Computers in Biology and Medicine 149 (2022) 105979

Contents lists available at ScienceDirect

Computers in Biology and Medicine

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

# Ensemble of Deep Neural Networks based on Condorcet's Jury Theorem for screening Covid-19 and Pneumonia from radiograph images

Gaurav Srivastava, Nitesh Pradhan *, Yashwin Saini

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

ARTICLE INFO

*Keywords:*
COVID-19
Pneumonia
Biomedical imaging
Chest X-ray images
Deep feature extraction
Ensemble Learning
Majority voting
Condorcet's Jury Theorem

ABSTRACT

COVID-19 detection using Artificial Intelligence and Computer-Aided Diagnosis has been the subject of several studies. Deep Neural Networks with hundreds or even millions of parameters (weights) are referred to as "black boxes" because their behavior is difficult to comprehend, even when the model's structure and weights are visible. On the same dataset, different Deep Convolutional Neural Networks perform differently. So, we do not necessarily have to rely on just one model; instead, we can evaluate our final score by combining multiple models. While including multiple models in the voter pool, it is not always true that the accuracy will improve. So, In this regard, the authors proposed a novel approach to determine the voting ensemble score of individual classifiers based on **Condorcet's Jury Theorem (CJT)**. The authors demonstrated that the theorem holds while ensembling the **N** number of classifiers in Neural Networks. With the help of CJT, the authors proved that a model's presence in the voter pool would improve the likelihood that the majority vote will be accurate if it is more accurate than the other models. Besides this, the authors also proposed a **Domain Extended Transfer Learning (DETL)** ensemble model as a soft voting ensemble method and compared it with CJT based ensemble method. Furthermore, as deep learning models typically fail in real-world testing, a novel dataset has been used with no duplicate images. Duplicates in the dataset are quite problematic since they might affect the training process. Therefore, having a dataset devoid of duplicate images is considered to prevent data leakage problems that might impede the thorough assessment of the trained models. The authors also employed an algorithm for faster training to save computational efforts. Our proposed method and experimental results outperformed the state-of-the-art with the DETL-based ensemble model showing an accuracy of 97.26%, COVID-19, sensitivity of 98.37%, and specificity of 100%. CJT-based ensemble model showed an accuracy of 98.22%, COVID-19, sensitivity of 98.37%, and specificity of 99.79%.

## 1. Introduction

The novel coronavirus is precedented as the 21st century's most significant threat to humankind, caused by a newly emerged virus, Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) [1]. Coronavirus (COVID-19) commenced its spread in Wuhan, China, in December 2019. It was declared a pandemic by the World Health Organization on 11th March 2020 [2]. This deadly pandemic lost millions of lives due to delayed diagnosis of patients with severe conditions [3].

Across nations, there are two different methods to diagnose COVID-19: real-time polymerase chain reaction (RT-PCR) or chest imaging. The RT-PCR is a nuclear-derived method used for detecting any specific genetic material present in any pathogen as a virus. The problem with RT-PCR is that it is not completely accurate [4,5]. Also, a small number of nurses work 16–17 h shifts a day at medical facilities [6] and the workers were reluctant to enter their employment due to infection worry. As a result, the testing of symptomatic patients was delayed. Although the testing kits have shown high sensitivity to the coronavirus, these tests are still carried out by lab technicians, which adds to the delay. During the peak of each wave that COVID brought with every new variant, hospitals all across the globe were filled with suspected COVID patients. To treat this virus, COVID patients need to wait hours with no cure at hand [7]. Chest imaging can be utilized in place of the conventional RT-PCR test for COVID-19 diagnosis because of its limitations, such as its slow response time.

Computed Tomography (CT) scans and Chest X-ray (CXR) images are the two primary methods for chest imaging. Although radiation is used in X-rays (radiography) to generate a 2-D image, the radiation produced by CT scans is far more damaging to human health than that of chest X-ray images [8]. As a result, we have opted for Chest X-ray (CXR) images instead of CT scans to detect COVID-19 in this

* Corresponding author.
*E-mail addresses:* mailto.gaurav2001@gmail.com (G. Srivastava), nitesh.pradhan@jaipur.manipal.edu (N. Pradhan).

https://doi.org/10.1016/j.compbiomed.2022.105979
Received 2 March 2022; Received in revised form 3 August 2022; Accepted 13 August 2022
Available online 25 August 2022

manuscript. In addition, CT scans are not cost-effective, require heavy infrastructure, and are not as readily available [9] whereas CXR images are more cost-friendly than CT scans. For our approach to making an impact, we needed a detection system that could work for most people, is cost-friendly, highly efficient, and less time-consuming.

During the pandemic, it has been observed that the patients also suffer from Viral Pneumonia [10]. It is a lung infection in which air sacs in an infected person's lungs start to fill with purulent material, causing extreme sickness, which could also lead to the patient's eventual death [11]. In most cases, timely treatment of pneumonia has a high success rate, but COVID-19 pneumonia can be deadly as there is no cure found for COVID-19 that positively works on all infected coronavirus patients [12]. In 2021, during the second wave of COVID-19, the situation in hospitals became even worse than before. Many infected people had severe breathing conditions, and their blood oxygen levels started plummeting, leading to deadly lung disabilities, and even organ failures for hundreds of people [13].

Computer-Aided diagnosis, combined with deep learning models, are often used to enhance the efficiency of the diagnosis and identification of COVID-19 infections from radiological images to minimize human intervention and error [14–16]. There are significant findings regarding these scans verifying that a differential diagnosis of them could be helpful. Recent research works have also shown that CXR images are effective for the early diagnosis of COVID-19 [17]. With all the recent advances in the field of artificial intelligence, we must aid the diagnostics processes of our hospitals using various techniques of Machine Learning and Deep Learning to better equip them for the unforeseen dangers to humanity in our very future [18,19]. Humankind's ability to fight such pandemics has had a great boom with advancing medical tech backed by the exponential growth of research on artificial intelligence [20,21].

Different Deep Convolutional Neural Networks perform differently on the same dataset. So, we leverage the ensemble learning approach to not rely on just one model. Ensemble learning aggregates the decisions made by individual models, hence improving the performance of meta-models compared to the base classifiers. Majority voting is one of the Ensemble approaches we use to enhance the base classifier performance. In majority voting, each voter, i.e., model, casts a vote, and the final decision is considered the majority vote among these. However, it is not always true that the more the number of models we include in our base classifiers group, our accuracy will improve. A less accurate model can make the cumulative decision of all models lesser when included in the voter's pool. In 1785, Marquis de Condorcet proposed several similar assumptions [22]. In this manuscript, with the help of Condorcet's Jury theorem, we prove that the theorem holds while ensembling the $N$ number of classifiers in neural networks. Condorcet's Jury theorem-based ensemble model has also shown competitive results compared to the state-of-the-art.

The recent work done by scientists and researchers to fight such a deadly pandemic and hold coronavirus back motivates us to conduct this research. Implementing preventive measures for the general public and engineering vaccines in such a short period shows how far our technological advancements have come in recent years. Ensemble technique and Transfer Learning have already been used to attain several breakthroughs in healthcare, and biomedical image processing with promising results [23–26]. To achieve the same, our fundamental goal is to design and develop a system using Deep Learning models and an Ensemble Learning method based on Condorcet's Jury Theorem that is completely automated for efficient computerized identification of COVID-19 in CXR images.

In this research, the author's primary contributions are:

1. On the CXR Dataset, the authors applied deep feature extraction techniques using pre-trained DCNN networks to extract deep features. These pre-trained networks are modified according to the CXR dataset. The authors investigated and selected the top-performing DCNN classifiers on the CXR dataset to create a meta-model.
2. To ensemble the base classifiers, the authors proposed a Domain Extended Transfer Learning based ensemble model as a soft voting ensemble method to compare it with hard voting, i.e., majority voting.
3. Using the concept of Condorcet's Jury Theorem, the authors proposed a novel approach to determine the majority voting ensemble score based on individual classifier scores. The authors demonstrated that the theorem holds while ensembling the $N$ number of classifiers in Neural Networks.
4. The authors employed a novel curated dataset, keeping in mind the problem of the duplicate image. This curated dataset was used to ensure the proposed model's robustness in real-world testing.

The remaining contents of the study can be summarized as follows. Section 2 is dedicated to analyzing the previous work of various scholars to detect COVID-19 and Pneumonia. Section 3 deals with the various Materials and Methods used and entailed in the proposed research. Furthermore, Section 4 dives deep into the proposed method. Next, Section 5 discloses and covers experimentation and the results found from the aforementioned experiments. Section 6 adds a brief discussion of proposed approaches in the paper. At last, we add the conclusion of our research with future aspects of it.

## 2. Related works

Implementing Deep Learning is on a spree in Bio-medical Imaging, autonomous navigation, visual recognition, and many other automation technologies today. However, it will be an ineffective use of such powerful technology if we do not use it to solve the more significant problems at hand. For example, from 2020, the novel coronavirus was the most considerable problem humans faced on this planet. To counter it with equal force, many types of research took place in all possible domains. Similarly, deep learning scientists too started using all possible ways to lend a helping hand to medical workers by a computer-aided diagnosis of pneumonia and COVID-19 using CXR images of likely infected individuals.

Ismael [27] proposed three different approaches for binary classification of COVID-19 on CXR images. For their first approach, deep feature extraction was classified with the help of a Support Vector Machines (SVM) classifier and many different combinations of kernel functions. The dataset used for training contained 180 COVID-19 and 200 healthy CXR images. The third approach, i.e., the end-to-end trained deep CNN model, produced an accuracy of 91.6%, a sensitivity of 90%, and a specificity of 93.33%. On fine-tuning, the ResNet50 model showed 92.6% accuracy, sensitivity of 87%, and a specificity of 97.78%. ResNet50 combined with the SVM classifier produces an accuracy of 94.7%, a sensitivity of 91%, and a specificity of 98.89%. Tang [28] proposed an approach to overcome overfitting, high variance problems, and generalization errors caused by using a single Deep Learning network. EDL-COVID ensembles the training results of various models based on open source network architecture. COVID-Net uses a weighted averaging ensembling approach that learns how different sensitivities of various deep learning models vary with different types of classes. These models are trained on COVIDx CXR datasets and give an accuracy of 95% and sensitivity of 96%, which is better than individual COVID-Net models. Finally, Jain [29] took 6432 CXR scans samples from the Kaggle repository, compared InceptionV3, Xception, and ResNeXt models, and reported their accuracy. The Xception model gave the highest accuracy of 97.97%, the sensitivity of 89%, and specificity of 99%. In conclusion, the author stated that this high accuracy obtained may be a cause of concern since it may result from overfitting.

Aminu et al. proposed a new deep learning architecture to detect COVID-19 which is suitable for training over limited data as the proposed architecture has less number of parameters [30]. To prevent the problem of overfitting caused by this lack of data, the L-2 regularization approach is used by the authors along with a global

average pooling layer. Using feature extraction over the CovidNet model and feeding them to various classifiers such as KNN, RF, and SVM, the effectiveness of the CovidNet model is measured. To attain even better performance from this approach, Bayesian optimization is also used to select the optimal parameters for selected classifiers. With the proposed model CovidNet, Aminu et al. achieved an accuracy of 95.81%, a sensitivity of 89.06%, and a specificity of 98.41%. Khan [31] proposed two deep learning frameworks: Deep Hybrid Learning (DHL) and Deep Boosted Hybrid Learning (DBHL), which benefit from data augmentation, TL-based fine-tuning, deep features boosting, and hybrid learning from two developed DCNN models named COVID-RENets-1 and 2. These models are used for hybrid learning and feature boosting of the CXR images of COVID and non-COVID patients from the training dataset, which helps merge both COVID-RENets models and, simultaneously, leave out deficits of these individual models. Proposed models achieved an accuracy of 98.53%, a sensitivity of 99%, and a specificity of 98%.

S-H. Wang et al. [32] proposed a hybrid algorithm in which they use wavelet Renyi entropy to extract deep features from images. The authors used a novel Three-Segment Biogeography-Based Optimization method to update the network weights and biases. The proposed approach was tested on 296 chest CT images with an accuracy of 86.12% ±2.75. The authors have also employed ten runs of 10-fold cross-validation to reduce randomness and get unbiased results. S.-H. Wang et al. [33] proposed a model named CCSHNet for detecting COVID-19 using Chest CT scans. This model comprises three proposed techniques. First, the authors proposed a transfer learning algorithm to extract deep features and set hyperparameters to remove the number of layers. Secondly, the authors proposed a selection algorithm to determine the best two models to create a fusion model. This algorithm selects the best two models identified by the transfer learning algorithm. Lastly, the authors proposed a discriminant correlation analysis algorithm to fuse the two features extracted by the fused models. These proposed techniques outperformed 12 state-of-the-art COVID-19 detection approaches.

A. Khan et al. examined Deep Learning (DL) techniques in depth and created a taxonomy based on diagnostic procedures and learning approaches [34]. This survey sheds light on interesting areas of research in DL for interpreting radiographic images, which might help to speed up the development of tailored DL-based diagnostic tools for dealing with novel COVID-19 variations and future difficulties. In addition, issues in establishing pandemic diagnostic procedures, cross-platform interoperability, and assessing imaging modalities are discussed, as well as the methodology and performance measurements employed in these approaches. S.H. Khan et al. proposed two custom architectures of Convolutional Neural Network, named COVID-RENet-1 and COVID-RENet-2 [35]. These models aim to classify COVID pneumonia and healthy individuals from a dataset of CXR images. Region and Edge-based operations were employed to obtain better information in an image, accompanied by the convolutional operations of the CNN architecture. These models achieved promising results that showed 98% accuracy, 0.96 Matthews correlation coefficient (MCC), and 0.98 F1-score. Furthermore, S.H. Khan et al. also compared their proposed approach with several existing models, resulting in high precision of 98% and sensitivity of 0.98. The comparative analysis of existing techniques can be seen in Table 1.

As per the recent studies we came across in our investigation, a significant downside of many of these proposed implementations is the shortfall of data, ultimately resulting in the overfitting of deep learning models. Overfitting and improper assessment could also occur due to duplicate images in training set in cases such as [29,30,35]. Furthermore, having duplicate images can also affect the proper evaluation of models because of the data leakage problem. Another common observation is the inefficient use of various pre-trained deep learning models, such as in [29]. The pre-trained networks can be modified to perform more efficiently on a specific dataset by cutting down some parameters without impacting the performance much. Based on this, we consider there is still a long way to go for deep learning researchers in COVID detection using CXR images as it would be of great use in equipping us against such pandemics in the upcoming time.

## 3. Materials and methods

### 3.1. Data description

Curated Dataset for COVID-19 Posterior–Anterior Chest Radiography Images (X-rays) [36] is a combined and filtered dataset formed after merging 15 different publicly available datasets. Initially, all these publicly available datasets combined accounted for 4558 COVID-19, 5403 Normal, 4497 Viral pneumonia, and 5768 bacterial pneumonia CXR images. The authors [36] used the Inception V3 model on this combined repository to remove duplicate and defective images. As a result, 1379 COVID-19, 1476 normal, 2690 viral pneumonia, and 2588 bacterial pneumonia duplicate CXR images were removed from the repository. Furthermore, the authors removed clusters of defective images using unsupervised learning methods based on cosine similarity distances. The final refined dataset contains 1281 COVID-19, 3270 Normal, and 1656 viral pneumonia CXR images.

### 3.2. Data preprocessing

In deep learning, the model expects the input image of the same size, but the images in our dataset are not in the same size or shape. Initially, the images in our CXR images dataset were of sizes ranging from 400 × 300 to 936 × 768. However, because the dataset's CXR images are not homogeneous and come in various sizes, we transformed all CXR images into a standard size.

#### 3.2.1. Image resizing

While downscaling the images, we can lose some information, so this has to be done very carefully as a part of the data preprocessing step by observing the dataset. For example, suppose we have a dataset of MRI scans for brain tumor classification. If we downscale the images to a very small size, the tumor will almost disappear from MRI scans, which can impact training accuracy. Also, resizing the image to a very large size like 512 × 512 can exceed the GPU memory. To make it both memory efficient and not lose any critical information from the image, we have to choose the best image size based on the experiments. Also, the Images in our dataset are not isotropic, they are of different sizes, and their aspect ratio varies. Pre-trained networks employed for extracting features expect the input data to be in a uniform size. Therefore, we must resize our images to a standard size to maintain this uniformity.

The main concern here is in what standard size we have to resize all of our images. Either we choose the smallest image size available and scale down all the images larger than that, or we choose the most prominent image size and stretch all the images smaller. While stretching the images, small image pixels are stretched as they are made larger. This might make it difficult for our model to pick up important details like object boundaries. Stretching can be an excellent approach to utilize the most pixels provided to the network if the input aspect ratio is unimportant. However, this also necessitates that we provide similar stretched images to our trained model. In addition, downsizing is less likely to hinder performance if we are detecting objects or classifying images where the area of the distinguishing attributes is the majority of our captured images.

So, experimenting with progressive resizing is a useful tactic. The models in our initial batch will be experimental. We start with smaller image size and examine the image size vs. accuracy and computational cost trade-off as we increase the image size. Additionally, starting with smaller image inputs, we may save time. We have resized and conducted our experiments on all 128×128, 196×196, and 256 × 256 image sizes, and we have observed that the accuracy remains constant for all three image sizes. However, on 128 × 128, training time is much lower, saving much computational cost. In addition, there was no overlap cropping technique used while resizing the images. The image resizing was done with the inbuilt python Pillow library function.

**Table 1**
Comparative Analysis of existing techniques and methods/models proposed by various researchers depicting their advantages and disadvantages.

| Author | Method used | Dataset Description | Advantage(s) | Disadvantage(s) |
|---|---|---|---|---|
| Ismael et al. [27] | ResNet50 + SVM | 180 COVID-19 and 200 Normal CXR images | Computational cost of ResNet50 + SVM is quite good with just 48.9 s. | The proposed technique may not be suitable for the large dataset. |
| Tang et al. [28] | EDL-COVID | 573 COVID-19, 6053 Pneumonia infected and 8851 Normal Chest X-rays. | Ensemble learning approach is used which reduces overfitting, high variance, and generalization errors caused by noise and a limited number of datasets. | The dataset is highly imbalanced as the COVID-19 class has only 573 images. |
| Jain et al. [29] | Transfer learning from InceptionV3, Xception, and ResNeXt models | 6432 Chest X-ray images | Various pre-trained networks have been used and presented for multi-class classification. Xception performed extremely well with an accuracy of 97.97%. | High accuracy obtained is due to overfitting and duplicate images in the used dataset |
| Aminu et al. [30] | CovidNet Architecture | 321 COVID-19, 500 Pneumonia, and 445 Normal CXR images | Proposed CovidNet architecture consists of a relatively small number of parameter which is efficient. | On the multi-class classification the weighted average ensemble based approach could be used. |
| Khan et al. [31] | Deep Boosted Hybrid Learning (DBHL) Framework. | 3224 COVID-19 infected and 3224 Normal Chest X-rays. | Hybrid learning from CNN models named COVID-RENets-1 and 2 has been used which leverages both the models. | Only binary classification has been performed. Also the balanced dataset might contain duplicate images. |
| S-H. Wang et al. [32] | Hybrid algorithm using wavelet Renyi entropy and three-segment biogeography-based optimization. | 148 COVID-19 and 148 Normal chest CT scans. | A novel approach is used to extract deep features from images and update the network weights and biases has been used. 10 runs of 10-fold cross validation have also been used. | Dataset used is very small scale. The neural network requires an adequate amount of data for training as well as the proper evaluation of trained models. |
| S-H. Wang et al. [33] | CCSHNet with deep fusion using transfer learning. | 284 COVID-19, 281 Pneumonia, 293 tuberculosis and 306 Normal Cardiac CT scans. | A novel CCSHNet has been introduced which produced remarkable results. | CXR images can be used in place of CT scans. |
| Saddam Khan et al. [35] | COVID-RENet-1 and COVID-RENet-2 CNN architectures. | 3224 COVID-19 and 3224 Normal CXR images | Region and Edge-based operations were employed to better obtain features in an image. | Multi-class classification can also be performed to test the proposed technique's robustness. Also the balanced dataset might contain duplicate images. |

#### 3.2.2. RGB ordering

Our dataset consists of grayscale images. In this study, we have used the pre-trained networks to extract the high-level features from the COVID-19, Pneumonia, and Normal CXR images. These networks have been previously trained on the large-scale ImageNet Database [37]. The problem is that the ImageNet database contains RGB images and the trained weights are also on those RGB images. This is because the network's input layer expects images to be RGB ordered. We have not modified the model's architecture because the weights were trained using a specific input set. The rest of the weights would essentially be meaningless if we were to replace the first layer with our own.

CNNs are designed to extract higher-level features as they go deeper using the lower-level features extracted from the preceding layers. By removing or altering the initial layers of a CNN, we are disrupting that hierarchy of features because the subsequent layers will not receive the features they are supposed to as their input. The second layer has been trained to expect the features of the first layer. By replacing the first layer with random weights, we are essentially throwing away any training that has been done on the subsequent layers, as they would need to be retrained. They could not recall any information they had acquired during their initial training.

So, In our case, we made our model function with those grayscale images because our dataset contains those. The image has been altered to seem like RGB ordered [38]. A third dimension was added, and the image array was repeated three times. The model's performance should be the same as on RGB images because we will have the same image across all three channels if we do this.

### 3.3. Condorcet's Jury Theorem

Condorcet's Jury Theorem is a mathematical theorem for calculating the relative probability of a group's accumulative decision-making. It states that if a majority of independent members in a group, individually, can make the correct decision rather than making a random choice, they are better at decision-making than just one member of that group [22]. This theorem in applications with Neural Networks helps ensemble the output of multiple trained deep learning models with good outcomes to give results better than any individual models.

Condorcet's Jury theorem applies to the following hypothetical situation: assume we have to choose between options + or −. Assume one of the two choices is 'right,' but we do not know which one [39]. Furthermore, imagine there are $n$ models in a set, and the entire set must make a decision. A majority vote is one feasible way. So, each model has a vote $X_i$, which has a value of either +1 or 1 based on its calculated weights, and the group choice is either + or − depending on whether $S_n = \sum_{i=1}^{n} X_i$ is positive or negative.

#### 3.3.1. Theorem

If individual votes $X_i, i = 1, \ldots, n$ are independent of one another, and each voter makes the correct decision with probability $p > \frac{1}{2}$, then as $n \to \infty$, the group's chance to reach a correct decision by majority vote approaches 1 as $n$ increases [40]. Fig. 3 shows that as the number of voters increases (value of n), the likelihood of reaching the right choice by majority vote increases.

#### 3.3.2. Proof

This is a consequence of the law of large numbers. Let $a = p - 1/2 > 0$. Since the problem is fair in + and -, we may, without loss of generality, assume the correct answer is + [22].

Then $EX_1$ is $> 0$ as shown in Eq. (1) and the weak law of large numbers states that $\frac{S_n}{n}$ converges in probability to $EX_1 = 2a$, where by converging in probability we mean that for any $\epsilon_1, \epsilon_2 > 0$ there is $N$ large enough such that for every n as shown in Eq. (2).

$$EX_1 = -\left(\frac{1}{2} - a\right) + \left(\frac{1}{2} + a\right) = 2a > 0 \tag{1}$$

$$n \geq N, P\left(\left|\frac{S_n}{n} - EX_1\right| < \epsilon_1\right) > 1 - \epsilon_2 \tag{2}$$

Taking $\epsilon_1 = 2a$, we see in Eq. (3) the probability of a correct decision tends to be 1.

$$P\left(S_n > 0\right) = P\left(\frac{S_n}{n} > 0\right) \geq P\left(\left|\frac{S_n}{n} - 2a\right| < 2a\right) \rightarrow 1 \tag{3}$$

The probability, $P_N$, that a model will deliver the correct answer, we calculated using Condorcet's jury theorem [41] as shown in Eq. (4).

$$P_N = \sum_{i=m}^{N} \left(\frac{N!}{(N-i)\cdot i!}\right)(p)^i(1-p)^{N-i} \tag{4}$$

where, $N$ = the number of models, $p$ = the probability of an individual model being right $m$ = the number of models required for a majority

Proof Based on the optimal Bayes classifier is available in the supplementary section of the manuscript.

## 4. Proposed work

### 4.1. Method overview

In deep learning, Ensembling is used to counter the high variance problem of neural networks, where multiple models are trained, and their predictions are added to improve results [42]. As observed in recent research, Ensemble methods give more accurate solutions as they add some bias to the prediction, which balances out the variance created by using just one neural network trained on the same dataset [43]. Many algorithms may combine various classifiers, with the majority vote being the most straightforward [44]. Despite its simplicity, it has been suggested that the majority vote is the optimal technique if the mistakes among the classifiers are not connected.

Each classifier casts a single vote, and the class with the most votes wins in hard voting (also known as majority voting). The ensemble's anticipated target label is the distribution mode of the labels' predictions. However, it is not always true that increasing the number of voters will improve the likelihood that the final choice will be correct. The group's overall prediction can occasionally be less accurate if we include a less reliable voter. In 1785, Marquis de Condorcet proposed several similar assumptions [22]. First, the theorem assumes that everyone in the group wants to choose by a majority vote. Second, each voter has an independent probability p of selecting one of the two possibilities that will result in the right choice. Third, the theorem asks how many voters we should include in the group. The result depends on whether p is greater than or less than 1/2.

The relative likelihood of a particular group of individuals reaching the right conclusion is the subject of Condorcet's jury theorem, a political science theory. The theorem has not yet been proven in neural networks, as per our best knowledge. We argue, however, that this theorem is true when trained models cast a vote. A model's presence in the voter pool will improve the likelihood that the majority vote will be accurate if it is more accurate than the other models. However, the likelihood that the majority vote will be accurate declines as a less accurate model is introduced to the group.

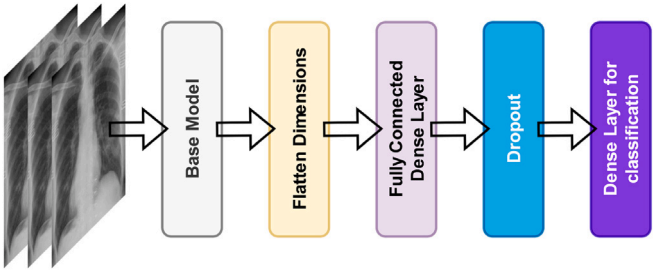


**Fig. 1.** Modified pre-trained DCNN Model Architecture used for extracting features from CXR images.

### 4.2. Deep feature extraction and model training

In this manuscript, we have used pre-trained networks — InceptionV3, InceptionResNetV2, ResNet50V2, DenseNet121, DenseNet201 [45–48] to extract high-level deep features from the CXR images. Furthermore, we have modified these architectures to work with the CXR dataset. From Fig. 1, after extracting the deep features, we flatten the dimensions, freezing all layers except the final layer of the network. After flattening, we get a feature vector consisting of all extracted features. To classify these features into their respective classes, we fed this obtained feature vector to a multi-layer perceptron network of a fully connected dense layer consisting of 1024 neurons.

These pre-trained networks tend to overfit when trained on relatively limited datasets since they are deep and massive networks. These pre-trained networks have already undergone training using the ImageNet database, which is a far larger dataset than the collection of chest X-ray images. So, we utilized a dropout layer to prevent the model from overfitting. *Dropout* is a regularization technique simulating several neural networks' concurrent training with various architectures. For the input layer, the dropout ($p$) value should be kept at about 0.2 or lower [49]. This is because dropping the input data can adversely affect the training. $p > 0.5$ is not advised, as it culls more connections without boosting the regularization. For intermediate layers, choosing $p = 0.5$ for large networks is ideal [49]. This is because the regularization parameter, $p(1-p)$ in Eq. (5), is maximum at $p = 0.5$.

$$E_R = \frac{1}{2}\left(t - \sum_{i=1}^{n} p_i w_i I_i\right)^2 + \sum_{i=1}^{n} p_i\left(1 - p_i\right) w_i^2 I_i^2 \tag{5}$$

So, we have added a dropout layer of 0.5 value to remove 50% of neurons in each iteration to avoid overfitting. Finally, to map these classified features to their respective classes, we have added a dense layer consisting of three neurons and the softmax activation function for classification purposes.

#### 4.2.1. Loss function

Since our dataset has multi-class, categorical Cross-Entropy is the loss function we have utilized here. Cross-Entropy refers to the variance between two probability distributions, so the cross-entropy loss changes according to the difference between the predicted probability and the actual label [50]. Eqs. (6) and (7) explains the computation of the cross-entropy loss function:

$$L_{\text{CE}} = -\sum_{i=1}^{n} t_i \log\left(p_i\right), \text{ for n classes,} \tag{6}$$

where $t_i$ is the truth label and $p_i$ is the Softmax probability for the $i$th class.

$$J(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(\hat{y}_i\right) + \left(1 - y_i\right)\log\left(1 - \hat{y}_i\right)\right] \tag{7}$$

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

#### 4.2.2. Optimizer

To optimize our DCNN models, we have employed an Adam optimizer. Adam is intuitively a combination of RMSProp and Stochastic Gradient Descent with Momentum. It uses the moving average of the gradient in place of the gradient itself, like in SGD with momentum, and it uses squared gradients to scale the learning rate like in RMSProp. Adam optimizer uses an exponentially decaying average of past gradients ($m_t$) and past squared gradients ($v_t$) as defined in Eqs. (8) and (9) respectively. The term $\beta_1$ and $\beta_2$ are the forgetting factors for the mean and non-centered variance of the gradient, respectively. Through the experiments, the authors identified that the fair values of $\beta_1$ and $\beta_2$ are 0.9 and 0.999, respectively.

$$m_t = \beta_1 m_{t-1} + \left(1 - \beta_1\right)\left[\frac{\delta L}{\delta w_t}\right] \tag{8}$$

$$v_t = \beta_2 v_{t-1} + \left(1 - \beta_2\right)\left[\frac{\delta L}{\delta w_t}\right]^2 \tag{9}$$

#### 4.2.3. Classifier

The softmax activation function and a multi-layer perceptron network are used to classify the obtained feature vector. Softmax function applies on a vector of logits, that the last fully connected layer of the CNN outputs. It transforms these logits into relative probabilities to sort out the desired classes in multiclass classification [51,52]. Softmax function is defined in Eq. (10).

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{10}$$

where, $\sigma$ = softmax, $\vec{z}$ = input vector, $e^{z_i}$ = standard exponential function for input, $K$ = number of classes in the multi-class, $e^{z_j}$ = standard exponential function for output.

#### 4.2.4. Learning rate schedule: ReduceLROnPlateau

ReduceLROnPlateau is used for model training here as a learning rate scheduler. It watches a quantity and reduces the learning rate if no progress is noticed after a 'patience' number of epochs [53]. When the decay is considered, the learning rate may be calculated as shown in Eq. (11).

$$\eta_{n+1} = \frac{\eta_n}{1 + dn} \tag{11}$$

Where $\eta$ is the learning rate, $d$ is a decay parameter, and $n$ is the iteration step.

When a measure stops improving, this callback reduces the learning rate. This callback tracks a quantity and reduces the learning rate by a "factor" value if no progress is noticed after a "patience" number of epochs, as shown in Eq. (12).

$$\text{new } lr = lr * \text{ factor} \tag{12}$$

### 4.3. Proposed ensemble methods

Two different voting schemes are common among voting classifiers:

1. In **soft voting**, every individual classifier provides a probability value that a specific data point belongs to a particular target class. The predictions are weighted by the classifier's importance and summed up. Then the target label with the greatest sum of weighted probabilities wins the vote.
2. In **hard voting** (also known as **majority voting**), every individual classifier votes for a class, and the majority wins. In statistical terms, the predicted target label of the ensemble is the mode of the distribution of individually predicted labels.

The authors proposed two ensemble techniques: Domain Extended Transfer Learning (DETL) Ensemble, which is based on soft voting, and Majority voting, based on Condorcet's Jury Theorem.

**Algorithm 1:** Algorithm for Domain Extended Transfer Learning Ensemble model training

**input:** Training set ($\delta 1$), Validation set ($\delta 2$), Testing set ($\delta 3$)
$\xi \longrightarrow$ iteration step $\xi \leftarrow$
$\lambda \longrightarrow$ iteration step $\lambda \leftarrow$
**output:** : Classification as COVID-19, Normal or Pneumonia CXR
**begin:**
**while** $\xi = 1 \leq \alpha$ **do**
  1. Set the input layer of CNN Model $\xi$;
  2. Set the head layers $CNN_{\text{flatten}}$, $CNN_{\text{dense}}$, $CNN_{\text{dropout}}$
  3. Initialize the CNN parameters: $\mu$, $\epsilon$, and $\beta$
  4. Train the CNN and compute the initial ($\omega^*$)
  **while** $\lambda = 1 \leq \epsilon$ **do**
    5. Randomly select a mini-batch (size : $\beta$) from $\delta 1$
    6. Forward propagation and compute the loss using Eq. (13)

$$J = \sum\left(\frac{1}{2} \times \left(Y_{\text{expected}} - Y_{\text{output}}\right)^2\right) \tag{13}$$

    7. Back propagate the error and update the weights using Eq. (14) with adam optimizer

$$W_n = W_n - \eta * \frac{\partial J}{\partial W_n} \tag{14}$$

    8. Repeat steps 5 to 7 until the total loss becomes minimum.
  **end**
  9. Save CNN weights ($\omega^*$) as $model_{\xi}.h5$
  **return** $model_{\xi}.h5$
**end**
**while** $\xi = 1 \leq \alpha$ **do**
  10. Freeze all layers of model $\xi$ except the output layer
**end**
11. Concatenate the output layer of all trained models
12. Set the dense layer ($CNN_{\text{dense}}$)
13. Set the output layer of the Ensemble Model
**while** $\lambda = 1 \leq \epsilon$ **do**
  14. Train the ensemble model.
**end**

#### 4.3.1. Domain Extended Transfer Learning (DETL) ensemble

In this proposed approach, first, we train the $N$ number of the best-performing model. We can discover the value of $N$ by experimenting with various models on the provided dataset. Because specific models may perform well on a dataset while others do not, experimenting is the best technique for choosing models. We now train and save the weights of all models one by one. After training all the models, we freeze all the layers except the top one and concatenate the output layers of all models, followed by a dense layer of 32 neurons, and finally, a three-neuron output layer for classification. This proposed approach is demonstrated graphically in Fig. 2.

The detailed training procedure of the DETL model can be seen in Algorithm 1 where $\delta 1$, $\delta 2$, and $\delta 3$ refer to the training, validation, and testing sets, respectively. $\alpha$ is the total no. of models to be trained to create a final ensemble model. $\mu$ is the learning rate of a model. It usually has a value close to zero. $\epsilon$ is the total number of iterations for which the CNN model has been trained (also known as epochs). $\beta$ is a second customizable hyper-parameter with values of $2^n$ (also known as batch size). ($\omega^*$) are the CNN weights.

#### 4.3.2. Majority voting based on condorcet's jury theorem

**Condorcet's Theorem in Neural Networks:** Given a set of models that must choose between a right conclusion with probability $0 \leq p \leq 1$ and a wrong one with probability $1 - p$, Condorcet's jury theorem [54] states:

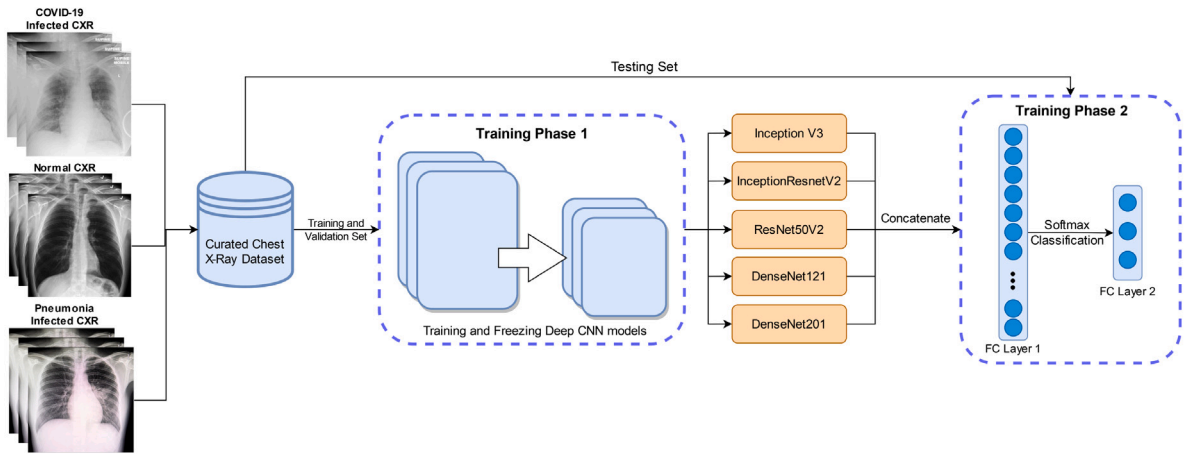


**Fig. 2.** Flowchart of the training procedure for both phases in DETL-based ensemble model. The first phase is the training of DCNN models and saving their weights. The second phase is training the last layer after concatenating the base classifiers.

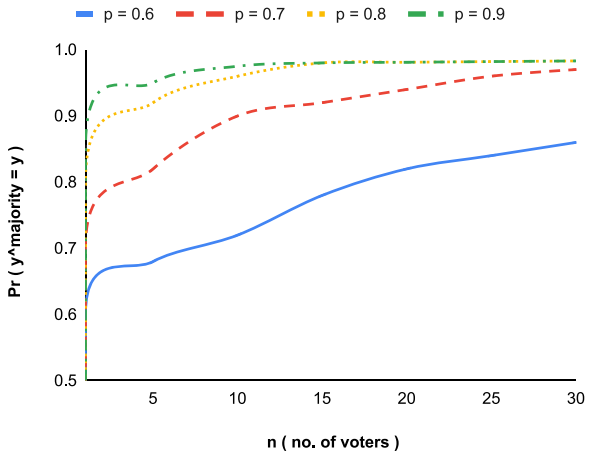


**Fig. 3.** Condorcet's Jury Theorem Curve depicting the probability of majority vote to be right vs. the no. of voters.

1. If $p > 1/2$ (i.e., each model is more likely to classify correctly than incorrectly), increasing the number of models improves the likelihood that the majority selects correctly. The probability of a correct decision approaches one as the number of models increases, as shown in Fig. 3.
2. if $p < 1/2$ (such that each model is less likely to vote erroneously than correctly), adding additional models reduces the likelihood that the majority selects appropriately, and the probability of a correct judgment is maximized for a model of size one.

In this proposed Algorithm, we input the $\alpha$ number of trained classifiers. We then record the predicted output labels by supplying each image in the testing test to all the classifiers. We now generate $X$ number of arrays for $X$ number of labels after recording the expected labels of each model for each image. Here, in our case, the value of $X$ is 3 since we have 3 output labels, i.e., normal, covid, and pneumonia. We then iterate through each of the classifier's output score array and count the number of votes for all normal, covid, and pneumonia classes. Finally, we iterate over all the arrays and store the majority vote count in the final score $f$ array. This proposed approach is demonstrated in Fig. 4.

The majority voting based on Condorcet's Jury Theorem is used to calculate the final score as per Algorithm 2. The notations used in Algorithm 2 are $\delta$ referring to the testing set. $\alpha$ is the total number of trained classifiers used in the voting ensemble. $\beta$ is the length of the testing set. In the output, we get a classification of the input image as either Normal, COVID-19 infected, or pneumonia infected.

**Algorithm 2:** Algorithm for final score calculation from Majority Voting Based on Condorcet's Jury Theorem

**Input:**
trained models ($\alpha$)
testing set ($\delta$)
size of the testing set ($\beta$)
**Output:** : Classification as COVID-19, Normal or Pneumonia CXR
**begin:**
1. Predict the score of each $\alpha$ and store it in an array of length $\beta$
2. Initialize $normal(n)$, $covid(c)$ and $pneumonia(p)$ arrays of length equal to length of $\beta$ with 0s.
**while** $i \leq \alpha$ **do**
  **while** $j \leq i$ **do**
    3. If $j == 0$ then $n++$
    4. If $j == 1$ then $c++$
    5. If $j == 2$ then $p++$
  **end**
**end**
6. Initialize $final\ score\ f$
**while** $i \leq \beta$ **do**
  **if** $n[i] >= c[i]$ **and** $n[i] >= p[i]$ :
    $f$.append(0)
  **elif** $c[i] >= n[i]$ **and** $c[i] >= p[i]$ :
    $f$.append(1)
  **elif** $p[i] >= n[i]$ **and** $p[i] >= c[i]$ :
    $f$.append(2)
**end**
7. Calculate the final score between original labels and final predicted labels based on Condorcet's Jury Theorem.

## 5. Experimental results

### 5.1. Dataset division

The authors divided the entire dataset into an 80% training set and the remaining 20% into a 50% validation set and a 50% testing set. As a result, we end up with 80% training data, 10% cross-validation data, and 10% final testing data, as shown in Table 2.

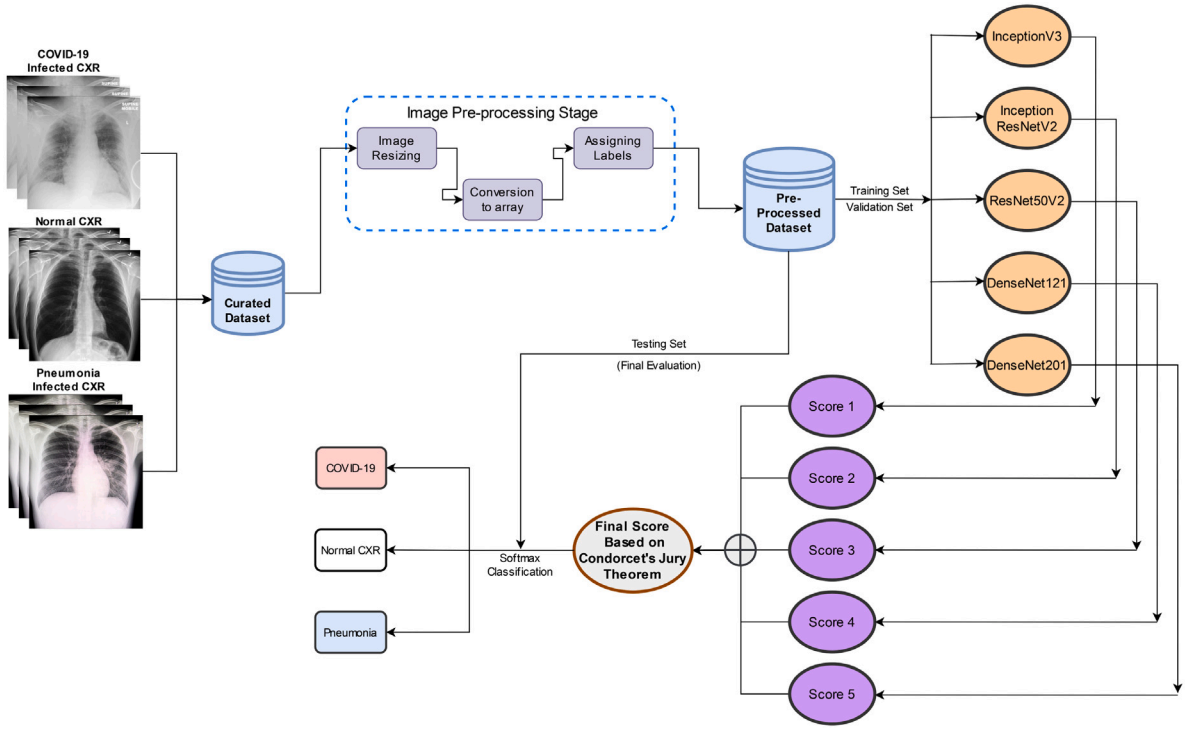


**Fig. 4.** Flowchart for majority voting score calculation of Ensemble model based on Condorcet's Jury Theorem.

**Table 2**
Class distribution of the dataset into training, validation, and testing set used to evaluate the proposed method.

| | Normal | COVID-19 Infected | Pneumonia infected |
|---|---|---|---|
| Training set | 2616 | 1025 | 1326 |
| Validation set | 327 | 128 | 165 |
| Testing set | 327 | 128 | 165 |

For training the base models i.e., InceptionV3, InceptionResNetV2, ResNet50V2, DenseNet121 and DenseNet201, we have used 80% of the total dataset. For training our meta learner (i.e., DETL Ensemble model), we cannot directly input the dataset as we did in our base models because the ensemble model requires input at five places while only one output is generated. So, we modified our training, validation, and testing set to provide the images as five inputs at a time. As a convention, we use different data partitions to train our base and meta learners. The base models are trained on an 80% training set, and the meta learner is trained on the remaining validation set to avoid overfitting. However, the overfitting in our trained DETL-based ensemble model did not occur, so we trained our meta learner on the training set itself.

The Jury based ensemble model does not require separate training as it just calculates the majority vote from trained base models and outputs the final prediction based on Condorcet's Jury Theorem. So, we have used the same data partitioning to train our base models.

### 5.2. Faster training algorithm

The input layer of a neural network is composed of artificial input neurons and brings the initial data into the system for further processing by subsequent layers of artificial neurons. The input layer in CNN contains image data and is represented as a three-dimensional matrix.

As a convention, we fed the batches of images into the model in every iteration. For example, suppose we take a batch size of 32, and the total number of images is 3200; one epoch will run for 100 iterations. In each iteration, firstly, the batch of 32 images will be converted to a 3-dimensional array as the input layer of a CNN takes a 3d matrix as input. Then the network calculates the predicted value and the loss by subtracting it from the actual output value. The error is then calculated, and the network propagates to update the weights.

In this whole procedure, the conversion of images to array is repetitive as it will start repeating from epoch 2. So, in this faster training algorithm, we convert all the images to an array before training itself and feeding it into the model. However, this will require some memory to store the 3d matrix in an array, but it can save much computational cost while training in a smaller dataset.

#### 5.2.1. Algorithm

This algorithm illustrates the pre-processing steps of the dataset for faster training. The notations used in Algorithm 3 are $\delta 1$, $\delta 2$, and $\delta 3$ referring to the training, validation, and testing datasets, respectively. $\mu$ is the total no. of classes we have. Because we have three classes, the value of $\mu$ here is 3. $\epsilon$ is the total no. of images in each class. The final dataset and label arrays are $\alpha$ and $\beta$, respectively. The image's width and height are the $w$ and $h$.

The algorithm defines the directories, initializes the dataset and label array, and sets the input image's width and height (refer to steps 1, 2, and 3 of Algorithm 3). Then, after reading each image, iterating through all the images in each class, the images are transformed into an array (refer to steps 5–7 of Algorithm 3). The image is then adjusted to the width and height specified in the input (refer to step 8 of Algorithm 3). After each image has been RGB ordered and added to the dataset array, a label is added to the label array. In our application, the labels 0, 1, and 2 represent normal CXR, COVID-infected CXR, and Pneumonia-infected CXR, respectively (refer to steps 10 of Algorithm 3).

### 5.3. Experimental setup

Tensorflow was used to implement the proposed method in Python. To evaluate the working of the code, minor epochs of training are done on a personal computer with an Intel(R) Core(TM) i7-6500U CPU 2.50 GHz, Nvidia 940M GPU with computational capability 5.0,

**Algorithm 3:** Dataset Preprocessing Algorithm for faster training

**input:** Chest X-Ray images
$\xi \longrightarrow$ iteration step for $\mu$ classes $\xi \leftarrow$
$\lambda \longrightarrow$ iteration step for $\epsilon$ classes $\lambda \leftarrow$
**output:** : Training set ($\delta 1$), Validation set ($\delta 2$), Testing set ($\delta 3$)
**begin:**
1. Read the directories
2. Initialize $\alpha$ and $\beta$
3. Set $w$ and $h$
4. Initialize $i = 0$
**while** $\xi = 1 \leq \mu$ **do**
  **while** $\lambda = 1 \leq \epsilon$ **do**
    5. Read the image
    6. Convert the image into an array
    7. Resize the image to $w \times h$
    8. Apply RGB Ordering to the image. Now the image dimensions are $w \times h \times 3$
    9. Append the array into the $\alpha$
    10. Append $i$ in $\beta$
  **end**
  11. $i++$
**end**
12. Split the dataset into 80% $\delta 1$ and 20% dump Set
13. Split the dump set to 50% $\delta 2$ and 50% $\delta 3$.

and 16 GB RAM. To obtain our final findings, we completed the whole training phase on Kaggle using a GPU Tesla P100-PCIE-16 GB computing capability: 6.0 and 16 GB GPU RAM. To achieve the best training, validation, and testing accuracy, each Model, was trained for 1000 epochs.

### 5.4. Results

#### 5.4.1. Gradient-weighted class activation mapping visualization

While deep learning has achieved remarkable accuracy in image classification, model interpretability remains one of the most prominent issues. Deep learning models are frequently considered "black box" approaches, with no clear understanding of where the network looks in the input image and how it arrived at its final output. This poses an intriguing question about how we can trust a model's judgments if we cannot fully evaluate how it arrived at those conclusions.

Selvaraju et al. [55] developed a Gradient-weighted Class Activation Mapping (Grad-CAM) to assist deep learning practitioners in visually debugging their models and accurately comprehending where they are looking in an image. Grad-CAM generates a heatmap representation for a specified class label (either the top, predicted label, or an arbitrary label we select for debugging). This heatmap may visually check where the CNN is looking in the image. Grad-CAM leverages any target idea's gradients, which flow into the final convolutional layer to create a coarse localization map highlighting the image's essential locations for concept prediction. We can visually validate where our Model is looking with Grad-CAM, ensuring that it looks at the correct patterns in the image and activates around them. Fig. 5 shows where the Model looks in Normal, COVID-19, and Pneumonia CXR images while predicting their output labels.

#### 5.4.2. Classifier performance

In the CXR Dataset, the authors did a 3-class classification to test the proposed technique. We used multi-class classification instead of binary classification since discriminating between three groups (COVID-19, Normal, and Pneumonia) is more challenging. This is because CXR infected with pneumonia and CXR infected with COVID-19 have a lot more parallelism. We will need some of the CXR dataset's finest pre-trained models to put the proposed method to the test. Because various

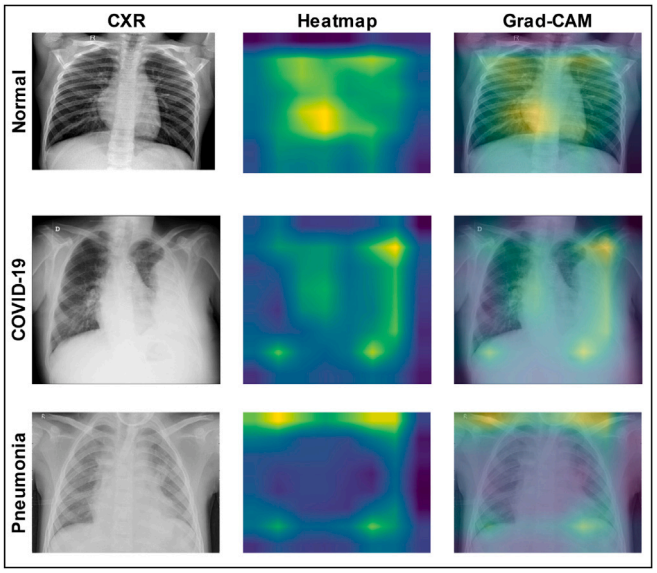


**Fig. 5.** Grad-CAM Visualization of COVID-19, Pneumonia and Normal CXR Images with Heatmap and Saliency maps generated by Grad-Cam.

**Table 3**
Validation (VA) and testing (TA) accuracies of various pre-trained DCNN models without and with scheduled learning rate (LRS).

| | Without LRS | | With LRS | |
|---|---|---|---|---|
| Classifier | VA (%) | TA (%) | VA (%) | TA (%) |
| VGG16 | 94.69 | 95.65 | 95.49 | 95.1 |
| VGG19 | 91.79 | 94.04 | 92.91 | 92.59 |
| Xception | 95.81 | 93.4 | 95.17 | 94.52 |
| InceptionV3 | 95.17 | 96.3 | 95.65 | 96.45 |
| InceptionResNetV2 | 95.81 | 95.97 | 96.14 | 97.42 |
| ResNet50 | 87.44 | 86.63 | 87.28 | 87.76 |
| ResNet50V2 | 96.46 | 96.62 | 97.26 | 97.9 |
| ResNet101V2 | 96.78 | 96.77 | 95.17 | 94.36 |
| ResNet152V2 | 95.97 | 96.62 | 94.85 | 95.65 |
| DenseNet121 | 96.94 | 95.97 | 97.26 | 97.75 |
| DensetNet169 | 95.97 | 95.16 | 95.81 | 96.13 |
| DensetNet201 | 96.94 | 94.84 | 95.49 | 97.26 |
| EfficientNetB1 | 92.59 | 94.36 | 95.65 | 95.97 |

models behave differently on different datasets, we cannot choose such models directly. Therefore, we experimented with various pre-trained models with varied parameters and chose the best.

On multi-class classification, InceptionV3, InceptionResNetV2, ResNet50V2, DenseNet121, and DenseNet201 performed exceptionally well. In addition, we used these models with a scheduled learning rate as a callback. This considerably improves the accuracy. The validation and testing accuracies with and without the learning rate schedule are shown in Table 3. The first column of Table 3 represents the classifier, whereas the second and third column represents the validation accuracy (VA) and final testing accuracy (TA) without Learning Rate Schedule (LRS), and the fourth and fifth column represents the VA and TA with LRS respectively.

To optimize our DCNN models, we have used an Adam optimizer with an initial learning rate of 0.001, the exponential decay rate for the first moment as 0.9, the exponential decay rate for the second moment as 0.999, and an epsilon value of 1e−7. Experimentation has determined that the learning rate and other hyperparameters are the most ideal settings per Table 4. These hyperparameters are selected utilizing the Grid search technique for model tuning and optimization. The batch size is set at 32, and the model training is halted at 1000 epochs. As a consequence, the best model weights are preserved. To conserve the finest weights, the authors employed an early stopping

**Table 4**
Hyperparameters used to train the DCNN and the proposed models.

| Hyper-parameters | Values |
|---|---|
| Optimizer | Adam |
| Dropout | 0.5 |
| Batch Size | 32 |
| Exponential Decay rate for 1st momentum ($\beta_1$) | 0.9 |
| Exponential Decay rate for 2nd momentum ($\beta_2$) | 0.999 |
| Epsilon ($\epsilon$) | 1e−7 |
| Initial Learning rate ($\alpha$) | 0.001 |
| Factor | 0.1 |
| Patience | 10 |
| Total no. of Epochs | 1000 |

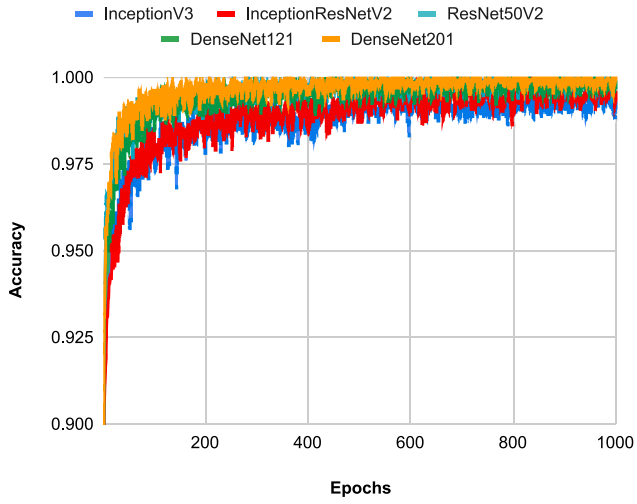


**Fig. 6.** Accuracy VS Epochs curve of top performing DCNN models during the training procedure employed before proposed ensemble method.

callback. When a monitored parameter stops improving, early stopping stops the training.

The accuracy of the developed DCNN models is related to the number of epochs. The accuracy value rises when the number of epochs increases from 1 to 1000. Around epoch 650, the accuracy of various implemented models appears to be constant, as shown in Fig. 6. The magnitude of loss is also dependent on the number of epochs. When the number of epochs is increased from 1 to 1000, the value of loss decreases as shown in Fig. 7.

After Ensembling the top-performing models using the proposed DETL approach, the final ensemble model's accuracy increased significantly. The loss and accuracy curves of the Proposed DETL Ensemble method are shown in Figs. 8 and 9 respectively.

Furthermore, we have ensembled the top-performing models using Condorcet's jury theorem, improving the model accuracy significantly. According to Condorcet's Jury Theorem, if each classifier votes with a probability $p > \frac{1}{2}$ then the final ensemble model's chance to reach a correct decision by majority vote approaches 1. Our 5 selected voters are InceptionV3, InceptionResNetV2, ResNet50V2, DenseNet121 and DenseNet201. We can observe their initial accuracy from Table 3. From Table 5, we can see that when the no. of voters was 2, initially, the accuracy was 95.33%. As we increased the no. of voters, the accuracy also increased.

From Table 3, we can also observe that the highest accuracy among all the voters is of ResNet50V2. According to the theorem, when a voter with a high probability is added, the chances of getting the decision right by a majority vote increase. So, from Table 5, we can see that when the ResNet50V2 is added to the group of voters, the accuracy increases by a margin of 2.41%.

From Fig. 11 and Table 5, it has been proved that in neural networks application, Ensembling $N$ no. of DCNN classifiers based on

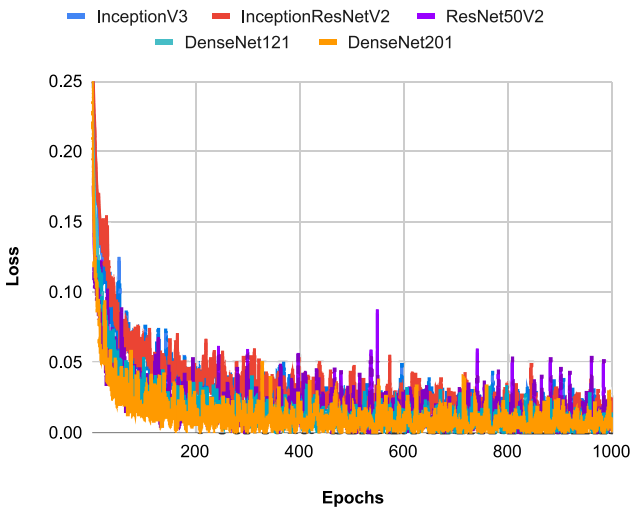


**Fig. 7.** Loss VS Epochs curve of top performing DCNN models during the training procedure employed before proposed ensemble method.

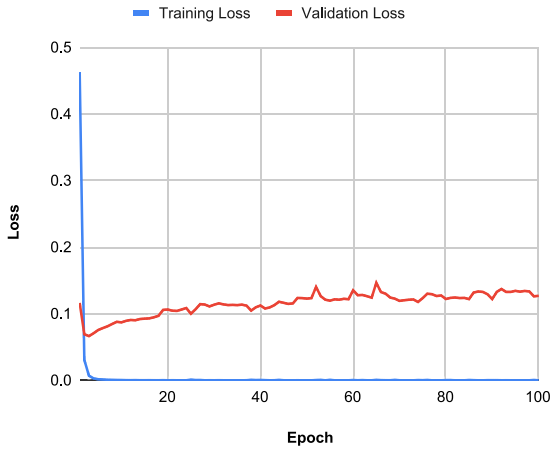


**Fig. 8.** Loss VS Epochs curve of DETL ensemble model during the training procedure.

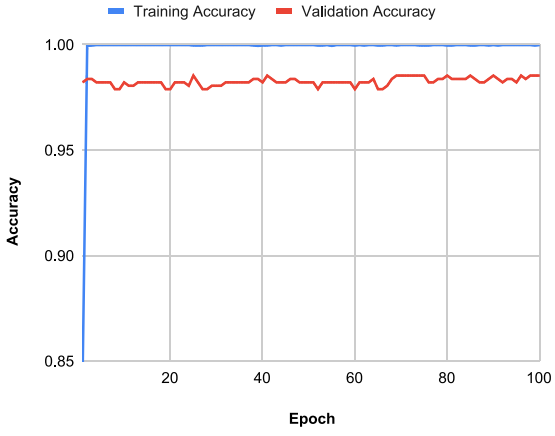


**Fig. 9.** Accuracy VS Epochs curve of DETL ensemble model during the training procedure.

Condorcet's Jury Theorem, the accuracy increases as the no. of DCNN classifiers increases. The proposed DETL and Jury Ensemble method shows a trailblazing accuracy of 97.26% and 98.22%, respectively.

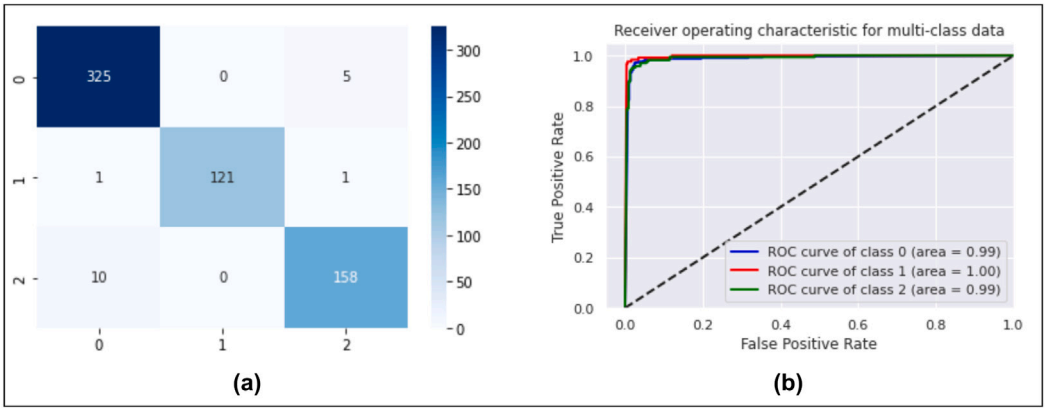


**Fig. 10.** (a) Confusion Matrix of DETL ensemble Model (b) ROC curve of DETL ensemble Model.

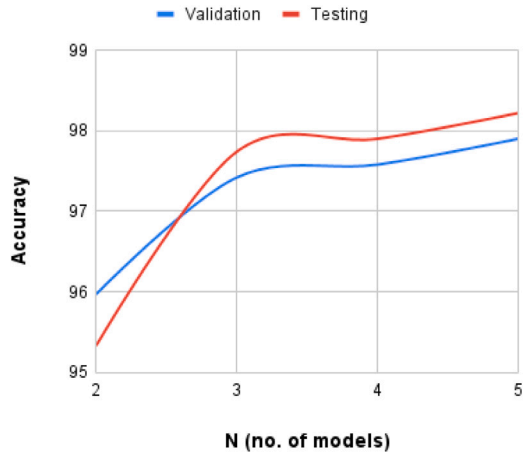


**Fig. 11.** Curve depicting the proof of Condorcet's Jury Theorem in Neural Networks. Chances of getting the right decision increase as more no. of accurate voters are added.

**Table 5**
Accuracy of Condorcet's Jury Theorem based ensemble model with N no. of classifiers.

| No. of classifiers | Classifiers | Jury's VA | Jury's TA |
|---|---|---|---|
| 2 | InceptionV3<br>InceptionResNetV2 | 95.97 | 95.33 |
| 3 | InceptionV3<br>InceptionResNetV2<br>ResNet50V2 | 97.42 | 97.74 |
| 4 | InceptionV3<br>InceptionResNetV2<br>ResNet50V2<br>DenseNet121 | 97.58 | 97.90 |
| 5 | InceptionV3<br>InceptionResNetV2<br>ResNet50V2<br>DenseNet121<br>DenseNet201 | 97.90 | 98.22 |

Tables 6 & 7 compare the proposed DETL and Jury Ensemble model's accuracy and other evaluation metrics against other models.

A *confusion matrix* is a critical metric used to accurately and easily depict how a model is working on classifying the positive and negative cases compared to the actual data. Fig. 12 shows the confusion matrices of top-performing models, where the rows depict instance classes and the columns represent actual classes. In Fig. 14, we can also see the confusion matrix of the ensemble model based on Condorcet's Jury Theorem.

From Fig. 13 we can see the area under the curve for pre-trained DCNN models on 3-class classification. The confusion matrix and ROC curve of the proposed DETL-based ensemble model can also be seen in Fig. 10.

### 5.5. Comparative analysis

#### 5.5.1. Based on accuracy

There are numerous publicly available datasets in this COVID-19 classification. Different authors have implemented and claimed the performance of their models and approaches on various datasets. Some of them also combined many datasets and utilized them to test their models and methods. As a result, we cannot make direct comparisons between our model and those research. So, to better its correctness, we compare our model to earlier studies and their accuracy in the 3-class classification of CXR images with their dataset information. Our proposed DETL ensemble model and Ensemble approach based on Condorcet's Jury Theorem beats the accuracy of various state-of-the-art models and approaches provided in earlier studies, as shown in Table 9.

#### 5.5.2. Based on efficiency

Convolutional neural networks have recently shown outstanding results in various computer vision applications. However, due to the high computational cost of CNN models, it is crucial to choose the models wisely based on both accuracy and efficiency. The training time is calculated by multiplying the time per epoch by the number of epochs required to achieve the specified degree of accuracy.

If $X_i, i = 1, \ldots, N$ is the training time taken for $N$ no. of classifiers and $Y_i$ is the training time for the ensemble model, then the total time taken for training DETL ensemble model is shown in Eq. (15).

$$\sum_{i=0}^{N}(X_i) + Y_i \tag{15}$$

If $X_i, i = 1, \ldots, N$ is the training time taken for $N$ no. of classifiers, and $\alpha$ is the training time for calculating the final score based on Condorcet's Jury Theorem. The total training time taken for the final ensemble model is calculated as Eq. (16).

$$\sum_{i=0}^{N}(X_i) + \alpha \tag{16}$$

where $\alpha$ is negligible, we can ignore it, and the total training time can be considered as shown in Eq. (17).

$$\sum_{i=0}^{N}(X_i) + \alpha \approx \sum_{i=0}^{N}(X_i) \tag{17}$$

**Table 6**
Table reports the validation accuracy (VA), testing accuracy (TA), Precision, and Recall for top-performing DCNN models and the proposed methods.

| Classifier/Ensemble | VA | TA | Normal Precision | Normal Recall | COVID-19 Precision | COVID-19 Recall | Pneumonia Precision | Pneumonia Recall |
|---|---|---|---|---|---|---|---|---|
| InceptionV3 | 95.65 | 96.45 | 0.98 | 0.97 | 0.94 | 0.98 | 0.95 | 0.95 |
| InceptionResNetV2 | 96.14 | 97.42 | 0.98 | 0.98 | 0.99 | 0.98 | 0.95 | 0.96 |
| ResNet50V2 | 97.26 | 97.90 | 0.98 | 0.99 | 1.00 | 0.98 | 0.96 | 0.96 |
| DenseNet121 | 97.26 | 97.75 | 0.98 | 0.99 | 0.98 | 0.98 | 0.97 | 0.96 |
| DenseNet201 | 95.81 | 97.26 | 0.97 | 0.99 | 0.98 | 0.98 | 0.96 | 0.94 |
| **DETL Ensemble Model** | 98.55 | 97.26 | 0.97 | 0.98 | 1.00 | 0.98 | 0.96 | 0.94 |
| **Jury Ensemble Model** | 97.90 | 98.22 | 0.98 | 1.00 | 0.99 | 0.98 | 0.98 | 0.95 |

**Table 7**
Table reports the sensitivity, specificity, and F1-score of Normal, COVID-19, and Pneumonia class for top-performing DCNN models and the proposed methods.

| Classifier/ Ensemble | Normal Sensitivity | Normal Specificity | Normal F1-Score | COVID-19 Sensitivity | COVID-19 Specificity | COVID-19 F1-Score | Pneumonia Sensitivity | Pneumonia Specificity | Pneumonia F1-Score |
|---|---|---|---|---|---|---|---|---|---|
| InceptionV3 | 96.96 | 97.55 | 97 | 97.56 | 98.55 | 96 | 94.64 | 98.21 | 95 |
| Inception ResNetV2 | 97.87 | 97.58 | 98 | 98.37 | 99.79 | 99 | 95.84 | 98.23 | 96 |
| ResNet50V2 | 98.78 | 97.58 | 98 | 97.56 | 100 | 99 | 96.42 | 98.67 | 96 |
| DenseNet121 | 98.78 | 97.56 | 98 | 97.56 | 99.59 | 98 | 95.84 | 98.89 | 96 |
| DenseNet201 | 98.78 | 96.86 | 98 | 97.56 | 99.58 | 98 | 94.04 | 98.67 | 95 |
| **DETL Ensemble Model** | 98.48 | 96.20 | 98 | 98.37 | 100 | 99 | 94.04 | 98.67 | 95 |
| **Jury Ensemble Model** | 99.69 | 97.56 | 98.79 | 98.37 | 99.79 | 98.77 | 95.23 | 99.34 | 96.67 |

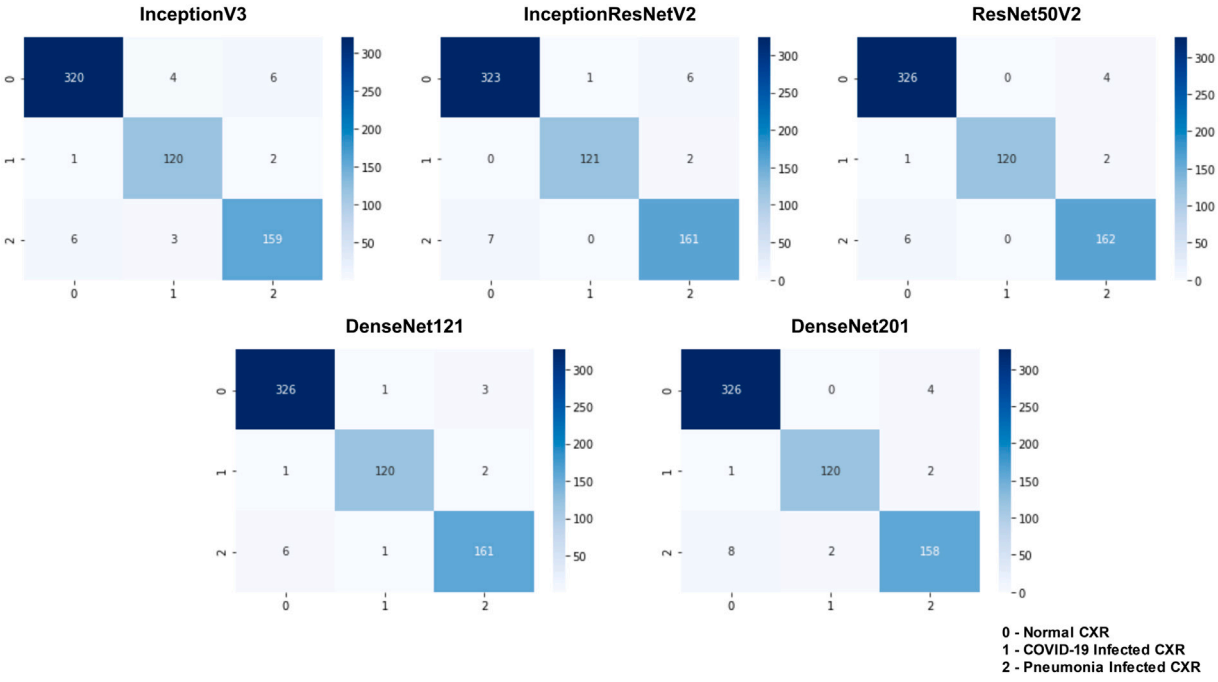


**Fig. 12.** 3-class Confusion Matrix of top performing DCNN models with class 0 as Normal, class 1 as COVID-19, and class 2 as Pneumonia Class.

**Table 8**
Computational efforts of the examined methods.

| Method | Training Time |
|---|---|
| InceptionV3 | 1 h 24 min 34 s |
| InceptionResNetV2 | 3 h 22 min 36 s |
| ResNet50V2 | 1 h 19 min 27 s |
| DenseNet121 | 1 h 29 min 29 s |
| DenseNet201 | 2 h 25 min 33 s |
| DETL Ensemble Model | 47 min 16 s |

Table 8 demonstrates the computational requirements of top-performing deep learning models and the DETL-based ensemble model.

## 6. Discussion

The Coronavirus (COVID-19) pandemic has created havoc on humanity, killing millions and creating severe physical and mental health problems. Therefore, COVID-19 detection using Artificial Intelligence and Computer-Aided Diagnosis has lately been the topic of various research to prepare humanity for the fast and efficient detection of the virus and its variations. To achieve this objective, the authors have proposed a novel method of detecting COVID-19 from CXR images.

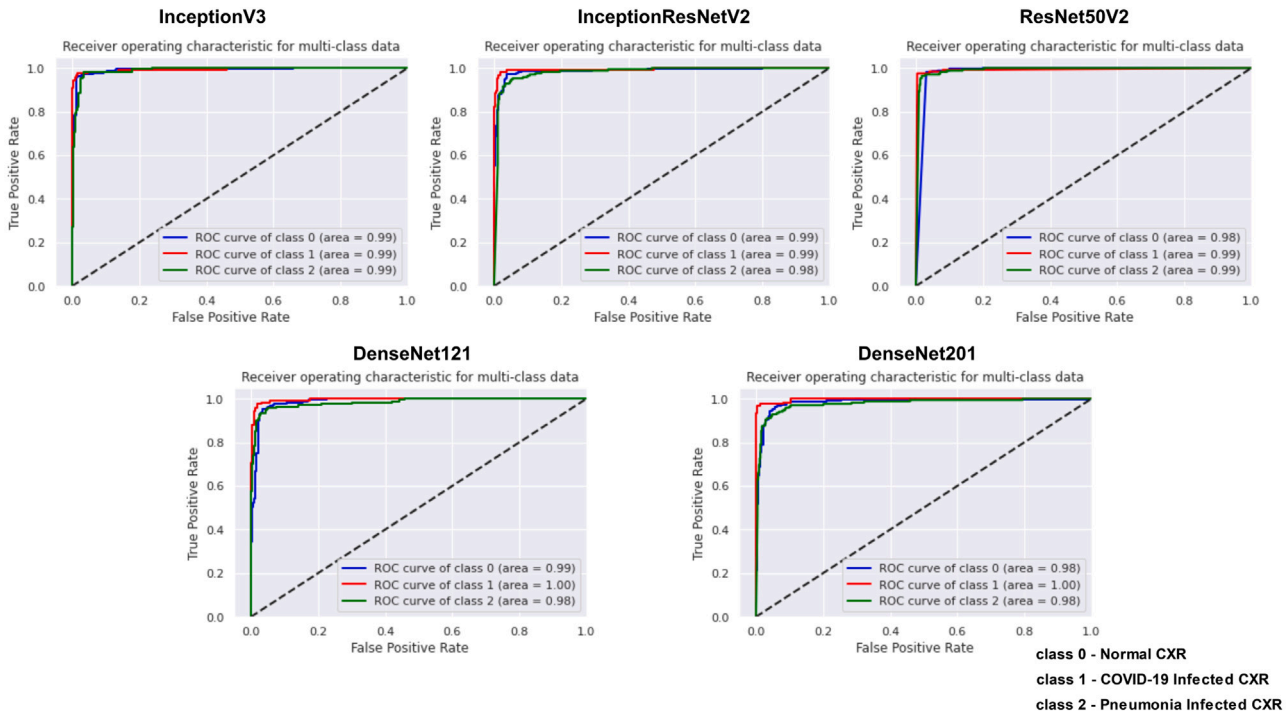


**Fig. 13.** 3-class ROC plots of top performing DCNN models with class 0 as Normal, class 1 as COVID-19, and class 2 as Pneumonia Class.

**Table 9**
Comparative study between the proposed and existing methods/models.

| Author | Method/Model | Dataset | Accuracy (%) | Sensitivity (%) | Specificity (%) |
|---|---|---|---|---|---|
| Ismael [27] | ResNet50 + SVM | 380 CXR images | 94.7 | 91 | 98.89 |
| Tang et al. [28] | EDL-COVID | COVIDx | 95 | 96 | – |
| T. Ozturk [56] | DarkCovidNet | 1127 CXR images | 87.02 | 85.35 | 92.18 |
| Ioannis D. [57] | Transfer Learning | 1427 CXR images | 96.78 | 98.66 | 96.46 |
| E. Luz [58] | EfficientNet Family | 13770 CXR images | 93.9 | 96.8 | – |
| E. Hussain [59] | CoroDet | 7390 CXR images | 94.2 | 94.2 | 96.2 |
| A. I. Khan [60] | CoroNet | 921 CXR images | 95 | – | 97.5 |
| Chuchan et al. [61] | Transfer Learning | 5232 CXR images | 96.39 | 99.62 | – |
| Brunese [62] | Transfer Learning with VGG-16 | 6523 CXR images | 97 | 96 | 98 |
| R. Abdrakhmanov [63] | Few-Shot Learning Approach | 6207 CXR images | 97.7 | – | – |
| D. Shome [64] | Covid-transformer | 6207 CXR images | 92 | – | – |
| F. J. Montalbo [65] | Truncating fined-tuned vision-based models | 6207 CXR images | 97.41 | – | – |
| E. Matsuyama [66] | Fine-tuned ResNet50 | 6207 CXR images | 87 | – | – |
| ***Proposed method*** | **Proposed DETL Ensemble Model** | 6207 CXR images | **97.26** | **98.37** | **100** |
| ***Proposed method*** | **Condorcet's Jury Theorem Based Ensemble Model** | 6207 CXR images | **98.22** | **98.37** | **99.79** |

In this manuscript, the authors have proposed two approaches to detect COVID-19: Domain Extended Transfer Learning (DETL) Ensemble model and Condorcet's Jury Based Ensemble model. In the DETL-based ensemble model, the authors have done the training in two phases. Firstly, the base learners are trained, and then the meta learner, i.e., the ensemble model, is trained. The proposed ensemble model takes the outputs of sub-models, i.e., the base learners, as input and attempts to learn how to best combine the input predictions to make a better output prediction. In the second proposed approach, the authors have employed Condorcet's Jury Theorem to ensemble the base learners. Then, the majority votes are calculated based on the predicted outcomes from all the voters (i.e., models).

Both approaches have shown a remarkable performance on the CXR dataset. The DETL ensemble model has demonstrated an accuracy of 97.26%, whereas the Jury theorem-based ensemble model has shown an accuracy of 98.22%. The authors also proved that Condorcet's Jury Theorem is valid while ensembling the $N$ number of classifiers in Neural Networks.

## 7. Conclusion and future directions

COVID-19 detection through Computer-Aided Diagnosis with the help of Deep Learning models is ongoing research. Many implementations are proposed on this as everyone on this planet was alarmed about the coronavirus situation around them. With 5.9 million deaths caused by this virus and still counting, and a massive 420 million population of people infected by COVID, early detection and prevention is our best chance against it until we find a permanent cure for this deadly virus.

In this manuscript, the authors have proposed an ensemble model based on Condorcet's Jury Theorem. A DETL-based ensemble model has also been proposed as a soft voting ensemble approach to compare it with Condorcet's Jury Theorem-based ensemble model as hard voting.

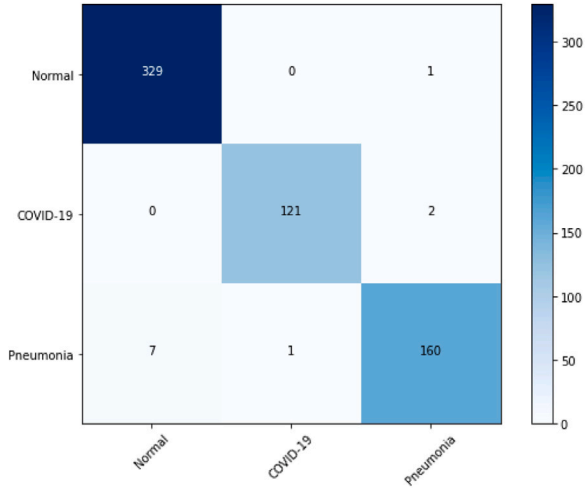


**Fig. 14.** 3-class Confusion Matrix depicting the performance of Condorcet's Jury Theorem based ensemble model.

Condorcet's Jury Theorem is an old game theory mathematical theorem for decision making. The authors demonstrated that the theorem holds while ensembling the $N$ number of classifiers in Neural Networks. With the help of the theorem, the authors proved that a model's presence in the voter pool would improve the likelihood that the majority vote will be accurate if it is more accurate than the other models. The proposed Ensemble model has been used to combine the results of various CNN models to improve this system's collective accuracy and efficiency in detecting COVID-19 in CXR images.

Also, the proposed method detects COVID-19 in patients where the disease has already progressed. However, the effect of stage/severity of the disease on classification is an unexplored field and leaves scope for researchers to work. In the future, this study can help researchers assess the top-performing models in recognizing the complex pattern of Chest X-ray images. Other classification problems, especially biomedical imaging, can benefit from the proposed Ensemble approaches. The proposed Ensemble model may also be used to detect 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, Code Implementations, Validation, Formal Analysis, Investigation, Writing – Original Draft, Writing – Review & Editing. **Nitesh Pradhan:** Conceptualization, Validation, Resources, Writing – Review & Editing, Supervision, Research Administration. **Yashwin Saini:** Literature Survey, Data Curation, Writing – Review & Editing.

## 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.

## Acknowledgment

We thank Manipal University Jaipur, India for providing the Deep Learning Research Lab to conduct our experiments.

## Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.compbiomed.2022.105979.

## References

[1] Y. Shi, G. Wang, X.-p. Cai, J.-w. Deng, L. Zheng, H.-h. Zhu, M. Zheng, B. Yang, Z. Chen, An overview of COVID-19, J. Zhejiang Univ. Sci. B 21 (5) (2020) 343–360.
[2] D. Cucinotta, M. Vanelli, WHO declares COVID-19 a pandemic, Acta Bio Medica: Atenei Parm. 91 (1) (2020) 157.
[3] P. Barach, S.D. Fisher, M.J. Adams, G.R. Burstein, P.D. Brophy, D.Z. Kuo, S.E. Lipshultz, Disruption of healthcare: Will the COVID pandemic worsen non-COVID outcomes and disease outbreaks? Prog. Pediatr. Cardiol. 59 (2020) 101254.
[4] A. Tahamtan, A. Ardebili, Real-time RT-PCR in COVID-19 detection: Issues affecting the results, Expert Rev. Mol. Diagn. 20 (5) (2020) 453–454.
[5] 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.
[6] S.R. Razu, T. Yasmin, T.B. Arif, M. Islam, S.M.S. Islam, H.A. Gesesew, P. Ward, et al., Challenges faced by healthcare professionals during the COVID-19 pandemic: A qualitative inquiry from Bangladesh, Front. Public Health (2021) 1024.
[7] G. Meyerowitz-Katz, S. Bhatt, O. Ratmann, J.M. Brauner, S. Flaxman, S. Mishra, M. Sharma, S. Mindermann, V. Bradley, M. Vollmer, et al., Is the cure really worse than the disease? The health impacts of lockdowns during COVID-19, BMJ Glob. Health 6 (8) (2021) e006653.
[8] N. Pradhan, V. Singh Dhaka, G. Rani, H. Chaudhary, Machine learning model for multi-view visualization of medical images, Comput. J. 65 (4) (2022) 805–817.
[9] 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.
[10] C. Hani, N.H. Trieu, I. Saab, S. Dangeard, S. Bennani, G. Chassagnon, M.-P. Revel, COVID-19 pneumonia: A review of typical CT findings and differential diagnosis, Diagn. Interv. Imaging 101 (5) (2020) 263–268.
[11] R.S. Gereige, P.M. Laufer, Pneumonia, Pediatr. Rev. 34 (10) (2013) 438–456.
[12] L. Gattinoni, D. Chiumello, S. Rossi, COVID-19 pneumonia: ARDS or not? Crit. Care 24 (1) (2020) 1–3.
[13] P. Mo, Y. Xing, Y. Xiao, L. Deng, Q. Zhao, H. Wang, Y. Xiong, Z. Cheng, S. Gao, K. Liang, et al., Clinical characteristics of refractory COVID-19 pneumonia in Wuhan, China, Clin. Infect. Dis. (2020).
[14] Y. Fang, H. Zhang, J. Xie, M. Lin, L. Ying, P. Pang, W. Ji, Sensitivity of chest CT for COVID-19: Comparison to RT-PCR, Radiology 296 (2) (2020) E115–E117.
[15] 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.
[16] S. Mohammed, F. Alkinani, Y. Hassan, Automatic computer aided diagnostic for COVID-19 based on chest X-Ray image and particle swarm intelligence, Int. J. Intell. Eng. Syst. 13 (5) (2020) 63–73.
[17] G. Srivastava, A. Chauhan, M. Jangid, S. Chaurasia, CoviXNet: A novel and efficient deep learning model for detection of COVID-19 using chest X-Ray images, Biomed. Signal Process. Control (2022) 103848.
[18] H. Lv, L. Shi, J.W. Berkenpas, F.-Y. Dao, H. Zulfiqar, H. Ding, Y. Zhang, L. Yang, R. Cao, Application of artificial intelligence and machine learning for COVID-19 drug discovery and vaccine design, Brief. Bioinform. 22 (6) (2021) bbab320.
[19] C. Park, C.C. Took, J.-K. Seong, Machine learning in biomedical engineering, Biomed. Eng. Lett. 8 (1) (2018) 1–3.
[20] A. Alimadadi, S. Aryal, I. Manandhar, P.B. Munroe, B. Joe, X. Cheng, Artificial intelligence and machine learning to fight COVID-19, Physiol. Genomics 52 (4) (2020) 200–202.
[21] M. van der Schaar, A.M. Alaa, A. Floto, A. Gimson, S. Scholtes, A. Wood, E. McKinney, D. Jarrett, P. Lio, A. Ercole, How artificial intelligence and machine learning can help healthcare systems respond to COVID-19, Mach. Learn. 110 (1) (2021) 1–14.
[22] P.J. Boland, Majority systems and the condorcet jury theorem, J. R. Stat. Soc. Ser. D 38 (3) (1989) 181–189.
[23] S. Tang, C. Wang, J. Nie, N. Kumar, Y. Zhang, Z. Xiong, A. Barnawi, EDL-COVID: Ensemble deep learning for COVID-19 case detection from chest X-Ray images, IEEE Trans. Ind. Inf. 17 (9) (2021) 6539–6549, http://dx.doi.org/10.1109/TII.2021.3057683.
[24] A. Castiglione, P. Vijayakumar, M. Nappi, S. Sadiq, M. Umer, COVID-19: Automatic detection of the novel coronavirus disease from CT images using an optimized convolutional neural network, IEEE Trans. Ind. Inf. 17 (9) (2021) 6480–6488, http://dx.doi.org/10.1109/TII.2021.3057524.
[25] W. Chen, X. Li, L. Gao, W. Shen, Improving computer-aided cervical cells classification using transfer learning based snapshot ensemble, Appl. Sci. 10 (20) (2020) 7292.
[26] L.D. Nguyen, R. Gao, D. Lin, Z. Lin, Biomedical image classification based on a feature concatenation and ensemble of deep CNNs, J. Ambient Intell. Humaniz. Comput. (2019) 1–13.
[27] A.M. Ismael, A. Şengür, Deep learning approaches for COVID-19 detection based on chest X-Ray images, Expert Syst. Appl. 164 (2021) 114054.
[28] S. Tang, C. Wang, J. Nie, N. Kumar, Y. Zhang, Z. Xiong, A. Barnawi, EDL-COVID: Ensemble deep learning for COVID-19 case detection from chest X-Ray images, IEEE Trans. Ind. Inf. 17 (9) (2021) 6539–6549.

[29] 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.

[30] M. Aminu, N.A. Ahmad, M.H.M. Noor, Covid-19 detection via deep neural network and occlusion sensitivity maps, Alex. Eng. J. 60 (5) (2021) 4829–4855.

[31] 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.

[32] S.-H. Wang, X. Wu, Y.-D. Zhang, C. Tang, X. Zhang, Diagnosis of COVID-19 by wavelet renyi entropy and three-segment biogeography-based optimization, Int. J. Comput. Intell. Syst. 13 (1) (2020) 1332–1344.

[33] S.-H. Wang, D.R. Nayak, D.S. Guttery, X. Zhang, Y.-D. Zhang, COVID-19 classification by CCSHNet with deep fusion using transfer learning and discriminant correlation analysis, Inf. Fusion 68 (2021) 131–148.

[34] A. Khan, S.H. Khan, M. Saif, A. Batool, A. Sohail, M.W. Khan, A survey of deep learning techniques for the analysis of COVID-19 and their usability for detecting omicron, 2022, arXiv preprint arXiv:2202.06372.

[35] S.H. Khan, A. Sohail, M.M. Zafar, A. Khan, Coronavirus disease analysis using chest X-ray images and a novel deep convolutional neural network, Photodiagnosis Photodyn. Ther. 35 (2021) 102473.

[36] 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).

[37] 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.

[38] S. Li, B. Qin, J. Xiao, Q. Liu, Y. Wang, D. Liang, Multi-channel and multi-model-based autoencoding prior for grayscale image restoration, IEEE Trans. Image Process. 29 (2019) 142–156.

[39] K.K. Ladha, The condorcet jury theorem, free speech, and correlated votes, Am. J. Political Sci. (1992) 617–634.

[40] D.M. Estlund, Opinion leaders, independence, and condorcet's jury theorem, Theory and Decision 36 (2) (1994) 131–162.

[41] D. Austen-Smith, J.S. Banks, Information aggregation, rationality, and the condorcet jury theorem, Am. Political Sci. Rev. 90 (1) (1996) 34–45.

[42] X. Dong, Z. Yu, W. Cao, Y. Shi, Q. Ma, A survey on ensemble learning, Front. Comput. Sci. 14 (2) (2020) 241–258.

[43] O. Sagi, L. Rokach, Ensemble learning: A survey, Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 8 (4) (2018) e1249.

[44] J. Kittler, M. Hatef, R.P. Duin, J. Matas, On combining classifiers, IEEE Trans. Pattern Anal. Mach. Intell. 20 (3) (1998) 226–239.

[45] 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.

[46] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[47] 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.

[48] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[49] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.

[50] Z. Zhang, M. Sabuncu, Generalized cross entropy loss for training deep neural networks with noisy labels, Adv. Neural Inf. Process. Syst. 31 (2018).

[51] Y. Wu, J. Li, Y. Kong, Y. Fu, Deep convolutional neural network with independent softmax for large scale face recognition, in: Proceedings of the 24th ACM International Conference on Multimedia, 2016, pp. 1063–1067.

[52] R.A. Dunne, N.A. Campbell, On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function, in: Proc. 8th Aust. Conf. on the Neural Networks, Melbourne, Vol. 181, Citeseer, 1997, p. 185.

[53] R. Ge, S.M. Kakade, R. Kidambi, P. Netrapalli, The step decay schedule: A near optimal, geometrically decaying learning rate procedure for least squares, Adv. Neural Inf. Process. Syst. 32 (2019).

[54] D. Berend, J. Paroush, When is condorcet's jury theorem valid? Soc. Choice Welf. 15 (4) (1998) 481–488.

[55] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, GRAD-CAM: Visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.

[56] 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.

[57] I.D. Apostolopoulos, T.A. Mpesiana, COVID-19: Automatic detection from X-Ray images utilizing transfer learning with convolutional neural networks, Phys. Eng. Sci. Med. 43 (2) (2020) 635–640.

[58] 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.

[59] 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.

[60] 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.

[61] V. Chouhan, S.K. Singh, A. Khamparia, D. Gupta, P. Tiwari, C. Moreira, R. Damaševičius, V.H.C. De Albuquerque, A novel transfer learning based approach for pneumonia detection in chest X-Ray images, Appl. Sci. 10 (2) (2020) 559.

[62] L. Brunese, F. Mercaldo, A. Reginelli, A. Santone, Explainable deep learning for pulmonary disease and coronavirus COVID-19 detection from X-Rays, Comput. Methods Programs Biomed. 196 (2020) 105608.

[63] R. Abdrakhmanov, M. Altynbekov, A. Abu, A. Shomanov, D. Viderman, M.-H. Lee, Few-shot learning approach for COVID-19 detection from X-Ray images, in: 2021 16th International Conference on Electronics Computer and Computation, ICECCO, IEEE, 2021, pp. 1–3.

[64] D. Shome, T. Kar, S.N. Mohanty, P. Tiwari, K. Muhammad, A. AlTameem, Y. Zhang, A.K.J. Saudagar, Covid-transformer: Interpretable covid-19 detection using vision transformer for healthcare, Int. J. Environ. Res. Public Health 18 (21) (2021) 11086.

[65] F.J. Montalbo, Truncating fined-tuned vision-based models to lightweight deployable diagnostic tools for SARS-CoV-2 infected chest X-Rays and CT-scans, Multimedia Tools Appl. 81 (12) (2022) 16411–16439.

[66] E. Matsuyama, H. Watanabe, N. Takahashi, Explainable analysis of deep learning models for coronavirus disease (COVID-19) classification with chest X-Ray images: Towards practical applications, Open J. Med. Imaging 12 (3) (2022) 83–102.