Deep Insights, Clear Diagnosis: VGG16's and Transfer Learning's Breakthrough in Pneumonia Detection Using Chest X-Rays

Adrika Chowdhury
Department of Computer Science
Asian University for Women
Chattogram, Bangladesh
adrika.chowdhury@auw.edu.bd

Tahiyat Islam
Department of Computer Science
Asian University for Women
Chattogram, Bangladesh
tahiyat.islam@auw.edu.bd

Tustee Mazumdar
Department of Computer Science
Asian University for Women
Chattogram, Bangladesh
tustee.mazumder@auw.edu.bd

***Abstract*— Pneumonia is indicated as a fatal lung disease that claims the lives of innocent children and also does not spare the elderly. Not being diagnosed early would bring difficulties in the patient's life, making it hard to recover from such a critical stage. Using X-ray images of the frontal side of the chest, doctors or other medical experts assess the presence of Pneumonia among patients. However, besides cancerous risks imposed by radioactive rays from these tests, diagnoses made by humans can lead to errors in accurate guessing due to contrasting intuitions made by the doctors themselves. This jeopardizes the patients' lives, as an improper decision or maltreatment could probably be made because of such an erroneous act. Hence, a deep learning-based diagnosis approach has been implemented in this study to make the diagnosis process efficient for both the doctor and the patient. This paper employs a well-known deep pre-trained convolutional neural network model using Python, namely VGG16, to diagnose Pneumonia. We also designed and implemented transfer learning through a custom Fully-Connected Neural Network (FCNN) with our base model. Our suggested model used chest X-ray images from a publicly available dataset imported from Kaggle, with 2 labels for each image ('Normal' and 'Pneumonia'). Besides this dataset and its preprocessing, data augmentation has also been implemented to train our overall model on other variances of the existing dataset. These approaches were applied to improve the accuracy in assessing the patients properly by assessing chest X-ray images through classification problems. Our test results showed a training accuracy level of 99.06% after we reached epoch 30. As a result, we realized that higher epochs lead to our model's improved performance, resulting in high accuracy and budget-friendliness in diagnosing Pneumonia. Our model's promising outcome could also help the residents of isolated areas with fewer resources for proper treatment.*

Keywords— Deep learning, CNN, VGG16, Pneumonia, Transfer Learning, Kaggle.

I. INTRODUCTION

Pneumonia is a respiratory infection characterized by the accumulation of fluid or pus and inflammation of lung tissues. Pneumonia in the general population is mostly caused by bacterial (*Streptococcus pneumoniae*) or viral (influenza) infections affecting the lungs. Here, the produced fluids accumulate within the air sacs of the lung, known as alveoli, which obstruct the normal breathing process in humans. This lethal illness manifests as fatigue, cough, thoracic discomfort, respiratory distress, and many more. Pneumonia can occur in any one of the lungs or both. Pneumonia can be classified into community-acquired Pneumonia (CAP), Hospital-acquired Pneumonia (HAP), and ventilator-related Pneumonia (VAP).

Younger children, particularly infants, and older people are more susceptible to this deadly disease. It was reported that a minimum of 80 thousand children under the age of 5 died in 2017 [1]. In addition, another analysis revealed that the number of fatalities among individuals aged 70 and beyond exceeded 1 million in 2019 [2]. Pneumonia alone causes millions of hospitalizations and deaths occurring to 50 thousand people and beyond [3].

The incidence of Pneumonia in overcrowded nations is much higher because most of the habitats in these countries have poor hygiene which is unsafe and a lack of medical resources [4]. These conditions are the direct source of infection and restrict the medical treatment making Pneumonia one of the widespread health risks in these countries. People with poor housing and lifestyles normally increase the chances of contracting the disease and as a result, the death rates also go higher. Mainly, Pneumonia is something that should be a matter of concern for all of us, but mostly for the people of age groups who are too weak (infants & elderly).

Early diagnosis is key to treating any disease, which also includes Pneumonia as well. This ensures proper well-being by retaining the survival rate of every individual irrespective of age and background. Pneumonia is detected using different methods such as Magnetic Resonance Imaging (MRI), Computed Tomography (CT), and x-rays of chests.

Chest X-ray (CXR) is the most commonly used and reliable procedure. It allows the doctors to inspect deeper into the patient's lungs. The white spots in the lungs in the CXR scanned image help detect Pneumonia among the patients. Nevertheless, using chest X-rays to diagnose Pneumonia has some shortcomings. Even though this is considered to be a reliable one, it possesses the risk of cancer closely associated with the emitted radiation from X-ray machines [5]. One of them is that individual radiologists interpret X-ray images differently leading to varied opinions about them [6]. Such interpretation variance can harm a patient's health, leading to inconsistent diagnoses and uncertainties in identifying the disease. The absence of a uniform

conclusion could lead to delayed treatment, thereby worsening patient outcomes.

On the other hand, the presence of properly skilled radiologists also can become a challenging task during this phase. Also, X-ray images might appear unclear [7]. This is why an automated system to detect Pneumonia among patients is invaluable. It will help eliminate human subjectivity and ensure a consistent and accurate diagnosis. Computerized systems are developed using deep learning principles that enable them to examine X-ray images more reliably, thus removing decision-making troubles for radiologists. Through standardizing diagnostic processes, these systems can also contribute towards timely treatment, improving patient results in the long run.

Deep learning has moved forward with unstoppable flourishment through technological advancements in modern times. Deep learning technology has found its way into all aspects of our lives, be it advanced medical diagnostics or image classification, promising new accurate solutions and innovations that can address the world's current challenges. Several deep learning models help with the process, which includes Convolutional Neural Networks (CNN). CNNs are usually used for tasks like image classification and recognition consisting of multiple layers including convolution and pooling layers in the middle. These layers in the middle facilitate important and complex features to be extracted from an input image for further analysis. At the end of CNNs, a Fully-Connected Neural Network (FCNN) lies, which encourages the classification of that input image on the basis of the extracted features. One of the primary advantages of employing CNNs is the ability to learn and identify patterns in an unseen image on their own and classify it correctly. This is possible due to iterative training to ensure more accuracy and precision with lesser loss.

Introducing a deep neural network model that leverages CNNs presents a promising approach to addressing the challenges and limitations associated with Pneumonia detection among patients. Such models could offer enhanced accuracy in identifying Pneumonia-affected areas within chest X-rays. For this problem, the deep neural network model could predict whether the X-ray image lies under one of the two categories: ‘Normal’ or ‘Pneumonia’.

As the depth of any neural network architecture increases, so does the model's accuracy. This becomes possible as more layers aid in more relevant feature extraction, increasing image classification accuracy. This matter is crucial for medical diagnosis as it could lead to a strong standpoint from which certain decisions could be made accurately, without compromising anyone’s health. We propose using a deeper neural network pre-trained VGG16 model containing a relatively deep architecture. We also applied the transfer learning method at the end.

Therefore, a neural network architecture that is deep enough will be more discerning for complex patterns, leading to the model having higher metrics levels. This is particularly vital in medical diagnosis where accurate diagnosis should be made not to compromise anybody’s life. Hence, we advocated for the adoption of VGG16 network architecture with transfer learning.

This novel technique can change the entire field of medicine thus bringing about invaluable advantages regarding individual health and well-being. Also, it is adaptable enough that doctors and other healthcare professionals can easily examine patient reports especially when dealing with CXR images to diagnose Pneumonia. This novel method could enhance healthcare facilities in the future.

II. Related Works

From little children, and young youths to older people, all of them are dying because of one of the most common cardiovascular diseases - Pneumonia. Consequently, many researchers & scientists have been constantly working on detecting this fatal disease by using deep learning techniques, especially through Convolutional Neural Networks (CNNs). A group of authors proposed a way of classifying Pneumonia X-ray images drawn from two public datasets comprising 100,000 images into two categories: ‘Normal’ and ‘Pneumonia’. To train the model using the deep transfer learning technique, the authors used three CNN models namely DenseNet-121, GoogLeNet, and ResNet-18. Furthermore, they have calculated the weights using four different evaluation techniques - F1-Score, AUC on Tanh (hyperbolic tangent) activation function. As a result, all these methods helped to get an accuracy rate of 98.81% and 86.85% for each dataset [4].

Similarly, another paper demonstrated Pneumonia classification with chest X-ray images on three pre-trained CNN models - MobileNetV2, Vision Transformer, and DenseNet169. Moreover, they have also used real-world datasets. Using fewer layers and features such as transition layers, batch normalization, ReLU activation, and transformer layers has made the model more effective than the regular existing model for Pneumonia detection [8]. In another research paper, Pneumonia was detected using a publicly available Kaggle dataset of 5300 chest X-ray images which follows the classification of Pneumonia as an abnormal case and a Normal case on the other side. Researchers used ResNet50, InceptionResNetV2, and InceptionV3 as their pre-trained models. To get efficient results and higher accuracy, they used two different optimizers - Adam and Stochastic Gradient Descent (SGD) alongside various transfer learning architectures. Ultimately, they have achieved 93.06% accuracy on RestNet50 [9].

In another study, Pneumonia detection was performed using the VGG16 model with a neural network providing an accuracy rate of 92.15% [10]. In addition, another group of authors detected Pneumonia from chest X-ray images by evaluating fifteen different CNN architectures trained on the same dataset. They have simply selected the best architectures among all, considering minimum loss and maximum accuracy [11].

Furthermore, another study on the same was carried out in the field of deep learning by another author group. Five pre-trained models—AlexNet, DenseNet121, ResNet18, InceptionV3, and GoogLeNet—were set up and assessed. ResNet18 performed better than the other two alone, with a 94.23% test accuracy. To increase accuracy, they gathered the predictions from all five models by choosing the class that was predicted the most often. This combined technique in medical image processing that includes several models was preferred due to its good precision as testified by its accuracy rate of 96.39%. Hence, compared with unimodal techniques, these heterogeneous techniques might lead to more reliable results in studies of a similar nature [12].

One more study was conducted using a dataset that consisted of 5856 frontal chest X-ray pictures (4273 Pneumonia cases and 1583 Normal cases), which aimed at Pneumonia identification. They took advantage of Xception and VGG16 architectures which are famous convolutional neural networks for this purpose. Additional fully connected layers were added to the Xception model after pre-training on ImageNet. Similarly, the Xception model itself was pre-trained using the ImageNet dataset before altering its architecture to accommodate the new fully connected layers. Furthermore, more layers together with pre-trained weights were appended onto the VGG16 model. Both models used techniques such as batch normalization, and data augmentation to avoid overfitting.

Consequently, among all other methods of diagnosis, Xception has been found to work best with Pneumonia cases with a precision of 91% and an accuracy of 87%. Besides that, Xception demonstrated good results for Pneumonia recall at 94% and normal accuracy at 86%. Every network has advantages. Hence, VGG16 was found to identify common cases properly, whereas Xception was found to be better at identifying Pneumonia [13].

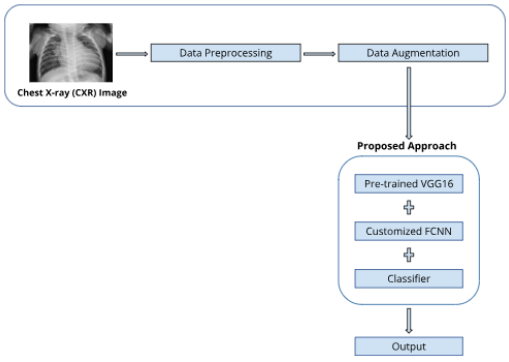
As we can see, many researchers have done many studies in deep learning to get better classification results on detecting Pneumonia over time. Also, it is known that deep learning is very suitable for large datasets and less complicated than image processing methods. Based on the insights of this past research, we delved into using an intricate model (VGG16), in which hyperparameter tuning and other necessary strategies were implemented to make our project a success. From the notion that increased complexity in an architecture would increase a model's accuracy by learning important features, we decided to stick to this approach for getting precise classification results.

We hereby discuss the procedure of conducting this project methodology through different deep-learning-based techniques. Besides that, we demonstrated the results found in our model training process.

III. Proposed Approach

For this research, we build our proposed approach by keeping the pre-trained VGG16 model as our base model and using transfer learning by appending a customized Fully Connected Neural Network (FCNN) at the end of the VGG16 model. Our proposed approach includes- importing libraries, preparing the dataset, setting up data loaders with ImageDataGenerators, defining and freezing the VGG16 base model, setting custom FCNN layer, compiling model, configuring callbacks (learning rate, early stopping, model checkpoint), training with callbacks, evaluating, sample prediction, displaying predictions, plot accuracy, predict single image. The Fig. 1 below illustrates our proposed approach in a flow diagram for convenience:

Fig. 1: Outline of our Proposed Approach



Then, the images from the Kaggle dataset were used to cross-validate our neural network. Through the combination of a pre-trained VGG16 model and customized FCNN layers, we trained our overall model through the following steps:

A. Architecture Details

1) Base Model: For our project, we implemented one of the popular and dense CNN (Convolutional Neural Network) architectures- VGG16. This model was developed by the Visual Geometry Group (VGG). VGG16's architecture allows it to extract highly valid features from an input image and further put it for classification tasks. Along with that, we used the technique of transfer learning by adding custom FCNN layers at the end. In this project, we have shown a detailed analysis of every part of the VGG16 model that we applied.

B. Convolutional Layers

Multiple layers of convolutions are used in VGG16 to obtain the essential characteristics of the input images. Along with that, our customized FCNN layers also remain. The overview of the layers and their corresponding functionalities are described below in detail:

1) Initial Convolutional Layers: Our base model (VGG16) has originally 16 layers, which consists of 13 convolutional layers and 3 fully connected layers. 3x3 filters are applied at each convolution layer for capturing intricate details from the input image. Along with that, max-pooling has been applied following each convolutional layer, helping to reduce the dimensions of the activation maps by retaining special features. This pooling function is also essential in overcoming overfitting.

The VGG16 model also comprises 3 fully connected layers at the final stage. This part plays a crucial role in deciding the final classification process based on the vital features the model has extracted till then. The output/last layer of this FCNN contains the number of possible classes for classification. Here, the neurons correspond to each class, whereas the output is the probability of the input image belonging to the best class.

By using a total of 14780481 trainable parameters, the final output shape calculated from the base model was found to be 7x7x512.

2) Global Average Pooling: The global average pooling reduces the overall dimension, but it does this work globally. It helps provide a global summary of the important features extracted by the CNN layers. This becomes crucial for our project as it encourages our model to retain the most significant characteristics throughout the entire chest X-ray image. Besides that, it also has a role in the reduction of overfitting.

This operation provides a vector output of shape 512, as there consists of 512 channels in its input.

3) Custom Fully Connected Layers: The input layer size (512) found after the global average pooling gets flattened to be fed into the FCNN. Here, our proposed model has used 1 fully connected hidden layer. The layer consists of 128 neurons connected with 512 neurons of the previous layer (input layer), having 128 x 512 weighted connections. This full connectivity allows the FCNN model to transform the format of the input data through the weights associated with each connection and the biases associated with each layer. The hidden layer undergoes nonlinear activation functions after the matrix multiplications among the parameters, typically ReLU. From this layer, we get output that becomes the input for the upcoming layers.

4) Output Layer: The final (output) layer consists of only one neuron interconnecting with 128 neurons of the previous dense layer. Binary classification was applied here using the Sigmoid activation function to detect whether or not someone has Pneumonia.

C. Activation Functions

Our research includes non-linear activation functions applied at every Convolution Layer and at the end of the FCNN layers throughout the VGG16 model (base model). ReLU (Rectified Linear Unit) activation functions were used in the convolutional and hidden layers of customized FCNN to introduce non-linearity by learning complex patterns. On the contrary, at the output layers of FCNNs, The Sigmoid function was utilized to make probabilistic predictions for tasks like binary classification. In this layer, the neuron will generate an output in terms of probability for the input image belonging to any of the classes. Such logistic regression function helps to limit the value found from linear function between 0 and 1, representing probabilities. The Sigmoid function is liable for classifying an input image as either 'Normal' or 'Pneumonia'.

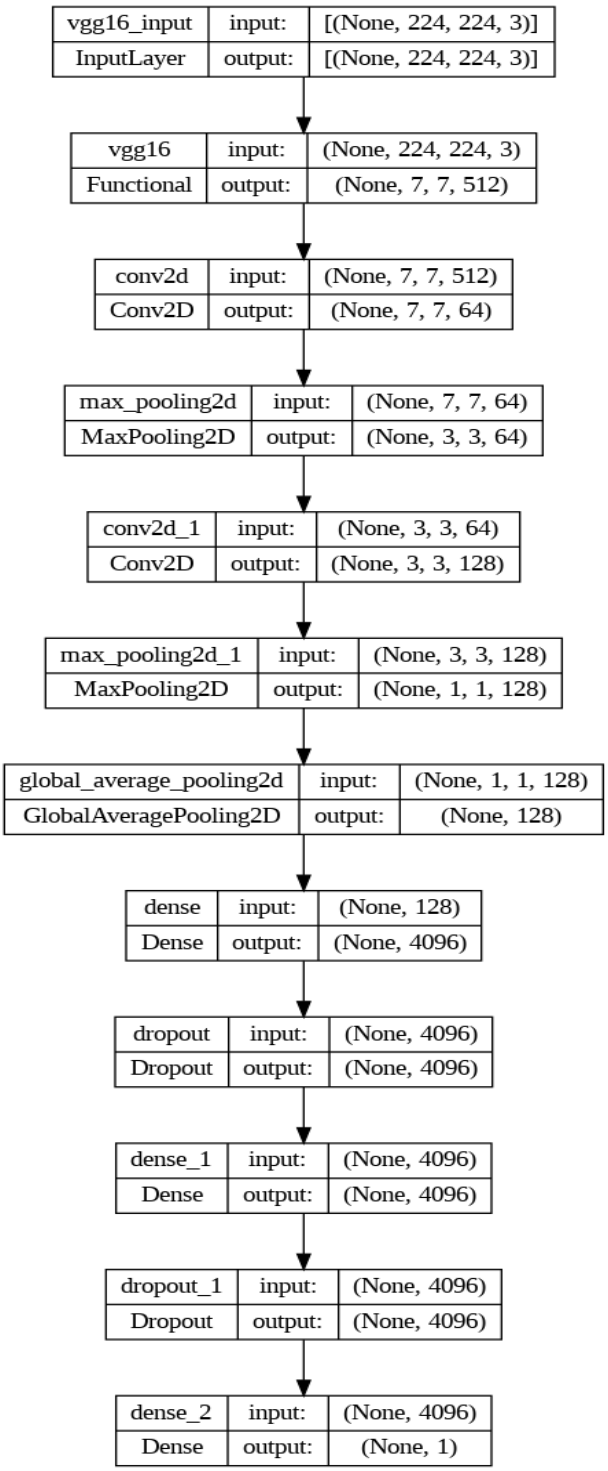
D. Loss Function

A loss function is an important part of the training process that works to maximize the model's performance by adjusting its parameters. The loss function helps the model decide how to reduce this gap by adjusting the weights. This is typically done by measuring the difference between the expected and actual true results. Several such loss functions consist of categorical cross-entropy, hinge loss, mean squared error (MSE), and others. However, these are not appropriate for our own particular goal. Such as Mean Squared Error (MSE) is not much suited for tasks like binary classification. Because this can't figure out the probabilities as binary cross-entropy does. On a side note, the loss function (categorical cross-entropy) was not required for our project. This is because our project involved binary classification, but not classification with multiple classes.

For the betterment of our model, we applied the binary cross-entropy loss function to our binary classification problem. Binary cross entropy is particularly good for a binary classification problem since it measures how well an algorithm performs against ground truth labels (0 and 1) where predicted variances are distributed over two classes. We want these losses minimized so that our models will be more efficient. By using binary cross-entropy we sort on the uncertainty characteristic of binary classification which gives much accurate and reliable predictions.

Here is an expansive overview of our proposed architecture for Convolutional Neural Network (CNN), providing details of the changes in design made to fit its performance and enhance its efficiency (Fig. 2):

Fig. 2: Structure of our Neural Network Model



Our proposed model had 17682625 total trainable parameters which used a memory of 67.45 MB. Whereas 14714688 total non-trainable parameters used a memory of 56.13 MB.

E. Hyperparameter Tuning Types

In hyperparameter tuning, we find the best or optimal set of hyperparameters to get the best performance for a machine/deep learning model. Tuning hyperparameters is all about getting the best combination to optimize a machine/deep learning model. Tuning these parameters is crucial when creating successful machine/deep learning models because architecture and training processes are critical. This study employed various techniques and strategies to optimize the hyperparameters of our convolutional neural network (CNN) model based on the VGG16 architecture to classify chest X-ray images.

Parameters are the variables that impact a model's performance during training. The most common ones are weights and biases, which have been used in the convolutional and dense layers of our proposed model. Nevertheless, hyperparameters are other variables outside of the weights and biases that govern the training process and model architecture. Some examples of it that are used in our model are- learning rate, batch size, and Adam optimizer (a gradient descent method).

By this means, we discussed below how the hyperparameters were tuned in our proposed model:

1) Learning Rate: We know model weight is a big factor as it is related to the loss gradient and the learning rate, a type of hyperparameter is an important thing that mainly controls how much is needed to change this model's weight. So for our model, we took the value of the learning rate at 0.0001. And to adjust this we used a learning rate scheduler. After training, we saw that for the first five epochs, the learning rate was constant. But after that, it gradually decreased. To improve convergence and stability, this hyperparameter is very important as it assists in fine-tuning the training model. Due to this learning rate, we can get more clear and accurate updates while training continues.

2) Batch Size: Batch size denotes the number of samples per epoch used during a model's training phase. For our model, the batch size was set to 32. This ensured our model made the training process efficient and stable. Furthermore, an optimal batch size guarantees more manageability and computational efficiency throughout the betterment of our model's performance.

3) Number of Epochs and Early Stopping: For training our model, the number of epochs was set to 30 at most. However, we implemented a technique called 'Early Stopping' which monitored the validation loss and stopped further epochs if the model didn't show any development in its performance. After that, the weights from the best model were then restored. It was helpful in the context of avoiding unnecessary memorization of the data by our model.

4) Optimizer Selection: There are several other forms of gradient descent algorithms applied in machine/deep learning models to optimize the parameters necessary for training. One of the optimizers is the Adam optimizer, which we implemented in our project. By using the Adam optimization technique, we have satisfied the main goal of the research- to fine-tune our model to get the highest accuracy and precision. while also resulting in improving training loss. Besides enhancing accuracy, the Adam optimization also ensured us with computational efficiency. Moreover, Adam worked well with hyperparameter tuning, helping our model by adjusting the learning rate. As we also trained our model with many features of Pneumonia, choosing Adam as the best optimizer has helped us with proper & accurate detections.

It is essential to implement hyperparameter tuning for developing an effective deep learning model for any real-life applications. Through tuning the important hyperparameters like learning rate, batch size, epoch number, optimizer, etc., we check for our model's performance to generalize well on unknown images. Such an approach ensured proper metrics values for our classification task.

IV. Experiments and Results

We used the Google Colab environment for our model to be trained and tested, utilizing both CPU (Central Processing Unit) and T4 GPU (Graphics Processing Unit) hardware accelerators. Although there was limited access to T4 GPU for free, it played a crucial role in the time-efficient computation of the epochs. Because of the limited accessibility of T4 GPU hardware, we used the CPU most of the time despite its time-consuming abilities.

Our approach involved the TensorFlow library tied with the Keras Application Programming Interface (API) to implement and develop our proposed model, thereby enabling effortless testing while using less resourceful configurations.

A. Defining the Data

1) Dataset: For our project, we made use of a publicly available Kaggle dataset which consists of chest X-ray images in JPEG file format. These images were originally collected from the Guangzhou Women and Children's Medical Center from the pediatric patients. The dataset is categorized into three sections for both categories. For our model training, we used a validation set on our own with the help of splitting the training set into training (70% of the original training set) and validation sets (30% of the original training set). After splitting: for the 'Normal' case, there is a training set containing 939 images, a test set containing 234 images, and a validation set with 402 images. For the 'Pneumonia' case, there is a training set containing 2713 images, a test set containing 390 images, and a validation set with 1162 images. The resolution of the chest X-ray images was 224×224 after rescaling/normalizing the original images.

For this dataset, the input dimension was 224×224×3 for 224×224 images with 3 channels corresponding to RGB. Its data type is 'uint8', as the pixel components in an image have values ranging from 0 to 255. The images are classified into two classes: Normal and Pneumonia. Keeping that in mind, the output format (output values) is kept as 0 (Normal) or 1 (Pneumonia), which are integer data types.

TABLE I: Distribution of Dataset

	Train	**Validation**	**Test**
Normal	939	402	234
Pneumonia	2713	1162	390

The Fig. 3 below provides a visual representation of a selection of chest X-ray images, illustrating both Normal and Pneumonia cases. The key difference among the X-ray images of both classes lies in the presence of white spots within the lungs of patients in the positive cases (Pneumonia). Whereas, these white spots are not evident in the

X-ray images of the negative (normal) cases. Our desired dataset became useful for both training and testing in our model, intending to detect Pneumonia by using CXR.

Fig. 3: Subsets from the dataset, each from *(a) Normal Cases* and *(b) Pneumonia Cases*

(a)

(b)

a) Data Preprocessing

Data preprocessing involves the normalization of the original image's pixel values within the range of 0 and 1 for an easier computational process. In this case, the pixel values are divided by 255 (1/255), the maximum pixel value. This process ensures that the data is more suitable for the deep learning neural network models to process efficiently.

b) Data Augmentation

The data augmentation method is applied to make our model learn about distinct angles, specific sizes, one-of-a-kind orientations, and so forth of the existing images in our dataset. To make our model adapt to large data, it artificially increases the training set by developing modified copies of an existing dataset, assisting in diversifying examples.

The data augmentation method is applied to make our model learn about distinct angles, specific sizes, one-of-a-kind orientations, and so forth of the existing images in our dataset. To make our model adapt to large data, it artificially increases the training set by developing modified copies of an existing dataset, assisting in diversifying examples.

Different methods of data augmentation were employed in our research. The image is rotated by 30° by the operation of rotation (rotation_range). By using horizontal and vertical image shifting operations (width_shift_range and height_shift_range), we moved our dataset images to left/right and up/down with a value of 20% of the total width and height. Also, using the shear transformation operation (shear_range) changed the shape of the image with the shear intensity of 0.2. On the other hand, the zooming operation (zoom_range) zooms in or out over the image by up to 20%. Besides that, flipping (horizontal_flip) is another option for mirroring an image horizontally and thereby creating a reflected version of images, making it possible for the model to generalize on unseen data.

TABLE II: Settings of Data Preprocessing and Augmentation on Images

Operations	Values
Rescale	1/255
Rotation	30°
Width Shift	0.2
Height Shift	0.2
Shear	0.2
Zooming	0.2
Horizontal Flip	True
Validation Split	0.3

B. Defining the Metrics

In this paper, we used some key metrics to evaluate the effectiveness of our deep learning model which are accuracy, loss, precision, recall, and F1 Score. All these metrics have their unique role in improving the model's performance. Below we have discussed these metrics and written the value of these metrics:

1) Accuracy: The metrics that calculate the ratio of correctly classified classes from all or total classes is called accuracy. Accuracy is also not always true if we don't have balanced datasets; though, it is a simple metric we used to get high performance. After 30 epochs, we got a test accuracy of 0.9022436141967773, which is around 90.22%. The formula to find accuracy is:

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

Where,

TP or True Positives = the number of correctly predicted positive samples.

TN or True Negatives = the number of correctly predicted negative samples.

FP or False Positives = the number of incorrect positive predictions.

FN or False Negatives = the number of incorrect negative predictions.

For our model, the values are:

TABLE III: Confusion Matrix of our Model

	Actual = Yes	**Actual = No**
Predicted = Yes	TP = 272	FP = 171
Predicted = No	FN = 118	TN = 63

2) Loss: Another important hyperparameter is the loss function which calculates the difference between the actuarial result (ground truth) and the predicted result. For our model, we got a test loss of 0.3518877327442169 or 35.19%. Our project is mainly a binary classification problem, And for this

type of case, we mainly utilize the Binary Cross-Entropy which is also known as Log Loss. This loss function is important during training mainly as it deals with wrong prediction.

The loss function is mainly calculated through his equation:

$$\text{Loss} = \mathcal{L}(y^i, \hat{y}^i) = -[y^i \log(y^i) + (1-y^i) \log(1-\hat{y}^i)]$$

Where,

i = sample number

$\hat{y}$ = predicted probability

y = true label (0 or 1)

To estimate the performance of the model, these true labels: 0 and 1 are very important as they differentiate its prediction against the actual truth class.

3) Precision: Precision is a hypermater-like accuracy but the difference is precision mainly sees the accuracy of the positive prediction by the model. Precision can be measured through this formula:

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

Precision is particularly useful when the cost of FPs is high. It indicates the proportion of correct identifications.

Test Precision is 0.8713318109512329 or 87.13% for our model. The precision is mainly necessary for the case where the false positive is high which mainly shows that it can't identify the positive cases.

4) Recall: The hyperparameter is significant to know how much positive prediction the model made which may include false positives also. High recall is the main goal though besides recall other hyperparameter values also should be looked at. The value of Recall for our model is 69.74% The formula to find Recall is:

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

5) F1 Score: Among all the evaluation methods, F1 Score is the most important one to consider. The higher the F1-Score is, the better the model predicts. For our model, we got an F1 Score: of 77.47%. F1 Score usually calculates the average between the recall and precision. The formula to find the F1-Score is:

$$\text{F1 Score} = \frac{2PR}{(P+R)}$$

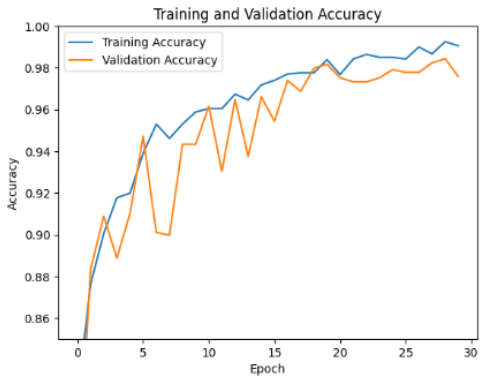
C. Present Results

- *Test Accuracy:* 90.22%
- *Test Loss:* 35.19%
- *Test Precision:* 87.13%

We will now see the performance of our trained model based on test, training, and validation datasets. The experiments included training the model with different metrics and evaluating its performance on the validation set. Here are the main outputs:

1) Training and Validation Accuracy: If we see the Fig. 4 graph of training and validation accuracies, we can see the plot indicates the training and validation accuracy over 30 epochs. The graph shows that both training and validation accuracy improves as the number of epochs increases. The training accuracy is higher than the validation accuracy which shows some possibility of overfitting. The training accuracy continuously improved and in the end, reached above 98%. On the other hand, the validation accuracy increases but it fluctuates so much. The test accuracy we got was 90.22% which represents that the model has good performance on new or unseen data.

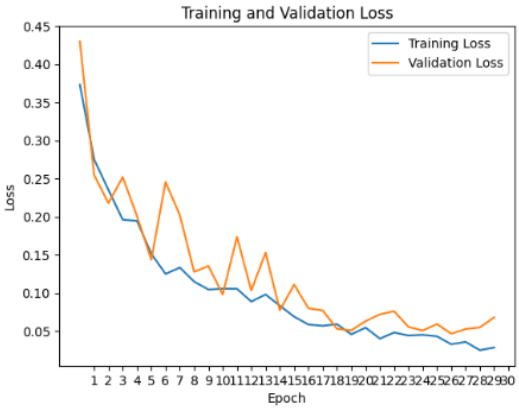
Fig. 4: Training and Validation Accuracy Graph



2) Training and Validation Loss: The second graph in Fig. 5 is about training and validation losses. This graph shows in the same way how the training and validation loss increased or decreased. In the graph, we can see that throughout the whole 30 epochs, there is a decline in both the validation and training loss which means our model has good learning and optimization. Although there are some fluctuations in validation loss, in the end the validation loss decreases severely which is a good factor for the model to do well in prediction. Also, we used early stopping so that our validation loss does not increase that much and cause overfitting to our model.

The test loss of 35.19% means that there is some error rate when our model makes predictions on the test data. This percentage mainly determines the difference between the predicted output and the actual output.

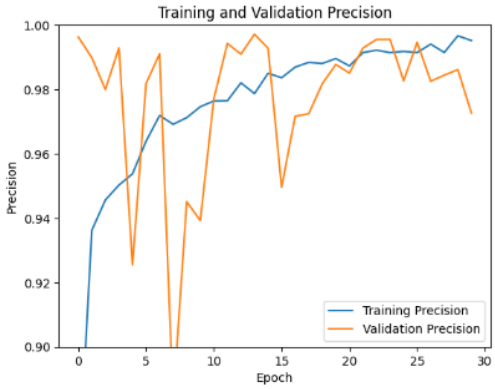
Fig. 5: Training and Validation Loss Graph



3) Training and Validation Precision: During calculation accuracy and loss, the precision was also measured during training. The third graph in the Fig. 6 is about Training and validation precisions. which shows the training and validation precision across the 30 epochs. Similar to accuracy, the precision for both training and validation data improves continuously but validation precision shows more fluctuation if we compare this with training precision. Both of them mainly reached around 98%.

The test precision of 87.13% means our model is quite good as it correctly identifies most of the positive classes among the positive predictions while the false positive rate is low. Our output for precision shows that our model will show good performance for new data, too.

Fig. 6: Training and Validation Precision Graph



The TABLE IV below presents the values of the metrics after some of the epochs:

TABLE IV: Metrics of our trained model

Epochs	Accuracy		Loss		Precision	
	Train	Validation	Train	Validation	Train	Validation
After 1 Epoch	86.60%	90.56%	0.2880	0.2064	0.9074	0.9128
After 6 Epochs	96.16%	96.61%	0.0999	0.0909	0.9754	0.9823
After 11 Epochs	96.41%	95.77%	0.1001	0.1094	0.9790	0.9945
After 16 Epochs	97.43%	97.14%	0.0672	0.0669	0.9861	0.9708
After 20 Epochs	98.45%	95.96%	0.0445	0.1077	0.9900	0.9954
After 25 Epochs	98.73%	97.92%	0.0333	0.0508	0.9926	0.9973
After 30 Epochs	99.01%	97.79%	0.0259	0.0603	0.9937	0.9955

D. Sample Predictions

To evaluate the model's performance, we also have one section where we gave a few sample predictions to the test sets. We showed the ground truth and predicted for each sample, and we can see that the model successfully can classify the cases of normal chest X-ray and also Pneumonia affected cases with high-level accuracy. Thus, our experiment represents that our trained model performed well on the given dataset.

TABLE V: Final Model Evaluation

	Accuracy	Loss	Precision
Training	99.06%	0.0285	99.52%
Validation	97.59%	0.0680	97.27%

In a nutshell, we presented our model's prediction result (TABLE V). We can see that the model did well in high accuracy and precision on both the training and validation sets. Furthermore, the training loss is low which is quite impressive as it shows that the model has minimized its error during training. Also with a test accuracy of 90.22%, we can say that our model is fit for unseen data. Though the test loss is high which is 35.19% which means in the future we can apply other necessary metrics and other techniques for the

improvement of the model. Again the precision maintains the balance as test precision is 87.13% which highlights that the model can correctly identify most of the positive samples. In summary, our model has a strong high accuracy and precision on the test set and also high predictive capability. However further work can be done to decrease the loss to get overall optimized performance.

V. Conclusion

Here, we aim to build a reliable and efficient Pneumonia classification system using a publicly available dataset from Kaggle. The primary goal of this project was to apply the pre-trained VGG16 model for chest X-ray image detection of Pneumonia, along with the transfer learning process. We used many techniques like data preprocessing as well as data augmentation to improve our model's performance. These included rotating, zooming, flipping horizontally, and shifting images. The augmentations were aimed at boosting the generalization ability of the model. The data has been preprocessed by normalizing the pixel values appropriately for use with VGG16 to ensure uniformity and better quality regarding input data through which diseases can be detected more accurately.

For 30 epochs, our experimental model underwent rigorous training and there was a continual improvement in the performance metrics. On the other hand, the VGG16 model has a deep structure and features pre-trained on ImageNet, enabling robust learning of Pneumonia identification aspects. From our model, we got a training accuracy of 99.06% and a training precision of 99.52%. On top of that, we attained a test accuracy of 90.22% and a test precision of 87.13% which implied that our model was computationally efficient for such a medical diagnosis. During our experiment, a smaller difference between the training and validation was found, indicating a good generalization tendency towards unseen data. With the advanced characteristics of our proposed model, this approach would enable doctors to examine chest X-rays more effectively. Additionally, this approach ensures that patients are given the right kind of treatment for their X-ray findings because it provides an accurate and clear diagnosis. Further studies should focus on how well the pre-trained VGG16 model with proper upgrades can help in diagnosing Pneumonia or any disease using a deep neural network approach developed by us here. Furthermore, for future work, more different types of optimizers along with any other pre-trained model could be utilized to further enhance the issue of overfitting through cross-validation. Thus, regular monitoring of our model's performance is a must, especially for using it on real-life applications like medical diagnosis.

Acknowledgment

We would like to express our gratitude to Paul Mooney, the author behind the publicly accessible dataset from Kaggle that we have used for our study. This dataset was useful for our model's training and testing.

Author Contributions

The authors wish to express their gratitude for the support that has made this work possible through a collaborative effort. The authors have made significantly equal contributions in the researching, writing, and analyzing process for the project.

References

[1] World Health Organization. (n.d.). *Pneumonia.*

[2] *Pneumonia.* (2019). Our World in Data.

[3] American Lung Association. (n.d.). *Five Facts You Should Know About Pneumonia.*

[4] Kundu, R., Das, R., Geem, Z. W., Han, G., & Sarkar, R. (2021). Pneumonia detection in chest X-ray images using an ensemble of deep learning models. *PLOS One, 16(9).*

[5] Pearce, M. S., Salotti, J. A., Little, M. P., McHugh, K., Lee, C., Kim, K. P., Howe, N. L., Ronckers, C. M., Rajaraman, P., Craft, A. W., Parker, L., & Berrington de González, A. (2012). Radiation exposure from CT scans in childhood and subsequent risk of leukaemia and brain tumours: a retrospective cohort study. *The Lancet, 380(9840),* 499–505.

[6] Neuman, M. I., Lee, E. Y., Bixby, S., Diperna, S., Hellinger, J., Markowitz, R., Servaes, S., Monuteaux, M. C., & Shah, S. S. (2012). Variability in the interpretation of chest radiographs for the diagnosis of pneumonia in children. *Journal of hospital medicine, 7(4),* 294–298.

[7] Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Ball, R. L., Langloztz, C., Shpanskaya, K., Lungren, M. P., & Ng, A. Y. (2017). CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning. *arXiv preprint arXiv:1711.05225*.

[8] Mabrouk, A., Redondo, R. P. D., Dahou, A., Elaziz, M. A., & Kayed, M. (2022). Pneumonia Detection on Chest X-ray Images Using Ensemble of Deep Convolutional Neural Networks. *Applied Sciences, 12(13),* 6448.

[9] Manickam, A., Jiang, J., Zhou, Y., Sagar, A., Soundrapandiyan, R., & Dinesh Jackson Samuel, R. (2021). Automated pneumonia detection on chest X-ray images: A deep learning approach with different optimizers and transfer learning architectures. *Measurement, 184, 109953.*

[10] Sharma, S., & Guleria, K. (2023). A Deep Learning based model for the Detection of Pneumonia from Chest X-Ray Images using VGG-16 and Neural Networks. *Procedia Computer Science, 218,* 357-366.

[11] GM, H., Gourisaria, M. K., Rautaray, S. S., & Pandey, M. (2021). Pneumonia detection using CNN through chest X-ray. *Journal of Engineering Science and Technology, 16(1),* 861-876.

[12] Chouhan, V., Singh, S. K., Khamparia, A., Gupta, D., Tiwari, P., Moreira, C., Damaševičius, R., & de Albuquerque, V. H. C. (2020). A Novel Transfer Learning Based Approach for Pneumonia Detection in Chest X-ray Images. *Applied Sciences, 10(2), 559.*

[13] Ayan, E., & Ünver, H. M. (2019). Diagnosis of Pneumonia from Chest X-Ray Images Using Deep Learning. *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT).*

[14] Chest X-Ray images (Kaggle). https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

GITHUB LINK: https://github.com/adrikachowdhury/DL_finalproject/blob/main/PneumoniaDetection.ipynb