Deep Learning Lab Project

Princy Gautam (B20BB051)

Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks

Introduction

The ongoing COVID-19 pandemic has underscored the need for reliable and efficient diagnostic methods. Traditional diagnostic techniques, such as PCR tests, can be time-consuming and resource-intensive. In this paper, Sekeroglu and Ozsahin explore the application of convolutional neural networks (CNNs) to analyze chest X-ray images for COVID-19 detection. By leveraging the power of deep learning, the authors investigate the potential of CNNs in accurately differentiating COVID-19 cases from other lung diseases. The findings of this study hold promise for the development of a rapid and accessible diagnostic tool, aiding in the timely identification and management of COVID-19 cases.

Motivation

Using X-ray images for the automated detection of COVID-19 might be helpful in particular for countries and hospitals that are unable to purchase a laboratory kit for tests or that do not have a CT scanner. AI tools have produced stable and accurate results in the applications that use either image-based or other types of data. Also, the financial costs of the laboratory kits used for diagnosis, especially for developing and underdeveloped countries, are a significant issue when fighting the illness.

Codeflow

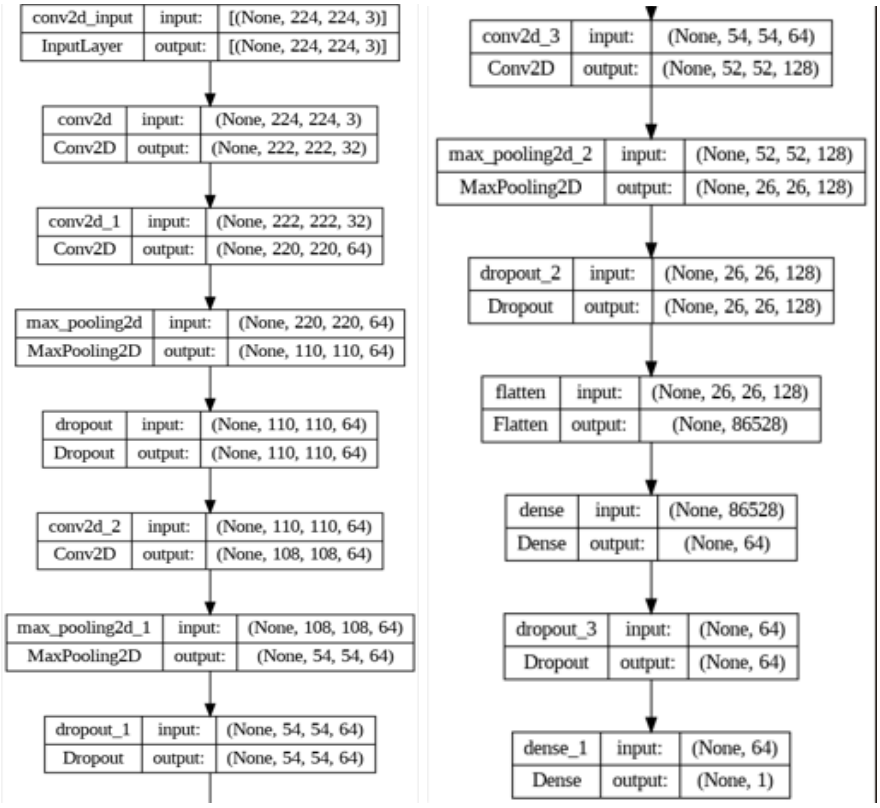
The main aim is to detect if an individual is infected with COVID virus or not by using the chest X-ray images of the individual. To accomplish this task, we design a Convolutional Neural Network (CNN) model which trains our model, the architecture for whom is provided below, on the X-ray images and then classifies the image as a positively or negatively infected person. Steps such as importing libraries, dataset and preprocessing it by rescaling and data augmentation. For rescaling, the image data is divided by 255, which normalizes the pixel values in the range (0,1). The train data is augmented, improving the model's generalization ability. Next, the CNN is defined, consisting of convolutional layers, max-pooling layers, dropout for regularization and finally fully connected layers to classify images. The model is compiled with loss function, optimizer, and evaluation metrics. The trained model is stored for future use. Evaluation is done by observing the performance of the model on both train and test sets. Further testing is done by loading the saved model and making predictions on additional images with recording actual labels and predicted labels. A confusion matrix is also calculated using the records. Also, an alternative approach to data augmentation using ImgAug library is shown and the model is trained and evaluated again with augmented images.

Model architecture

The CNN is defined using Keras framework. The architecture is as follows:

- The model takes in input RGB images of size 224x224.
- Then there are convolutional layers with different filter sizes such as 32, 64, 128 and a 3x3 kernel size.
    - Filters are applied to extract features from the input images using the ReLU activation function which introduces non -linearity.
- The max - pooling layers (2x2) reduce the spatial dimensions of the feature maps, focusing on the most relevant information.
- Dropout layers with 0.25 or 0.5 dropout are used which prevents overfitting by randomly disabling some neurons during the training phase.
- Flatten layer transforms the multi-dimensional feature maps into a single vector.
- The final dense layer contains a single unit with sigmoid activation providing the model's output for binary classification(+/-).
- Loss function: Binary cross entropy loss since it's a binary classification task.
- Optimizer: Adam
- The standard CNN design progressively extracts features through convolutional layers, pooling layers reducing spatial dimensions and dropout for regularization. The flattened features are then fed into fully connected layers for binary classification.

Results



The model achieved 98% accuracy on validation set and other parameters as observed below:

```
<ipython-input-10-3b784b638eea>:3: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Mode
  hist = model.fit_generator(
Epoch 1/10
7/7 [==============================] - 13s 2s/step - loss: 0.8452 - accuracy: 0.5625 - val_loss: 0.6740 - val_accuracy: 0.8667
Epoch 2/10
7/7 [==============================] - 11s 2s/step - loss: 0.6086 - accuracy: 0.6562 - val_loss: 0.5618 - val_accuracy: 0.7167
Epoch 3/10
7/7 [==============================] - 11s 2s/step - loss: 0.4609 - accuracy: 0.7768 - val_loss: 0.3853 - val_accuracy: 0.9500
Epoch 4/10
7/7 [==============================] - 10s 1s/step - loss: 0.2938 - accuracy: 0.8750 - val_loss: 0.1807 - val_accuracy: 0.9500
Epoch 5/10
7/7 [==============================] - 11s 1s/step - loss: 0.2393 - accuracy: 0.9107 - val_loss: 0.1505 - val_accuracy: 0.9667
Epoch 6/10
7/7 [==============================] - 11s 2s/step - loss: 0.1802 - accuracy: 0.9330 - val_loss: 0.0860 - val_accuracy: 0.9667
Epoch 7/10
7/7 [==============================] - 11s 2s/step - loss: 0.3124 - accuracy: 0.8795 - val_loss: 0.1299 - val_accuracy: 0.9667
Epoch 8/10
7/7 [==============================] - 11s 2s/step - loss: 0.1722 - accuracy: 0.9330 - val_loss: 0.1063 - val_accuracy: 0.9833
Epoch 9/10
7/7 [==============================] - 11s 2s/step - loss: 0.1591 - accuracy: 0.9286 - val_loss: 0.0840 - val_accuracy: 0.9667
Epoch 10/10
7/7 [==============================] - 11s 2s/step - loss: 0.1147 - accuracy: 0.9688 - val_loss: 0.0690 - val_accuracy: 0.9833
```

Number of training images = 224
Number of validation images = 60

References

https://journals.sagepub.com/doi/full/10.1177/2472630320958376
https://github.com/aleju/imgaug
https://www.sciencedirect.com/science/article/pii/S2666827021000694
https://www.geeksforgeeks.org/python-data-visualization-using-covid19-india-api/
https://www.analyticsvidhya.com/blog/2021/03/binary-cross-entropy-log-loss-for-binary-classification/
https://towardsdatascience.com/binary-image-classification-in-pytorch-5adf64f8c781
https://github.com/ieee8023/covid-chestxray-dataset