Predicting Mortality Risk of Covid-19 Patients Using Chest X-Rays

Akeem Olowolayemo*, Mohammed Yasin, Mohammed Raashid Salih
Department of Computer Science, KICT, International Islamic University Malaysia.

*Corresponding author: akeem@iium.edu.my
(Received: 18thJune 2022; Accepted: 17th January 2023; Published on-line: 28th January 2023

Abstract— The outbreak of COVID-19 in late 2019 presents a challenging dimension exhibited by its fast and high rate of infection, even though its severity on infected patients is somewhat feeble, especially in people with strong immunity. Due to this rapid infection rate and the limited capacity of healthcare infrastructures, an optimal allocation of health facilities and resources becomes imperative. Consequently, forecasting an individual's infection severity is crucial to efficiently determine whether the patient requires hospitalization or may be treated as an outpatient to free resources for those desperately deserving. Without such systems, health resources would be inefficiently utilized, resulting in needlessly lost lives. This study attempts to provide a model to determine the mortality of an infected patient on arrival to the health facilities to determine whether or not it is necessary to admit them to intensive care. A Convolutional Neural Networks (CNNs) model based on the ResNet-18 architecture was trained on chest X-rays of COVID-19 patients to estimate their mortality risk, with the best model achieving a training accuracy of 99.6 percent while the validation accuracy achieved 86.7% along with high sensitivity.

Keywords— Deep Learning, Convolutional Neural Networks(CNNs), Image Classification, X-Rays, COVID-19, Mortality.

I. Introduction

COVID-19 began in late 2019 and has since decimated the planet, its rate of infection and spread is unprecedented due to globalization. It is a particularly effective illness due to its fast rate of infection even though it has an extremely low mortality rate. COVID-19's massive infections have pushed the global healthcare facilities and resources to their limits, as a large number of people have been infected in a short period. Several countries such as China, India, Italy, etc., for example, had to convert stadiums or other public spaces into makeshift specialized hospitals [1], [2].

The severity of the infected patients' symptoms, which can range from mild to severe, is an important factor to consider [3], and certain co-morbidities, such as age and diabetes, have been identified as high-risk while other elements are still being determined. This means that not everyone needs to be admitted to the hospital. This is possibly the most important aspect of the research. Consequently, the healthcare system's limited capacity and the unidentified requirement for hospitalization motivated the need to develop a prioritizing procedure to decide which patients should be hospitalized.

The intensity of the infected individuals' symptoms, which can range from moderate to severe, is an important factor to consider. This means that not everyone needs to be admitted to the hospital, and while some co-morbidities, such as age and diabetes, have been identified as high-risk [4], other factors have yet to be found. As a result, it's impossible to say whether a person has to be admitted to the hospital or if they can recover at home.

These two aspects of the pandemic, namely the healthcare system's limited capacity and the unknown requirement for hospitalization, have necessitated the development of a prioritizing algorithm to decide which patients require hospitalization. Unfortunately, in the current scenario, some patients are admitted to hospitals, wasting valuable resources when they do not require them, while others have no access owing to a shortage of resources. As a result, doctors had to make some difficult but important judgments about who gets to access such resources [5].

Chest X-rays (or CXRs) may hold the key to determining those who should be admitted. CXRs are routinely used by radiologists to visually assess lung abnormalities in both emergency and non-emergency situations. COVID-19 is diagnosed likewise, having pneumonia as its most common symptom. CXRs have been demonstrated to have adequate sensitivity and specificity in detecting such lung abnormalities [6].

The challenge of suboptimal hospitalization has provided an opportunity for deep learning to spread its wings. By serving as an acceptable prioritizing mechanism, a model that can estimate the prospective mortality risk of a patient infected with COVID-19 based on CXRs would be a great contribution to the medical community at large. By allocating medical resources optimally to minimize unnecessary deaths.

The goal of this study is to develop computer vision techniques to estimate mortality risk in COVID-19 patients using X-ray dataset. The data for the study was gathered from reputable sources of COVID-19-related X-ray data, which was then transformed into a format suitable for the CNNs model.

It is intended that by using a model that can properly estimate mortality risk, unnecessary deaths can be avoided by allocating limited medical resources wisely, reducing the number of deaths. This should also reduce the number of COVID-19 patients admitted to hospitals, limiting future exposure by reducing the number of individuals in hospitals and ensuring the safety of medical personnel. Finally, this lowers the incidence of peak infection, delays the onset of the peak, and lowers the total number of infections during the pandemic.

II. Related Work

Previous studies, such as [7], acknowledged COVID-19's high infection rate and the negative consequences it has in terms of limited medical resources and loss of life. The main motivation for this research is that early detection of patients with a poor prognosis is critical for early prevention before serious symptoms appear. The work presents a fully autonomous deep learning approach for COVID-19 diagnosis and prognostic analysis using commonly used computed tomography (or CT). CT was chosen because it is far more sensitive than RT-PCR x-rays, even in asymptomatic patients, and it can be obtained rapidly and cheaply. COVID-19 was distinguished from other types of pneumonia with AUC by the model, which performed excellently. The model performed considerably well, able to distinguish COVID-19 from other pneumonia with AUC (area under the ROC Curve) of 0.87 and 0.88, respectively, and viral pneumonia with an AUC of 0.86. More crucially, the deep learning system was able to classify patients into high-risk and low-risk groups depending on the length of their hospital stay.

In contrast to [7], the study in [8] used the opposite approach, utilizing chest X-rays instead of CT scans. The study, like the previous one, emphasized the scarcity of mechanical ventilators. To that end, the research offers a deep learning model for predicting the requirement for mechanical ventilation in hospitalized COVID-19 patients using chest X-ray images. They contend that X-rays are more practical than CT scans because they are more readily available and have a lower risk of machine contamination. The model performed with high accuracy, sensitivity, and specificity of 90.06%, 86.34%, and 84.38%, respectively. The authors equally utilized the predictions of two Pulmonary and Critical Care experts on the need for mechanical ventilation to compare with their models. The model outperformed the experts by an incremental accuracy of 7.24%–13.25%.

When it comes to COVID-19, the work in [9] points out a potential stumbling block for models trained on X-rays: accessibility and availability. While the various models that have been constructed show promising outcomes, they are not computationally efficient, which is a barrier for medical practitioners on the ground, according to the report. The research provides a strategy for providing an efficient, yet effective, model for pattern detection of COVID-19 chest X-rays in answer to this difficulty. This is accomplished by combining the EfficientNet family of ANNs with a hierarchical classifier. The study showed promising results, with the model having a 93.9 percent accuracy and 5 to 30 times fewer parameters than other studied models.

The study in [10] using an exploratory approach acknowledges [9] concerns about the lack of efficient and effective models to assist physicians, with two additional key aspects, the bias that is caused by a lack of quality control and biased evaluation of public COVID-19 datasets, in particular. The paper recognizes the enormous number of machine learning solutions that have been presented to address the challenge of COVID-19 diagnosis and prognosis. It also recognizes the brief time range during which this occurred, as well as the presence of undetected bias, which precludes such models from performing well on independent test data when compared to training data. The study warns against using such public models inappropriately and encourages researchers to choose better datasets as well as highlight important practices in dataset selection.

III. Methodology

Data cleaning, missing label imputation, picture augmentation, data pipeline building, image modification, model generation, and model evaluation are all part of the approach for this study. When it comes to image classification, object identification, and recognition, as well as computer vision applications using deep neural networks, convolutional neural networks (CNNs) are the de facto architectural standard. As a result, for this research, a CNN-based model called Residual Networks (or ResNets) [8] has been adopted while chosen ResNet-18 as the preferred architecture. ResNet-18 (18 total number of layers) was chosen due to its ability to rapidly converge. The ResNet-18 CNNs' steps are detailed in the following subsections.

A) ResNet-18

In this study, the ResNet-18 CNNs were utilized on a dataset of COVID-19 chest X-rays (CXR) images that had been preprocessed. The steps involved in preparing the dataset to be used by the ResNet-18 CNNs for classification are detailed in the dataset subsection. The COVID-19 chest X-rays (CXR) images are first processed through a 7x7, 64-channel convolutional layer. The convolutional layer is in charge of extracting features. While performing convolutional operations on the chest X-ray, a kernel (or filter) of a specific size is dragged across the image resulting in a feature map. The convolutional operation is represented thus:

$$\mathrm{C(i,j)} = (C_{XR} * K)(i,j) = \sum_{m,n} C_{XR}(i+n, j+m)K(m,n) \qquad (1)$$

where K is the kernel operating on the X-Ray C_{XR}.

Several convolutional layers are piled in series, as seen in the modules, where one layer's output becomes the input to the next. This is done so that higher-level features can be extracted. Hyper-parameters like kernel dimensions and stride (the number of steps taken by the kernel) are also taken into account. The kernel dimensions in this example are 7x7 and the stride value is 2.

Because pictures are inherently non-linear, the Rectified Linear Activation function (ReLU) is applied after each convolutional layer to infuse non-linearity into the model. When compared to other non-linear functions, this enhances the performance of the model, to converge faster[11]. The ReLU as seen in fig. 1, is defined as

$$\mathrm{ReLU}(x) = \max(0, x) \qquad (2)$$

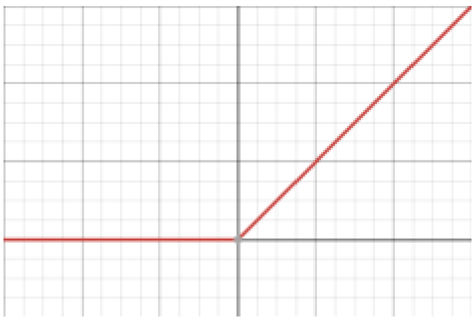


Fig. 1 The ReLU Graph

B) *Pooling Layer*

Following the initial convolution, the chest X-ray feature map tensor is subjected to a 3x3 max pooling procedure. The pooling layer performs the critical function of subsampling, which is to lower the dimensionality of the convolutional layer's feature maps. This is crucial to reduce the network's parameters and, as a result, the amount of processing required. Pooling can be done in a variety of ways, including max, average, and min pooling, which yields the maximum, average, and minimum values of the pixels of the chest X-ray corresponding to the size of the kernel, respectively. Like the convolutional layer before it, the pooling layer takes into account characteristics like layer size and stride. This initial max-pooling operation is 3x3 in size and has a stride of 2.

C) *Residual Connections*

When compared to regular CNN design, the ResNet-18 model's layers reveal unconventionality in the form of residual connections. Because of vanishing gradients, deeper networks perform worse. Instead of learning functions that do not reference the input, residual connections are employed to learn functions that do. These are referred to as residual functions.

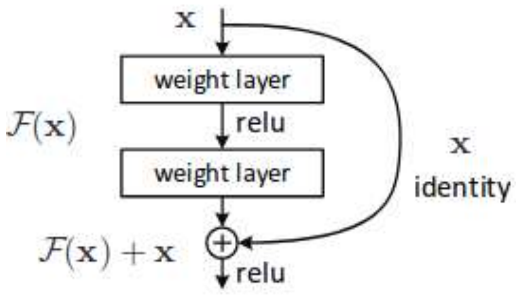

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 2 Residual Building Block [12].

The desired output is represented by H(x), while the input value is represented by x in the diagram above. In terms of x, the aforementioned residual function is defined as

$$F(x) = H(x) - x \qquad (3)$$

As a result, the targeted output is renamed as

$$H(x) = F(x) + x \qquad (4)$$

The residual connection, which acts as a shortcut connection, and element-wise addition are used to produce F(x) + x (equal to H(x)) at the output. The residual function in this two-layer example is calculated as follows:

$$F(x, \{W_i\}) = W_2\, \sigma(W_1 x) \qquad (5)$$

Where σ is the ReLU activation function.

The reasoning behind this redefining of the intended output H(x) in terms of the input x is that the referenced residual mapping (H(x)-x) is easier for the stacked non-linear layers to optimize than the unreferenced H(x). The residual F(x) naturally tends to 0 in the (extreme) circumstance when the optimal desired output is the identity mapping, where the output is the same as the input (which is x). Fitting the stacked non-linear layers to the identity mapping directly is substantially more difficult.

Residual connections can be divided into two categories:

When the input and output dimensions are the same, the identity shortcut is used. It is symbolized by

$$y = F(x, \{Wi\}) + x \qquad (6)$$

When the dimensions of the input and output are not the same due to a change in the number of channels between layers, the projection shortcut is used. To ensure dimension parity, a linear transformation is used in this example. This has the unintended consequence of adding more parameters, which are represented by Ws.

$$y = F(x, \{Wi\}) + Ws\, x \tag{7}$$

D) Modules 1-4

The chest X-ray tensor travels through four separate modules connected in series after max pooling. Before reaching the fully connected layer, CNNs typically consist of repeated modules that include the convolutional layer, the ReLU, and the pooling layers. Each module has four 3x3 convolutional layers with the same number of channels, as well as ReLU activation functions and residual connections. Each module's convolutional layers have 64 channels in the first, 128 channels in the second, 256 channels in the third, and 512 channels in the fourth. The projection shortcut indicated in equation 7 is used since the number of channels varies between modules. The stride value of 2 is used by all of the levels described.

E) Fully Connected Layer

After passing through the last module and before reaching the fully connected layer of 1000 neurons, the chest X-ray feature map tensor encounters the global average pooling procedure. The completely connected layer, which appears at the network's end, is in charge of classification. It resembles the typical neural network architecture, in which every neuron in one layer is connected to every neuron in the next, resulting in a dense network. Because fully linked layers cannot directly act on the chest X-ray tensors, the tensor of feature mappings is retrieved from the layers before being turned into a vector via a method known as flattening. The layer is sometimes known as the flattening layer for this reason. The completely connected layer identifies the patient as likely surviving or not surviving based on features collected from the chest X-ray. The softmax operation must be completed before the classification may begin. It assures that the probability of the output classes totals up to one. and is defined as

$$\mathrm{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{8}$$

The output of each neuron in the fully connected layer is obtained by

$$f(\boldsymbol{x}, \boldsymbol{w}, b) = \boldsymbol{x}^T \boldsymbol{w} + b \tag{9}$$

Which is typical of artificial neural networks. Here, $\boldsymbol{x}$ represents the tensor of feature maps derived from the chest X-ray, $\boldsymbol{w}$ is the weight associated with the particular neuron, and b is the bias value.

F) Dataset

For the experiment with deep neural networks, it is important to gather a lot of data for each labeled class. This paper intended to predict the mortality risk of a COVID-19 patient using chest X-rays. Obtaining a public COVID-19 chest X-ray dataset with a label indicating mortality was difficult due to the nature of the study within the ongoing pandemic situation. The dataset finally utilized was discovered on The Cancer Imaging Archive (TCIA) [12] from Stony Brook University and collected by [13] with the necessary labels. The Stony Brook University (SBU) dataset on TCIA was much richer compared to the IEEE8023 dataset utilized previously in [14], containing only 169 usable COVID-19 chest X-rays images. The SBU dataset had 562,376 radiography images of 1384 COVID-19-positive patients collected over 15 months including X-rays, CT scans, MRIs of the chest, abdomen, kidney, pelvis, and more with different modalities. However, the images in the dataset were in DICOM medical imaging format, which is the standard format used in many hospitals and laboratories to store and process medical images. Due to the high resolution and amount of metadata information found in DICOM image files, the file sizes often end up being much larger than other common imaging formats. In the case of the SBU dataset, each image was around 15MB in size, and as a consequence the whole dataset was over 500GB in size. Fortunately, not all of the images in the dataset were relevant to this study because, as mentioned above, it also had radiography images of organs other than the chest. Nonetheless, even with those images filtered out and only the chest X-rays selected, the dataset still amounted to over 192GB.

To further shrink the dataset to a manageable size, the download was divided into seven chunks. At the completion of each chunk of downloads, the images that have been downloaded were converted to PNG format with a maximum width restriction of 512 pixels and then uploaded to Kaggle as a dataset. This process was repeated for all the seven chunks until all of the images were uploaded to Kaggle. This significantly reduced the size of the dataset to just over 2GBs.

The filtered dataset had 13638 images, which was still large, a lot more than the IEEE8023 dataset. However, upon inspecting the dataset, it was noticed that many of the X-ray images had duplicates that were an enhanced version of an already existing image. This posed a problem as it made it difficult to split the dataset into training and testing sets without causing the duplicate images from the former to outflow into the latter. To overcome this, a new label was added to distinguish between the original and the enhanced X-ray images in the dataset. The images in the dataset were taken in series, each series consisting of 1 to 6 images. For a series that had only 2 X-ray images, it was easy to find and mark the enhanced version of the X-ray images as they were in folders with a larger starting number as the name. However, the series with more than 2 images had no such clear distinguishing features, so over 1600 images that were in this series had to be labeled manually. The labels were stored in a new metadata file. After having the duplicate images labeled and filtered, the number of unique images in the dataset was found to be 7681, barring the images that were in lateral view, of which 4803 were of surviving patients and 2878 were of deceased patients.

G) *Model Training and Evaluation*

Since the study was based on the previously developed model utilizing IEEE8023 dataset in [14], the best model from the previous study and the corresponding class-balancing technique were then utilized in the training and testing with the SBU dataset. This is done with a view to getting the final model whose performance metrics on the test set were then compared with the best model of the previous study based on IEEE8023 dataset to demonstrate the improvements.

H) *Creating the Test Set*

The test set was created from the SBU dataset by carrying out a 90-10 split. The first part consisting of 90% of the data was set aside for training and testing, while the second part consisting of 10% of the data was used as the test set for both the first model from [14] and the second phase of training, utilized the best training model from the SBU dataset. Using the same test set allowed us to compare the performance metrics of the best model from the first dataset (IEEE8023 dataset) and the model from the second dataset (SBU) directly.

I) *Algorithm*

Training a convolutional neural network on images requires them to be of the same size. A 224x224 cropped resize was chosen to be applied to all the X-rays as that resolution was the standard in image classification. Besides that, the images were normalized. Image normalization is like the normalization of numerical data. The idea is to center a range of data around 0. Images are represented as tensors, which are three-dimensional arrays of numbers representing the three-color channels Red, Green, and Blue for each pixel. These numbers can also be normalized and that is what happens during image normalization. Each of the channels is normalized based on the values of all the pixels in that channel. So, in total, there would be three separate normalizations happening, one for each dimension of the tensor.

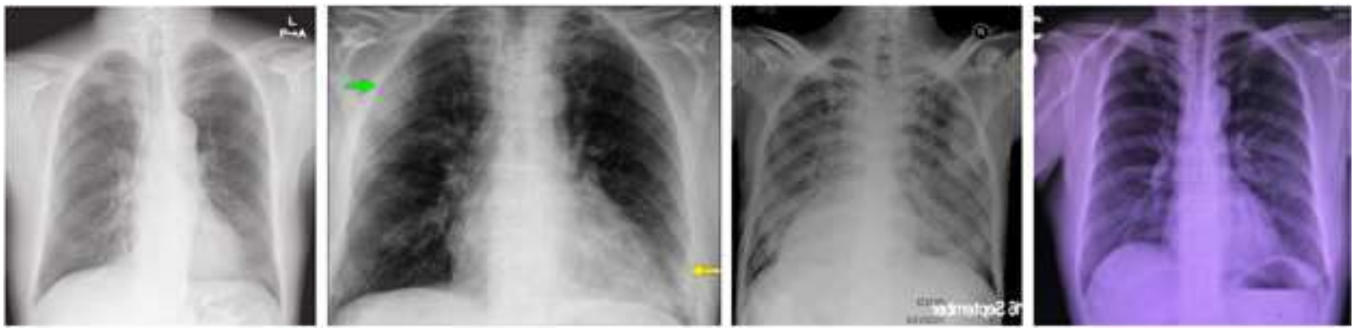


Fig. 3 Sample demonstrating x-ray variety.

For this paper, a ResNet-18 convolutional neural network pre-trained model which consists of 18 layers was used as the base model. The standard practice is to normalize images based on the mean and standard deviation of ImageNet, which is a database consisting of millions of images on which pre-trained models such as ResNet-18 were trained. Furthermore, the images in the training set would also be augmented to create more variations. This is done by manipulating certain features of the existing images with image transformation and then using the newly generated images for training. The image transformations applied include zoom, warp, brightness adjustment, and rotation. Additionally, training a convolutional neural network one image at a time is very slow and time-consuming. The alternative is to train using a batch of images at every epoch and GPUs are fast and efficient in carrying out batch processing of this nature. Images can be resized, normalized, augmented, and batched by using the DataBlock API of FastAI.

Additionally, a pre-trained model already has weight assignments at each of the layers based on what the model has been previously trained to recognize. But the last layer of the pre-trained model is very specific to the classification task that it has been originally designed to do, which is why it is removed and replaced with a new layer with random weight assignments so that the model can be used for the specific tasks relevant to this paper. Besides that, the cross-entropy loss was used as the loss function, and the learning rate optimizer used was the popular Adam optimizer.

Moreover, for the training, a stratified K-Fold cross-validation technique with a 5-fold split was used. This means the data was split and evaluated 5 times while also maintaining the ratio of the target classes in each split, and each time a different portion of the data was chosen to be the validation set such that at the end of the 5 iterations, the model was trained and evaluated on all of the available data. This was appropriate in the case of this paper because the amount of training data was small, and hence testing the methodology on the whole dataset and getting the average performance metrics provided a better representation of the efficacy of the proposed techniques.

After creating the model learner, the optimal learning rate is found using a learning rate finder and the model is first fine-tuned with all but its last layer frozen for 5 epochs. Then, all the layers were unfrozen, and the learning rate finder is again used to find the optimal learning rate, which then is used to train the model for another 30 epochs. This is done for every split of data, so in total for each of the three intended models, the algorithm ran 5 times and stored the accuracy and predictions at each run. The models are then evaluated by averaging the performance metrics on the validation set for all the splits and a cumulative confusion matrix was generated to determine the true positivity rate, true negativity rate, false positivity rate, and false negativity rate of each of the models on the whole data. Additional performance metrics are also obtained by evaluating the model on the test set.

```
Algorithm 1 Model generation
  Input: COVID-19 CXR dataset
  Output: COVID-19 mortality risk classifier model
  Initialize: $batch\_size \leftarrow 64$, $freeze\_epochs \leftarrow 5$, $unfreeze\_epochs \leftarrow 30$, $batches \leftarrow []$
 1: $data \leftarrow$ Cohen et. al dataset
 2: for $i \leftarrow 0$ to $i < LengthOf(data)$ do
 3:     $batch \leftarrow []$
 4:     for $n \leftarrow 0$ to $n < batch\_size$ do
 5:         $image \leftarrow ResizeImage(image)$
 6:         $image \leftarrow NormalizeImage(image)$
 7:         $batch \leftarrow batch + image$
 8:         $n \leftarrow n + 1$
 9:     end for
10:     $batches \leftarrow batches + batch$
11:     $i \leftarrow i + n$
12: end for
13: $model \leftarrow$ Load pre-trained ResNet-18 CNN model
14: Remove last layer from $model$
15: Add layer with randomized weights to $model$
16: Freeze all layers except last layer of $model$
17: $lr \leftarrow$ find and store optimal learning rate
18: for $epoch \leftarrow 0$ to $freeze\_epochs$ do
19:     for all batch in batches do
20:         $TrainCNN(model, batch, lr)$
21:     end for
22:     $accuracy \leftarrow$ Calculate accuracy of $model$ on validation set
23:     $loss \leftarrow$ Calculate loss of $model$ on validation set
24:     $print(accuracy, loss)$
25: end for
26: Unfreeze all layers of $model$
27: $lr \leftarrow$ find and store optimal learning rate
28: for $epoch \leftarrow 0$ to $unfreeze\_epochs$ do
29:     for all batch in batches do
30:         $TrainCNN(model, batch, lr)$
31:     end for
32:     $accuracy \leftarrow$ Calculate accuracy of $model$ on validation set
33:     $loss \leftarrow$ Calculate loss of $model$ on validation set
34:     $print(accuracy, loss)$
35: end for
36: Repeat Steps 2-35 with:
        1. Class weight adjustment
        2. Random weighted resampling
37: Determine best class balancing technique based on validation accuracy
38: Save best model checkpoint
39: $data \leftarrow$ Stony Brook University dataset
40: $model \leftarrow$ Best model determined in Step 37
41: Train using Steps 16-35 applying the best class balancing technique
42: SaveModel($model$)
```

Fig. 4 Model generation algorithm.

J) *Training and Testing the Model with SBU Dataset*

The best model from the first dataset (IEEE8023 dataset) of the training and testing was then fine-tuned on the SBU dataset using the same algorithm as described before maintaining the same class balancing technique associated with the best model. The fine-tuned model was then evaluated to get the required set of metrics that would be used to compare it with the model from the first dataset (IEEE8023 dataset) in [14].

K) *Evaluation*

Since K-Fold cross-validation essentially trains a new model at each iteration, the performance metrics had to be calculated each time a new model was generated during the cross-validation. As a result, at end of the cross-validation, there were five sets of performance metrics, one for each model. To get the overall performance metrics, the scores were averaged over the five sets. The scores that were calculated were the balanced accuracy, precision, recall, F1-score, and AUC-ROC score. Besides this, a cumulative confusion matrix was generated by summing up the confusion matrix of each of the models on the validation set. From this confusion matrix, the true positive, true negative, false positive, and false-negative rates for the overall model were derived.

The complete algorithm for model generation is shown in Fig. 4.

IV. Results and Discussion

The second phase of training and testing used the model trained on the original unimputed data with class weight adjustment, which was determined to be the best model from the results of the first dataset (IEEE8023 Dataset) Model, as the base model which was then fine-tuned on the SBU dataset. Table I summarizes the performance results from this iteration of training averaged over five folds.

On the validation set, the model achieves an accuracy of 86.7% with a ROC AUC score of 0.94 and an F1-Score of 0.832. The true positive rate was found to be decent, around 91.2% while the false negative rate was still not quite low, standing at 18.4%

TABLE I
Performance Metrics of the SBU Dataset Model

Metrics	Training	Validation
Balanced Accuracy	0.996	0.867
ROC AUC Score	1.000	0.938
Average Precision	0.992	0.849
Average Recall	0.996	0.816
F1 Score	0.994	0.832
True Positive Rate	-	0.919
True Negative Rate	-	0.816
False Positive Rate	-	0.081
False Negative Rate	-	0.184

The ROC Curve and Precision-Recall curves are shown in Fig. 5 and Fig. 6 respectively. As observable, there is a significant improvement in the smoothness and stability of the curves of the second model based on the SBU dataset, compared to the curves of the first dataset. This is due to a large number of images in the validation set of the SBU dataset compared to the first dataset model (IEEE8023 Dataset[14].

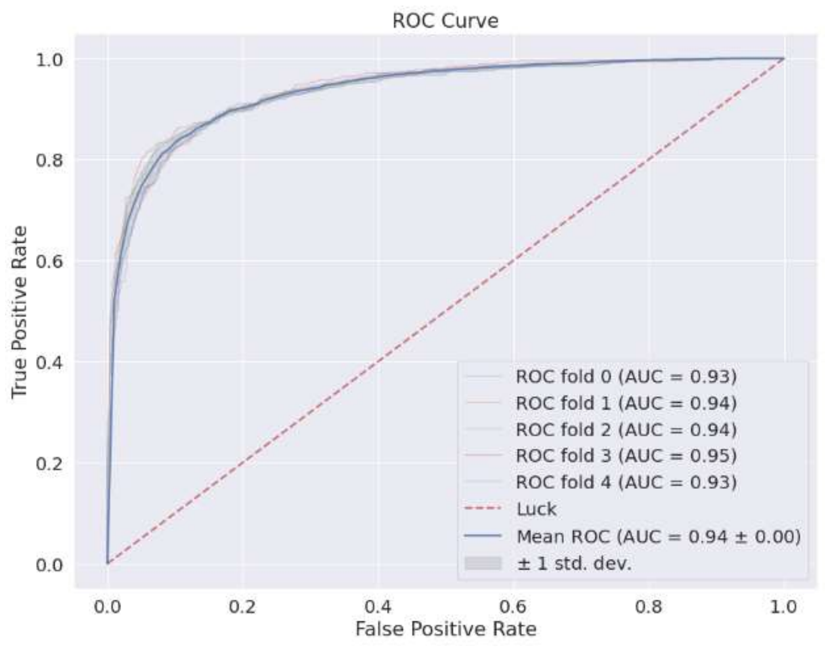


Fig. 5 ROC curve of the second phase (SBU dataset) model.

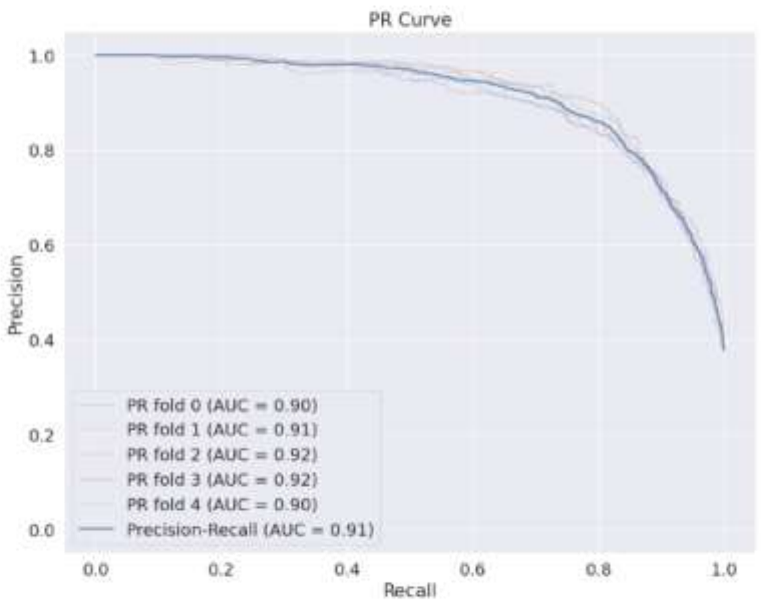


Fig. 6 Precision-Recall curve of the second phase (SBU dataset) model.

L) *First IEEE8023 Dataset Model vs. Second Model (SBU Dataset) Performance Comparison.*

The performance metrics discussed so far were from the models' evaluation on the validation set of the respective datasets. However, as noted before, a test set was also set aside for making a comparison between the performance of the best model from the first phase (IEEE8023 Dataset) and the model from the second dataset (SBU dataset). The table below summarized the model's performance metrics on the test set.

TABLE II
COMPARISON OF PERFORMANCE METRICS BETWEEN THE FIRST IEEE8023 DATASET MODELS & THE SBU DATASET MODEL

Metrics	**First Model (IEEE8023 dataset)**	**Second Model (SBU dataset)**
Balanced Accuracy	0.512	0.860
ROC AUC Score	0.534	0.928
Average Precision	0.444	0.829
Average Recall	0.080	0.815
F1 Score	0.136	0.822
True Positive Rate	0.080	0.815
True Negative Rate	0.944	0.906
False Positive Rate	0.056	0.094
False Negative Rate	0.920	0.185

The discrepancy in the performance between the models from the two stages can be easily noticed. The best model from the first stage achieves an accuracy of just 51.2%, significantly down from the 92% accuracy seen earlier on the validation set metrics. But it's not just the accuracy, every metric shows a huge decrease in performance here compared to the metrics seen earlier using the validation set. This could be attributed to the model's failure to generalize to unseen data, as expected, due to the limited number of images in the IEEE8023 dataset [14]. On the other hand, the model from the second stage still maintains high accuracy of 86%, down by just 0.7% from the accuracy seen on the validation set. This shows that the second model managed to generalize well to the unseen test data, which again could be attributed to the size of the dataset, in this case, the large size of the SBU dataset.

The confusion matrices on the test set are also shown below for both models.

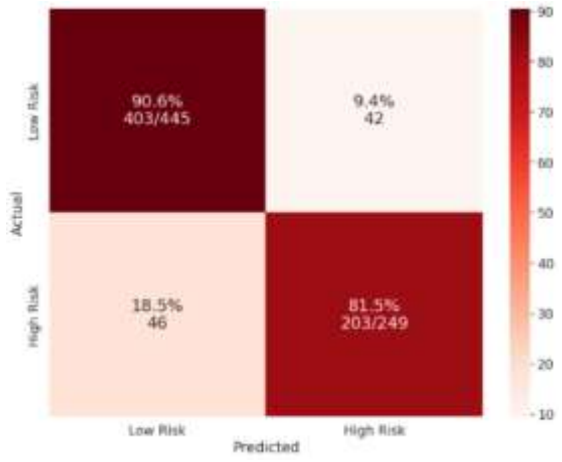


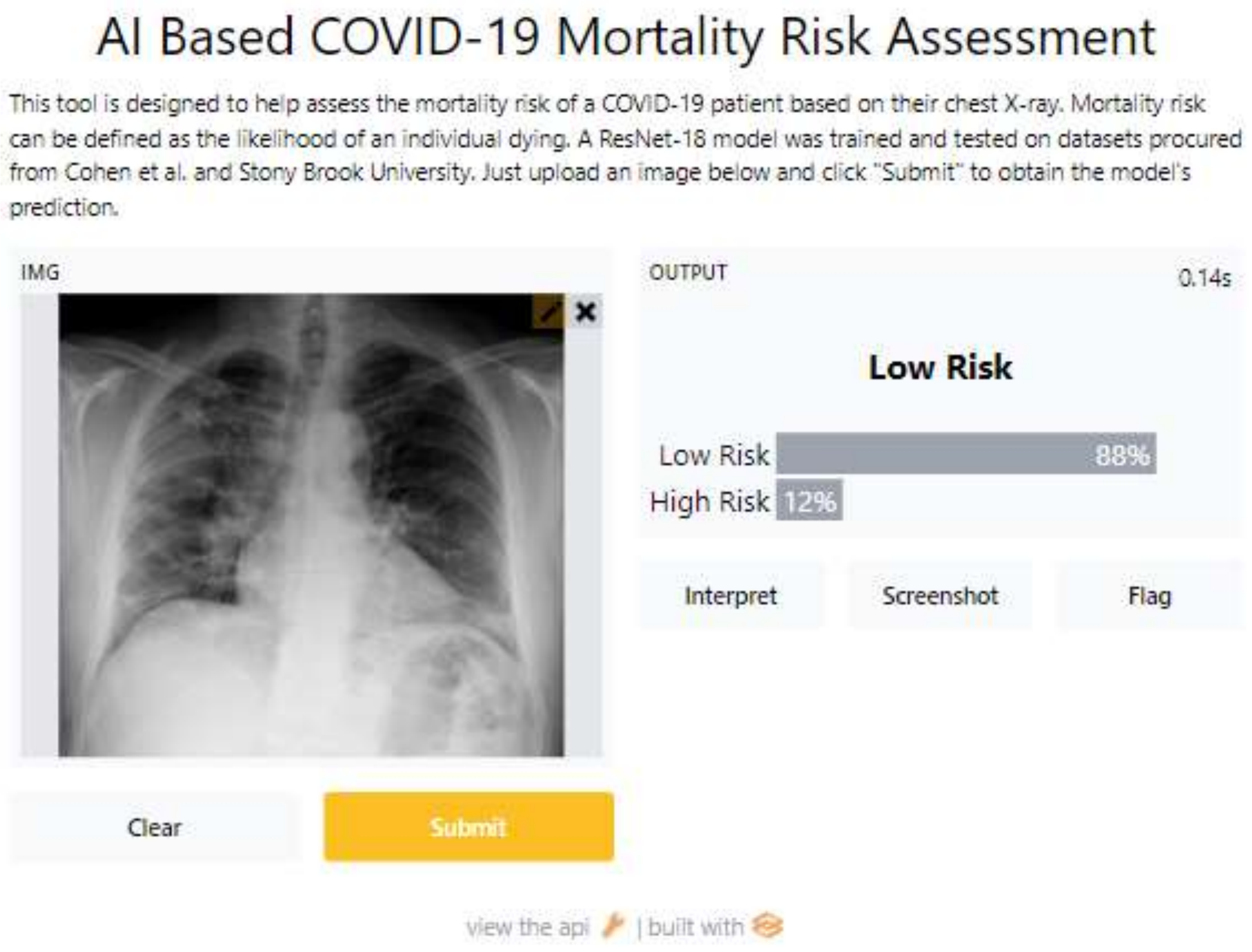
Fig. 7 Test set confusion matrix of the second phase (SBU dataset) model.

This second model does much better with 81.5% of high-risk patients classified correctly and only 18.5% of high-risk patients classified incorrectly as low risk.

From the above, it can easily be seen that the model from the second dataset achieved a very significant and clear improvement in performance over the best model from the first stage with the IEEE8023 dataset [14].

M) Model Deployment

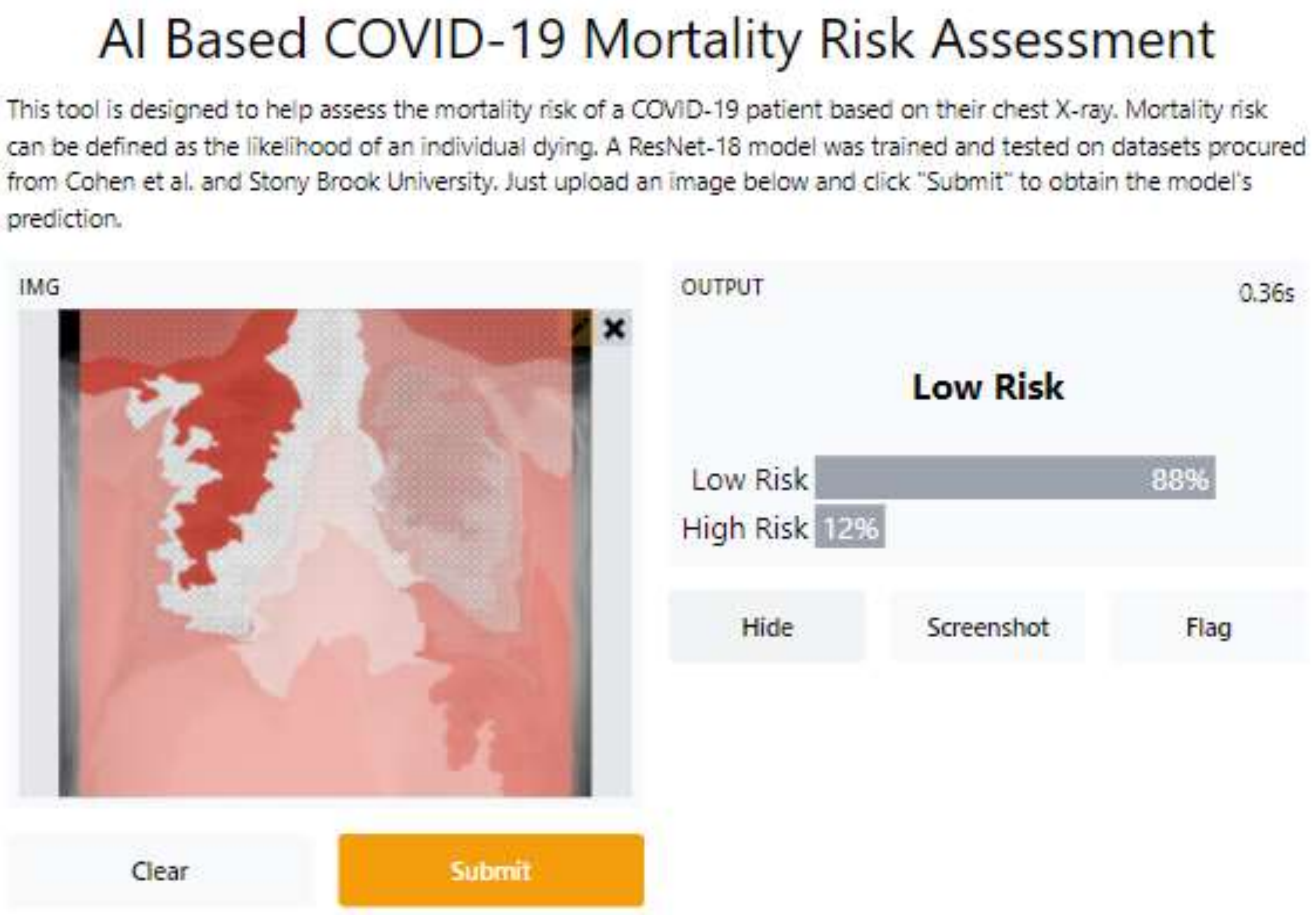
The model was deployed for the purpose of demonstration and external testing. Gradio was used for the front-end interface, while Heroku was used for hosting and deployment. Gradio is a Python-based library that allows one to easily create a web interface for machine learning models. Heroku is a platform as a service (PaaS) that facilitates the convenient development and operation of applications on the cloud. The interface provides the user with the ability to upload their image to get predictions from the model(Fig. 8). Besides that, one can also visualize the activation layer as an overlay to see which features of the image led to the prediction (Fig. 9). It can be accessed here at https://covid-mortality.herokuapp.com/



AI Based COVID-19 Mortality Risk Assessment

This tool is designed to help assess the mortality risk of a COVID-19 patient based on their chest X-ray. Mortality risk can be defined as the likelihood of an individual dying. A ResNet-18 model was trained and tested on datasets procured from Cohen et al. and Stony Brook University. Just upload an image below and click "Submit" to obtain the model's prediction.

Fig. 8 Gradio interface with prediction.



AI Based COVID-19 Mortality Risk Assessment

This tool is designed to help assess the mortality risk of a COVID-19 patient based on their chest X-ray. Mortality risk can be defined as the likelihood of an individual dying. A ResNet-18 model was trained and tested on datasets procured from Cohen et al. and Stony Brook University. Just upload an image below and click "Submit" to obtain the model's prediction.

Fig. 9 Gradio interface with activations visualized.

V. Conclusion

In this paper, a COVID-19 mortality risk prediction deep learning model was trained using the Stony Brook University (SBU) dataset, retrieved and transformed from The Cancer Imaging Archive (TCIA) repository. The first stage of the study reported [14] utilized the IEEE8023 dataset – obtained from a public GitHub repository. The model using the original unimputed data with class weight adjustment achieved the best performance among the six models tested. This model was utilized in this current study using the Stony Brook University (SBU) dataset. Afterward, the first model i.e., the best model from the first phase of training, and the second model i.e., the model that was finetuned on the SBU dataset, were compared using a test set created from a portion of the SBU dataset.

The result showed that the second model trained on the SBU dataset performed much better in terms of accuracy, significantly outperforming the first model trained on just the IEEE8023 dataset, with an accuracy of 86%. This was despite the chest X-rays were not discriminated against based on the patients' duration of infection at the time of X-ray. The model also achieved a high recall value of 81.5% with a false negative rate of 18.4% on the test set.

It may be possible to further enhance the model by trying other pre-trained models with different architectures, collecting more X-ray images of COVID-19 cases and filtering the cases based on the duration of infection, and then fine-tuning the existing model on the new data. Firstly, larger models tend to be suitable for data that have a large number of variations and patterns to be learned, so they may produce a better accuracy when it comes to modeling X-ray images, although this need not be the case. Secondly, since the infection worsens gradually for the first few days, the chest X-rays that were taken a few days after the infection could be more indicative of the patient's condition and mortality risk, and hence may be more suitable to be used for training. Lastly, the limited number of data, especially X-ray images of non-survival cases, required us to use oversampling methods such as image augmentation to bring up the number of images in the minority class. Although image augmentation is a decent technique to create more variations in the dataset, it still does not adequately account for the variations that may be seen in the real world, hence collecting more X-ray images of non-survival cases and fine-tuning the model on them could potentially yield better results.

VI. Acknowledgement

The authors hereby acknowledge the review support offered by the IJPCC reviewers who took their time to study the manuscript and find it acceptable for publishing

VII. Conflict of Interest

The authors declare that there is no conflict of Interest

References

[1] W. Zhu *et al.*, "Establishing and Managing a Temporary Coronavirus Disease 2019 Specialty Hospital in Wuhan, China," *Anesthesiology*, vol. 132, no. 6, pp. 1339–1345, 2020, doi: 10.1097/ALN.0000000000003299.

[2] "Covid-19: Army opens hospitals to civilians, setting up temporary facilities | India News - Times of India." https://timesofindia.indiatimes.com/india/covid-19-army-opens-hospitals-to-civilians-setting-up-temporary-facilities/articleshow/82318086.cms (accessed Jun. 18, 2022).

[3] Centers for Disease Control and Prevention, "Coronavirus Disease 2019 (COVID-19) – Symptoms." 2021.

[4] Centers for Disease Control and Prevention, "COVID-19 and Your Health." 2020.

[5] R. Anand, "Patients turned away as ICU wards fill up in Malaysia." 2021.

[6] N. Islam *et al.*, "Thoracic imaging tests for the diagnosis of COVID-19.," *Cochrane database Syst. Rev.*, vol. 11, p. CD013639, Nov. 2020, doi: 10.1002/14651858.CD013639.pub3.

[7] S. Wang *et al.*, "A fully automatic deep learning system for COVID-19 diagnostic and prognostic analysis," *Eur. Respir. J.*, vol. 56, no. 2, 2020, doi: 10.1183/13993003.00775-2020.

[8] A. R. Kulkarni *et al.*, "Deep learning model to predict the need for mechanical ventilation using chest X-ray images in hospitalised patients with COVID-19," *BMJ Innov.*, vol. 7, no. 2, pp. 261–270, 2021, doi: 10.1136/bmjinnov-2020-000593.

[9] E. Luz *et al.*, "Towards an effective and efficient deep learning model for COVID-19 patterns detection in X-ray images," *Res. Biomed. Eng.*, 2021, doi: 10.1007/s42600-021-00151-6.

[10] B. G. S. Cruz, M. N. Bossa, J. Sölter, and A. D. Husch, "Public Covid-19 X-ray datasets and their impact on model bias - a systematic review of a significant problem," *medRxiv*, 2021, doi: 10.1101/2021.02.15.21251775.

[11] M. D. Zeiler *et al.*, "On rectified linear units for speech processing," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 3517–3521, doi: 10.1109/ICASSP.2013.6638312.

[12] K. Clark *et al.*, "The Cancer Imaging Archive (TCIA): Maintaining and Operating a Public Information Repository," *J. Digit. Imaging 2013* 266, vol. 26, no. 6, pp. 1045–1057, Jul. 2013, doi: 10.1007/S10278-013-9622-7.

[13] "Stony Brook University COVID-19 Positive Cases (COVID-19-NY-SBU) - The Cancer Imaging Archive (TCIA) Public Access - Cancer Imaging Archive Wiki." https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=89096912 (accessed Jun. 18, 2022).

[14] Y. Mohammed, R. S. Batha, A. Olowolayemo, and W. K. Shams, "Deep Learning Models for Prediction of Mortality Risk in Patients with Covid-19 Using Chest X-Rays," *Submitt. to Malaysian J. Comput. Sci.*, 2022.