is equal to y at the output layer, using z as shown in Eq. (2),

$$a = \psi(z) \tag{2}$$

Combining equation (1) and (2) we get Eq. (3),

$$a_j^l = \sigma\left(\sum_k w_{jk}^l a_k^{l-1} + b_j^l\right) \tag{3}$$

where, w_i are the weights, b is the bias and ψ is said to be the activation function.

3.2.1. Deep transfer learning

The technique of obtaining deep features and fine-tuning pre-trained CNN models is known as deep transfer learning [56]. Transfer learning aids the deep learning process for this suggested method's image classification problem when there are a limited amount of training images [57]. The weights from a pre-trained network can be utilized to speed up or improve the learning process instead of training a model from scratch. The model's first layers may be thought of as descriptors of image features, while the latter levels are related to particular categories. As a result, several layers may be utilized in numerous applications. Following that, the task of transfer learning is to determine how and which layers of a pre-trained model should be employed [58]. Even when transferring weights from entirely unrelated domains, this strategy has proven to be helpful in a variety of computer vision applications. After changing the architecture to suit the chosen problem, we initialize a new layer and define the learning process. We trained the model using this Deep Transfer Learning process and updated the weights using the appropriate optimizer and Softmax function for classification [36].

3.2.2. Fine tuning

Fine-tuning is the process of making slight modifications to the existing approach to achieve the desired results. In deep learning, we use fine-tuning to significantly reduce the time and resources used to improve the model's efficiency. We do this by utilizing the initial layers of pre-trained models and their weights on our dataset, provided they have a similar input, like a dataset consisting of images. Deep learning models require substantial data for training, which creates much hassle. This can be solved by altering specific parameters of an already trained model to perform a new similar task. We have fine-tuned various pre-trained models to do a comparative study analyzing top-performing models on our dataset.

3.3. Convolutional neural networks (CNNs)

Convolutional Neural Network (CNN), a class of artificial neural networks popular for such Computer Vision functions, is being used extensively for biomedical imaging and has been primarily used in our work. CNN has been modeled to learn and process different features and patterns automatically. Convolutional, pooling, and fully connected layers are the three layers or building elements that make up the system. The first two layers, the convolutional and pooling layers, are employed to extract features. Fully connected layers, the last layer, do classification, which is mapping to the final output [36].

The first layer of a CNN network is the convolutional layer, the core building block which is responsible for the majority of the computational work [59]. The use of kernels and filters does this. The input is the image. So for an image whose RGB value is of depth n, a filter of depth n would only be applied. The output would be a 2-dimensional matrix for convolution with a 3-dimensional filter.

In addition to CNNs, kernel convolution is a critical component of various other computer vision technologies. It is a technique in which we apply a small number matrix called a kernel or filter on our image, then transform it using the filter's values. For example, this formula is used to compute feature map values as shown in Eq. (4).

$$G[m,n] = (f * h)[m,n] = \sum_j \sum_k h[j,k] f[m-j, n-k] \tag{4}$$

where, f stands for the input image and h stands for our kernel. The result matrix's row and column indexes are denoted by m and n, respectively.

We had created an output feature map using a convolution method. In the convolution layer, each output feature map is blended with many input feature maps, as shown in Eq. (5).

$$x_j^l = f\left(\sum_{i \in M_j} x_j^{l-1} * k_{ij}^l + b_j^l\right) \tag{5}$$

where, x_j^l is the current layer's output, x_j^{l-1} is the previous layer's output, k_{ij}^l is the current layer's kernel, and b_j^l are the current layer's biases. A collection of input maps is represented by M_j. The convolution results are then processed by a nonlinear activation function.

The second layer is Pooling layer, as shown in Eq. (6). It is responsible for lowering the number of parameters by downsampling feature maps. It is applied throughout the layers in the 3D volume. A popular

Table 1
Modified EfficientNet architecture.

Layer name	Layer type	Input	Output
efficientnet-b0_input	Input Layer	(None, 224, 224, 3)	(None, 224, 224, 3)
efficientnet-b0_input	Model	(None, 224, 224, 3)	(None, 7, 7, 1280)
avg_pool	Global Average Pooling 2D	(None, 7, 7, 1280)	(None, 1280)
batch_norm	Batch Normalization	(None, 1280)	(None, 1280)
dropout	Dropout	(None, 1280)	(None, 1280)
dense	Dense	(None, 1280)	(None, 512)
batch_norm_1	Batch Normalization	(None, 512)	(None, 512)
activation	Swish Activation	(None, 512)	(None, 512)
dropout	Dropout	(None, 512)	(None, 512)
dense	Dense	(None, 512)	(None, 128)
batch_norm_2	Batch Normalization	(None, 128)	(None, 128)
activation_1	Swish Activation	(None, 128)	(None, 128)
dense_2	Dense	(None, 128)	(None, 2)

pooling layer employs a non-overlapping 2 cross 2 max filter with a stride of 2. The maximum value in the features inside the region would be returned by a max filter.

$$x_j^l = \text{down}\left(x_j^{l-1}\right) \tag{6}$$

where the down-sampling is represented by the down(.) function.

The fully connected layer, which includes Flattening, is the last layer. The whole pooling feature map matrix is then transformed into a single column and input into the neural network. We create a model by combining these characteristics using fully connected layers. Finally, the output is classified using the softmax activation function as shown in (7).

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{7}$$

where, σ = softmax, $\vec{z}$ = input vector, e^{z_i} = input's standard exponential function, K = number of output classes in a multi-class classification, e^{z_j} = output's standard exponential function.

3.4. Deep feature extraction

In the feature extraction method, we remove the fully connected layers of a pre-trained CNN model while keeping the remaining network of convolution and pooling layers, which act as a feature extractor. Any machine learning classifiers and fully connected layers can be added to the feature extractor. This enhances the network's performance on our chosen dataset as it is better suited for our model. By eliminating the dataset's final fully connected layer, we extracted features using pre-trained models. After that, we employed a fully connected layer of two neurons with the Softmax Activation function as a classifier to accomplish the classification task. Following three standard models pre-trained on ImageNet were used for feature extraction.

3.4.1. InceptionV3

The inception block is the central concept of the inception network [60]. In a traditional or convolutional neural network layer, the previous layer's output would be used as the input for the following layer, and so on until the prediction. However, the inception block separates the individual layers. Instead of passing it through one layer, it concatenates the outputs from all of these distinct procedures after passing the previous layer input through four separate operations in parallel. In addition, internal layers can determine which filter size is necessary to learn the required information, thanks to the Inception layer. As a result, the layer adapts even if the image's size varies.

Inception net V3 is a 48 layers deep convolutional neural network and uses the inception module [61]. An Inception Module consists of an Input layer, 1×1 3×3 5×5 convolution layer, Max pooling layer, and Concatenation layer [62].

3.4.2. DenseNet169

DenseNets has based on the premise that the next layer's input is the concatenation of all the preceding layers' inputs [63]. It is a convolutional neural network that connects all the layers directly (with identical feature-map sizes), resulting in dense connections between them [64]. To retain the feed-forward nature, each layer gets fresh inputs from all preceding levels and delivers its feature maps to all following layers [65]. With multi-layer feature concatenation, DenseNet has a larger capacity. However, dense concatenation introduces a new problem: it necessitates more GPU memory and training time. DenseNet-169 was chosen despite its 169 layer depth since it has a less number of parameters compared to other models and the design handles the vanishing gradient problem adequately [66].

3.4.3. EfficientNet

EfficientNet is a convolutional neural network scaling and design method that consistently scales all depth, breadth, and resolution parameters using a compound coefficient [67]. Not only do EfficientNets increase model accuracy, but they also improve model efficiency by decreasing parameters and FLOPS (Floating Point Operations Per Second) [68]. Luz, E. [27] modified EfficientNets by adding some batch normalization, dropout, activation, and dense layers. In this work, the same architecture has been used. Detailed Architecture of modified EfficientNet is shown in Table 1.

3.5. Proposed model—CoviXNet

After experimenting on all the pre-trained models, a novel CNN model CoviXNet was constructed for COVID-19 detection. A CNN comprises two essential components: feature extraction and classification. The model's initial layers may be thought of as descriptors of image features, while the latter levels are related to particular categories. Several convolution layers are used in feature extraction, followed by max-pooling and an activation function. The classifier is typically made up of fully connected layers and a softmax activation function. If there are more no. of classes in the dataset, there will be more no. of features for a model to learn. So the feature extraction component of a CNN should be deeper and more complex to learn the complex features. Since there are only 3 classes, a model has to discriminate between them, so we have used 3 convolutional, max pooling, and activation layers consisting of 32, 32, and 64 neurons. These parameters, such as no. of layers and no. of neurons, are determined by performing experiments balancing efficiency and accuracy.

After flattening, the feature vector consists of 4096 neurons, and then for classification, the network has been scaled down by adding a dense layer of 64 neurons. Unlike the networks pre-trained on ImageNet Dataset, where the model has to classify between 1000 classes, we have only 3 classes in our dataset. So, scaling down the dimensions of the classification layer does not hamper the performance much. We have also determined these parameters by performing a set of experiments balancing both efficiency and accuracy.

Table 2
Table describing the model architecture of the proposed model CoviXNet and its various parameters.

Layer name	Layer type		Input shape	Output shape	Parameters
input	Input Layer	64 × 64 × 3 images	(64, 64, 3)	(64, 64, 3)	0
conv2d-1	Conv2D	32 Filters, S = (3, 3)	(64, 64, 3)	(62, 62, 32)	896
activation-1	Activation	ReLU	(62, 62, 32)	(62, 62, 32)	0
max_pooling2d-1	MaxPooling2D	pool size = (2, 2)	(62, 62, 32)	(31, 31, 32)	0
conv2d-2	Conv2D	32 Filters, S = (3, 3)	(31, 31, 32)	(29, 29, 32)	9248
activation-2	Activation	ReLU	(29, 29, 32)	(29, 29, 32)	0
max_pooling2d-2	MaxPooling2D	pool size = (2, 2)	(29, 29, 32)	(14, 14, 32)	0
conv2d-3	Conv2D	64 Filters, S = (3, 3)	(14, 14, 32)	(12, 12, 64)	18496
activation-3	Activation	ReLU	(12, 12, 64)	(12, 12, 64)	0
max_pooling2d-3	MaxPooling2D	pool size = (2, 2)	(12, 12, 64)	(6, 6, 64)	0
flatten	Flatten	Flatten layer	(6, 6, 64)	(2304)	0
dense-1	Dense	Dense Layer of 64 units	(2304)	(64)	147520
activation-4	Activation	ReLU	(64)	(64)	0
dropout	Dropout	0.5	(64)	(64)	0
dense-2	Dense	Dense Layer of 2 units	(64)	(2)	130

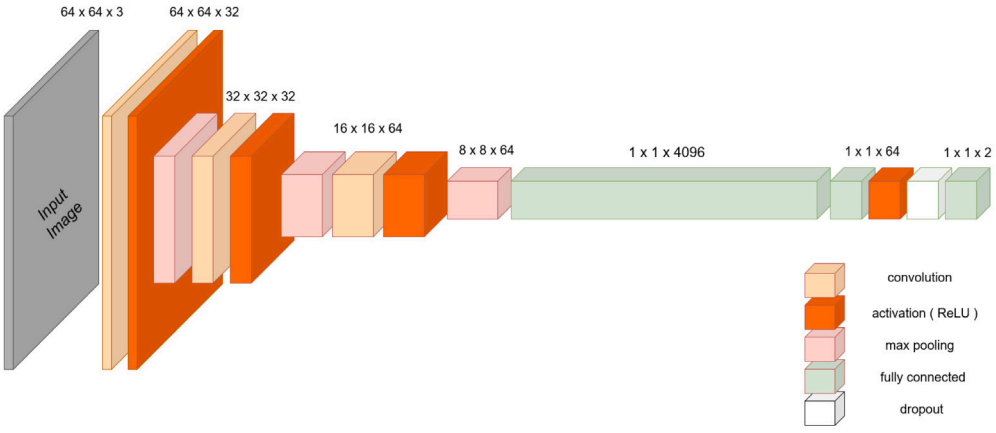

Fig. 3. 3-Dimensional Architecture of CoviXNet.

The proposed model has 15 layers, beginning with the input layer and continuing with 3 convolutional layers, 3 ReLU layers, and 3 max-pooling layers. The first two convolutional layers consist of 32 neurons, and the third convolutional layer is of 64 neurons, each of filter size 3 * 3. The HeUniform kernel is used to initialize the weights of each convolutional layer. A ReLU activation layer follows each convolutional layer, followed by max-pooling layers of pool size 2 * 2. Then a flatten layer is used to transform all data into a 1-dimensional array. Then a Dense layer consisting of 64 neurons is used, and after that, there is a ReLU activation layer. Then a dropout layer eliminating 50% neurons in every iteration is used to take care of overfitting issues. The layer is then mapped with a two-neuron output layer, one for covid-19 positive images and the other for regular CXR images, using the softmax function for classification purposes. Detailed architecture with a description of each layer is shown in Table 2 whereas 3-Dimensional visual representation is shown in Fig. 3.

3.6. Model training

This algorithm is based on end-to-end training of the proposed model CoviXNet. The notations used in Algorithm 1 are described here, with $\delta 1$ and $\delta 2$ referring to the training and validation data sets, respectively. μ is the model's learning rate, which indicates how rapidly it adapts to the problem. It usually has a value close to zero. ϵ is the total number of iterations (also known as epochs) for which the CNN model has been trained. Finally, β is another customizable hyper-parameter that usually has the form of 2n.

The algorithm starts by establishing the model's top input layer (refer to 1st step of Algorithm 1) and adding it as the head layer to the remaining 5 layers (refer to 2nd step of Algorithm 1). Then a for loop with several iterations ranging from 1 to the total number of iterations (ϵ) to train and update the weights using forward and backward propagation (refer to steps 6–8 of Algorithm 1).

The end to end training of the proposed model CoviXNet has been done for 100 epochs (ϵ), and Adam Optimizer has been used with a learning rate (μ) of 0.001, $\beta 1$ - 0.9, $\beta 2$ - 0.999 and an ϵ value of 1e–07. In addition, the loss function Categorical Cross Entropy has been employed.

3.6.1. Loss function

Loss function is used in helping the model measure how far an estimated value is from the true value [69]. This assists in defining what a good prediction is for the model to achieve. Cross entropy loss as shown in Eq. (9) is the loss function used for the work.

$$J(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(\hat{y}_i\right) + \left(1 - y_i\right)\log\left(1 - \hat{y}_i\right)\right] \tag{9}$$

where, model parameters, such as the neural network's weights, are denoted by $\mathbf{w}$, y_i is actual output label and $\hat{y}_i$ is predicted output label.

4. Experimental results

4.1. Dataset division

To avoid overfitting and model selection bias, it is ethical to divide the dataset into three sections—training, validation, and testing set. Our parameter estimates have more variance when we have less training data. Likewise, our performance metric will have more variance if we have less testing data. Therefore, we should divide the data in such a way that none of the variances is excessive. As a starting point and Pareto's 80-20 Split intuition [70], we chose the training set as 80% of

Algorithm 1 Training Procedure of the Proposed Model

Input : Training and Validation Set ($\delta 1$ & $\delta 2$)
Output : CNN weights
begin;
1. Set the input layer of the CNN architecture and feed the input size.
2. Set the head layers, CNN (Conv2D, Maxpooling2D, Flatten, Dense, Dropout)
3. Initialize the CNN parameters : μ (learning rate), ϵ (no. of epochs), β (batch size)
4. Resize each image in accordance with the model's required input shape.
5. Train the CNN and compute the initial weights

for $\eta = 1$ **to** ϵ **do**
6. Randomly select a mini batch from (size : β) from training set ($\delta 1$)
7. Forward propagation and compute the loss. The loss function used is described in Section 3.6.1.
8. Back propagate the error and update the weights using equation 8.

$$W_n = W_o - \eta * \frac{\partial J}{\partial W_o} \tag{8}$$

where, W_n is the new updated weight, W_o is the original weight, η is the learning rate, and $\frac{\partial J}{\partial W_n}$ is the partial derivative of the original weight.
end

Table 3
Table depicting the data distribution of Database-1.

	Normal	Covid-19 infected	Pneumonia infected
Training Set	2616	1025	1326
Validation Set	327	128	165
Testing Set	327	128	165

Table 4
Table depicting the data distribution of Database-2.

	Normal	Covid-19 infected	Pneumonia infected
Training Set	2289	897	1159
Validation Set	327	128	165
Testing Set	654	256	332

the whole data and the remaining 10%–10% for validation and testing, respectively.

To test the robustness of CoviXNet architecture properly when lesser data is provided for training, we have also evaluated CoviXNet on 70% training, 10% validation, and 20% testing data. Allocating more data in the final testing set ensures the robustness of the model and reduces possible failure in real-world testing.

Even though the final dataset contains adequate data, we cannot say that it is sufficient for training a deep learning model, particularly when it comes to end-to-end CNN training. As a result, we performed our experiments on both the original and augmented datasets. Table 3 shows the distribution of the Covid-19 Infected, Pneumonia Infected and Normal CXR images for training, validation and testing set for 80-10-10 split whereas Table 4 shows distribution for 70-10-20 split. After Performing Data Augmentation on Training set we have generated a total of 15,691 Normal CXR, 6148 Covid-19 Infected and 7951 Pneumonia Infected images as shown in Table 5. These are the total no. of images roughly generated after performing data augmentation. We have generated 6 times the original no. of images from the original training set.

4.2. Experimental setup

All the pre-trained models and our proposed Model were implemented with Tensorflow in python. The training is performed for minor epochs on a personal computer with Intel(R) Core(TM) i7-6500U CPU 2.50 GHz, Nvidia 940M GPU with compute capability 5.0, and 16 GB RAM. The whole training part is done on Kaggle with GPU Tesla P100-PCI-E-16GB compute capability: 6.0 and 16 GB GPU RAM. Each Model was trained for 100 epochs to obtain the best training, validation, and testing accuracy.

Table 5
Table depicting the number of Chest X-ray images in the dataset after data augmentation.

	Normal	Covid-19 Infected	Pneumonia infected
Training Set	15691	6148	7951
Validation Set	327	128	165
Testing Set	327	128	165

Table 6
Table depicting the experimental results obtained on pre-trained networks VGG16 and VGG19 with Adam and SGD optimizers on Database-1 and Database-2.

No.	Name	Optimizer	VA - 1	TA - 1	VA - 2	TA - 2	Parameters
1	VGG16	Adam	98.46	98.02	99.11	99.01	14,715,714
2	VGG16	SGD	98.46	98.46	98.45	98.68	14,715,714
3	VGG19	Adam	99.12	98.46	98.45	98.25	20,025,410
4	VGG19	SGD	97.8	98.24	97.56	98.68	20,025,410

4.3. Results

We Experimented with various available pre-trained deep learning models for our dataset and use case. Based on their analysis, various optimizers have been experimented on and used for better performance in different models. We first implemented VGG16 and VGG19 with other optimizers as shown in Table 6. This showed good accuracy but with many parameters. We then implemented the Inception and InceptionResNet family models, which are very popularly used; results are shown in Table 7 : Inception Networks. Then the DenseNets and the ResNets (Residual Networks) were implemented with Adam and SGD optimizers, the results of which are shown in Tables 8 and 9 respectively. Modified EfficientNet and Efficient family (B1–B7) were then implemented, and their comparative results are shown in Table 10. Finally, Xception and AlexNet were also implemented with the most commonly used optimizers, the results of which are shown in Table 11. These were experimented on, as they have demonstrated excellent performance in recognizing medical images before.

4.3.1. Top performing models

Our primary focus was binary classification, i.e., determining if a particular image was infected with covid-19 or not. However, viral pneumonia cases detected during a COVID-19 pandemic are thought to have a significant chance of developing COVID-19 infections, too, as stated in [32]. As a result, we also performed 3-class classification on our top-performing models, i.e., classifying the given image as covid-19 infected, pneumonia infected, or normal.

With InceptionV3 on 2-class classification, we achieved an accuracy of 99.78%, a sensitivity of 100%, and a specificity of 99.21%. The DenseNet169 model shows an accuracy of 99.56%, a sensitivity of 100%, and specificity of 98.43%. Both Modified EfficientNet B0 and Modified EfficientNet B1 models demonstrated an accuracy of 99.78%, a sensitivity of 100%, and specificity of 99.21%. On 3-class classification we achieved an accuracy of 96.45%, 96.13% and 95.97% with InceptionV3, DenseNet169 and modified EfficientNetB1 respectively. ResNet50V2 achieved a remarkable accuracy of 97.9% on 3-class classification.

In the case of medical datasets, it is recommended to use the original dataset if there is no shortage of datasets. Also, using fabricated images is not recommended in the medical field as they impact the model performance in real-world testing. Further to properly evaluate the robustness of proposed models when lesser data is provided for

Table 7
Table depicting the experimental results obtained on pre-trained networks InceptionV3 and InceptionResNetV2 with Adam, SGD, and RMSProp optimizers on Database-1 and Database-2.

No.	Name	Optimizer	VA - 1	TA - 1	VA - 2	TA - 2	Parameters
1	InceptionV3	Adam	98.68	98.9	98.89	99.23	21,806,882
2	InceptionV3	SGD	98.68	99.78	99.78	99.34	21,806,882
3	InceptionResNetV2	Adam	96.05	95.82	99.11	98.68	54,339,810
4	InceptionResNetV2	RMSProp	99.12	98.9	98.23	98.90	54,339,810
5	InceptionResNetV2	SGD	99.12	99.56	99.11	99.56	54,339,810

Table 8
Table depicting the experimental results obtained on pre-trained networks DenseNet121, DenseNet169, and DenseNet201 with Adam and SGD optimizers applied on Database-1 and Database-2.

No.	Name	Optimizer	VA - 1	TA - 1	VA - 2	TA - 2	Parameters
1	DenseNet121	Adam	99.34	98.24	99.78	99.23	7,039,554
2	DenseNet121	SGD	99.34	99.34	100	99.34	7,039,554
3	DenseNet169	Adam	99.34	99.56	98	99.56	12,646,210
4	DenseNet169	SGD	99.12	99.24	99.11	99.01	12,646,210
5	DenseNet201	Adam	99.56	99.34	99.11	98.58	18,325,826
6	DenseNet201	SGD	99.34	98.9	99.78	99.23	18,325,826

Table 9
Table depicting the experimental results obtained on pre-trained networks ResNet50 and ResNet152 with Adam and SGD optimizers applied on Database-1 and Database-2.

No.	Name	Optimizer	VA - 1	TA - 1	VA - 2	TA - 2	Parameters
1	ResNet50	Adam	97.36	98.46	99.78	99.02	23,591,810
2	ResNet50	SGD	95.82	95.16	99.11	99.23	23,591,810
3	ResNet152	Adam	96.92	97.58	99.33	99.56	58,375,042
4	ResNet152	SGD	97.36	97.14	99.11	99.34	58,375,042

Table 10
Table depicting the experimental results obtained on pre-trained networks EfficientNet B0-B7 and modified versions, Modified EfficientNet B0-B7 with Adam optimizer on Database-1 and Database-2.

No.	Name	Optimizer	VA - 1	TA - 1	VA - 2	TA - 2	Parameters
1	EfficientNetB0	Adam	98.9	99.34	97.34	98.80	4,052,126
2	Modified EfficientNetB0	Adam	99.12	99.78	98.67	98.25	4,779,038
3	EfficientNetB1	Adam	98.9	98.9	98.45	99.01	6,577,794
4	Modified EfficientNetB1	Adam	99.34	99.78	98.67	99.01	7,304,706
5	EfficientNetB2	Adam	99.12	99.56	98.45	99.01	7,771,380
6	Modified EfficientNetB2	Adam	99.12	99.56	98.45	98.25	8,564,084
7	EfficientNetB3	Adam	99.47	98.46	98	98.46	10,786,602
8	Modified EfficientNetB3	Adam	99.34	99.12	98	98.03	11,645,098
9	EfficientNetB4	Adam	98.68	98.24	98.89	99.34	17,677,402
10	Modified EfficientNetB4	Adam	99.34	99.34	98.23	99.13	18,667,482
11	EfficientNetB5	Adam	99.12	99.34	99.56	98.91	28,517,618
12	Modified EfficientNetB5	Adam	99.12	99.78	99.56	99.34	29,639,282
13	EfficientNetB6	Adam	99.12	97.8	97.78	98.46	40,964,746
14	Modified EfficientNetB6	Adam	99.12	99.34	98.67	99.02	42,217,994
15	EfficientNetB7	Adam	98.46	98.24	98.89	99.01	64,102,802
16	Modified EfficientNetB7	Adam	98.9	98.9	98.45	99.23	65,487,634

training, we have also evaluated them on 70% training, 10% validation, and 20% testing data. Allocating more data in the final testing set ensures the robustness of the model and reduces possible failure in real-world testing. In Tables 6–11 there is two databases, Database-1 and Database-2. In Database-1 the images are augmented and the split done is 80% training, 10% validation and 10% testing whereas in Database-2 the images are not augmented and the split done is 70% training, 10% validation and 20% testing data.

4.3.2. Proposed model

On 2 class classification, our proposed model (CoviXNet) demonstrated an accuracy of 99.56%, a sensitivity of 99.7%, and specificity of 99.15% on a 90% training and 10% testing dataset Split. We also did an 80% training and 10% testing dataset split, and the accuracy came out to be 100% on the training set and 99.23% on the testing set, as shown in Table 12. Although accuracy is not fluctuating much still, it ranged from 98.90% to 100%. So to ensure we are getting correct accuracy, we did a 10 cross-fold validation, shuffling the training and testing data on each fold. We got an accuracy of 99.47% (+/−0.36%). To test this on more images, we increased cross-validation data by 10% and again performed 10 cross-fold validation and got an accuracy of 99.39% (+/−0.27%) which is more or less the same as shown in Table 13. On 3 class classification, also CoviXnet achieved an accuracy of 96.61%. CoviXNet Model worked exceptionally well in our study, and it has a total of 176 K Parameters, which is approximately 124 times less than InceptionV3, 72 times than DenseNet169, 23 times EfficientNetB0, and 37 times than EfficientNetB1.

The accuracy of the developed models is related to the number of epochs, as shown in Fig. 4. When the number of epochs is increased from 1 to 200, the accuracy value rises and falls unevenly. Around epoch 200, the accuracy of various implemented models appears to be constant. As seen in Fig. 5, the magnitude of loss is also dependent on the number of epochs. When the number of epochs increases from 1 to 35, the value of loss increases and decreases unevenly. At epoch 35, the loss value appears constant for various implemented models. The accuracy and loss curves of CoviXNet on 3-class classification are shown in Figs. 6 and 7.

We employed a variety of metrics to properly evaluate the proposed model, which is described below.

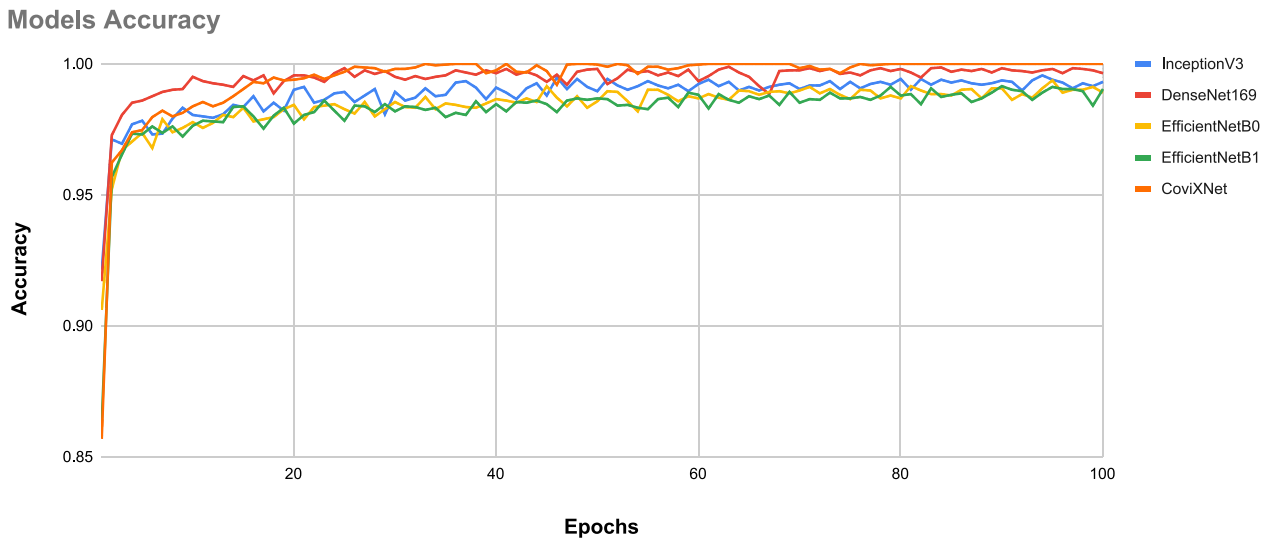

Fig. 4. Graph showcasing the accuracies of top performing models and CoviXNet.

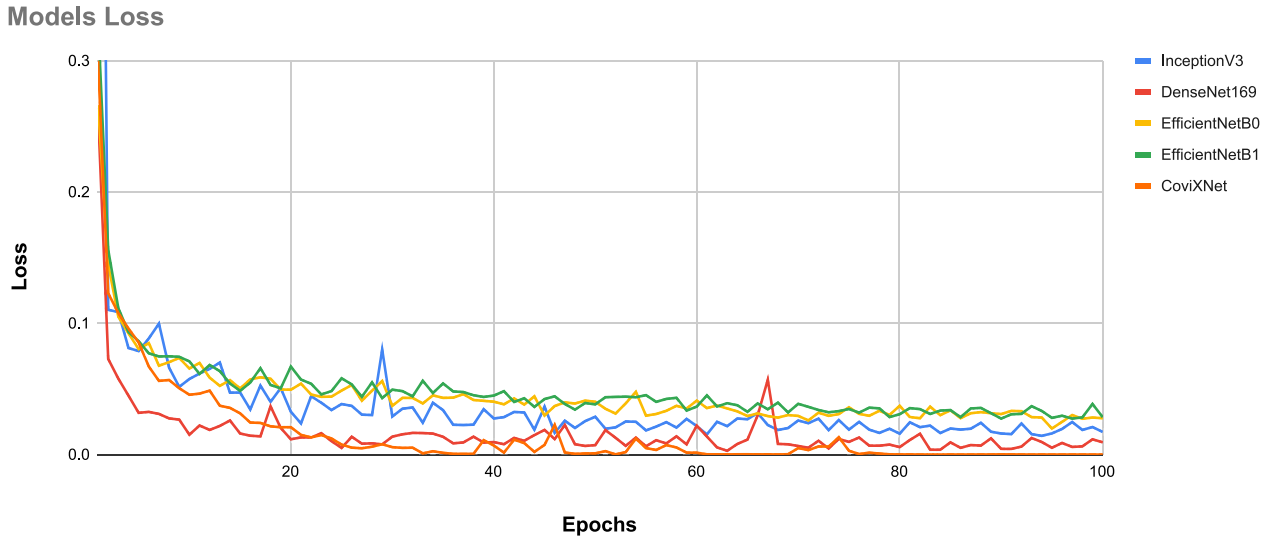

Fig. 5. Graph showcasing the loss of top performing models and CoviXNet.

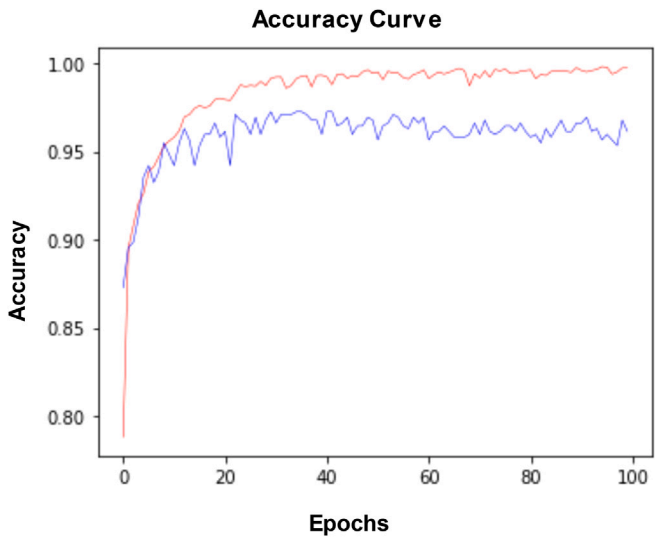

Fig. 6. Accuracy curve of CoviXNet on 3-class classification.

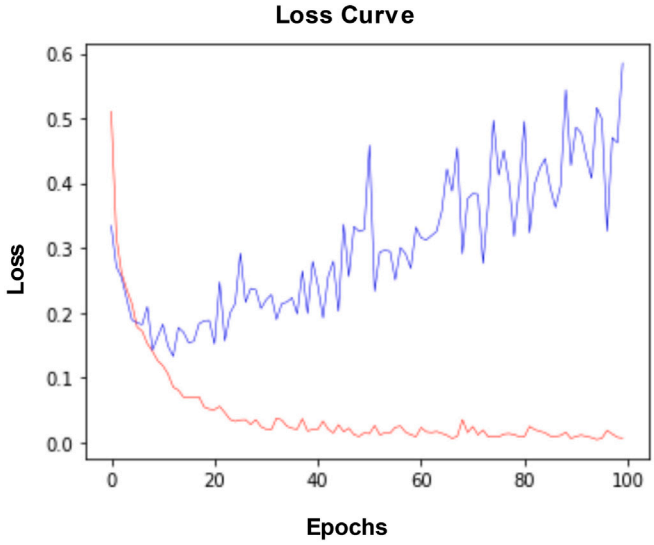

Fig. 7. Loss curve of CoviXNet on 3-class classification.

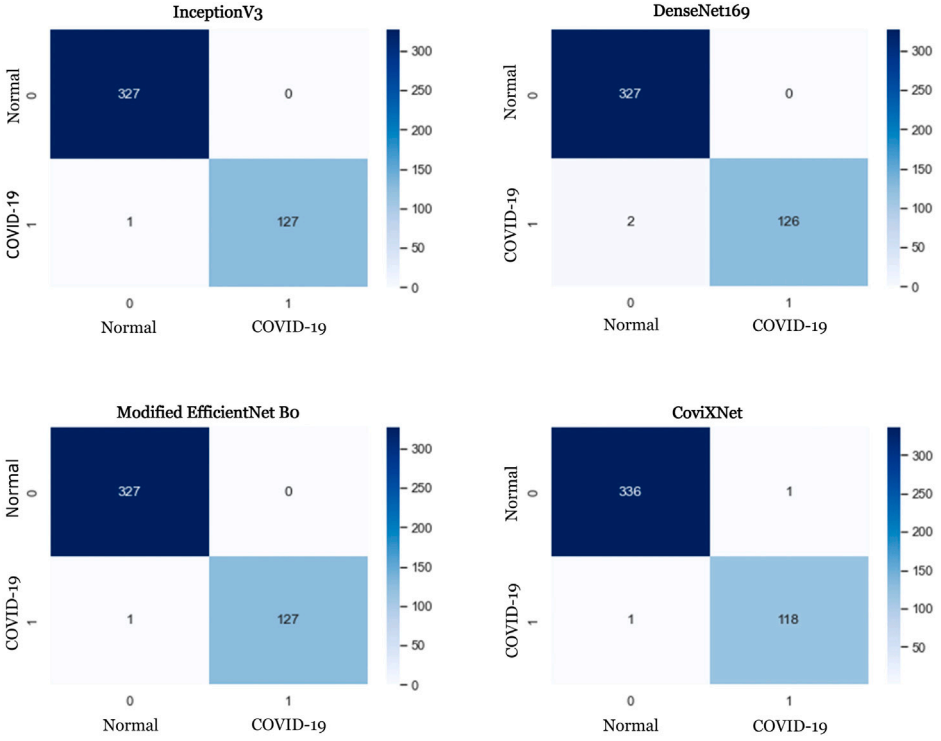

Fig. 8. Figure showcasing the Confusion Matrices of best performing models and CoviXNet.

1. **Confusion Matrix :** This is a visualization approach for the model's performance in classification tasks. This is used to accurately and easily depict how the machine learning model is classifying the positive and negative cases compared to the real data. Here the rows depict instance classes, and the columns depict actual classes. From Fig. 8 we can see that the model classifies covid-19 infected images correctly. At the same time, it makes mistakes on 1–2 images when classifying normal images, whereas CoviXNet classified one image of each class incorrectly. Fig. 9 represents the CoviXNet confusion matrix on 3-class classification
2. **AUC - ROC curve :** This tells us how well the model can differentiate between the classes. The AUC ROC curve is utilized at various threshold values to solve classification problems. It is a critical statistic for assessing the model's performance. From Fig. 10 we can see the area under curve for InceptionV3 and Modified EfficientNetB0 is 1.00 whereas for DenseNet169 and CoviXNet is 0.99 on 2-class classification. The area under curve for CoviXNet on 3-class classification is also shown in Fig. 11.
3. **Precision :** It is also an important metric to analyze the model performance which shows how precise the model was for the given classification. It is a proportion of accurately predicted positive instances by the model to the total number of positive cases anticipated. This is represented by a formula shown in Eq. (10).

$$\text{Precision} = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{10}$$

4. **Recall :** This allows us to compare our model's effectiveness in predicting true positives vs total positive instances. It is the ratio of accurately anticipated positive results by our model to the total number of positive cases in reality as shown in Eq. (11).

$$\text{Sensitivity} = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{11}$$

5. **Specificity :** This allows us to compare our model's performance in predicting true negatives to the overall number of negative cases in the output. It is the ratio of accurately predicted COVID negative findings by our model to the total number of negative instances in reality as shown in Eq. (12).

$$\text{Specificity} = \frac{True\ Negatives}{True\ Negatives + False\ Positives} \tag{12}$$

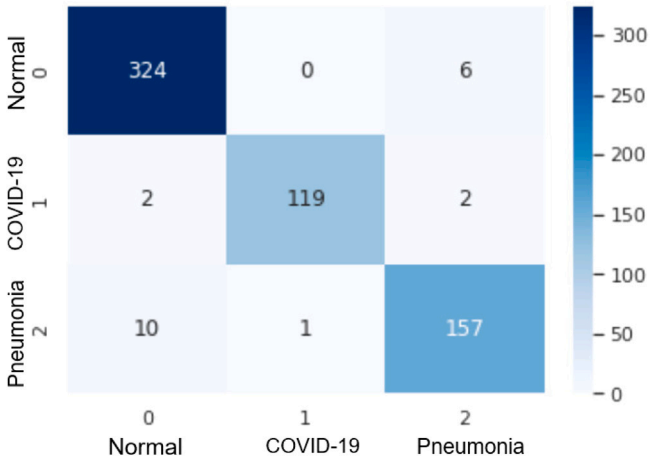

Fig. 9. Figure showcasing the Confusion Matrix of CoviXNet on 3-class classification.

These metrics of top performing models and CoviXNet model is shown in Table 14.

4.4. Comparison with other models

There are numerous publicly available datasets in this COVID-19 classification. Different authors have implemented and claimed their models' performance on different datasets. Some of them also merged various datasets and used them to evaluate their models and approaches. As a result, we cannot directly compare our model to

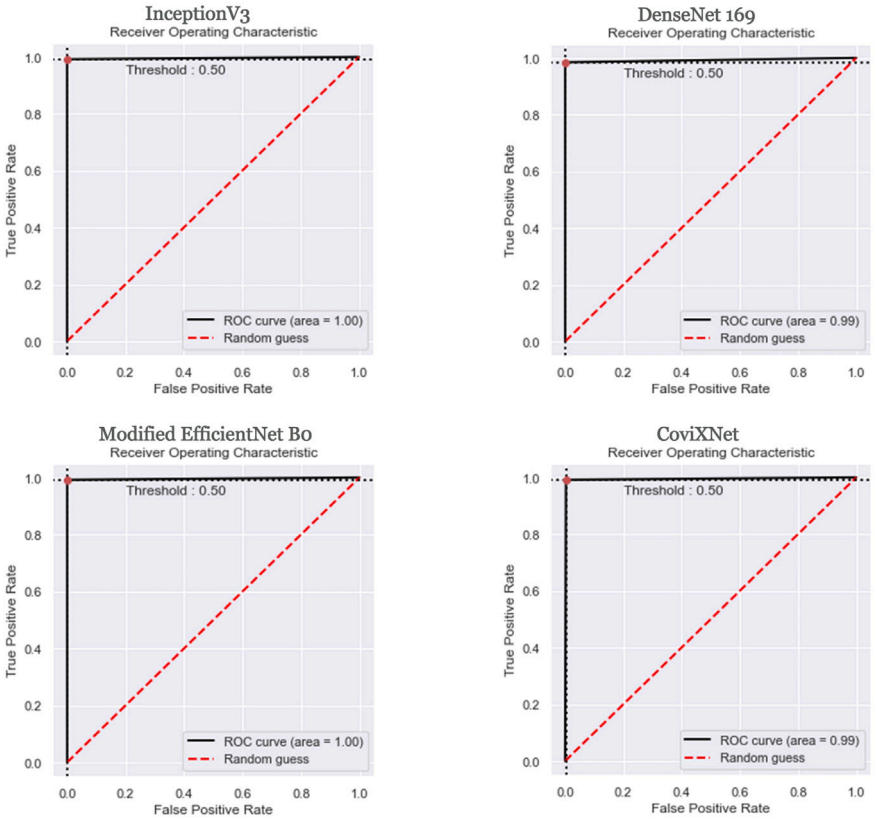

Fig. 10. Figure showcasing the ROC Plots of all top performing models and CoviXNet.

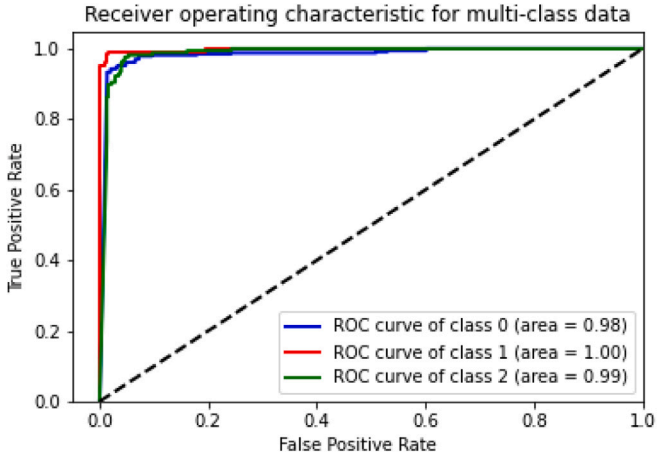

Fig. 11. Figure showcasing the ROC Plot of CoviXNet on 3-class classification.

Table 11
Table depicting the experimental results obtained on pre-trained networks Xception and AlexNet with Adam and SGD optimizers on Database-1 and Database-2.

No.	Name	Optimizer	VA - 1	TA - 1	VA - 2	TA - 2	Parameters
1	Xception	Adam	99.12	99.12	99.56	98.91	20,865,578
2	Xception	SGD	99.34	98.90	98.89	99.13	20,865,578
3	AlexNet	Adam	98.02	97.8	99.56	99.67	24,740,610
4	AlexNet	SGD	98.68	98.9	98.89	99.34	24,740,610

those studies. Still, to obtain a better sense of our proposed model's accuracy, we compare it to previous works with their used datasets and accuracies on 2-class and 3-class classification on Chest X-ray images. Table 15 shows that InceptionV3, ResNet50V2, and our proposed model

Table 12
Table depicting the experimental results obtained on Proposed model CoviXNet on different Dataset splitting.

Split	Training accuracy	Testing accuracy
90% Training and Cross validation data and 10% Testing Data	100%	99.56%
80% Training and Cross validation data and 20% Testing Data	100%	99.23%

Table 13
Table depicting the experimental results obtained on Proposed model CoviXNet on performing 10 Folds.

No. of folds (K)	Shuffling data	Split	K fold accuracy
10	True	90% Training and cross validation data and 10% testing Data	99.47% (+/−0.36%)
10	True	80% Training and cross validation data and 20% testing data	99.39% (+/−0.27%)

CoviXNet outperformed many of the state-of-the-art models proposed in other research.

5. Discussion

In this paper, experimentation on various pre-trained models was performed, with the results of the best-performing models explained in Section 3. With InceptionV3 and Modified EfficientNetB0 & B1, an accuracy of 98.78% was achieved in detecting Covid-19 from CXR images. With ResNet50V2, 97.90% accuracy was observed on 3-class

Table 14
Table depicting classification report of the top performing models and the Proposed model CoviXNet.

Model	Accuracy	Precision	Sensitivity/Recall	Specificity	F1
InceptionV3	99.78	1.0	1.0	0.9921	1.0
DenseNet169	99.56	0.99	1.0	1.0	1.0
Modified EfficientNetB0	99.78	0.99	1.0	0.9921	1.0
Modified EfficientNetB1	99.78	1.0	1.0	0.9921	1.0
CoviXNet	99.56	1.0	0.997	0.9915	1.0

Table 15
Comparison of our result with some other studies.

Author	Models	Covid-19 images	Pneumonia images	Normal images	2-class accuracy (%)	3-class accuracy (%)
Panwar et al. [25]	nCOVnet Model	192	–	337	88.10	–
Ismael et al. [26]	ResNet50 + SVM	180	–	200	94.7	–
Luz et al. [27]	EfficientNet B3-X	183	5521	8066	–	93.9
Hussain et al. [28]	CoroDet Model	2843	1439	3108	99.1	94.2
Ozturk et al. [31]	DarkCovidNet	127	500	500	98.08	87.02
Our Study	InceptionV3 ResNet50V2	1281	1656	3270	**99.78**	**97.90**
Our Study	CoviXNet	1281	1656	3270	**99.47**	**96.61**

classification. With the proposed CNN model CoviXNet, which aims to detect Covid-19 from Chest X-ray images, a 10-fold accuracy of 99.47% was achieved on 2-class classification and 99.61% on 3-class classification.

The numerous beneficial studies conducted on the subject so far have mostly faced the lack of adequate labeled and heterogeneous data for the proper evaluation of the model performance. This is generally a common problem in medical imaging. Many of the earlier works on Covid-19 detection were unable to find appropriate data, especially for Covid-19 positive lung images, as it had relatively been a much newer disease. The other issue researchers faced was finding a balance between accuracy and computational cost. Many of the state-of-the-art models implemented in previous studies were either trading off on feasibility and computational power by using very deep neural networks to achieve remarkable performance in terms of accuracy or were unable to justify the precision of their methods by using smaller CNN models.

The method proposed, CoviXNet, is a lightweight CNN-based architecture that has shown substantial ability to detect Covid-19 using Chest X-ray images on both binary and 3-class classification. Furthermore, it has demonstrated performance comparable, and in some cases, even better than that achieved by state-of-the-art models on the same dataset while using lesser parameters. This makes the proposed method computationally efficient and increases its feasibility for real-world application.

6. Conclusion and future directions

In this paper the work has explored the utilization of Chest X-ray images which are a feasible, efficient and cost effective method of testing, for the efficient and accurate detection of COVID-19 in patients. We have experimented and analyzed many of the pre-trained models on our dataset and also performed 3-class classification on top performing models. All duplicate images and data leakage issues encountered when training a CNN were addressed in our study.

Furthermore a novel 15 layer CNN architecture CoviXNet has been constructed which has shown a remarkable performance in terms of both accuracy and efficiency. CoviXNet performed extremely well on a diverse and well processed dataset which we believe, outperforms many of the previous proposed models in terms of both accuracy and efficiency.

In the future, this comparative study can help researchers assess the top performing models in recognizing the complex pattern of Chest X-ray. Proposed CoviXNet architecture can also be helpful in other classification tasks in Bio-medical imaging. The CoviXNet model can also be experimented for detection of other lung abnormalities.

We also believe the facilitation of this model in medical equipment and GUI will be extremely helpful for the hospitals and doctors for efficient detection of COVID-19.

CRediT authorship contribution statement

Gaurav Srivastava: Conceptualization, Methodology, Software, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Aninditaa Chauhan:** Conceptualization, Methodology, Software, Writing – original draft, Investigation, Data curation, Formal analysis, Writing – review & editing, Visualization. **Mahesh Jangid:** Conceptualization, Validation, Resources, Writing – review & editing, Supervision. **Sandeep Chaurasia:** Validation, Resources, Writing – review & editing, Supervision.

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.

Acknowledgments

We thank the Department of Computer Science and Engineering, Manipal University Jaipur, India for providing the Deep Learning Research Lab to conduct our experiments.

References

[1] worldometer, Coronavirus Death Toll, https://www.worldometers.info/coronavirus/coronavirus-death-toll/.
[2] Y. Li, L. Yao, J. Li, L. Chen, Y. Song, Z. Cai, C. Yang, Stability issues of RT-PCR testing of SARS-CoV-2 for hospitalized patients clinically diagnosed with COVID-19, J. Med. Virol. 92 (7) (2020) 903–908.
[3] T. Ai, Z. Yang, H. Hou, C. Zhan, C. Chen, W. Lv, Q. Tao, Z. Sun, L. Xia, Correlation of chest CT and RT-PCR testing for coronavirus disease 2019 (COVID-19) in China: A report of 1014 cases, Radiology 296 (2) (2020) E32–E40.
[4] R.C. Nelson, S. Feuerlein, D.T. Boll, New iterative reconstruction techniques for cardiovascular computed tomography: How do they work, and what are the advantages and disadvantages? J. Cardiovasc. Comput. Tomogr. 5 (5) (2011) 286–292.
[5] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[6] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[7] L. Deng, D. Yu, Deep learning: Methods and applications, Found. Trends Signal Process. 7 (3–4) (2014) 197–387.

[8] M.H. Hesamian, W. Jia, X. He, P. Kennedy, Deep learning techniques for medical image segmentation: Achievements and challenges, J. Digit. Imaging 32 (4) (2019) 582–596.

[9] A.A. Ardakani, A.R. Kanafi, U.R. Acharya, N. Khadem, A. Mohammadi, Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: Results of 10 convolutional neural networks, Comput. Biol. Med. 121 (2020) 103795.

[10] Z. Akkus, A. Galimzianova, A. Hoogi, D.L. Rubin, B.J. Erickson, Deep learning for brain MRI segmentation: State of the art and future directions, J. Digit. Imaging 30 (4) (2017) 449–459.

[11] D. Das, K. Santosh, U. Pal, Truncated inception net: COVID-19 outbreak screening using chest X-rays, Phys. Eng. Sci. Med. 43 (3) (2020) 915–925.

[12] E.T. Chin, B.Q. Huynh, L.A. Chapman, M. Murrill, S. Basu, N.C. Lo, Frequency of routine testing for coronavirus disease 2019 (COVID-19) in high-risk healthcare environments to reduce outbreaks, Clin. Infect. Dis. 73 (9) (2021) e3127–e3129.

[13] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A convolutional neural network for modelling sentences, 2014, arXiv preprint arXiv:1404.2188.

[14] Q. Li, W. Cai, X. Wang, Y. Zhou, D.D. Feng, M. Chen, Medical image classification with convolutional neural network, in: 2014 13th International Conference on Control Automation Robotics & Vision, ICARCV, IEEE, 2014, pp. 844–848.

[15] L. Xu, J.S. Ren, C. Liu, J. Jia, Deep convolutional neural network for image deconvolution, Adv. Neural Inf. Process. Syst. 27 (2014) 1790–1798.

[16] P.-H. Dinh, Multi-modal medical image fusion based on equilibrium optimizer algorithm and local energy functions, Appl. Intell. 51 (11) (2021) 8416–8431.

[17] P.-H. Dinh, Combining gabor energy with equilibrium optimizer algorithm for multi-modality medical image fusion, Biomed. Signal Process. Control 68 (2021) 102696.

[18] P.-H. Dinh, An improved medical image synthesis approach based on marine predators algorithm and maximum gabor energy, Neural Comput. Appl. (2021) 1–19.

[19] P.-H. Dinh, A novel approach based on grasshopper optimization algorithm for medical image fusion, Expert Syst. Appl. 171 (2021) 114576.

[20] P.-H. Dinh, A novel approach based on three-scale image decomposition and marine predators algorithm for multi-modal medical image fusion, Biomed. Signal Process. Control 67 (2021) 102536.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2009, pp. 248–255.

[22] U. Sait, K. Lal, S. Prajapati, R. Bhaumik, T. Kumar, S. Sanjana, K. Bhalla, Curated dataset for COVID-19 posterior-anterior chest radiography images (X-Rays), Mendeley Data 1 (2020).

[23] R. Jain, M. Gupta, S. Taneja, D.J. Hemanth, Deep learning based detection and analysis of COVID-19 on chest X-ray images, Appl. Intell. 51 (3) (2021) 1690–1700.

[24] S. Basu, S. Mitra, N. Saha, Deep learning for screening covid-19 using chest x-ray images, in: 2020 IEEE Symposium Series on Computational Intelligence, SSCI, IEEE, 2020, pp. 2521–2527.

[25] H. Panwar, P. Gupta, M.K. Siddiqui, R. Morales-Menendez, V. Singh, Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet, Chaos Solitons Fractals 138 (2020) 109944.

[26] A.M. Ismael, A. Şengür, Deep learning approaches for COVID-19 detection based on chest X-ray images, Expert Syst. Appl. 164 (2021) 114054.

[27] E. Luz, P. Silva, R. Silva, L. Silva, J. Guimarães, G. Miozzo, G. Moreira, D. Menotti, Towards an effective and efficient deep learning model for COVID-19 patterns detection in X-ray images, Res. Biomed. Eng. (2021) 1–14.

[28] E. Hussain, M. Hasan, M.A. Rahman, I. Lee, T. Tamanna, M.Z. Parvez, CoroDet: A deep learning based classification for COVID-19 detection using chest X-ray images, Chaos Solitons Fractals 142 (2021) 110495.

[29] S. Karakanis, G. Leontidis, Lightweight deep learning models for detecting COVID-19 from chest X-ray images, Comput. Biol. Med. 130 (2021) 104181.

[30] A.I. Khan, J.L. Shah, M.M. Bhat, CoroNet: A deep neural network for detection and diagnosis of COVID-19 from chest x-ray images, Comput. Methods Programs Biomed. 196 (2020) 105581.

[31] T. Ozturk, M. Talo, E.A. Yildirim, U.B. Baloglu, O. Yildirim, U.R. Acharya, Automated detection of COVID-19 cases using deep neural networks with X-ray images, Comput. Biol. Med. 121 (2020) 103792.

[32] K. Hammoudi, H. Benhabiles, M. Melkemi, F. Dornaika, I. Arganda-Carreras, D. Collard, A. Scherpereel, Deep learning on chest x-ray images to detect and evaluate pneumonia cases at the era of covid-19, J. Med. Syst. 45 (7) (2021) 1–10.

[33] A.M. Ayalew, A.O. Salau, B.T. Abeje, B. Enyew, Detection and classification of COVID-19 disease from X-ray images using convolutional neural networks and histogram of oriented gradients, Biomed. Signal Process. Control (2022) 103530.

[34] S.H. Khan, A. Sohail, A. Khan, M. Hassan, Y.S. Lee, J. Alam, A. Basit, S. Zubair, COVID-19 detection in chest X-ray images using deep boosted hybrid learning, Comput. Biol. Med. 137 (2021) 104816.

[35] U. Muhammad, M.Z. Hoque, M. Oussalah, A. Keskinarkaus, T. Seppänen, P. Sarder, SAM: Self-augmentation mechanism for COVID-19 detection using chest X-ray images, Knowl.-Based Syst. (2022) 108207.

[36] F. Ertam, G. Aydın, Data classification with deep learning using tensorflow, in: 2017 International Conference on Computer Science and Engineering, UBMK, IEEE, 2017, pp. 755–758.

[37] A. Haghanifar, M.M. Majdabadi, S. Ko, COVID-19 chest X-Ray image repository, 2021, http://dx.doi.org/10.6084/m9.figshare.12580328, URL https://figshare.com/articles/dataset/COVID-19_Chest_X-Ray_Image_Repository/12580328.

[38] T. Rahman, M. Chowdhury, A. Khandakar, COVID-19 radiography database, 2021, Kaggle, https://www.kaggle.com/tawsifurrahman/covid19-radiography-database.

[39] M.E. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M.A. Kadir, Z.B. Mahbub, K.R. Islam, M.S. Khan, A. Iqbal, N. Al Emadi, et al., Can AI help in screening viral and COVID-19 pneumonia? IEEE Access 8 (2020) 132665–132676.

[40] T. Rahman, A. Khandakar, Y. Qiblawey, A. Tahir, S. Kiranyaz, S.B. Abul Kashem, M.T. Islam, S. Al Maadeed, S.M. Zughaier, M.S. Khan, M.E. Chowdhury, Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images, Comput. Biol. Med. 132 (2021) 104319, http://dx.doi.org/10.1016/j.compbiomed.2021.104319, URL https://www.sciencedirect.com/science/article/pii/S001048252100113X.

[41] W. El-Shafai, F. Abd El-Samie, Extensive COVID-19 X-Ray and CT chest images dataset. Mendeley data, V3, 2020.

[42] W.H. Khoong, Covid-19 xray dataset (train & test sets), 2020, Kaggle, https://www.kaggle.com/khoongweihao/covid19-xray-dataset-train-test-sets.

[43] P. Patel, Chest X-ray (Covid-19 & pneumonia), 2021, Kaggle, https://www.kaggle.com/prashant268/chest-xray-covid19-pneumonia.

[44] Çağatay Berke Erdaş, CovID-19 X-ray, 2020, Kaggle, https://www.kaggle.com/cagataybrk/covid19.

[45] m.a. Zainab Ali, COVID-2019 dataset with chest X-Ray images, 2021, Kaggle, https://www.kaggle.com/zainaali/covid2019-dataset-with-chest-xray-images.

[46] S. ghosh, COVID 19 xray image dataset with huge samples, 2020, Kaggle, https://www.kaggle.com/mr3suvhro/covid-19-xray-image-dataset-with-huge-samples.

[47] A. Patel, Covid19 chest xray, 2021, Kaggle, https://www.kaggle.com/aaryapatel/covid19-chest-xray.

[48] M.Z. Islam, COVID-19 chest X-Ray image repository, 2020, Kaggle, https://www.kaggle.com/mdzabirulislam/covid19-chest-xray-image-repository.

[49] N. Sajid, COVID-19 patients lungs X ray images 10000, 2020, Kaggle, https://www.kaggle.com/nabeelsajid917/covid-19-x-ray-10000-images.

[50] D. Deshpande, COVID-19 detection X-Ray dataset, 2020, Kaggle, https://www.kaggle.com/darshan1504/covid19-detection-xray-dataset.

[51] P. Mooney, Chest X-Ray images (Pneumonia), 2018, Kaggle, https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia.

[52] D. Kermany, K. Zhang, M. Goldbaum, et al., Labeled optical coherence tomography (Oct) and chest X-ray images for classification, Mendeley Data 2 (2) (2018).

[53] srikar, Covid-19 Xray images, 2020, Kaggle, https://www.kaggle.com/akkinasrikar/covid19-xray-images.

[54] D. Soni, Data leakage in machine learning, 2019, Towards Data Science, https://towardsdatascience.com/data-leakage-in-machine-learning-10bdd3eec742.

[55] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 1–48.

[56] L. Torrey, J. Shavlik, Transfer learning, in: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques, IGI global, 2010, pp. 242–264.

[57] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, A survey on deep transfer learning, in: International Conference on Artificial Neural Networks, Springer, 2018, pp. 270–279.

[58] P. Marcelino, Transfer learning from pre-trained models, 2018, Towards Data Science.

[59] S. Albawi, T.A. Mohammed, S. Al-Zawi, Understanding of a convolutional neural network, in: 2017 International Conference on Engineering and Technology, ICET, Ieee, 2017, pp. 1–6.

[60] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.

[61] B. Raj, A simple guide to the versions of the inception network, 2018, Towards Data Science.

[62] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[63] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.

[64] I. Allaouzi, M.B. Ahmed, B. Benamrou, An encoder-decoder model for visual question answering in the medical domain., in: CLEF (Working Notes), 2019.

[65] P. Ruiz, Understanding and visualizing DenseNets, 2018, http://www.pabloruizruiz10.com/resources/CNNs/DenseNets.pdf.

[66] C.-F. Wang, The vanishing gradient problem, 2019, Towards Data Science.

[67] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 6105–6114.
[68] A. Shahid, EfficientNet: Scaling of convolutional neural networks done right towards data science, 2020, Towards Data Science, https://towardsdatascience.com/efficientnet-scaling-of-convolutional-neural-networks-done-right-3fde32aef8ff.
[69] K.E. Koech, Cross-entropy loss function, 2020, Towards Data Science, https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e.
[70] R. Dunford, Q. Su, E. Tamang, The pareto principle, 2014.