Detection of Covid-19 using Convolutional Neural Networks and Chest X-Rays

Karim Jabbour, Martin Gallois, Siqi Zhang, Yichen Wu, Yinan Wang

***Keywords* -** COVID-19, CNN, Classification, Visualization, Machine Learning, Feature Map

Abstract. The coronavirus, also known as COVID-19, continues to infect more than 500,000 people daily even after the widespread vaccination against it. The common method used to detect COVID-19 is polymerase chain reaction (PCR) tests, however some facilities might not have enough equipment and trained professionals for the amount of tests needed. Therefore our team proposes to detect COVID-19 through chest X-rays (CXR), which can minimize transportation of patients with portable radiography units which are easy to disinfect among other benefits. Our project consisted of two parts. First, using a Convolutional Neural Networks (CNN), our project aims at determining if X-ray images represent healthy patients or COVID-19 patients and propose this innovative method of COVID-19 detection. Second, we tried to implement a Web-App for a user-friendly interface. By the end of this project, we were able to determine whether a patient has COVID-19 with an accuracy of 90% and have made a website for better user experience.

1. Introduction

COVID-19 is an infectious disease that causes patients to experience respiratory illnesses such as coughing or difficulties breathing. Being highly infectious as it is, COVID-19 has infected and continues to infect millions of people worldwide. The common method used to diagnose COVID-19 is PCR tests, which are invasive and might not be accessible at every facility. Under such circumstances, our team recognizes the value of diagnosing COVID-19 through CXR because COVID infection results in distinct radiographic visual characteristics in CXRs[1]. With a well trained model, diagnoses can be made in a shorter period of time with this less invasive method.

2. Methodology

The following methods are segmented into three key areas:

1. Acquiring of data and implementation of the Convolutional Neural Network

2. Design and building of the Application/Website to allow for easy user-experience.

2.1 Dataset

In order to build the model we needed two datasets: CXRs from healthy patients and CXRs from COVID-19 infected patients [2,3]. For pre-processing, images were filtered such that only the posteroanterior (PA) chest was used. 196 images were used from each data-set and each set was split with a ratio of 0.5 (training and validation) such that 4 directories were output as the table shown below (Table. 1)

	Training Sets		Validation Sets	
# of samples	COVID-19 -ve	COVID-19 +ve	COVID-19 -ve	COVID-19 +ve
	98	98	98	98

Table. 1 The Summary Table of the Split of Data

2.2 CNN

The CNN model [4] was implemented using the Keras library within Python.

The CNN consists of 4 convolutional layers with dropout layers preventing it from overfitting and ReLU as activation function after each convolutional layer to make the model learn faster.

Some note-worthy elements:

1. Binary Cross Entropy was used as the loss function since the CNN classified into two categories only

2. The Gradient descent optimizer used was 'Adam' (simply based on popularity; no other optimizers were tested).

2.3 Website

A website [5] (Fig. 1) is developed using python dash ploty [6] to provide a user-friendly interface to interact with the CNN model. The user would be able to upload the lung CXR image to the website and use the CNN model to identify if the lung is infected by COVID-19.

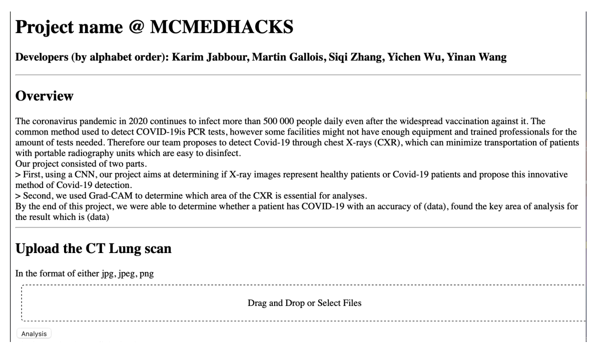


Fig. 1. The User Interface of the Website

2.4 Feature Map Visualization

To understand the function of each layer of the CNN, we generated a set of feature maps. Each feature map represents the result of applying a filter kernel to an input CXR image[7].

Specifically, we convoluted the input CXR image with each kernel from the CNN model and then generated the corresponding convoluted images.

In order to compare the difference between the COVID positive and negative cases, we also generated the convoluted images for both cases.

3. Result

Due to the nature of the neural networks and the randomization of images within our data in sampling, results may vary slightly within each run. Nevertheless, below is a solid set of results that demonstrate what one build results in.

3.1 Confusion Matrix

The matrix demonstrates in Fig. 2 that using a 196 set of images, we have 90 true positives, 96 true negatives, 8 false negatives, and 2 false positives. This result (Fig. 2) indicates little error in prediction using our datasets.

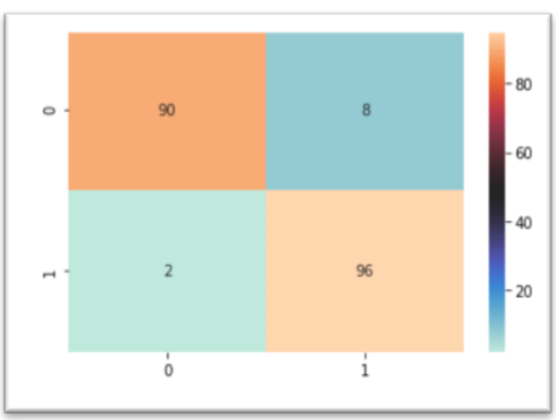


Fig. 2 Confusion Matrix where that 0 represents COVID-19 positive and 1 represents COVID-19 negative

3.2 Accuracy [8]

The below accuracy plot (Fig. 3) shows increasing accuracy as the model builds through the CNN layers with each epoch over time. Final accuracy is around 90%, which is great.

Both lines slow-down in growth towards the end which indicates that we are on the verge of “over-fitting”.

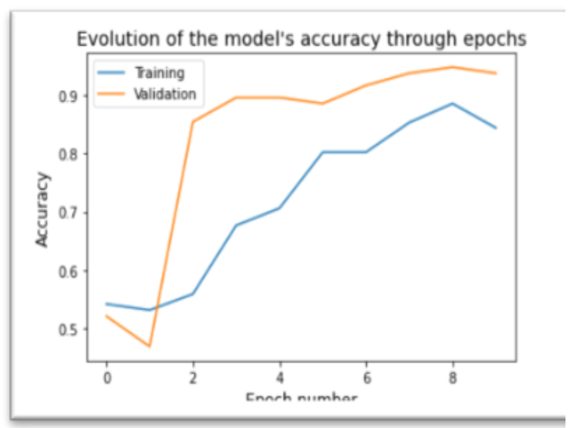


Figure 3. Accuracy for the training and validation sets for each epoch

3.3 Entropy Loss

The below loss plot (Fig. 4) shows decreasing loss with each epoch, which is expected. The lines do not separate which is a good sign that over-fitting is not taking place.

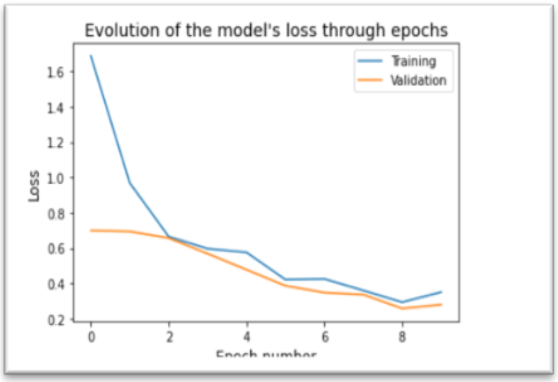


Figure 4. Entropy loss for the training and validation sets for each epoch

We will also leave the following summary table of our model (Fig. 5); however, we will not be delving into details with it due to time and resource constraints of the Hackathon.

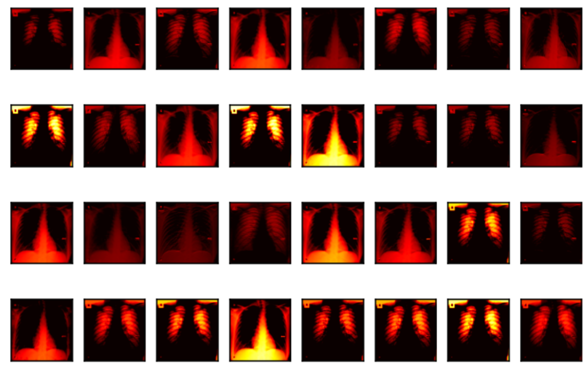
```
model.summary()

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 222, 222, 32)      896
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 220, 220, 64)      18496
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 110, 110, 64)      0
_________________________________________________________________
dropout (Dropout)            (None, 110, 110, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 108, 108, 64)      36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 54, 54, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 54, 54, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 52, 52, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 26, 26, 128)       0
_________________________________________________________________
dropout_2 (Dropout)          (None, 26, 26, 128)       0
_________________________________________________________________
flatten (Flatten)            (None, 86528)             0
_________________________________________________________________
dense (Dense)                (None, 64)                5537856
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 5,668,097
Trainable params: 5,668,097
Non-trainable params: 0
_________________________________________________________________
```

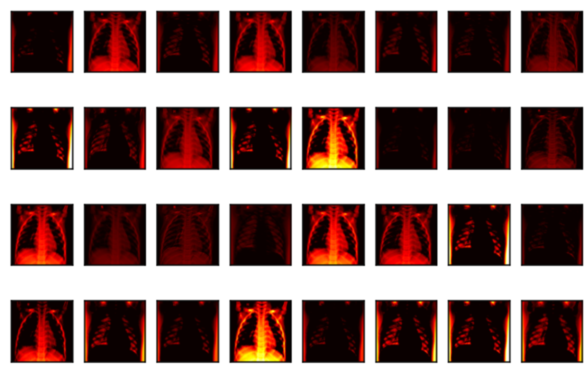
Figure 5. CNN Model Structure

3.4 Feature Map Visualization

The results for the feature map visualization for the first layer of the CNN network is shown below (Fig. 6).



(a) COVID-19 positive feature map



(b) COVID-19 negative feature map

Figure 6. Feature maps for COVID-19 detection

4. Conclusion

The hackathon project was able to deliver on its promises of building a CNN that distinguishes healthy from COVID-19 chest X-rays.

The website did serve as a visual guide as to what we were to implement provided the time and resources. With more time, it would have been fully functional.

The built neural network, while powerful, fails to distinguish between COVID-19 patients and patients with other forms of Lung infections and pneumonia.

Some future possibilities to improve upon the model include implementation of transfer learning along with larger datasets to possibly predict COVID-19 cases from other infections.

Reference

[1]https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7448820/

[2]https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia

[3]https://github.com/ieee8023/covid-chestxray-dataset.git

[4]https://www.youtube.com/watch?v=nHQDDAAzIsI

[5]https://caffeineoverflow-covid.herokuapp.com/?fbclid=IwAR0ktxanZL_H49WfDNGHi6y1RnqusZzZvuxV-tHPwNvASIJUa8hx8AOvwkE

[6]https://plotly.com/dash/

[7]https://machinelearningmastery.com/how-to-visualize-filters-and-feature-maps-in-convolutional-neural-networks/

[8]https://machinelearningmastery.com/display-deep-learning-model-training-history-in-keras/?fbclid=IwAR2EVApo5K_ALTOc3Hzuo8B2kvNU5iJ6tgLyhSBZ5BSCoVBpPpj-512eNOM