Using Convolutional Neural Networks to Detect Pneumonia

Aarav Awasthi

Received April 22, 2023
Accepted July 12, 2024
Electronic access July 31, 2024

This paper presents a study on the effectiveness of a neural network in detecting pneumonia; more specifically, Convolutional Neural Networks are becoming more and more precise at analyzing patterns in images. This experiment utilizes a convolutional Neural Network to detect traces and types of pneumonia in chest x-ray images. The accuracy and validation accuracy data were analyzed when the network was tested and it was observed that the neural network was highly effective in detecting pneumonia. The consistency in the accuracies observed between epochs 15 and 30 is particularly noteworthy, as it suggests the model's reliability over time. The findings of this study have the potential to advance the development of more precise diagnostic tools for pneumonia detection, ultimately leading to improved patient outcomes. In accordance, hopefully scientists, doctors, and engineers will use this method of detecting pneumonia in patients. In future use, this study could be expanded on to include more lung disease detection.

Keywords: Pneumonia, CNN, Detection, Chest X-Ray

Introduction

Pneumonia, a global health concern, poses challenges to diagnose despite advances in medical imaging technology. Pneumonia is a growing concern in our society. If not treated promptly, it can lead to severe and sometimes irreversible health complications, even causing death. Millions of people worldwide succumbed to this disease in 2018[1]. This project aims to improve pneumonia diagnosis by developing a deep learning-based model that can detect subtle patterns in chest x-ray images indicative of the disease[1]. The proposed approach involves building a neural network using machine learning techniques, enabling a computer to learn from data and experience similarly to humans. Specifically, the model will utilize a convolutional neural network (CNN), a neural network designed for image processing due to its ability to recognize image patterns[2]. Given the complexity of chest x-ray images and the variability in pneumonia presentations, a CNN-based classification system is expected to provide the most accurate and robust disease detection[3]. By leveraging machine learning and neural network technology, this project has the potential to significantly improve pneumonia diagnosis and reduce the burden of this disease on global health.

Literature Review

As stated before, detecting and treating Pneumonia is a challenging task that requires an efficient, and accurate way of addressing. Therefore, finding an effective way to identify pneumonia in chest X-rays is crucial. According to Puneet Gupta, a potential solution is to build a convolutional neural network to tackle this issue. He puts forth that pre - trained CNN's (which they are often called), have the capacity to classify large sets of image data[1]. This project will utilize Convolutional neural networks, which can recognize complex patterns during image analysis[2]. The convolutional neural network will be able to detect the specific areas of abnormalities in the lungs and analyze which lungs are normal and healthy and which are abnormal and infected. Now, when a lung is found to be affected by Pneumonia, it can be either viral pneumonia or bacterial pneumonia[3]. Viral pneumonia is the most dangerous and the hardest one to diagnose; it requires physical examination, chest x-rays (hard to spot), and study of symptoms. Bacterial Pneumonia is usually caused by the bacterium Streptococcus while Viral Pneumonia is caused by the Syncytial virus[4]. In Gupta's approach, he utilized a VGG-16, VGG-19, and his own model that he built from scratch to test which model is the most accurate in detecting pneumonia. He found that the VGG-16 model had the most accurate validation and test data. As the names imply, VGG-16 is a model with 16 layers while VGG-19 is a model with 19 layers. He classifies the data by determining whether the x-ray is of Pneumonia or not. He also claims that this model can be built upon in the future to also be able to detect other lung-related illnesses such as Covid-19[1]. On the other hand, Lingzhi Kong and Jinyong Cheng utilized a different method of image classification called transfer learning which can reuse knowledge from previous models into other areas of a similar background. He also utilizes an Xception algorithm with a long-short-term memory algorithm (LSTM). This helps in reducing hyperparameter errors or vanishing gradients while also allowing for more accurate visualizations[5]. Overall, these two methods show

how CNN's can be used in different ways to ultimately solve the same problem in detecting Pneumonia. Gupta's method can be used for deep learning techniques and in other lung-related illnesses while Kong's work can be used when there are a lot of gradients and hyperparameters needed to test. Both of these studies have been peer-reviewed and verified to show that the methods work.

The way a convolutional neural network is structured is that it has 7 layers; the first layer is the input layer. This is where an input image is processed through the network. It holds the pixel values[2], which are combined to form 3 pieces of data: width, height, and channels. The number of input channels depends on whether the image is grayscale (1) or RGB (3). In our case, chest X-rays are classified as grayscale images. So, this means they only need one channel to go through. The next layer is the actual convolution layer, it is the core building block of the entire Convolutional network. How this layer works is that it applies a certain size kernel and slides it across input data to extract certain features of the image[2]. Each of the filters in this layer extracts a different type of data feature. Next, for the third layer, comes the activation layer, which introduces non-linearity among the data. Common types include ReLU and Sigmoid. After that, comes the pooling layer, which helps mediate overfitting by reducing the dimensions of the data. Some common types of pooling include Max Pooling and Average Pooling which take the max and average value respectively from the feature. You can then add any additional layers like a separable convolution or a batch normalization, these helps normalize and feature your data to specific locations. You can also add a fully connected layer, which connects all the previous neurons to the neurons in the next layer. However, a fully connected layer comes with the advantage of being able to learn all the information at once from the previous layers. The final layer is called the output layer, and as the name implies, it is where the final output is produced. The number of neurons in this layer usually depends on how many classes/categories of data have been passed through the input layer[2].

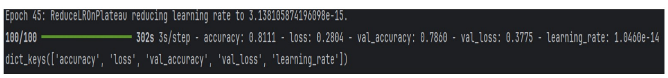
Methodology

This project utilized many libraries such as Numpy for data processing and analysis, Matplotlib for data visualization through graphs, TensorFlow for a full machine learning library, and Keras to get optimizers, pre-built layers, and loss functions. The chosen dataset is publicly available on Kaggle: "pneumonia_dataset". There are 3 classes for classifying the chest x-ray images: Normal, Bacterial Pneumonia, and Viral Pneumonia. The structure of the neural network is to first pass in these three classes as inputs. This is done by accessing the file pathways for where these images are held and setting up a loop to loop through the main file which then accesses the Normal, Bacterial, and Viral Pneumonia files individually and then displays them. The array is then 'flattened' to reduce a 3D array to a 1D array to make it easier for the network to analyze. Next, the data generators are prepared, the dataset is preloaded, and it is ready for training. The method for this is called datagen. Additionally, the test dataset is preloaded and pre-processed. This part will return the training image generator, test image generator, the test dataset, and the labels to that data. Finally, the number of epochs for how long the network will run is set to 30. To implement the project, resize all images to 150 x 150 and set the batch size to 50 for processing per epoch. The resize will make it easier to pool and analyze the images pixel by pixel, while setting the batch size to 50 will let the network process an optimum amount of samples before updating itself. After that, the actual neural network has to be set up. The project utilizes a neural network that has 7 layers, out of which 5 are convolutional layers, and the remaining 2 are output and dense functions. Among the 5 convolutional layers, only one is normal, while the others are separable. Separable layers break down a normal convolutional layer into two simple steps and can run without much computational cost. Each convolution layer has a pooling layer built into it, and 3 of them have batch normalization. In the last 2 layers, dense functions are used to combine all the data from the previous layers, and then the output layer finalizes the data into the three classes. The ReLU and Sigmoid activation functions are utilized in this neural network. The neural network utilized the ReLU and Sigmoid activation functions. The ReLU activation function was used with the convolutional and dense layers, while the Sigmoid function was called upon in the output layer. The data optimizer, Adam, was imported from the Keras libraries, which helps in reducing the loss by adjusting the weights of the network. Also, a code block was included to save the most valuable weights in the network and to monitor the loss. When the loss gets too high, the early_stop function in the code block is triggered.

To train the neural network, the model.fit.generator function must be called. This function utilizes the previously defined data generators and runs them for the epoch amount. After each epoch, the early_stop and lr_reduce functions are employed to check the size of the loss. When the loss exceeds a certain threshold, these functions cease running the network and make appropriate adjustments. This process is time-intensive due to the substantial amount of strain and effort required by the neural network to process the large dataset for 45 epochs. After completion, the following code block imports a library from sklearn.metrics for calculating the accuracy of the neural network and evaluating its performance. Running the test data prompts the neural network to predict the images, which are stored in a variable. Based on the value of this variable, the accuracy function assesses the neural network's accuracy on the testing data. Finally, the accuracy on both the training and testing datasets can be visualized using the Matplotlib library and several functions to create a line graph for the accuracy. In summary, the code imports, processes, trains, and tests data and the neural network to optimize accuracy in classifying Pneumonia types.

Results

The convolutional neural network was tested by importing the dataset and creating separate folders for training, testing, and validation. The data was then processed, and the number of epochs was set to 45. The class weight was set to 1, 1, and 0.5, with the third class having the highest number of images. The batch size was set to 50. The model was defined using hist and generator functions, and these variables were passed as arguments. Initially, the code was run without class weights, which resulted in class imbalance. The class_weight function was used to resolve this issue. The final validation accuracy was approximately 0.78, or 78 percent, when the code was executed.


```
Epoch 45: ReduceLROnPlateau reducing learning rate to 3.138105874196098e-15.
100/100 ━━━━━━━━━━━━━━━━━━━━ 302s 3s/step - accuracy: 0.8111 - loss: 0.2804 - val_accuracy: 0.7860 - val_loss: 0.3775 - learning_rate: 1.0460e-14
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss', 'learning_rate'])
```

We can see here that the accuracy on the test data was 81 percent, which is higher than the validation accuracy, which was approximately 78%. This is actually reasonable because a neural network of this size and with this amount of data should not easily be getting 95 - 99% accuracy. That indicates that something is wrong with class imbalance, where it only focuses on one set of data heavily and it gets a 'false' accuracy on it[2].

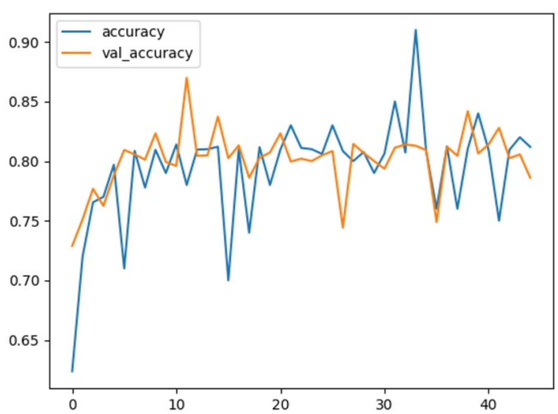


Fig. 1 Accuracies vs Epochs over time (Graph 1)

The graph above shows how our accuracy and validation accuracy changed per epoch. Both of them follow a log graph pattern. It goes up a bit, but then becomes less and less steep. Between epochs 15 and 30, the accuracies were near a steady 80% (except for the spikes up and down). This steadiness throughout the neural network is important for it to be able to function efficiently and accurately. Upon analyzing the accuracy and validation accuracy data presented in this study, it can be inferred that the neural network utilized was highly effective in detecting pneumonia. The consistency in accuracies observed between epochs 15 and 30 is particularly noteworthy, as it suggests the model's reliability over time. These findings have the potential to advance the development of more precise diagnostic tools for pneumonia detection, ultimately leading to improved patient outcomes. However, there are still some limitations to this approach: since the neural network has only be shown images from a specific dataset with labeled data, it might have trouble identifying Pneumonia if it is present in a different area in the lungs or even on a different dataset.

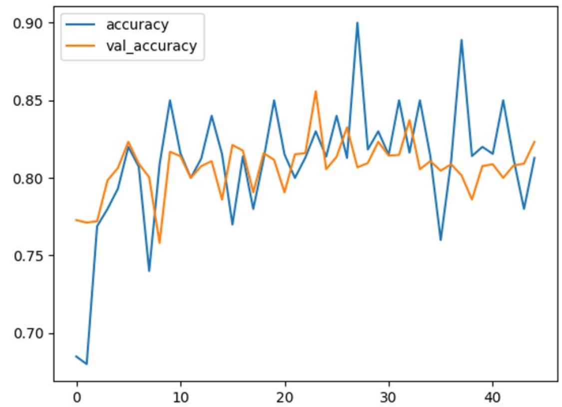


Fig. 2 Accuracies vs Epochs over time (Graph 2)

Conclusion

To conclude, this study has demonstrated the effectiveness of using a convolutional neural network for detecting pneumonia in chest x-ray images. The model achieved 81% accuracy and 78% validation accuracy rates, which indicates its high reliability in detecting pneumonia. The consistency in the accuracies observed between epochs 15 and 30 suggests that the model is stable over time. With the growing concern of pneumonia as a global health issue, the development of more precise diagnostic tools is crucial. The proposed deep learning-based model has the potential to significantly improve pneumonia diagnosis and reduce the burden of this disease on global health. Nevertheless, the limitation outlined previously could easily be solved; future work could focus on expanding the dataset, exploring different network architectures, and investigating the model's performance on diverse populations. It could also be used in more lung diseases. Overall, this research provides a promising direction for the development of automated pneumonia detection systems using deep learning techniques.

References

1 P. Gupta, *International Journal for Modern Trends in Science and Technology*, **7**, 77–80.

2 R. Nash, *An Introduction to Convolutional Neural Networks*, `https://doi.org/10.48550/arXiv.1511.08458`, ArXiv.

3 T. Rajasenbagam, *Journal of Ambient Intelligence and Humanized Computing*.

4 J. Mendoza, *Computational Intelligence*, **36**, 370–401.

5 L. Kong and J. Cheng, *PloS One*, **16**, 0258804.