Model for COVID-19 detection from chest X-Rays
Project Report

Team Members

Abolfazl Meyarian, Role: Implementation of Deep CNN (Convolutional Neural Networks) Models

Chandrakanth Mandalapu, Role: Data Preparation

Antonio Motta, Role: Data preparation

Varaha Narasimha Aditya Tatipaka, Role: Preparation of UI interface & Testing

Abstract

The goal of this project was to create an intelligent web-based software for detecting COVID-19 from X-Ray images of the chest powered by Deep Learning Models. This model is intended to be used by clinicians through an online interface to aid them in their diagnosis of COVID-19. Our model can identify critical regions in the lung in detection of the disease, which could be insightful to the clinician.

The website uses scripts which call our python-based model to make a decision regarding the condition of the patient. At the user's interface, first the user uploads an X-Ray image of the lung, then, the image is pre-processed and sent to our intelligent engine to decide regarding the patient's condition. The results are shown on the user panel on the website.

Data Specification

Mention supervised learning

The dataset used in this project is publicly available on Kaggle (link). It consists 2.11 Gb of data, with a total of 6939 chest X-ray posteroanterior images distributed in three classes of equal weight: 2313 of covid-19 positive patients, 2313 of patients with pneumonia, and 2313 of normal lungs. To test our model, 60 images from each of those three categories were reserved.

The raw data from this dataset was first pre-processed in order to deal with the preliminary challenge—image format. To target this, any RGBA image was converted to RGB format as the pre-trained model that we used takes RGB images of size 224x224x3. We also scaled the values of image pixels from a range of 0 to 255 to the range of 0 to 1, because in this way training the model would be easier as the range of values generated by it is more limited. We also performed rotation by a maximum of 30 degrees. Since the images had different dimensions we used the nearest-neighbor algorithm to resize them all into a unified size.

Design and Milestones

Network Configuration

In order to implement our model, we have used the Tensorflow library in Python and we also used OpenCV, Pandas, MatPlotlib, and Numpy, in pre-processing and preparation of our data as well as visualization of the results. Due to the lack of enough data, we took the advantage of the transfer learning method in which a shared backbone is used to extract the features from the images. The backbone is a pre-trained model on the Imagenet database. There are numerous CNN architectures, such as ResNet 50, InceptionV3, MobileNetV2, DenseNet121, etc. In our work we tried different architectures but finally chose to go forward with InceptionV3 (shown in Fig. 1) due to the higher level of accuracy of this model.

In order to map the extracted features to the labels (i.e. Covid, Normal, Pneumonia), we have used two fully-connected layers with 4096 neurons, ReLU activation functions and batch normalization. In the last layer of the network, we used a fully-connected layer with three neurons, each corresponding to one class, which was equipped with Softmax activation function in order to create a probabilistic form of output for the network.

In total, InceptionV3 itself has around 24 million parameters, however, due to the specific requirements of this classification task, we needed to have an additional 268 million parameters added by the fully-connected layers.

In order to train our networks, we used Google Colab Pro environment, which provides access to a Tesla P100 GPU with 16 GB of memory and also provides 16 GB of memory.

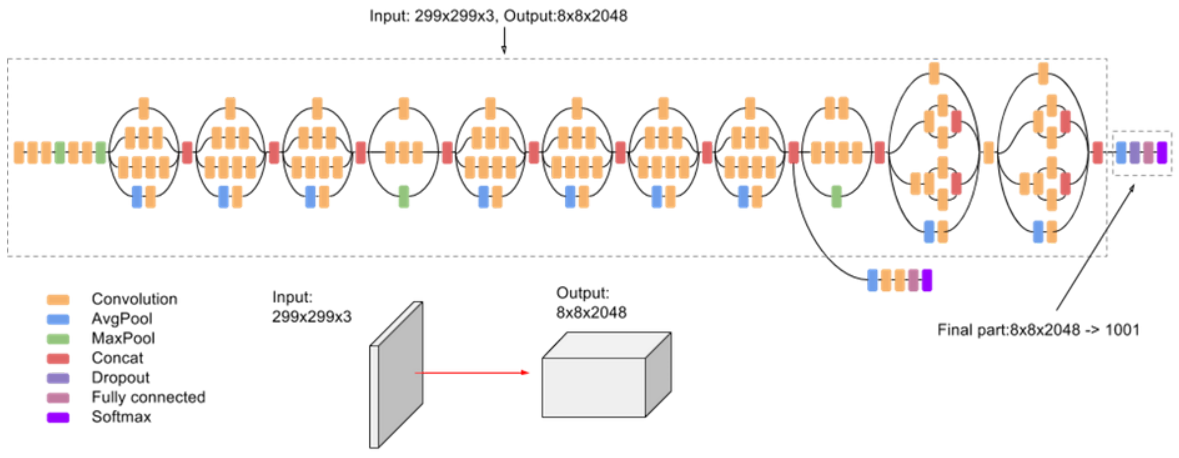


Fig. 1. Illustration of Inception-V3 Architecture

Cases of Failure

Input data:

Initially we intended to use either image gradients or a combination of image gradients with RGB images as the main source of information for the model, however, during the training phase we realized that stacking two different types of data, even with normalization, makes the training harder and does not converge to an optimal performance eventually. In the case of using image gradients only, we realized that the model tends to overfit on the training data and showed poor performance on the test samples.

Google Colab:

Given the high volume of processes on Google Colab Pro GPU servers, we had difficulties training the model during the daytime. A regular training that usually takes two minutes was performed on the servers in one hour on average. This drawback caused delay in the delivery of results and also prevented us from testing different approaches under comprehensive testing schemas.

Training Strategy:

Given the volume of the augmented version of the dataset that we found initially, we decided to fine-tune the whole weights of a pre-trained network in addition to the parameters in the added FC layers. However, we realized that this leads to the problem overfitting towards the positive cases of COVID-19. Therefore, we decided to freeze the parameters in the backbone and just to optimize the added FC layers.

Dataset:

One limitation with the dataset chosen initially was that there were no samples of normal lungs in the database and the test samples did not contain any information about the specifics of the lung disease in case of Negative COVID-19. Therefore, we found another dataset with 3 categories of normal, pneumonia, and COVID-19, with an equal number of samples for all classes. As a consequence of this choice, we are able to identify now whether a patient has COVID or not, and in the case of a negative result, is the lung healthy or it is infected by bacteria.

Results and Analysis

Table 1.

	Softmax Cross Entropy Loss	Accuracy (%)	F1-Score (%)	Precision	Recall
Validation	0.32	90.50	0.8485	0.8310	0.8772
Testing	0.23	92.78	0.8587	0.8361	0.8919

Testing our method on the test images we obtained 92.78% accuracy. The high level of precision with a rate of 0.8361 shows that the model classifies the majority of samples correctly. The high rate of F1-score also is an indicator of the low number of false positive cases.

In order to provide a heatmap of the network's attention we used the idea provided by (link) which tries to find the most activated pixels in classification of a given sample according to the gradients.

Samples of different pages of the developed web application by our team is shown below:

According to the map, the network is paying equal attention to all of the sections of an image. We believe that the lack of specificity of the network as the backbone was frozen and the FC layers were only trained.

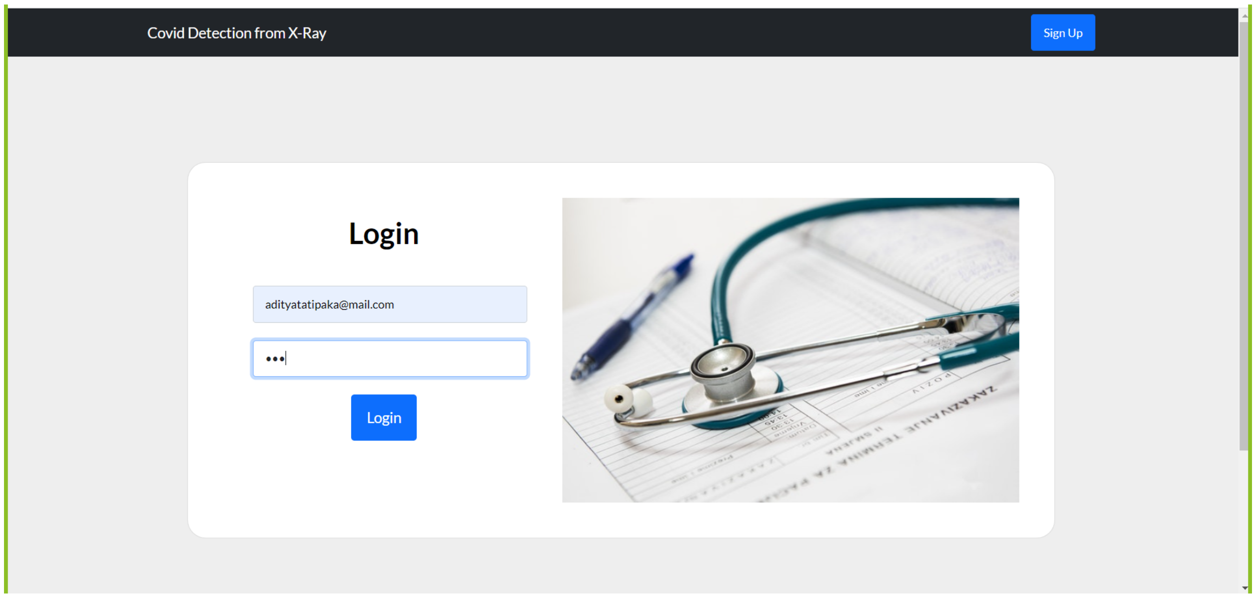
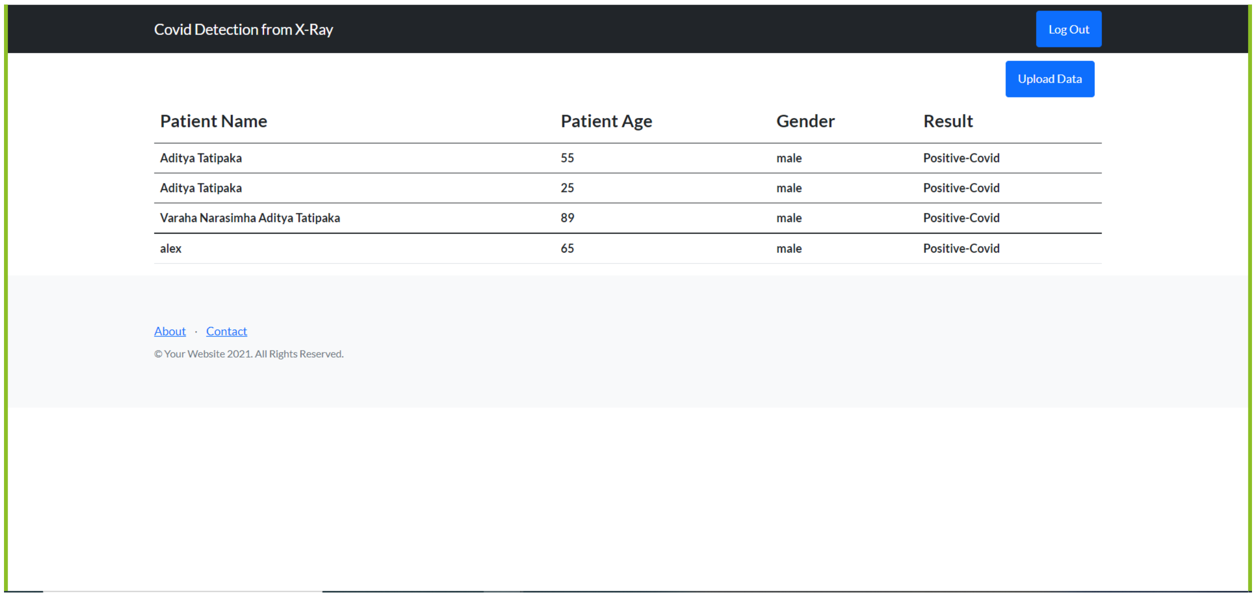


Fig. 2. Login view

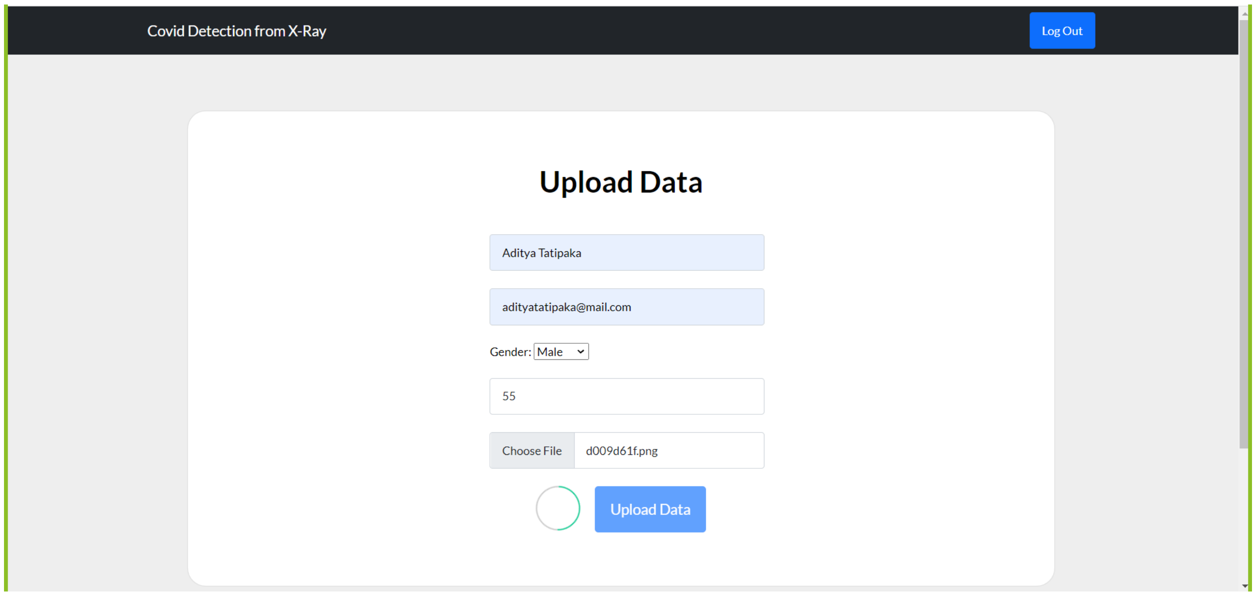

Covid Detection from X-Ray

Patient Name	Patient Age	Gender	Result
Aditya Tatipaka	55	male	Positive-Covid
Aditya Tatipaka	25	male	Positive-Covid
Varaha Narasimha Aditya Tatipaka	89	male	Positive-Covid
alex	65	male	Positive-Covid

About · Contact

© Your Website 2021. All Rights Reserved.

Fig. 3. A view for physician which provides a list of visited patients


Covid Detection from X-Ray

Upload Data

Aditya Tatipaka

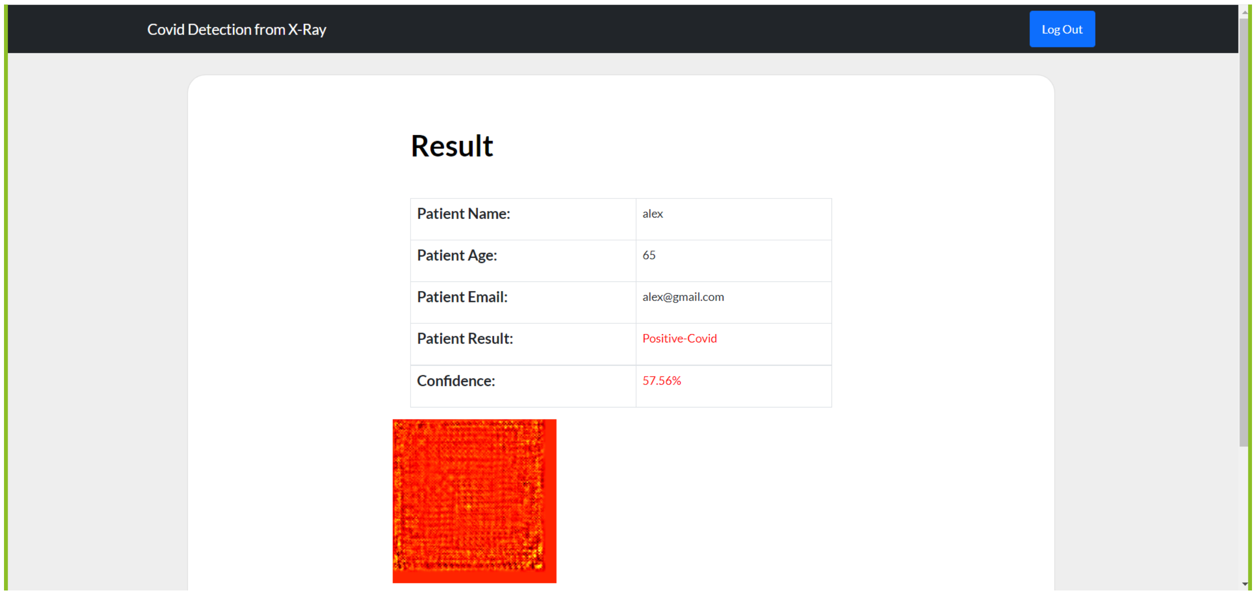
adityatatipaka@mail.com

Gender: Male

55

Choose File d009d61f.png

Fig. 4. A view of the page for uploading the samples



Result

Patient Name:	alex
Patient Age:	65
Patient Email:	alex@gmail.com
Patient Result:	Positive-Covid
Confidence:	57.56%

Fig. 5. A view of the result page for a specific patient. The image on the bottom of the page shows the attention map.

Repository / Archive:

https://github.com/mlghost/Software_Dev_AI

Link to the files on Google Drive:

https://drive.google.com/drive/folders/1Hh3ZbnsbGokrm2QXhmGrurzVOr2kjtPT?usp=sharing

In order to run the training of the network you will need to install the latest version of Tensorflow (preferably v2.5), Numpy, Matplotlib, Pandas, and Scikit-Image and run the SoftwareDevAI_Model.ipynb from our github repo cell by cell. The code is usable both in GPU and CPU modes.

In order to run the user interface the user needs to install Node.js and run the following commands in the below order using the terminal in the root directory.

- npm install
- npm start

Once you have run these commands please open a web-browser and open localhost:3012 to access the login page.

Please sign-up first and then login.

Once Logged in you can View your previous patient's records or can upload new data using the Upload Data button. Please fill in the particulars of the patient and upload the image.

It may take some time to load the result, once the result is loaded you can get to know whether the patient has covid or not.

***Note: You may face issues in connecting to Database so please contact us so that we can white list your IP Address.**