DETECTING COVID-19 USING CHEST X-RAY IMAGES

Muhammad Osama Khan
mok232@nyu.edu

Muhammad Muneeb Afzal
mma525@nyu.edu

Sean Robert Hornbuckle
sh4986@nyu.edu

1 INTRODUCTION

Over the past six months the entire world has been experiencing a pandemic due to the spread of the SARS-CoV-2 virus and the subsequent illness it causes (CoVID-19). While initially the potential for the virus to become a global pandemic seemed small to some health officials, we have seen an explosion in cases in the United States over the past two months.

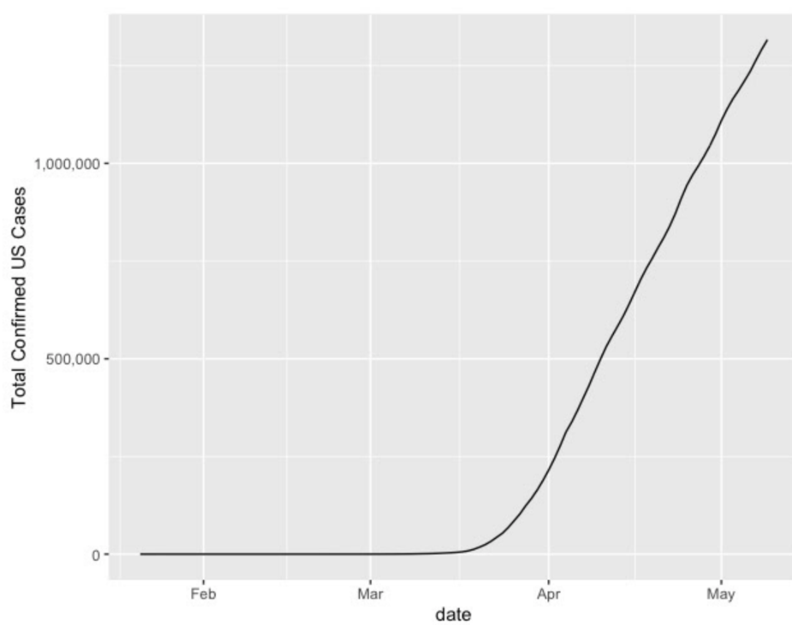

Figure 1: Confirmed COVID-19 cases in the United States against time.

In addition to the high and increasing number of cases, we are also seeing a fairly high death rate. The graph below shows the number of confirmed CoVID deaths we have experienced in the US over the past month and half alone. In mid-march we were under 100 deaths and at the time of this writing, less than 2 months later, we are looking at nearly 80,000 confirmed deaths. To add to this, we can see from the graph below that the total number of deaths has yet to level off.

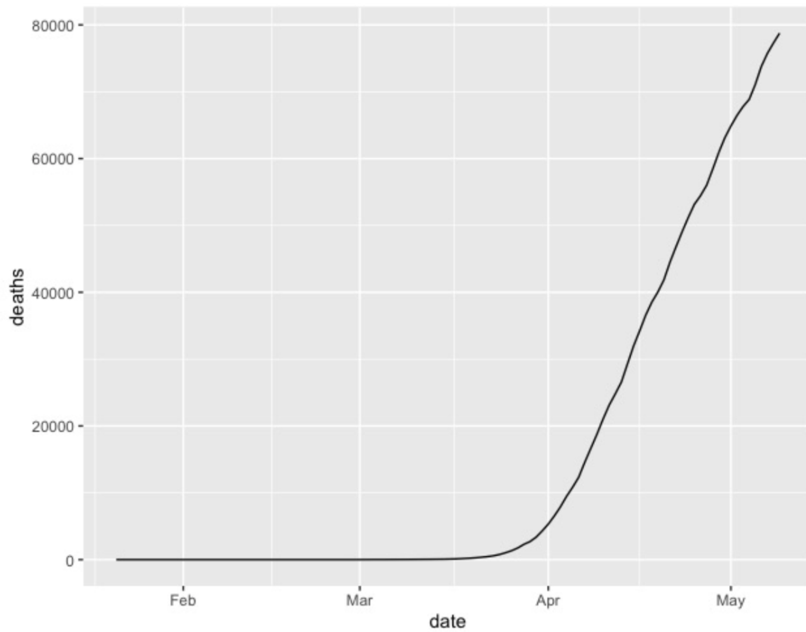

Figure 2: Confirmed COVID-19 related deaths in the United States against time.

2 COVID-19 and Pneumonia

With so many deaths happening over such a short amount of time, any insight we can gather on how to treat the virus more effectively could potentially lead to a vast amount of lives saved. Although this virus can be deadly to all it infects, those with preexisting lung conditions are of particularly high risk. As the disease progresses, one of the more severe infections that can arise is pneumonia.

Since pneumonia is an ailment whose effects we have seen multiplied by COVID-19, we are going to perform analysis on the lungs of healthy patients and those with pneumonia in order to train and develop a ML model that is able to detect the presence of pneumonia from chest X-rays.

2.1 Why is it useful?

Any additional information that can be provided that can aid in the detection of pneumonia could lead to faster treatment times and ultimately improved patient recovery. Additionally, it will allow physicians to make informed decisions about limited resources (e.g. beds, ventilators, CT-scans etc.) and hence, can serve as a digital second opinion or pre-screen, allowing the physicians to act with more confidence or prioritize treatment, while they wait for the analysis of a radiologist.

3 Methodology and Results

In this section, we outline our approach for detecting pneumonia from chest X-ray images.

3.1 Dataset

To train our models, we used a Chest X-ray Dataset from Kaggle. The dataset contains examples of chest X-rays of healthy patients and those suffering from COVID-19 related pneumonia. The train and test sets are comprised of 5286 and 624 images respectively.

3.2 Implementation Overview

We devised 2 strategies to detect pneumonia from chest X-ray images. Firstly, we used machine learning algorithms. For this part, we needed to first go through a data pre-processing procedure to clean the data and transform it into a form that we could use. We then applied various ML packages and techniques in order to allow the software to identify the pneumonia patients from the X-rays. Once the models were trained using this input data, we were ready to read-in images produced by an X-ray machine and provide the technician with a prediction. Secondly, we used deep learning techniques to identify pneumonia from chest x-ray images. In this section, we either created our own architecture or used transfer learning.

3.3 ML Based Methods

For ML implementation, the images was converted to vectors. Our input is an image which is an array of size $w \times h \times 3$. Note that since we are using RGB images, our last dimension is 3 . It is not trivial for a machine learning classifier to interpret an array of pixels and give them a semantic meaning. To this end, we need to process the image so as to extract important parts of the image in the form of a vector that can help our machine learning classifier model to distinguish between an infected and normal person.

3.3.1 Feature Descriptors

We extracted different features of the images before applying our MLmodels. The general pipeline for feature extraction is shown in Figure 3.

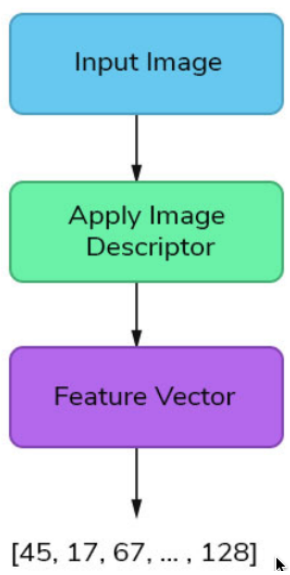

Figure 3: Shows the process of feature extraction of the input image.

The output snippet from our code (Figure 4) below shows that using PySpark, in parallel, we converted all the images into features descriptors. Note that without the parallel computing performed in PySpark, this task would take longer to complete.

image_path	new_label	descriptc
'Users/mma525/Doc...	0	[9, 221, 17, 251,.
'Users/mma525/Doc...	0	[129, 58, 159, 24.
'Users/mma525/Doc...	0	[249, 189, 17, 62.
'Users/mma525/Doc...	0	[5, 121, 121, 101.
'Users/mma525/Doc...	0	[100, 37, 61, 101.
'Users/mma525/Doc...	0	[232, 109, 63, 10.
'Users/mma525/Doc...	0	[93, 114, 152, 10.
'Users/mma525/Doc...	0	[204, 180, 19, 24.
'Users/mma525/Doc...	0	[32, 237, 25, 249.
'Users/mma525/Doc...	0	[36, 48, 17, 97, .
'Users/mma525/Doc...	0	[56, 72, 20, 104,.
'Users/mma525/Doc...	0	[36, 100, 185, 69.
'Users/mma525/Doc...	0	[29, 205, 26, 239.
'Users/mma525/Doc...	0	[24, 140, 27, 254.
'Users/mma525/Doc...	0	[116, 104, 61, 12.
'Users/mma525/Doc...	0	[252, 72, 214, 18.
'Users/mma525/Doc...	0	[120, 37, 101, 10.
'Users/mma525/Doc...	0	[45, 117, 24, 235.
'Users/mma525/Doc...	0	[182, 99, 225, 10.
'Users/mma525/Doc...	0	[124, 180, 152, 2.

Figure 4: Shows the output snippet of extracting image features in parallel

The particular feature extractors we used are described below.

Flattened Image

Firstly, we tried the approach of flattening the image. However, before flattening, the image was resize to $50 \times 50 \times 3$ which gives us a vector of length 7500 as our feature. Note that if we did not resize the image, we will get a very large vector from which our classifier model will not be able to learn properly. Moreover, it will be very computationally expensive to process such a large vector.

ORB (Oriented FAST and rotated BRIEF)

ORB is a robust local feature detector that is used in computer vision tasks. Local feature extractors, like ORB, determine interest points in the image and consider the regions surrounding those interest points for analysis. For ORB, we obtain key points and a descriptor for each image. The keys points for an example image are shown in Figure 5. Using OpenCV, we applied an ORB algorithm with a number of features equal to 50. It yielded a feature descriptor of size 1600. This extracted feature served as the input to our ML model later.

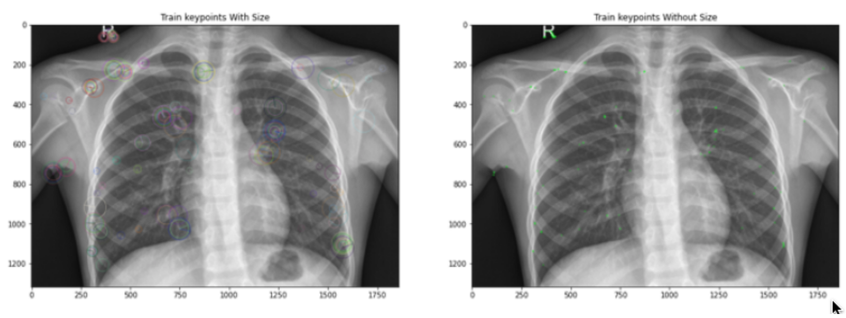

Figure 5: Illustrates the ORB algorithms at work, detecting key points in the images

Haralick Texture, Hu Moments and Color Histogram

The three feature extraction algorithms: Haralick Texture, Hu Moments and Color Histogram, are global feature extractors. Instead of finding interest points, they reason about the image in a more holistic manner.

Haralick Texture algorithm learns about the texture information in the image by quantifying the spatial variation in the grey level values. It must be noted that before applying this algorithm, the image is first converted to grayscale. Hu Moments algorithm uses image moments, which are the weighted average of the intensity of the pixels in the image to understand the information related

Table 1: Shows different feature descriptors and their performance (for both metrics higher is better)

Feature Descriptor	Flattened Image	ORB	Haralick + Hu Moments + Color Histogram
Train AUC	0.999	0.999	0.921
Train Accuracy	0.998	0.999	0.747
Validation AUC	0.981	0.72	0.918
Validation Accuracy	0.966	0.651	0.735
Test AUC	0.85	0.674	0.861
Test Accuracy	0.734	0.639	0.625

to shape and orientation in the image. Color Histogram feature extractor describes the distribution of colors in the image. Each of these feature extractors yields a vector. These vectors are then concatenated to obtain our final global feature extractor.

3.3.2 Results

The three features were then fed to our Logistic Regression model which finally classified the image (whether they belong to an infected patient or not). The results are shown in Table 1 and Figure 6. The two metrics used are Area Under ROC (AUC) and accuracy. Both metrics are computed out of 1 and the higher score is better.

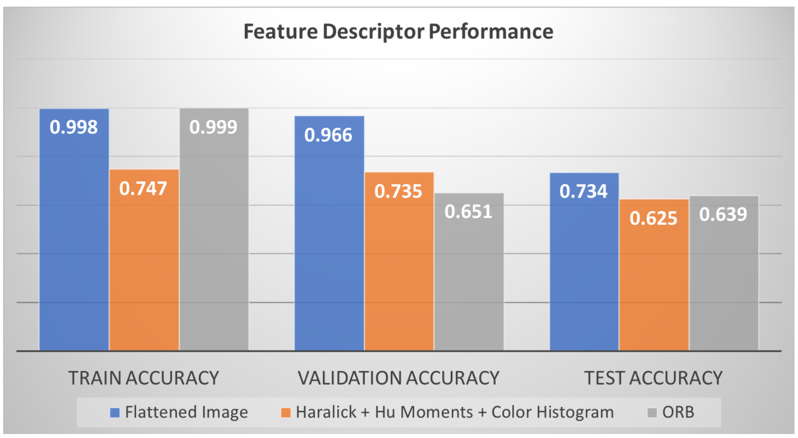

Figure 6: Feature Descriptor Accuracy

3.3.3 Analyses

The results show that for the test set, we obtain the highest AUC for the combined feature: *Haralick, Hu moments and Color Histogram* and the highest accuracy for flattened image. Note that AUC is a metric that evaluates the overall performance of a binary classifier by using ROC curve and takes into account all classification thresholds. Therefore, it is a more robust way to evaluate the quality of a binary classifier.

As expected, we get the highest value of AUC for Haralick, Hu moments and Color Histogram since the combination of these three features extract global information about the images rather than relying on local features such as interesting points. Surprisingly, we got high accuracy for flattened images. This may be because we did not throw away information about the image and let the model learn itself. However, working with flattened images was very computationally expensive and thus took a lot of time to train and test. Thus, it is not scalable in contrast to other methods. ORB extractor did not produce great results since it is a local feature detector that focuses mainly on parts of an image near the keypoints. Hence, it was able to learn the holistic features about the images that could help us classify the infected and normal patients.

3.4 DL Based Methods

In addition to machine learning, deep learning models were also implemented to screen patient X-rays for COVID-19 pneumonia. With regards to deep learning, we both designed our own model and implemented a transfer learning approach, which drew from pretrained models - trained on large datasets such as ImageNet. Designing our own model allowed us to freely choose the neural network architecture. However, it also had the disadvantage that it was more computationally expensive to train from scratch and produced significantly worse results since it is difficult to train effectively on small datasets. In contrast, the transfer learning approach had the advantage that it was computationally cheaper (since it did not involve training from scratch) and produced better results.

Asides from the 2-layer neural network approach, the other 5 techniques we used produced accuracy results in the mid to high 70% range. Overall, the Xception based technique produced the best results, with a 78.8% accurate prediction (see Table 2 and Figure 7).

Table 2: Comparison of custom model vs transfer learning.

Learning Type	Model	Accuracy (%)
Custom Model	2 Layer NN	39.1
Transfer Learning	InceptionV3	78.2
Transfer Learning	Xception	78.8
Transfer Learning	ResNet50	75.4
Transfer Learning	VGG16	74.8
Transfer Learning	VGG19	76.1

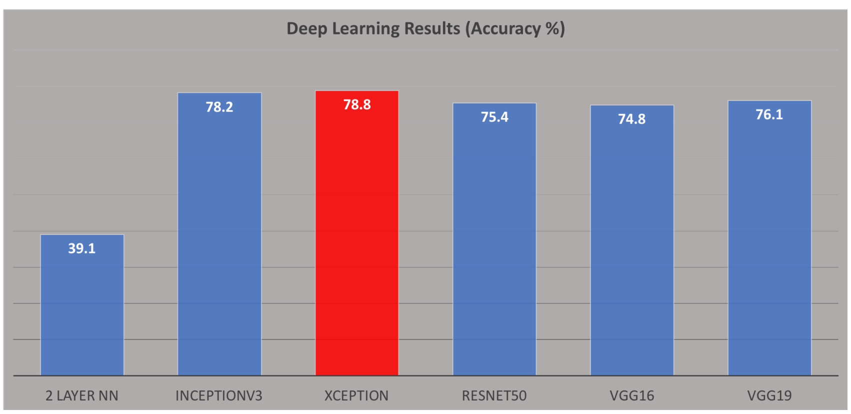

Figure 7: Deep learning models results.

3.4.1 Analyses

When analyzing the results from the deep learning approach, although the overall accuracy was 78.8%, the false positives outnumbered the false negatives by a nearly 2:1 ratio (see Table 3). Due to the application of our software, false positives are more tolerable than false negatives. It is less of an issue to have a healthy patient flagged than to miss a sick patient.

Table 3: Best deep learning model results. Results are reported on the test set.

Model	Accuracy (%)	False Positives	False Negatives
Xception	78.8	81	43

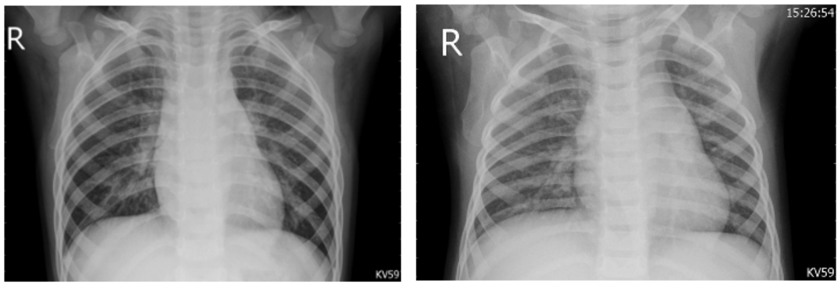

Figure 8: Visualization of false positives.

The figure above shows examples of X-ray images that were incorrectly classified as positives (having pneumonia) by our model. Since pneumonia usually causes the air sacs in the lungs to expand, one might conjecture that this is a result of the cloudy regions (which could be image artifacts) seen in the rib cage area in these x-rays. However, chest x-ray analysis is a specialized task performed by radiologists and hence, it's difficult to draw strong conclusions about the performance of the network and it should only be used for pre-screening.

3.4.2 Deployment

Once the models are trained, the deployment of our software by clients is fairly straightforward. Trained models are deployed as Spark SQL UDFs. This means that users don't need to have any ML or high-level coding expertise and can directly interact with the model using SQL queries. This will allow technicians to use the software, instead of radiologists, which will keep billing rates low and preserve client resources.

4 Improvements

Although we were able to produce acceptable results with our models, there are future improvements that can be made to further increase accuracy. Perhaps the most straightforward way to improve our models is to remove bad images from our dataset. Upon manual examination, the X-ray dataset we used contained some images that were not actually chest X-rays, but were used to train our models. Another simple, yet computationally expensive improvement, is to avoid resizing the images to 299 $\times$ 299 (this was done since the pretrained models expect this input size and to reduce computational cost). Our target clients will likely have the resources to purchase or rent the level of processing power needed to avoid resizing. One final recommended improvement is to use pre-trained models for grayscale images and pre-trained models for medical images since those models are likely to extract much more relevant features to the task at hand than a network trained on a generic image dataset such as ImageNet.

5 Technologies Used

The following technologies were used in the development of this project:

- Pyspark
  - Pyspark ML Features
  - Pyspark MLlib Clustering
  - Sparkdl Library
- Numpy
- Keras
- OpenCV
- Mahotas
- R and R-Studio
- Databricks + AWS Clusters

6 Conclusion

Implementing our software package into your hospital/clinic workflow will allow for instant pre-screening of CoVID-19/Pneumonia patients. Patients flagged by the pre-screening process can have their X-rays and other prognostic information evaluated at a higher priority, potentially saving both money and lives. By choosing to share your X-ray images along with the final patient diagnosis with us, you will help us improve upon our modeling, which will in turn produce more accurate field results.