Covid-19 Identification from Chest X-Ray with a study on Google Search Correctness estimation

Sibashis Chatterjee
CCE AI & ML with Python
CCE, IISC Bangalore
West Bengal, Kolkata
sibashis1992@gmai.com

***Abstract*—The last couple of years one of the most important researches have been to be able to detect COVID-19 fast and accurately. This document proposes a way of detecting of COVID-19 using Machine Learning models run on Chest X-Ray. This model is expected to be able to distinguish between COVID-19 infected, healthy or other viral infected lungs from the X-Ray.**

Keywords—Radiology, X-Ray, COVID-19, detection, machine Learning, Scrapping, Tensorflow, CNN

I. Introduction

COVID-19 pandemic has affected the lives of almost all human beings living in the modern world. This is probably the most global event in the recent past that needed the best of us to respond in the most effective way. We hope to have gotten past the deadliest phase of the pandemic but at the cost of a very high number of lives. The infection changes its behavior too often to be able to be treated in standard operating procedure (SOP) [1]. At the onset of the pandemic it became the utmost priority to be able to detect the infection correctly and detect it fast [2]. There are specialized tests (e.g. Rapid Antigen Test [RAT] or Reverse transcription polymerase chain reaction Test [RT-PCR]) to detect the virus infection but at the peak of infection the supply chain for the test kits gets disrupted. At those times it can be of great help if we have a way to detect COVID-19 infections with the help of one of the most common and available medical equipment like a X-Ray machine [3]. In this analysis a Machine Learning model was created using Convolutional Neural Network (CNN) implemented in Tensorflow library.

II. Data Sources

A. Google Image Search

One of the most common places for data collection is Google. For collecting data of COVID-19 infected lungs X-Ray initially an attempt was made to look at Google Image Search. First glance using human eyes show that there are a lot of good quality images of infected lungs there. Though it's not always very clear whether the X-Ray is actually of infected lungs or not. Manual Google search would be a mammoth task hence A developer account was created at Google and their python SDK was used to interact with the Custom Search API [4] to query and get back programmatically readable search results.

B. Published Research Datasets

Many universities funded research on detecting COVID-19 and different mechanisms were proposed. Couple of studies from Qatar University [5] and University of Montreal were considered in this project [6,7]. These Data sets are selected because they have X-Ray images of Covid-19 infected, Normal and Viral Pneumonia infected lungs, this helps us make our model more enriched with different kinds of Infections so that specific types of COVID-19 infections can be predicted with more accuracy.

III. Data Collection

Data is collected through *Google Custom Search API* [4] and Python's out of the box *Request* API.

A. Prepare Python Development Ecosystem

The first step is to install the Python development ecosystem in the workstation. A laptop or desktop computer with Windows, Linux or Mac can be used. An M1 Mac 2021 was used for the project and analysis.

- Install Python (version >= 3.11)
- Upgrade PIP for easy package installation
- Install basic packages like numpy, pandas etc.
- Create a virtual environment if needed
- Install Jupyter notebook for a better presentation of the work

B. Google APIs

Google APIs are used for scrapping and downloading images from Google image search. The steps are as follows:

- Register with Google Developer Community [8] for generating API keys that are used by the Google SDKs. This is **secret** information and must not be published to code repositories. For this analysis an API key type credential named "IISC CCE AIML Course Project Google Search API Key" was created.

- Create a Programmable Search Engine [9] for making the API calls through Python SDK. For this analysis one was created with the name "IISC CCE Image Search". The search engine ID generated in this console also should be considered **secret** and must not be published to code repositories.

- Install the Google Search API Python SDK [10].
- Use the documentation of Custom Search API [11] to create a programmatic readable search object and iterate over the pages in search results to collect the desired number of image data against a query string. For this analysis this query was used “covid 19 infected lungs x ray”.

C. *Python Requests*

- Python comes with an out of the box library called **`requests`** that is used for downloading zipped archives of data from university published sources.
- Python out of the box **`zipfile`** utility was used to unzip the archives.
- Python utilities **`shutil`** and **`os`** package were used to move the data collected from different sources to proper folders to aggregate and correctly label test and train data sets.

IV. Data Cleaning

Data collected from university released sources were correctly sorted, filtered and labeled. This was very high quality data that did not require any cleaning. The data collected from Google Image Search was extremely noisy on the other hand. There were multiple stages of data cleanup required for these images.

A. *Corrupt Images*

The corrupt images cause trouble while the model attempts to predict. To avoid this we used the Python Image Library (PIL) to read all the downloaded images as valid image data There were around 23 images out of the 200 downloaded that were corrupted or of invalid MIME type that can’t be encoded into RGB number matrix [12]. These images were dropped from test cases.

B. *Improper X-Ray images*

There were some images with improper information that are not suited for the trained model. The model was trained using individual X-Ray images as shown in Fig. 1a whereas google downloaded photos had multiple X-Rays, labels and legends in those images as shown in Fig. 1b. Samples can be seen below:

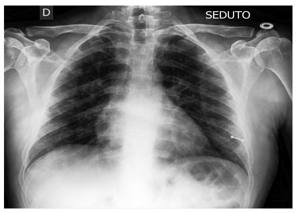


Fig. 1a: Valid Image with single X-Ray

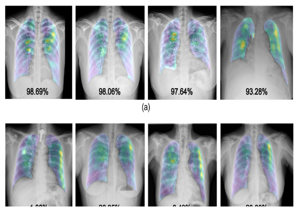


Fig. 1b: Invalid Image with multiple X-Rays

V. Model Training

1. ***Training Data Directory Structure:*** The training dataset from multiple sources were collated in a single directory structure as shown below:

```
/path/to/data
    | → train
    |       | → Covid
    |       |       | → 01.jpeg
    |       |       | → …
    |       | → Normal
    |       |       | → 01.jpeg
    |       |       | → …
    |       | → Viral Pneumonia
    |       |       | → 01.jpeg
    |       |       | → …
    | → test
    |       | → Covid
    |       |       | → 01.jpeg
    |       |       | → …
    |       | → Normal
    |       |       | → 01.jpeg
    |       |       | → …
    |       | → Viral Pneumonia
    |       |       | → 01.jpeg
    |       |       | → …
```

Fig. 2: Training Directory Structure

2. ***Image Data Preprocessing:*** Following preprocessing steps were applied while reading the training data.

A) Keras utility `ImageDataGenerator` was used for generating batches of tensor image data with real-time data augmentation. Images were read with a *1/255* reclaiming factor.

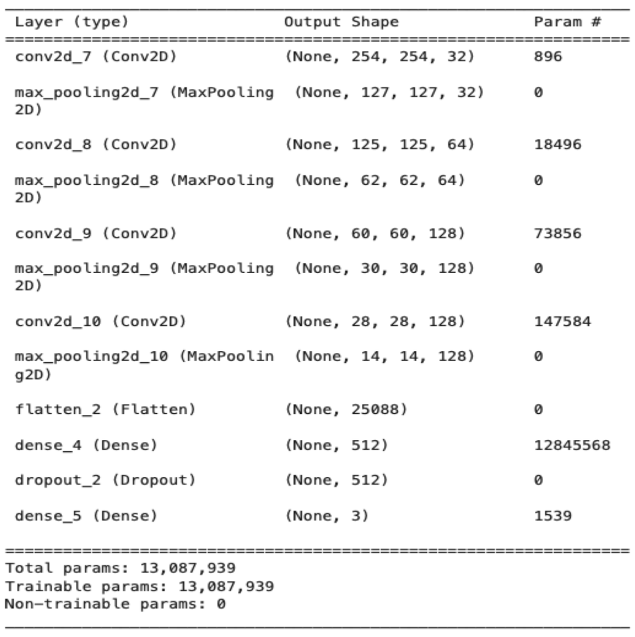
B) The `flow_from_directory` method in the above mentioned class expects the training data to be classified into labels. This is done by creating multiple sub-directories in the training data directory and each of these sub-directories are named by individual labels of the classifier as shown in Fig. 2. The images were converted to *256 X 256* dimension and read in batches of *32* images in categorical mode. *3979* images were found belonging to *3* classes for training and *66* images in *3* classes for cross validation.

3. ***Model Configuration:*** After multiple trial and errors the highest accuracy was reached with the following hyperparameters:

- A *sequential Keras* model with
  - *4* 2D convolution layers (*1* input and *3* hidden layers)
  - Layers connected with Max pooling operation for 2D spatial data.
  - Flatten the last layer
  - *2* densely connected Neural Network layers

    - All nodes with `ReLU` activation except the final layer
    - The final layer with `softmax` activation
- Compile the model with
    - `Adam` optimizer with `categorical_crossentropy` loss function and optimization on `accuracy`

4. ***Model Fitting:*** Model fitting was performed with the `train` directory as training data and `test` directory as the cross validation data. *10* epochs were used. Evaluation of the model was performed and *~86%* accuracy observed. Refer Fig. 3 for a summary of the neural network.



```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_7 (Conv2D)           (None, 254, 254, 32)      896

 max_pooling2d_7 (MaxPooling  (None, 127, 127, 32)     0
 2D)

 conv2d_8 (Conv2D)           (None, 125, 125, 64)      18496

 max_pooling2d_8 (MaxPooling  (None, 62, 62, 64)       0
 2D)

 conv2d_9 (Conv2D)           (None, 60, 60, 128)       73856

 max_pooling2d_9 (MaxPooling  (None, 30, 30, 128)      0
 2D)

 conv2d_10 (Conv2D)          (None, 28, 28, 128)       147584

 max_pooling2d_10 (MaxPoolin  (None, 14, 14, 128)      0
 g2D)

 flatten_2 (Flatten)         (None, 25088)             0

 dense_4 (Dense)             (None, 512)               12845568

 dropout_2 (Dropout)         (None, 512)               0

 dense_5 (Dense)             (None, 3)                 1539

=================================================================
Total params: 13,087,939
Trainable params: 13,087,939
Non-trainable params: 0
_________________________________________________________________
```

Fig. 3: Model Summary

VI. Analysis of Google Image Search Results

This model was used to do analysis of the cleaned Google Image Search results. The objective of this analysis was to investigate how the model performs on unseen data. The search images were preprocessed in a similar fashion as mentioned in section V(2). The model fitted in section V(4) was used to classify the search images Half of the images were labeled wrongly as Normal or Viral Pneumonia; these are considered negative test cases and the other half were labeled as Covid; these are considered positive test cases. The analysis of prediction yields the results tabulated in table I.

TABLE I. Google Image Classification

Test case type	Predictions			
	Total predictions	*Correct Predictions*	*Incorrect Infection*	*Incorrect Normal*
Positive	97	78 (80.41%)	5 (5.15%)	14 (14.43%)
Negative	80	64 (80%)	7 (8.75%)	9 (11.75%)

Assuming Google images are actually all COVID positive cases

For further analysis of the internal layers, we can see the features extracted in different layers. Some of the extracted features are visualized in Fig. 4.

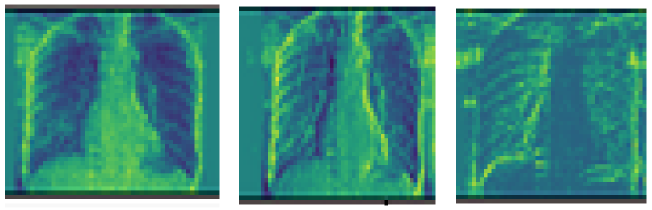


Fig. 4: Visualization of extracted features in different layers of model

Acknowledgment

We thank CCE faculty of IISC, Bangalore for arranging this CCE course on AI and ML with Python and giving us the opportunity of working on this model. We thank Google Inc. for releasing APIs and SDKs for interacting with their search programmatically and we thank Kaggle and research teams across universities for releasing and hosting data sets. Above all, we thank all the first responders and front line workers who helped us get through the COVID pandemic.

References

[1] Lopez-Leon, S., Wegman-Ostrosky, T., Perelman, C. *et al.* More than 50 long-term effects of COVID-19: a systematic review and meta-analysis. *Sci Rep* **11**, 16144 (2021). https://doi.org/10.1038/s41598-021-95565-8

[2] Ji T, Liu Z, Wang G, Guo X, Akbar Khan S, Lai C, Chen H, Huang S, Xia S, Chen B, Jia H, Chen Y, Zhou Q. Detection of COVID-19: A review of the current literature and future perspectives. Biosens Bioelectron. 2020 Oct 15;166:112455. doi: 10.1016/j.bios.2020.112455. Epub 2020 Jul 21. PMID: 32739797; PMCID: PMC7371595.

[3] Albahli S, Ayub N, Shiraz M. Coronavirus disease (COVID-19) detection using X-ray images and enhanced DenseNet. Appl Soft Comput. 2021 Oct;110:107645. doi: 10.1016/j.asoc.2021.107645. Epub 2021 Jun 25. PMID: 34191925; PMCID: PMC8225990.

[4] Custom Search JSON API Documentation, Google Developers' Community (https://developers.google.com/custom-search/v1/reference/rest/v1/cse/list)

[5] Anas M. Tahir, Muhammad E.H. Chowdhury, Amith Khandakar, Tawsifur Rahman, Yazan Qiblawey, Uzair Khurshid, Serkan Kiranyaz, Nabil Ibtehaz, M. Sohel Rahman, Somaya Al-Maadeed, Sakib Mahmud, Maymouna Ezeddin, Khaled Hameed, Tahir Hamid, COVID-19 infection localization and severity grading from chest X-ray images, Computers in Biology and Medicine, Volume 139, 2021, 105002, ISSN 0010-4825, https://doi.org/10.1016/j.compbiomed.2021.105002. (https://www.sciencedirect.com/science/article/pii/S0010482521007964)

[6] Chest X-Ray data from University of Qatar, COVID-QU-Ex Dataset (https://www.kaggle.com/datasets/anasmohammedtahir/covidqu)

[7] X-Ray images published in Kaggle courtesy to the University of Montreal (https://www.kaggle.com/datasets/pranavraikokte/covid19-image-dataset)

[8] Google Developer Community at Google Developer Community (https://console.cloud.google.com/apis/)

[9] Programmable Search Engine for programmatic searching (https://programmablesearchengine.google.com/controlpanel/all)

[10] Google Search API Python SDK for querying search API (https://pypi.org/project/google-api-python-client/)

[11] Documentation of Custom Search API with descriptions (https://developers.google.com/custom-search/v1/reference/rest/v1/cse/list)

[12] PIL Image converter to RGB matric (https://pillow.readthedocs.io/en/stable/reference/Image.html#PIL.Image.Image.convert)