# Survey on Convolutional Neural Network Models that Detect COVID-19 using Chest X-Ray Images

You Tube Video Link
Ujjal Saha , Ashish Pradhan , Dilip Ravindran
*The Grainger College of Engineering, Department of Computer Science, University of Illinois – Urbana Champaign*
*201 North Goodwin Avenue, Urbana, Illinois 61801-2302, United States of America*
ujjals2@illinois.edu , apradh6@illinois.edu , dilipr2@illinois.edu

***Abstract*— It is critical to accurately detect positive COVID-19 cases because of its high R0 factor and mortality rate compared to other widespread airborne diseases like the common flu. Recently, there have been many Convolutional Neural Network (CNN) models that detects positive COVID-19 from Chest X-Ray images. In this paper, four of the proposed models that have claimed high accuracy, precision and recall for binary classification have been reimplemented. They are EMCNet, DarkCovidNet, CoroNet and an unnamed model proposed by Haque et al. The purpose of the reimplementation is to demonstrate the veracity of the claims made by these models while using a common set of data source, software package, environment etc. Standard performance metrics validating the models' effectiveness were compared, along with their computational performance. Critical analysis and any further improvements that can lead to a more practical solution were discussed. It was observed that all four models could be replicated fairly accurately, with similar metrics as mentioned in the original publication. All the models had accuracy of over 96% on a balanced test dataset, which was achieved within 100 epochs. CoroNet and DarkCovidNet performed better than Haque et al and EMCNet, which could be due to the fact that they were derived from existing architecture, Xception and DarkNet respectively, whereas the other two were created from scratch. This paper intends to provide an unbiased and neutral review of these existing proposed models in the form of a comparative survey in order to aid the researchers and experts in choosing the most relevant one to detect COVID-19 patients from Chest X-Ray images.**

***Keywords—COVID-19, Deep Learning, Healthcare, Survey, CNN, Convolutional Neural Network, Classification, Chest X-Ray***

## I. Introduction

On the last day of 2019, the World Health Organization (WHO) China Office was informed about pneumonia cases from unknown cause in Wuhan City located in Hubei province of China. The WHO started monitoring[1] the cases and publicly reported clusters of pneumonia cases in Wuhan, China. On Jan 10, 2020 WHO issued its first test toolkit for countries to check their ability to detect novel coronavirus. The novel coronavirus infection [1] started to rapidly spread from Wuhan to the rest of the world within a short span of a few months. On Feb 11, 2020, the WHO named the virus COVID-19 (Coronavirus Disease 2019). On Mar 11, 2020, an assessment was made by WHO and COVID-19 was declared a global pandemic. COVID-19 was classified as highly infectious, can be easily spread from human to human through air droplets and classified as deadly. Not just that, classifying positive cases quickly was attributed as a lifesaving event in order to stem the spread, prevent deaths as the most important form of defense against the virus is to socially isolate positive cases. The current COVID-19 detection process [2] is a lab-based testing process which takes anything between few hours to even days to produce test results. As the virus primarily attacks the lungs and hampers pulmonary functions, leading to inflammation and even respiratory failure, Chest X-Ray (CXR) images is the only known form of data which researchers can actually visualize and therefore diagnose with.

Many CNN models for COVID-19 detection using CXR images approaches have been put forward in the last few months by different group of researchers and experts with each approach claiming high precision and accuracies. Each of these approaches were independent research, therefore using their distinct datasets, image resizes, image preprocessing, training etc. Hence, there is a possibility that while looking at these proposed models and its metrics, they cannot be compared to one another. The comparison of these approaches can only be done if and only if they were tested under same platform with same set of data sources, environments, software implementation, same training to test data ratio etc.

This paper is putting forward a comparative survey data where it will present a comparison of the already proposed approaches of CNN based COVID-19 detection from CXR images. This paper will reimplement four of the proposed models that claim to have accuracy more than 95%. The reimplementation will be done on a common platform using common positive COVID-19 CXR data source, non-COVID-19 CXR data sources, common image resize dimension, common environment, common software packages etc. Once the reimplementation is done, the metric data will be extracted from those models and presented in the form of a final result for analysis.

**The major contributions of this paper are as follows:**

1) Reimplement already proposed four CNN models using common set of inputs and common environmental setup
2) Present the vercity of their claims in the form of metrics comparison chart (precision, accuracy, auc and f1 score)
3) Present the overall performance of these models
4) Present unbiased observations and challenges encountered (if any) during the reimplementation.

[1] https://www.who.int/emergencies/diseases/novel-coronavirus-2019/events-as-they-happen

The remaining paper is summarized as follows. Section 2 represents recent research for the detection of COVID-19 from CXR radiology images using CNN. Section 3 provides a full description of dataset, data preprocessing, reimplementation methodology. Section 4 provides performance analysis of these models with respect to evaluation metrics and comparative evaluation. Finally, Section 5 discusses meaning, importance and relevance of the results with the related work along with the observations. Section 6 concludes the paper.

## II. Related Work

Since COVID-19 outbreak, many researchers have been trying their best to find automated COVID-19 detection model using CNN deep neural networks. This section explores eight such recent proposed models related to CNN deep neural network-based COVID-19 detection that will help determining certain criteria for shortlisting papers for reimplementation. Below Table 1 represents overview of the eight papers from recent research that proposes COVID-19 detection using CXR.

**Table 1:** Recent research papers related to COVID-19 detection with input datasets, Convolutional layers along with their accuracy and precision claims

| Research Models | COVID CXR | Normal CXR | Other CXR | CNN Model | Accuracy | Precision |
|---|---|---|---|---|---|---|
| **COVINet [3]** | 78 | 28 | 79 | 7 Layer CNN | 97.2% | 98.2% |
| **Sarki et al [4]** | 296 | 1341 | 3785 | 16 Layer CNN | 100% | Unknown |
| **Ioannis et al [5]** | 224 | 504 | 700 | VGG16 & more | 98.75% | Unknown |
| **EMCNet [6]** | 2300 | 2300 | | 20 Layer CNN | 98.91% | 100% |
| **Haque et al [7]** | 496 | 1356 | | 4 Layer CNN | 97.5% | 97.5% |
| **ConvNet [8]** | 225 | 4292 | 1583 | 6 Layer CNN | 98.5% | Unknown |
| **CoroNet [9]** | 284 | 310 | 657 | 36 Layer Xception | 99% | 98.3% |
| **DarkCovidNet [10]** | 127 | 500 | 500 | 19 Layer CNN | 99% | 98.08% |

COVINet [3] was proposed as an architecture keeping in mind the low number of COVID images available and therefore used synthetic data for the experiment. Synthetic images were generated by augmenting Dataset 1 (79 images of virus and bacterial pneumonia) and Dataset 2 (78 COVID Images and 28 normal images) into a 10,000-size dataset using Keras library ImageDataGenerator class. Image was resized to a dimension of 120 x 120 x 3. COVINet used 7 layers in its CNN modeling with 3 convolutional layers and one each of maxpooling, ReLU and dropout. It showed accuracy, precision, recall, F-score, AUC, sensitivity, and specificity as the performance evaluation metrics with accuracy of 97.2% and precision of 98.2%. This model compared their results with other benchmark image classification models like AlexNet and VGG16 [16]. The authors highlighted that this model took the lowest training time compared to others, even though VGG16 [16] was better in performance.

Sarki et al [4] proposed another COVID-19 detection approach using CNN. This experiment also used synthetic image data using Keras library ImageDataGenerator class. Their initial dataset contained 296 images, 83 female and 175 male positive COVID-19 cases. Furthermore, bacterial pneumonia x-ray images were obtained for multi class classification. The gathered data consists of 1341 healthy images, 296 images with positive and suspected COVID-19, and 3875 images with viral and bacterial Pneumonia positive images. Therefore, data imbalance can be observed in the gathered dataset, which can give misleading classification results. After removing overexposed, underexposed images, they ended up with 140 images from each category. Contrast Enhancement was done using masking. Cross Validation with k=10 resampling was done for training the model. The model used 15 layers in their CNN models with 5 convolutional layers, with a max pooling layer after every convolutional layer and then flattened after eventually running through FC layers for 3-class classification. This model does heavy preprocessing resulting in higher accuracy than VGG16 [16] for multi class classification with accuracy of 100%. This paper however is in review stage and not yet published.

Ioannis et al [5] presented an approach for COVID-19 detection. The dataset utilized in this experiment is a collection of 1427 X-Ray images. 224 images with confirmed COVID-19(Cohen), 700 images with confirmed common pneumonia, and 504 images of normal conditions. They utilized Transfer Learning, a process that involves training CNN using other datasets by identifying common characteristics such as borders, textures, or patterns. The pre-trained model retains both its initial architecture, and all the learned weights. The paper mentioned that the combination of accuracy, sensitivity, and specificity must be the criterion for choosing the best model as their dataset was imbalanced (only 224 out of 1427 are COVID Images). This paper did not implement any new model or layer, used the standard models VGG19 [12], Inception [13], Xception [11], MobileNet [14] etc. out of which VGG19 [12] showed the highest accuracy of 98.75%.

EMCNet [6] combined COVID-19 images from multiple data sources. A total of 2300 positive COVID-19 x-ray images were taken from 6 different data sources. These multiple data sources sometimes source data from each other so there could be instances of data being duplicated. Therefore, data imbalance can be observed in the gathered dataset, which can give misleading classification results. 2300 non COVID-19 CXR images were collected from the NIH CXR images data source. EMCNet then combines the images (4600 images) and splits them into three sets with the ratio of 70%:20%:10%. The first set is the training set (3220 images), second is the validation set (920 images) and third set is the test set (460 images). Images were of various sizes, but they were resized to 224 x 224 pixels and data normalization was also performed. EMCNet proposes 20 layers in their CNN models. Three epoch variations were used: 50, 60 and 100. The model used 6 convolution layers, each followed by the pair of maxpool and dropout layers with flatten and FCL at the end. The EMCNet claimed accuracy of 96.52% and precision of 100%. The results were also compared with RestNet [15], GoogleNet [13] and few others.

Haque et al [7] also combined COVID-19 images from three different data sources with the possibility of duplicated images as the sources known to have sourced image from one another. Non COVID-19 images were sourced from NIH database. Image dataset was small and consisted of 295 COVID-19 CXR images and 659 Non-COVID CXR Images. Images were resized to 224 x 224 pixels. This model has simple architecture compared to other seven models that were studied in this paper. The model mainly consists of four components: input layers, one convolutional layer, fully connected layers and output layers. For internal comparative study the paper also designed two more models but that didn't come out to be as efficient as the first

model. The model's performance evaluation metrics consisted of accuracy, precision, recall and f1-score and claimed accuracy of 97.5% and precision of 97.5%.

ConvNet [8] study uses 6100 publicly available X-ray images. The dataset has 1583 healthy, 4292 pneumonia (2790 bacterial and 1502 viral) and 225 confirmed COVID-19 cases. All images were resized to 640 x 480 pixels for uniformity. The experiments were done with k-fold cross validation using 8 folds. 12.5% of the data was used for testing and 87.5% for training. Four randomly selected images each for healthy and coronavirus-infected patients were assigned as the validation set. It used a 6-layer model with 2 convolution layers with maxpool in between each followed by two fully connected layers. This experiment claimed accuracy of 98.5%.

CoroNet [9] is another COVID-19 detection model which is based on Xception [11]. Xception is a deep convolutional neural network architecture that is known to be developed on top of Inception [13] model. CoroNet used a small dataset containing 284 COVID-19 CXR images and 997 Non-COVID-19 CXR Images. The images were resized to 224 x 224 pixels. . CoroNet is claimed to be computationally less expensive and achieved promising results on the dataset. It claims 99% accuracy with precision of 98.08% in the binary COVID-19 classification. The metric score is lower for 3-class or 4-class classification. The paper also compared its metrics with other models and claimed to outperform every model including the standard ones such as VGG19 [12], Inception [13], RestNet [15] etc. The paper also claims to perform good on larger datasets if tested.

DarkCovidNet [10] is another CNN model that has proposed for COVID-19 detection from CXR with the open-source code. The input dataset was small, only 125 positive COVID-19 CXR, 500 normal CXR and 500 pneumonia class CXR. This model used concepts from the DarkNet [17] which is an open source neural network framework and real time object detection system called YOLO [18] (You only look once). Inspired from DarkNet-19 [19], the model implemented 17 layers model with 3 convolution layers while introducing filtering variations on each layer. Evaluation was done using 5-fold cross-validation procedure. This model achieved binary and multiclass classification with an accuracy of 98.08% and 87.02%, respectively and while maintaining precision 98.08%. The model also claimed it can be tested on large dataset.

## III. Methodology

Based on the review of eight papers that were discussed in above section, certain qualifying criteria were set to decide which papers would be picked for reimplementation. They are: 1)Paper must be published 2) Detailed architecture information about the CNN layers, 3)Credible dataset sources, 4)Thorough discussion on result and analysis 5) Relevant metrics should have higher values (accuracy and precision 90% above).

Considering all the qualifying criteria four papers were selected EMCNet [6], Haque et al [7], CoroNet [9] and DarkCovidNet [10] for reimplementation. Compared to other papers, these covered a sizeable part of the implementation details along with good description of datasets and explaining result metrics. Their accuracy and precision were also more than 90%. Hence, it was decided that they will be reimplemented to validate performance and metrics claims using standardized set of inputs, ratio of training vs validation vs test data, library package, GPU resource allocation and then comparing the result analysis with the metrics generated by the reimplementation.

### *A. Description of datasets*

The proposed CNN models for COVID-19 detection were implemented using image data from different data sources, synthetic data and using different preprocessing criteria. As the goal of this paper is to set up a common ground for the experiment, so a common set of images were used as input image parameters to all the models, the data must be carefully selected and processed for the test. The availability of images is very limited in count; therefore COVID-19 positive X-Ray images were sourced from multiple sources.

**COVID-19 chest X-Ray image data source:**

– 3600 positive COVID-19 CXR images were collected from Kaggle COVID-19 Radiograpy Database. This is the hybrid database of positive COVID-19 CXR Images source from many other repos and datasets.
*Data Source: Kaggle COVID-19 Radiography Database*[2]

**Non COVID-19 chest X-Ray image data source:**

– 3600 Non COVID-19 chest X-Ray images were collected from NIH Chest X-Ray dataset.
*Data Source: NIH Chest X-Ray Dataset*[3]

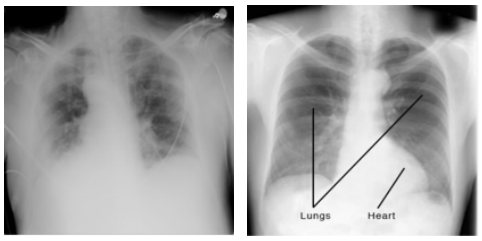



**Figure 1:** The chest X-ray on the right is normal. The chest X-ray Image on left is from COVID-19 infection (Image Courtesy: Kaggle CXR Data Source).

The input dataset for the all the models to be tested will therefore contain a total of 7200 images (3600 COVID-19 Images and 3600 non COVID-19 images). Out of total 7200 images, 60% (4320 images) will be used as training set, 20% (1440 images) will be used as validation set and remaining 20% (1440 images) will be used as test dataset. Partition of dataset into training, validation, and testing sets is described Table 2.

**Table 2:** Partition of the dataset into training, validation and testing set.

| Dataset | | COVID-19 | Non COVID-19 | Total |
|---|---|---|---|---|
| **Training Set** | **60%** | 2160 | 2160 | 4320 |
| **Validation Set** | **20%** | 720 | 720 | 1440 |
| **Testing Set** | **20%** | 720 | 720 | 1440 |
| **Combined** | **100%** | **3600** | **3600** | **7200** |

### *B. Data preprocessing*

Since sample images were not from a single data source, they were of various sizes. All the images were resized to 224 × 224 pixels. Furthermore, data normalization was performed to achieve an even distribution, making them better suited for input images for the training of the models.

[2] https://www.kaggle.com/tawsifurrahman/covid19-radiography-database
[3] https://nihcc.app.box.com/v/ChestXray-NIHCC

### C. Assumptions on the datasets

There is no strong evidence found that would establish the classification labels of the CXRs. Labels for the images such as gender, age, country, continent, etc. were not always clearly specified in the dataset. Also, COVID-19 has stages in infecting lung and the dataset has no evidence at which stage of COVID-19 images were taken and also no evidence whether there are multiple CXR of the same person are present or not. Because of the hybrid nature of COVID-19 CXR dataset (sourced from multiple other sources) there is also a possibility that there might be duplicates CXR images.

### D. Reproducing the approaches

The model architectures that were mentioned in the papers were strictly followed. Wherever, there was incomplete information about the tuning of the hyperparameters, a grid search algorithm was used to find out the best values. Data Preprocessing was uniformly done before being used as training data for the models. The visualizations as well as the calculation of the metrics were done in using a standardized approach with same packages utilized for calculations across the four models. Table 3 shows the convolution layer related information of the four models and Table 4 shows few granular details about layers.

**Table 3:** Convolutional Layer information of the selected four Models

| EMCNet [6] | CoroNet [9] | DarkCovidNet [10] | Haque et al [7] |
|---|---|---|---|
| Conv2D | Xception | ConvBlock1* | Conv2D |
| ReLU | Flatten | MaxPooling2D | ReLU |
| Conv2D | Dropout | ConvBlock2* | Conv2D |
| ReLU | Dense | MaxPooling2D | ReLU |
| MaxPooling2D | Dense | TripleConvBlock1** | MaxPooling2D |
| Dropout | | MaxPooling2D | Conv2D |
| Conv2D | | TripleConvBlock2** | ReLU |
| ReLU | | MaxPooling2D | MaxPooling2D |
| MaxPooling2D | | TripleConvBlock3** | Conv2D |
| Dropout | | MaxPooling2D | ReLU |
| Conv2D | | TripleConvBlock4** | MaxPooling2D |
| ReLU | | MaxPooling2D | Dropout |
| MaxPooling2D | | ConvBlock3* | Flatten |
| Dropout | | ConvBlock4* | FCL |
| Conv2D | | ConvBlock5* | Sigmoid |
| ReLU | | Flatten | |
| MaxPooling2D | | FCL | |
| Dropout | | | |
| Conv2D | | * | |
| ReLU | | ConvBlock# | |
| MaxPooling2D | | Conv2D | |
| Dropout | | BatchNorm2D | |
| | | LeakyReLU | |
| Flatten | | | |
| FCL | | ** | |
| Dropout | | TripleConvBlock# | |
| FCL | | ConvBlock# | |
| | | ConvBlock# | |
| | | ConvBlock# | |

**Table 4:** Convolutional Layer Summary of the selected four Models

| Selected Models | CNN Layers | Max Pooling | Activation Function | FC Layers | Other Information |
|---|---|---|---|---|---|
| **EMCNet [6]** | 6 | After every Convolution Layer | ReLU Sigmoid | 2 | Padding = 'Valid', Dropout after every conv. layer |
| **CoroNet [9]** | 36 | After first three layer, And last layer | ReLU | 1 | Adds a dropout layer and two FC layer at the end of Xception Layer |
| **DarkCovidNet [10]** | 16 | After convolution layers 1,2,5,8, 11 | Leaky ReLU | 1 | Three convolutional layer blocks: one conv layer, one batch normalization and Leaky ReLU |
| **Haque et al [7]** | 4 | After every Convolution Layer | ReLU Sigmoid | 1 | Dropout of 0.5 before connecting to FC Layer |

[4] https://colab.research.google.com/
[5] https://pytorch.org/docs/stable/index.html

### E. Metrics for performance evaluation

For the performance evaluation of EMCNet, the metrics used in this paper were accuracy, precision, recall, and F1-score. The formulas for deriving the values of these metrics are shown in (1), (2), (3), (4).

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$FI\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

In the above equations, TP means "true positive," which represents the successful prediction of actual class label of COVID-19 cases by the system. TN means "true negative," which describes the successful prediction of the actual class label of normal cases by the system. FP means "false positive," which indicates that the model misclassifies normal cases as COVID-19, but they are actually not COVID-19-infected patients. FN represents "false negative," which highlights that model misclassifies COVID-19 cases as normal. For each mode. a confusion matrix will be used to represent TP, TN, FP and FN for actual and predicted values.

## IV. Experimental Results

The experiments involve extracting features from images using CNNs and classifying COVID-19 using 4 proposed models. The models were trained on a batch size of 32 with a learning rate of 0.0001 over 100 epochs. Common datasets and image pre-processing techniques were used to evaluate the relative performance of the models. Accuracy, precision, recall and f-score were used to evaluate the performance of the models. CoroNet and DarkCovidNet provide good details about the architecture in the paper. Since the authors published the source code, the missing details of the implementation could be found.

### A. Experimental Setup

The implementation of all the four models was run on Google Collaboratory[4] (Colab). Google Colab is a Jupyter notebook-based cloud service (platform as a service) that allows researchers, students to combine executable code and rich text in a single document, along with images, HTML and more, making it ideal platform for developing deep learning models. Google Colab has integrated features to import image dataset, train an image classifier on it, and evaluate the model, all in just a few lines of code. As Google Colab notebook is only accessible via web browsers, Colab notebooks execute code on Google's cloud servers, that leverages the power of Google hardware, including GPUs and TPUs. Below snapshot (Figure 2) shows the Google Colab Free instance GPU information. The reimplementation was done using Python language leveraging PyTorch[5] 1.8.1open-source package in Google Colab.

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 465.19.01    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   35C    P8     9W /  70W |      0MiB / 15109MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
```

**Figure 2:** Google Colab free instance GPU Information

### B. *Result Analysis*

Each model was originally trained and tested on different set of images from various sources. This reimplementation standardizes the number of input images, image sizes, train/validation/test ratios, ratio of non-Covid CXR samples vs Covid CXR samples. The comparison chart in Table 5 shows the parameters of the input images used originally vs provided in this experiment.

**Table 5:** Comparison of the CXR input dataset for all the four models with the original vs provided in this experiment

| Selected Models | COVID-19 Input Images | | Non COVID-19 Input Images | | Train/Valid/Test Ratio (in %) | | Image Size (pixels) | |
|---|---|---|---|---|---|---|---|---|
| | Original | Provided | Original | Provided | Original | Provided | Original | Provided |
| EMCNet [6] | 2300 | 3600 | 2300 | 3600 | 70:20:10 | 60:20:20 | 224 x 224 | 224 x 224 |
| CoroNet [9] | 284 | 3600 | 967 | 3600 | Unknown | 60:20:20 | 224 x 224 | 224 x 224 |
| DarkCovidNet [10] | 125 | 3600 | 1000 | 3600 | 80:20:0 | 60:20:20 | Unknown | 224 x 224 |
| Haque et al [7] | 295 | 3600 | 659 | 3600 | 80:20:0 | 60:20:20 | 224 x 224 | 224 x 224 |

The development medium and environment for the models were diverse, therefore to maintain consistency and uniformity Google Colab and PyTorch were used throughout for all the tasks. Table 6 shows the development tools originally used vs the tools that were used during reimplementation.

**Table 6:** Comparison of the development platform and development package for all four model original vs in this reimplementation

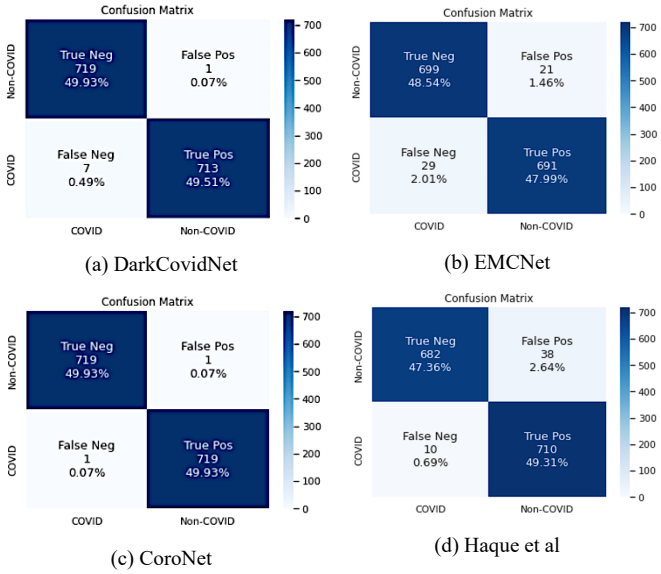
| Selected Models | Development Environment | | Development Package | |
|---|---|---|---|---|
| | Original | Provided | Original | Provided |
| EMCNet [6] | Google Colab (GPU) | Google Colab (GPU) | Unknown | PyTorch 1.8.1 |
| CoroNet [9] | Google Colab (GPU) | Google Colab (GPU) | TensorFlow 2.0 | PyTorch 1.8.1 |
| DarkCovidNet [10] | Unknown | Google Colab (GPU) | Fastai | PyTorch 1.8.1 |
| Haque et al [7] | Unknown | Google Colab (GPU) | Unknown | PyTorch 1.8.1 |

The models have different architectures, i.e., in terms of layers, activation function, parameters etc. Their best performance (achieving best results) was yielded at different level of epochs and learning rates than they claimed. During the experiment, the hyperparameters had to be set to a common value for comparison and also to bring out the best performance for each model. Table 7 shows the epoch count, learning rate and batch size that was set for all for models to a common value.

**Table 7:** Comparison of the Hyperparameters used by the four original models vs provided in the reimplementation to achieve their claimed metrics

| Selected Models | Epochs | | Learning Rate | | Batch Size | |
|---|---|---|---|---|---|---|
| | Original | Provided | Original | Provided | Original | Provided |
| EMCNet [6] | 50 | 100 | Unknown | 0.0001 | Unknown | 32 |
| CoroNet [9] | 80 | 100 | 0.0001 | 0.0001 | 10 | 32 |
| DarkCovidNet [10] | 100 | 100 | 0.003 | 0.0001 | 32 | 32 |
| Haque et al [7] | 25 | 100 | Unknown | 0.0001 | Unknown | 32 |

The four models were tested on a dataset of 1440 images (720 COVID and Non-COVID images each) which were not a part of training or validation to get the corresponding metrics. The confusion matrix (Figure 3) was created to visualize the image classification. CoroNet misclassified just 1 image each in the COVID and Non-COVID class whereas EMCNet misclassified 21 as COVID and 29 as Non-COVID images.




(a) DarkCovidNet (b) EMCNet

(c) CoroNet (d) Haque et al

**Figure 3:** Confusion Matrix of the four models

Figure 4 shows how coronet maintained a high accuracy right from the first epoch whereas the others gradually increased proportional to the number of epochs. CoroNet accuracy was high from the first epoch and it was maintained throughout whereas Haque et al and EMCNet accuracy increased gradually with each passing epoch. DarkCovidNet had a fluctuating accuracy with big differences across each epoch.

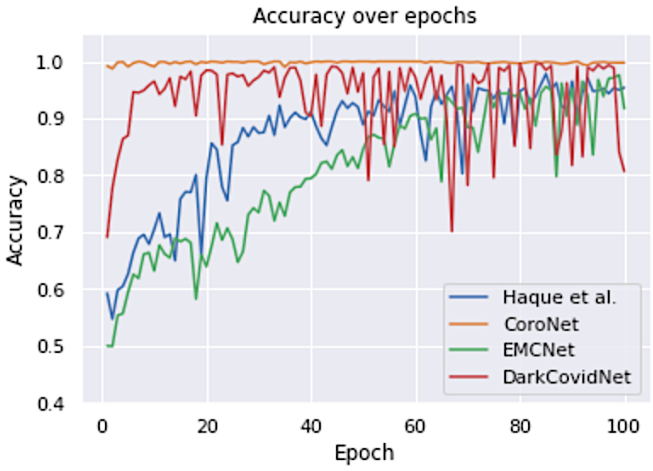
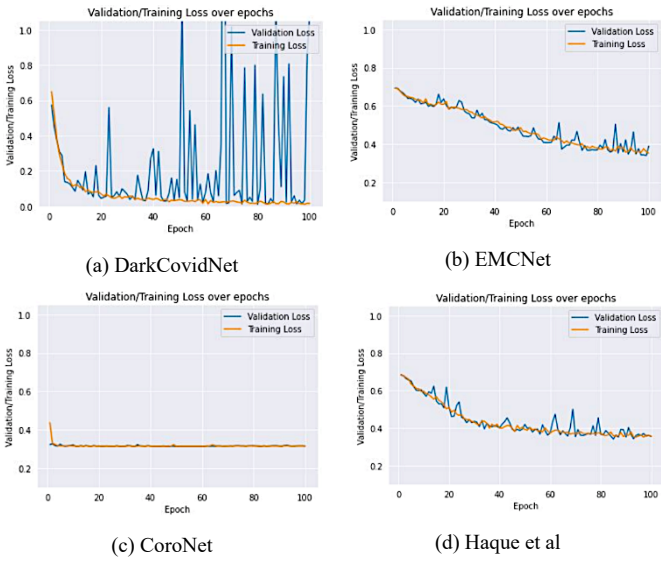



**Figure 4:** Accuracy over epoch chart for all the four models

Figure 5 shows the training and the validation loss across the epochs for all the four models. CoroNet has a fairly steady loss value across the epochs whereas DarkCovidNet losses fluctuate frequently. EMCNet and Haque et al losses were consistent with the progress of a good deep learning model, both the training and validation losses gradually decreasing after each epoch.




(a) DarkCovidNet (b) EMCNet

(c) CoroNet (d) Haque et al

**Figure 5:** Validation/Training Loss over epoch for all four models

Table 8 shows the claimed vs the observed metrics for comparison. CoroNet had the highest accuracy of 99.86% and EMCNet had the lowest accuracy at 96.53%. CoroNet also scored the highest in the other metrics while EMCNet scored the lowest amongst the four.

**Table 8:** Comparison of the claimed and observed values of accuracy, precision, F-score and Recall. All numbers are in percentages (%).

| Selected Models | Accuracy | | Precision | | Recall | | F-Score | |
|---|---|---|---|---|---|---|---|---|
| | Claimed | Observed | Claimed | Observed | Claimed | Observed | Claimed | Observed |
| EMCNet [6] | 96.52 | 96.53 | 100 | 96.53 | 96.52 | 96.53 | 98.23 | 96.53 |
| CoroNet [9] | 99.00 | 99.86 | 98.30 | 99.86 | 99.30 | 99.86 | 98.50 | 99.86 |
| DarkCovidNet [10] | 98.08 | 99.44 | 98.03 | 99.45 | 95.13 | 99.44 | 96.51 | 99.44 |
| Haque et al [7] | 98.30 | 96.67 | 96.72 | 96.74 | 100 | 96.67 | 98.30 | 96.67 |

The following chart (Table 9) shows the performance in terms of runtime showing how much training time each of the implemented model took to complete the execution.

**Table 9:** Training time of the four models when reimplemented

| Model | Training Runtime (sorted: lowest first) |
|---|---|
| DarkCovidNet [10] | 1 hour 46 mins |
| Haque et al [7] | 1 hour 57 mins |
| EMCNet [6] | 2 hours 15 mins |
| CoroNet [9] | 2 hours 58 mins |

## V. Discussion

To avoid the risk of implementing incomplete and semi-informative models, EMCNet, DarkCovidNet, CoroNet and the unnamed model by Haque et al were chosen from the numerous other papers which were reviewed during literature survey about the topic. These 4 were published in reputed journals and had information which made implementing the models easier than the rest, e.g., the data sources, details of the model layers, selection of the optimizers and effective reasoning behind the selection of their metrics to name a few. CoroNet and DarkCovidNet were the pick of the papers as the architecture were well detailed and published scores were close to observed scores. They were also derived from established architecture of Xception and DarkNet respectively which could have been a factor for their higher scores.

Standardizing the data preprocessing, computational resources, code libraries as well as utilizing the same images for training, testing and validation leveled the playing field for comparison. The high performance of CoroNet could be due to a higher number of convolutional layers. Maxpool and Activation layers after each convolution could also have played a factor in increasing the accuracy and other metrics. A higher recall for the positive COVID cases is of paramount importance compared to the other metrics as their accurate classification can lead to the correct diagnosis of the disease. DarkCovidNet and CoroNet both were better in this metric with above 99% compared to approximately 96% for the rest. However, looking at the training vs validation loss of DarkCovidNet, it can be seen that there is considerable fluctuation in the validation loss. This can be due to 2 reasons: an overcomplex model with many layers and the lack of a drop out layer anywhere in the architecture. Both results in overfitting which could be the case for this model.

A few challenges were encountered during the replication of the models in a standardized environment. Translating codes written in TensorFlow and Fastai to PyTorch was an area where considerable effort was invested. A major caveat that was noticed in all the papers was that they essentially lacked mentioning the exact hyperparameter values that the authors used to tune their models. For example, Haque et al had no mention of the learning rate, which is considered the most important hyperparameter while training the model. Due to this, although the metrics could be replicated almost completely, the exact reproduction of the models was difficult.

The primary limitation of this study was the lack of metadata of the images sourced from various sources. This was a concern in all the papers that were reviewed. There is no way to distinguish the images except for them being COVID-19 or Non-COVID-19. No demographic information was provided, even the serial number of the images are mere identifiers and are not exclusive to different patients, i.e., two CXRs could have been of the same patient but taken at different times. Not only that, but there could also very much be cases of duplication of images, all of which can make the model overfit and less robust. A central database with effective metadata can go a long way in making the existing models train more effectively.

There is a lot of scope of further work in this study. Creating a dataset with suitable metadata would give further validation about the robustness of the model. Though currently the metrics are already commendable, an improved dataset would make these models more credible. Even though DarkCovidNet had a good evaluation, there were some aspects which needs further analysis, like the fluctuation of the validation loss and check if an increase in the epochs can mitigate it. Finally, this survey can be further expanded by including similar papers on this topic as well as expanding the model to include multi-class classification of other pulmonary diseases like pneumonia.

An important takeaway from this study was that the models need not be highly complex with a large number of layers to make it efficient classifications. DarkCovidNet with 99.44% accuracy has 16 CNN Layers whereas Haque et al with 96.67% has just 4 of them. An overcomplex layer can lead to overfitting as well which is suspected in the case of DarkCovidNet. However, it was noted that the complex models achieved higher accuracy and other metrics earlier in the model training with CoroNet based on Xception which has over 30 layers maintained over 98% accuracy from the first epoch itself.

## VI. Conclusions

In this study, a standardized comparison of the four most effective COVID models was done. Using the same dataset and computational environment, various metrics such as accuracy, precision, recall, AUC, f1-score as well as the training time were calculated in order to see if they could be reproduced as mentioned in the original papers.

CoroNet was the best performing model with 99.86% accuracy, 99.86% precision and 99.86% recall. EMCNet was the least performing model with 96.53% accuracy, 96.53% precision, 96.53% recall. Even though DarkCovidNet was one of the more complex ones with 17 convoluted layers, it took the least amount of time with it completing in 1 hour and 46 minutes. CoroNet took 2 hours and 58 minutes to run which made it the slowest of the four. Furthermore, DarkCovidNet and EMCNet could be easily reimplemented due to the authors including all the essential parts of the model, including the parameters and the architecture. Clarity about the hyperparameters should be mentioned in the papers which could have led to better reimplementation of the rest. Overall, this study briefly summarizes the recent research done in the study of Chest X-Ray imaging using CNN models and reiterates how this field could be a potent tool in fighting the long battle in eradicating COVID-19 from the world.

## VII. Acknowledgment

Our sincere thanks to Prof Jimeng Sun, TAs and other course staff of CS598 Deep Learning for Healthcare, Department of Computer Science, University of Illinois – Urbana Champaign, Illinois for providing us an opportunity to do this project and also for all the help and guidance along the way. We are also thankful to the chest X-Ray dataset providers for the quality of data shared in the public domain for research and development, Google Colab for providing the platform to run neural network models in GPU enabled notebook platform free of charge, Also grateful to our peers for the collaboration and discussions throughout the course on Piazza and Slack.

## REFERENCES


[1] C. C. Lai, T. P. Shih, W. C. Ko, H. J. Tang and P. R. Hsueh. Severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) and coronavirus disease-2019 (COVID-19): The epidemic and the challenges. International Journal of Antimicrobial Agents. Vol 55, Issue 3, Mar 2020, 105924. https://doi.org/10.1016/j.ijantimicag.2020.105924

[2] B. Udugama, P.Kadhiresan, H. N. Kozlowski, A. Malekjahani, M. Osborne, V. Y. C. Li et al. (2020). Diagnosing COVID-19: The Disease and Tools for Detection. ACS nano, 14(4), 3822–3835. https://doi.org/10.1021/acsnano.0c02624

[3] M. Umer, I. Ashraf, S. Ullah, A. Mehmood & G. S. Choi. COVINet: a convolutional neural network approach for predicting COVID-19 from chest X-ray images. Jan 2021, J Ambient Intell Human Comput (2021). doi: https://doi.org/10.1007/s12652-021-02917-3

[4] R. Sarki, K. Ahmed, H. Wang, Y. Zhang and K. Wang. Automated Detection of COVID-19 through Convolutional Neural Network using Chest x-ray images. Feb 2021, [Preprint – Not Reviewed Not Published]. doi: https://doi.org/10.1101/2021.02.06.21251271

[5] I. D. Apostolopoulos and T. Bessiana. COVID-19: Automatic detection from X-Ray images utilizing Transfer Learning with Convolutional Neural Networks. Physical and Engineering Sciences in Medicine 43:635-40, Mar 2020 doi: https://doi.org/10.1007/s13246-020-00865-4

[6] P. Saha, M. S. Sadi and M. M. Islam. EMCNet: Automated COVID-19 diagnosis from X-ray images using convolutional neural network and ensemble of machine learning classifiers. Informatics in Medicine Unlocked, Vol 22, 2021, https://doi.org/10.1016/j.imu.2020.100505

[7] K. F. Haque and A. Abdelgawad. A Deep Learning Approach to Detect COVID-19 Patients from Chest X-ray Images. AI. 1. 418-435. Sep 2020, doi: http://dx.doi.org/10.3390/ai1030027

[8] B. Sekeroglu and I. OzsahinFirst. Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks. Sep 18, 2020, PubMed https://doi.org/10.1177/2472630320958376

[9] A. I. Khan, J. L. Shah and M. M. Bhat . CoroNet: A deep neural network for detection and diagnosis of COVID-19 from chest x-ray images. Comput Methods Programs Biomed. Nov 2020 196: 105581. https://doi.org/10.1016/j.cmpb.2020.105581

[10] T. Ozturk, M. Talo, E. A. Yildirim, U. B. Baloglu, O. Yildirim and U. R. Acharya. Automated detection of COVID-19 cases using deep neural networks with X-ray image. Computers in Biology and Medicine Vol 121, June 2020, 103792. https://doi.org/10.1016/j.compbiomed.2020.103792

[11] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp 1800-1807, doi: 10.1109/CVPR.2017.195

[12] Liu, Bao-Di & Meng, Jie & Xie, Wen-Yang & Shao, Shuai & Li, Ye & Wang, Yanjiang. (2019). Weighted Spatial Pyramid Matching Collaborative Representation for Remote-Sensing-Image Scene Classification. Remote Sensing. v11. https://doi.org/10.3390/rs11050518

[13] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1-9, doi: https://doi.org/10.1109/CVPR.2015.7298594.

[14] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv, abs/1704.04861.

[15] S Æ Jónsson, E Gunnlaugsson, E Finssonn, D L Loftsdóttir, G H Ólafsdóttir, H Helgadóttir, J S Ágústsson, 0447 ResTNet: A Robust End-to-End Deep Learning Approach to Sleep Staging of Self Applied Somnography Studies, Sleep, Volume 43, Issue Supplement_1, April 2020, Page A171, https://doi.org/10.1093/sleep/zsaa056.444

[16] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), 2015, pp. 730-734, doi: https://doi.org/10.1109/ACPR.2015.7486599

[17] J. Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016. 3

[18] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779-788, doi: https://doi.org/10.1109/CVPR.2016.91.


## ABOUT THE AUTHORS

Ujjal Saha is a graduate student from University of Illinois – Urbana Champaign pursuing MCS-DS and CS598 DLH is his 7th course in the program. By profession Mr. Saha is a Sr. Site Reliability Engineer working for a Fortune 500 software company with 14+ years of experience in software engineering. Mr. Saha was actively involved 100% during the whole life cycle of this project.

Ashish Pradhan is a graduate student from University of Illinois – Urbana Champaign pursuing MCS-DS and CS598 DLH is his 6th course in the program. By profession Mr. Pradhan is Data Analyst working for a Non Profit Foundation with 5+ years of experience in software engineering. Mr. Pradhan was actively involved 100% for the whole life cycle of this project.

Dilip Ravindran is a graduate student from University of Illinois – Urbana Champaign pursuing MCS-DS and CS598 is his 3rd course in the program. By Profession Mr. Ravindran is Sr. Software Engineer working in a Multinational IT Solutions company with 10+ years of experience in software engineering. Mr. Ravindran was actively involved 100% for the whole life cycle of this project.

## CONTRIBUTIONS

The team members went through the project requirement documentation individually and then discussed to decide the project topic that would be worked upon. As a team around 8 papers were selected for initial literature survey. Mr. Saha studied research papers [3] [6] and [9]. Mr. Pradhan studied research papers [4] and [10]. Mr. Ravindran studied research papers [4] and [10]. The papers were studied thoroughly by the individuals to understand the implementation and then a knowledge sharing session was done where each member talked about their study. During the study each member also did the write-ups of their understanding for each paper which later got included in the related works paper documentation. More papers were read by the team members during information gathering for the reimplementation work. During a team discussion, criteria were set to shortlist papers for reimplementation. Four papers were selected for reimplementation and performance comparison with the original paper claims vs our implementation observations. Before the actual development work to begin, Mr. Saha started with the paper draft documentation, Mr. Pradhan did a proof-of-concept to finalize the development environment for the project and Mr. Ravindran worked on preparing the common input dataset of chest x-ray images that will be used as input for all the four model. The initial paper draft, environment and dataset were then reviewed by the team members, After the dataset and environment setup has been finalized, all team members joined a peer programming session and reimplemented one model (CoroNet) to conclude the best practices, common approaches and metrics that will be followed by the team for implementing remaining three models. Then each team members individually implemented EMCNet (by. Mr. Saha), DarkCovidNet (by Mr. Pradhan) and Haque et al (by Mr. Ravindran). After the models were implemented, team members reviewed other team member's code, metrics and suggested changes required to be done. Finally in a group session all four models were re-run for another round of fine-tuning, metrics generation, comparative analysis and conclusion. A final run of all four models were done with no changes. After all results were generated, the remaining portions of the paper documentation were completed in a team meeting. The team also prepped the presentation slides and presentation video in a team meeting setup. Overall, it was a great experience to work as a team and collaborate for this project to be successfully completed. All the team members worked hard for completing the project on time.