**3. Challenges of COVID-19 Detection:**

* CT scans' limited specificity might make it difficult to discover non-COVID-19 cases. Furthermore, the rays from CT scanners might pose complications for individuals who need many CT scans throughout their illness.
* Color information, with its variables such as color composition, light beams, and reflections, causes various issues.
* X-ray imaging is far more common and less expensive than traditional diagnostic examinations.
* X-rays have several advantages to CT scans, including being faster, safer, easier, and less damaging.
* Problems include the possibility of disease transmission when utilizing a CT scan scanner, as well as the technology's expensive cost, can generate major issues for patients and healthcare systems.
* The CT scan's limited specificity can make it difficult to discover non-COVID-19 instances.

# ****4. Methodology****

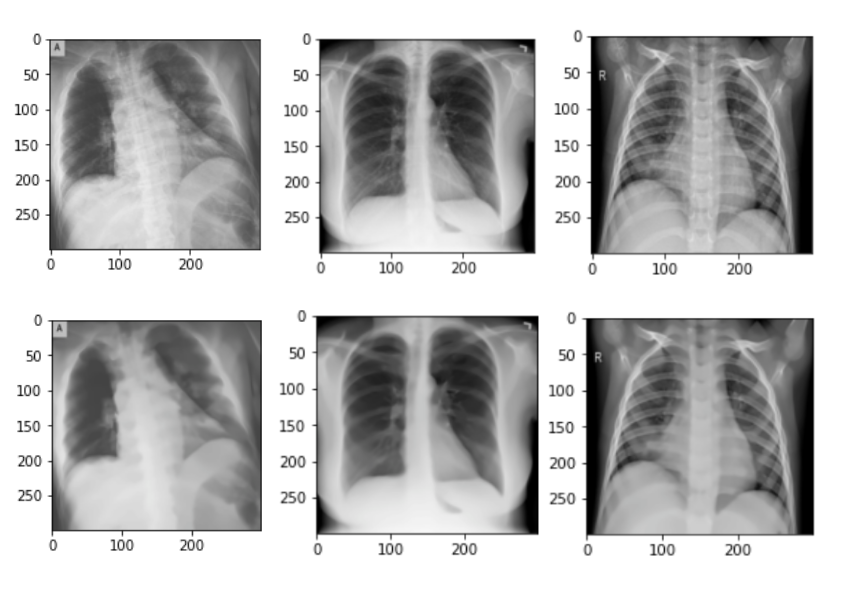
## **4.1 Dataset Class Balancing**

# Unbalanced datasets are a prevalent problem that will inevitably occur. This issue emerges when one group of classes has a significant advantage over another. The machine learning model becomes more biased towards the majority class as a result of this. It leads to a misclassification of minority groups.

# Weighted neural networks or cost-sensitive neural networks are backpropagation algorithms that may be adjusted to weight misclassification mistakes in proportion to the relevance of the class. In datasets with a strongly skewed class distribution, this allows the model to pay more attention to samples from the minority class than the majority class. The majority class's decrease in error is substantially scaled down to very tiny numbers, which may have only a slight or no influence on model weights.

## **4.2 Image Enhancement**

To eliminate even a little amount of noise, picture smoothing methods such as Gaussian blurring, Bilateral Filtering, and others are used. The Gaussian filter is a filter that is commonly used in image processing to smooth out images, reduce noise, and compute derivatives. It's a convolutional filter with Gaussian matrices as the underlying kernel. The reduction of noise from the original input picture is shown in Figure 2. A Gaussian function in two dimensions has the following formula:



**INPUT IMAGE**

**OUTPUT IMAGE**

Figure 2: Removing Noise from the Input Image

## **4.3 Data Augmentation**

Data augmentation is the process of manipulating existing data to create new data objects. Rotating, resizing, cropping, and other techniques can be used to do this with images. This is accomplished by using domain-specific approaches to transform examples from the training data into new and unique training examples. It's critical to investigate the process's resilience in preserving the same label after transformation while using Data Augmentation. Rotations and flips, for example, are usually resilient on detection tests like cat vs dog, but not on digit identification tasks like 6 against 9. In image classification, object recognition, and segmentation, data augmentation may be utilized entirely to train Deep Learning models. Table 1 shows the parameters that we utilized for data augmentation on our dataset.

|  |  |
| --- | --- |
| **Data Augmentation Parameter** | **Parameter Value** |
| Rotation Range | 5 |
| Height Shift Range | 0.3 |
| Width Shift Range | 0.3 |
| Shear Range | 0.2 |
| Zoom Range | 0.2 |
| Channel Shift Range | 10 |
| Horizontal Flip | True |
| Fill Mode | Nearest |

Table 1.  Data augmentation parameters and values.

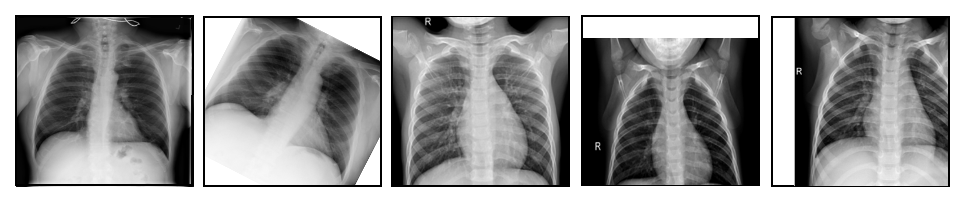


Fig. 3: During training, data augmentation techniques were used. (a) Original Image, (b) Rotation, (c) Zoom, (d) Vertical Shifts (e) Horizontal Shifts.

**4.4 Dataset**

In the era of this latest technology incredible computational power and effective applications begin with data. Various organizations in the world are coming out with the COVID19 dataset. These data are of various types like textual counts, genome sequence, 2d structure, and various features like patient's age, sex, location, symptoms, etc. We have chosen an x-ray image dataset for our diagnosing purposes. The Dataset we have used is the Covid-19 Radiography Database which has 3616 COVID-19 positive patients, as well as 10,192 Normal and 1345 Viral Pneumonia pictures, were included in the study.

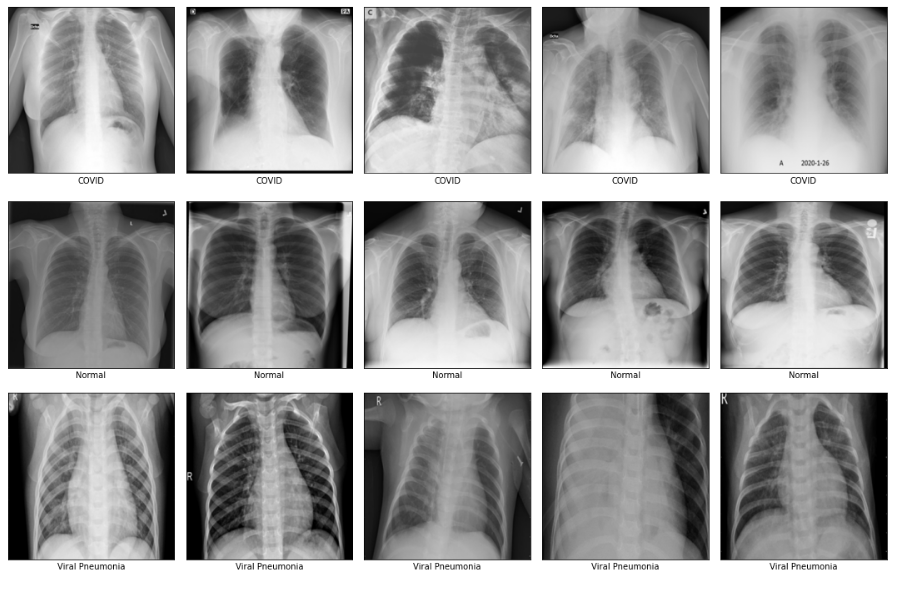


Figure 4. Samples selected from the dataset

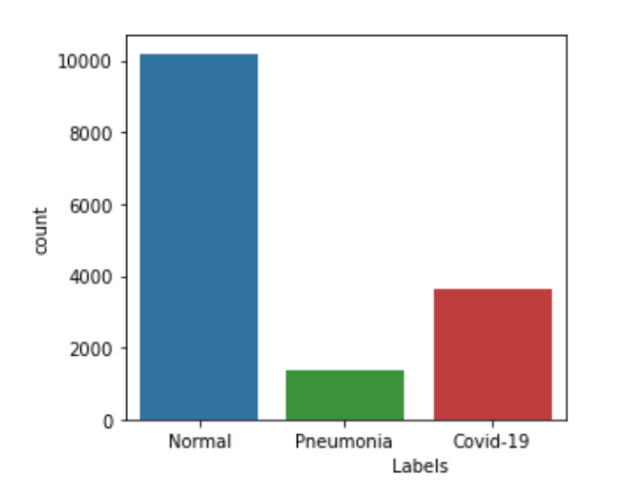


Figure 5. Label Count of the classes

## **4.5 Data Pre-processing**

Preprocessing new data and making it fit for a machine learning model is the first and most critical step. The primary goal of preprocessing our dataset is to enhance the original images by eliminating noise and artefacts caused by the gel application prior to image capture. To achieve a high classification rate, we eliminated noise and objects from the images. We have used data normalization, data elimination, feature extraction, and eventually, numerical data to convert the string data of the mark.

**4.6 CNN Hyper Parameter Tuning**

The CNN hyper parameter tuning was also done. CNN hyper parameter optimization, on the other hand, seeks to identify the best range of values for a given dataset before training can begin in a suitable period of time (e.g., the number of epochs). The introduction of residual connections overcomes the degradation problem caused by deep structures while simultaneously shortening the training period. It is capable of obtaining superior results than other CNN designs.

**4.7 Custom Model**

Transfer learning is a branch of machine learning that aims to transfer data from a source project to a target project by using correlations in outcomes, functions, or models. It may obey a range of distributions, and data annotation does not require a large number of annotations. The simulation model's characteristics and weights will be used to train new models and completely new tasks. The training model's characteristics and weights will be used to train new models and complete new activities in the model. Transfer learning allows you to use previously trained models' experience (features, weights, and so on) to train newer models and also solve challenges including getting fewer data for the newer mission. For transfer learning, the Inception-Resnet-V2 architecture with pre-trained weights was used. We froze the weights of earlier 100 layers in the custom model. The learned network does not change the parameters of the frozen layers. Many initial layer weights may be frozen to speed up network training and avoid overfitting to the dataset. Over a million pictures from the ImageNet collection were used to train the convolutional neural network Inception-ResNet-v2. The 164-layer network will categorise images into 1000 different object groups, such as the keyboard, mouse, pencil, and various animals. As a result, the network has amassed a library of rich attribute representations for a wide range of images. Multiple-sized convolutional filters and residual connections are merged in the Inception-Resnet block.

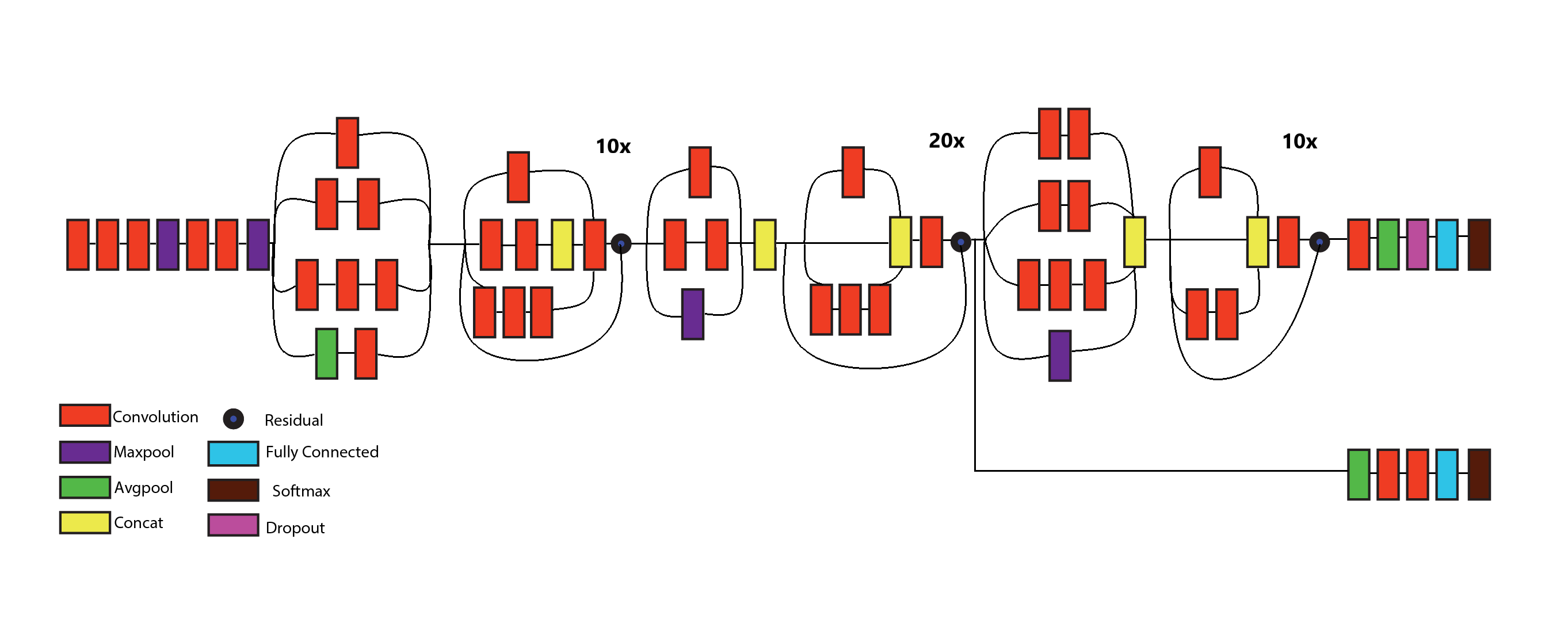
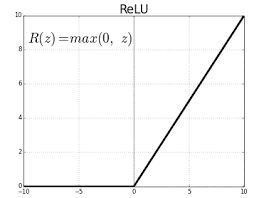


Figure 6 (a) The architecture of the Custom System

Residual connections provide for model shortcuts, allowing this architecture to achieve even higher performance. It has also allowed for significant simplification of the Inception blocks. This design is a hybrid of inception and residual blocks that improves performance. To enhance the training outcome, Inception makes better use of computational resources and can extract more features with the same amount of computation. The output of the preceding layer is combined with the network in the network since the calculation of the 5x5 convolution kernel is too big. The 1x1 convolution is used throughout the Inception module for two reasons: the first is to superimpose further convolutions over receptive fields of the same scale in order to derive richer features from the constellation diagram, and the second is to reduce the measurements and computational cost. The network's 1x1 convolution, which comes before the 3x3 and 5x5 convolutions, is used to reduce dimensionality.

Activation function used in the dense layers is the rectified linear unit (ReLU). Mathematically, it is defined as y = max(0, x).

Graphically, it looks like the following:



To minimize parameters, the fully linked layer was replaced by global average pooling, with the exception of the Inception module. The network also incorporates a batch normalization (BN) layer at the same time. The BN layer will normalize each mini-batch constellation map as it is extended to a neural network layer, preventing gradient disappearance.

a is a set of constellations that serves as the training ground for any given constellation. In the back-propagation algorithm, we must also calculate Jacobians.

In the network, Adam is used to maximize the network parameter and minimize the loss. When working with large problems with a lot of data or parameters, the method is extremely efficient. It is efficient and requires less memory.

, where  and I suppose  for some n.

The equation for Dropout is given below:

Figure 6 (a) The following steps of our Methodology

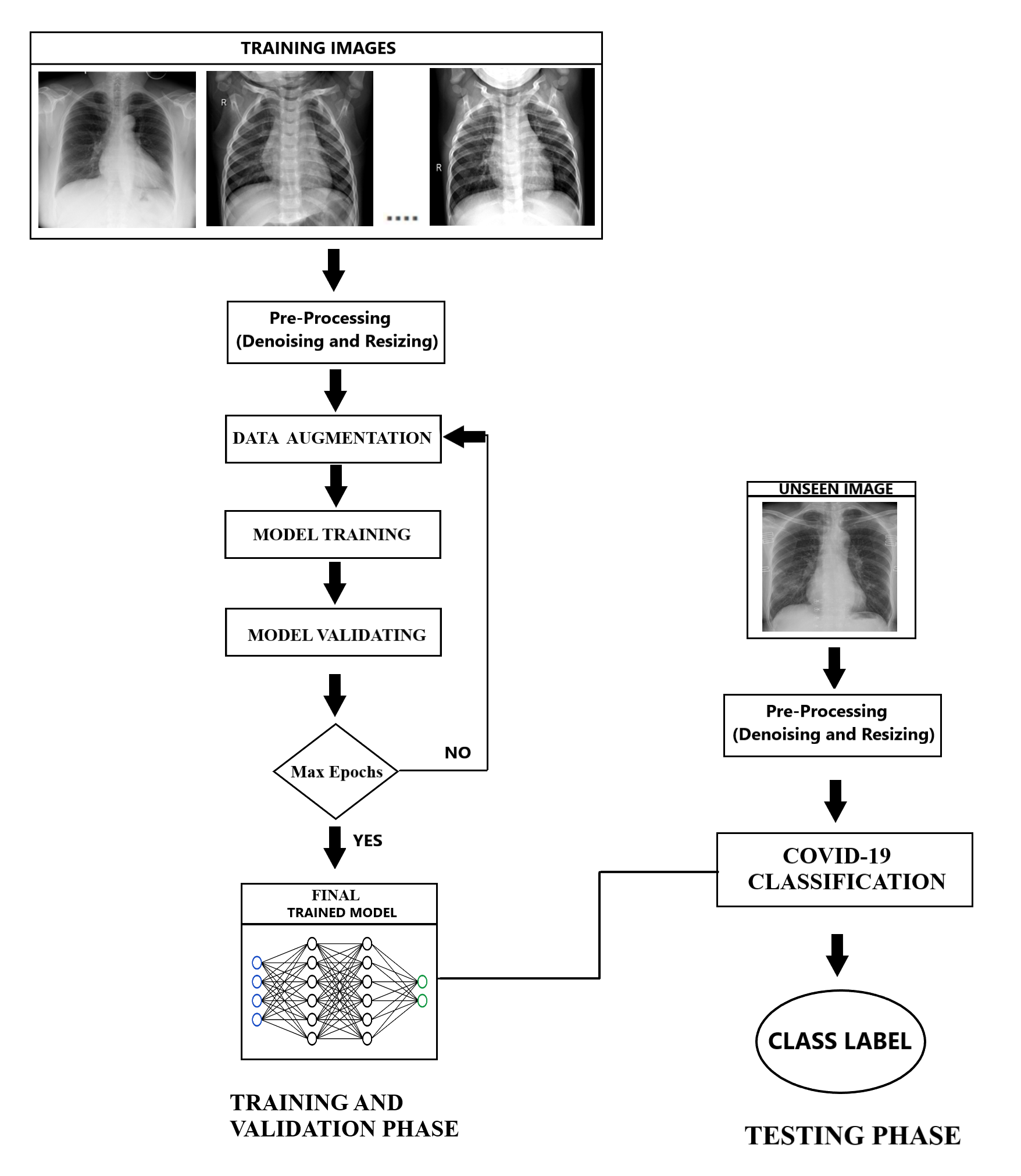


Figure 6. Schematic Flow Diagram of the Proposed System

**4.7 Comparisons Model**

**4.7.1 VGG19**

VGG19 is a deep learning architecture model that was previously trained to detect image representations on a large-scale image dataset (ImageNet). The model achieves 92.7 percent top-5 evaluation accuracy in ImageNet. When compared to other more complex models, it achieves competitive classification accuracy. It's linked to the fact that it has a strong structure. There are five sets of conv layers: two have 64 filters, two have 128 filters, four have 256 filters, and the next two sets each have four conv layers with 512 filters. Each series of conv layers contains max-pooling layers. 2x2 filters with a stride of 2 are used in max-pooling layers (pixels). The output of the final pooling layer is flattened and fed to a completely connected layer with 4096 neurons that is used for classification. The output goes to another fully connected layer with 4096 neurons, whose output is fed into another fully connected layer with 1000 neurons. Many of these layers have ReLU turned on. Finally, there's the softmax layer.

**4.7.2 ResNet50**

ResNet is the abbreviation for Residual Networks, a form of neural network. It's a 50-layer convolutional neural network (CNN). ResNet's core idea is to present a so-called "character alternative way association" that bypasses at least one layer. There are 23,587,712 parameters in the standard ResNet50 model, which can classify images into 1000 object categories. Among them, there are 23,534,592 trainable and 53,120 non-trainable parameters. Convolution with a kernel size of 7 \* 7 and 64 individual kernels, both with a stride of size 2, and max-pooling with the same stride scale as the kernels.

* + 1. **MobileNetV2**

There are two types of blocks in MobileNetV2. The first is a one-stride residual block. Another type is a two-stride block. Both types of blocks have three layers. The first layer is 1x1 convolution with ReLU6 this time. The depth wise convolution is the second sheet. Another 1x1 convolution is used in the third layer, but this time there is no non-linearity. The inner layer encapsulates the model's ability to modify from lower-level thoughts like pixels to higher-level descriptors like image levels, while the bottlenecks encode the model's ability to modify from lower-level thoughts like pixels to higher-level descriptors like image levels.

**5. Experimental Results & Discussion**

**5.1 Experimental Setup and Tools**

As a backend, Keras and Tensorflow were used to introduce the proposed process. Experiments were conducted on a computer with a Ryzen 7 Quad-Core processor, 16 GB RAM, 6 GB NVIDIA Geforce GTX 1660 Ti Max-Q RAM, and Windows 10 operating system. The networks were conditioned for 100 epochs and the Adam optimizer was used for optimization. The entire preparation took about 3 hours to complete.

# 5.2 Experimental Results

# All of the models were trained with Early Stopping callbacks for 100 epochs (patience = 20 epochs). The Adam optimizer, which combines SGD with momentum and RMSProp for faster convergence, is used with a learning rate = 0.0001 as the parameter. For all three models, the same optimizer is used, and the models are then saved as .h5 files. Figure 7 depicts the loss and accuracy plots of various models during the training and testing stages. The loss and accuracy plots for the custom model are shown in the top part of Figure 7, while the plots for VGG19, ResNet50, and MobileNetV2 are shown in the middle and lower parts, respectively.

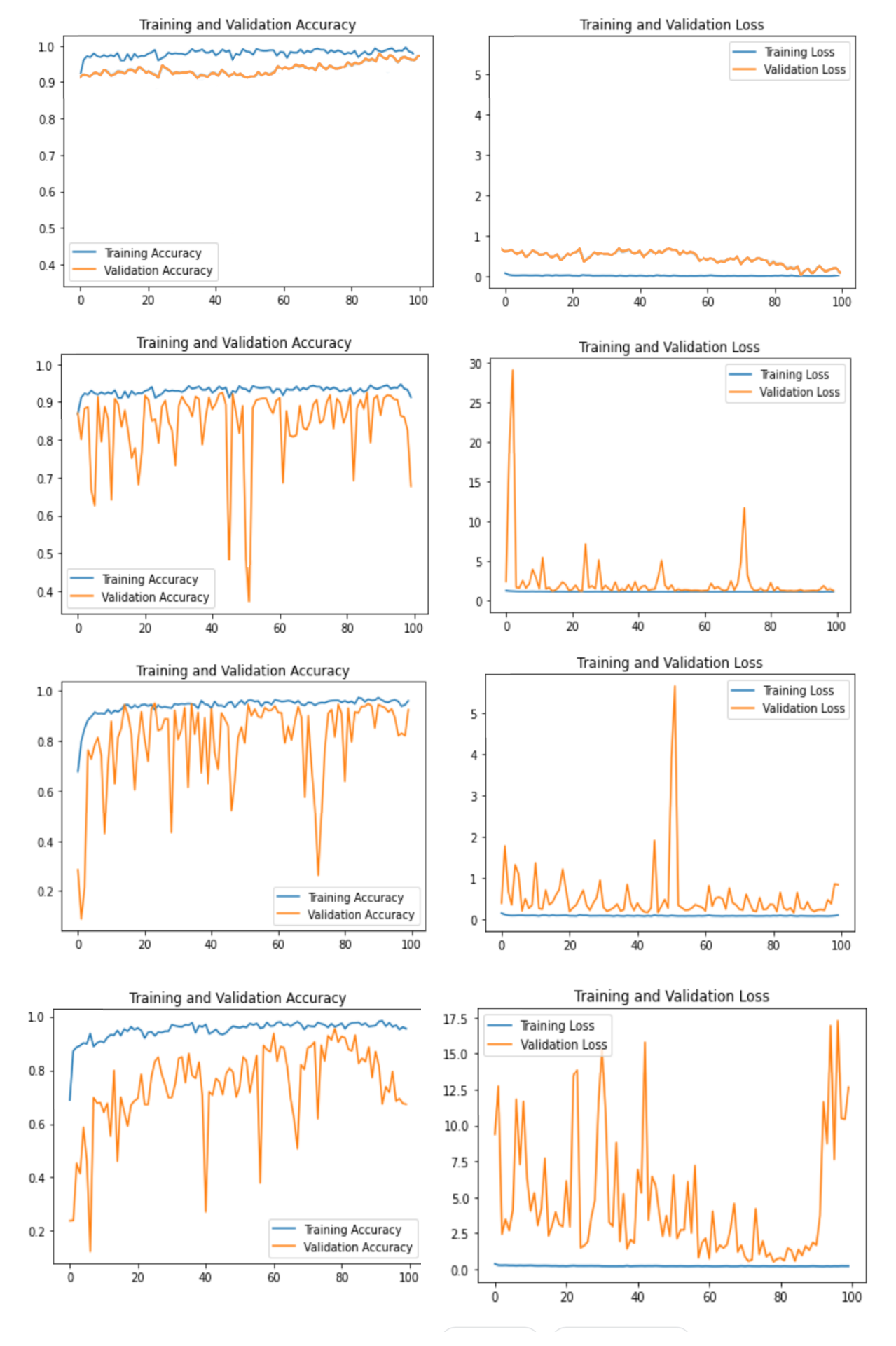


Figure 7. Accuracy and Loss of Comparison of Custom Model with VGG 19, ResNet 50, and MobileNetV2

**Table 2.** Performance Comparison

**MODEL CLASS PRECISION RECALL F1-SCORE ACCURACY**

COVID-19 0.99 0.99 0.99 0.99

Custom Normal 0.99 0.99 0.99 0.99

Viral Pneumonia 0.99 0.98 0.98 0.98

COVID-19 0.98 0.99 0.98 0.98

VGG19 Normal 0.90 0.89 0.89 0.89

Viral Pneumonia 0.88 0.88 0.88 0.88

Normal 0.87 0.88 0.87 0.87

ResNet50 Normal 0.92 0.93 0.93 0.93

Viral Pneumonia 0.93 0.92 0.92 0.92

COVID-19 0.91 0.93 0.91 0.92

MobileNetV2 Normal 0.79 0.80 0.79 0.79

Viral Pneumonia 0.75 0.75 0.75 0.75

Normal 0.74 0.76 0.75 0.74

SqueezeNet Normal 0.98 0.98 0.98 0.98

Viral Pneumonia 0.96 0.98 0.97 0.96

Normal 0.98 0.98 0.98 0.98

The different accuracy metrics for the bespoke approach are shown in Table 2. Precision, recall, and F1 score are all included. The harmonic weighted average of accuracy and recall is the F1 score. This score accounts for both false positives and false negatives. It transmits the delicate balance between accuracy and recall.

*F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)*

**5.3 The ROC curve and Confusion Matrix**

The receiver operating characteristic (ROC) curves of the classification of COVID-19, normal, and Viral Pneumonia in Fig. 8 indicate the performance of the deep-learning model. We can observe the values of each individual CLASS. The area under the ROC curve (AUC) was calculated to be 99.8%

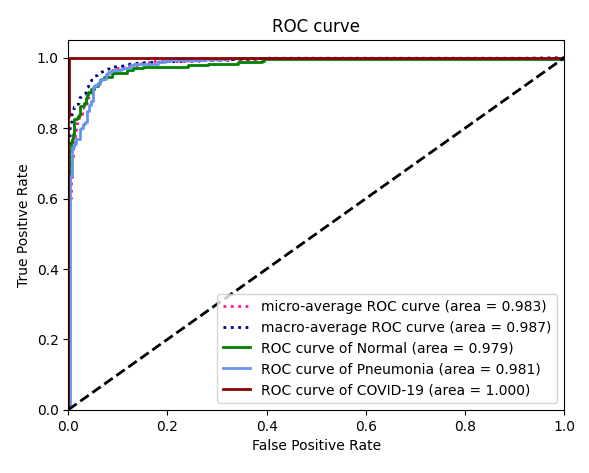


Figure 8. ROC Curve

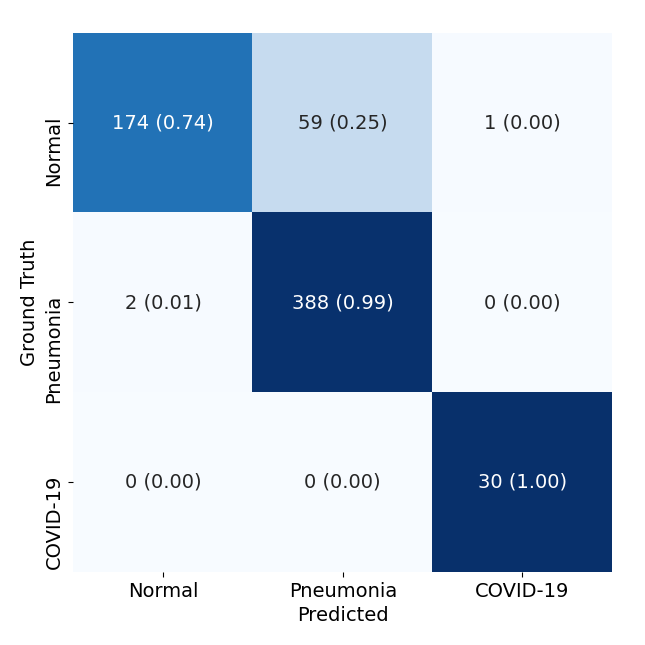


Figure 8. Confusion Matrix of the Custom Model

A confusion matrix is a table that is used to explain how well a classification model performs. It's a N \* N matrix, with N indicating the number of classes. Confusion matrix that compares model class predictions to actual classes. Type II error, also known as False Negatives, occurs in the second quadrant, whereas type I error, also known as False Positives, occurs in the third quadrant.

The confusion matrix in Fig. 8. In the confusion matrix, there are four key terms:

* **True Positives (TP):** The occurrence in which we predicted true and real yields occurred as well.
* **True Negatives (TN):** The occurrence in which the expected yield was false and the actual yield was also false.
* **False Positives (FP):** The occurrence in which we expected true and received a false result was also false.
* **False Negatives (FN):** The occurrence in which we expected a false yield but received a true yield was also true.

**Web Application or Mobile App, Serving the Model**