DETECTNG COVID-19 IN X-RAY IMAGES WITH KERAS USING DEEP LEARNING

G.Boopathi Raja, S.Purushotaman, K.Roshni, S.Sateesh Kumar, B.Ebika

Assistant Professor(Sr.Gr), Dept. of ECE, Velalar College of Engineering and Technology, Thindal, Erode, Tamilnadu, India.
B.E Final Year Students, Dept. of ECE, Velalar College of Engineering and Technology, Thindal, Erode, Tamilnadu, India.

ABSTRACT: In this Project, a classification method of early detection of Detecting COVID-19 is crucial in reducing mortality. Magnetic resonance imaging (MRI) may be a viable imaging technique for Detecting COVID-19 detection has been studied for computed tomography(CT) images . However to the best of our knowledge ,no detection methods have been carried out for the MR images .In this project, a Detecting COVID-19 detection method based on deep learning is proposed for thoracic MR images with parameter optimizing, spatial three-channel input construction, and transfer learning, a faster R-convolution neural network(CNN) is designed to locate the Detecting COVID- 19 region.

KEYWORDS: Computed Tomography (CT), GLCM, Neural Network, Magnetic Resonance Imaging (MRI)

INTRODUCTION

The emergence of Corona virus Disease 2019 (COVID-19) in early December 2019 has caused immense damage to health and global well-being. Currently, there are approximately five million confirmed cases and the novel virus is still spreading rapidly all over the world. Many hospitals across the globe are not yet equipped with an adequate amount of testing kits and the manual Reverse Transcription-Polymerase Chain Reaction (RT-PCR) test is time-consuming and troublesome. It is hence very important to design an automated and early diagnosis system which can provide fast decision and greatly reduce the diagnosis error. The chest X-ray images along with emerging Artificial Intelligence (AI) methodologies; in particular Deep Learning (DL) algorithms have recently become a worthy choice for early COVID-19 screening. This paper proposes a DL assisted automated method using X-ray images for early diagnosis of COVID-19 infection. We evaluate the effectiveness of eight pre-trained Convolution Neural Network (CNN) models for classification of COVID-19 from normal cases. Also, comparative analyses have been made among these models by considering several important factors such as batch size, learning rate, number of epochs, and type optimizers with an aim to find the best suited model. The models have been validated on publicly available chest X-ray images and the best performance is obtained by ResNet-34 with an accuracy of 98.33%. This study will be useful for researchers to think for the design of more effective CNN based models for early COVID-19 detection.

The outbreak of the COVID-19 pandemic caused the death of a large number of people. Millions of people are infected by this virus and are still getting infected day by day . As the cost and required time of conventional RT-PCR test to detect COVID-19, researchers are trying to use medical images linke X-Ray and Computed Tomography (CT) images to detect it with the help of Artificial Intelligence (AI) based systems.

LITERATURESURVEY

A. COVID-Net

A tailored deep convolutional neural network design for detection of COVID-19 cases from chest XX-ray images. The Corona virus Disease 2019 (COVID-19) pandemic continues to have a devastating effect on the health and well-being of the global population. A critical step in the fight against COVID-19 is effective screening of infected patients, with one of the key screening approaches being radiology examination using chest radiography. It was found in early studies that patients present abnormalities in chest radiography images that are characteristic of those infected with COVID-19. Motivated by this and inspired by the open source efforts of the research community, in this study we introduce COVID-Net, a deep convolution neural network design tailored for the detection of COVID-19 cases from chest X-ray (CXR) images that is open source and available to the general public. To the best of the authors' knowledge, COVID-Net is one of the first open source networkdesignsforCOVID-19 detection from CXR images at the time of initial .We also introduce COVID x, an open access benchmark dataset that we generated comprising of 13,975 CXR images across 13,870 patient patient cases, with the largest number of publicly available COVID-19 positive cases to the best of the authors' knowledge. Furthermore, we investigate how COVID-Net makes predictions using an explain ability method in an attempt to not only gain deeper insights into critical factors associated with COVID cases, which can aid clinicians in improved screening, but also audit COVID-Net in a responsible and transparent manner to validate that it is making decisions based on relevant information from the CXR images. By no means a production-ready solution, the hope is that the open access COVID-Net, along with the description on constructing the open source COVIDx dataset, will beleveraged and build upon by both researchers and citizen data scientists alike to accelerate the development of highly accurate yet practical deep learning solutions for detecting COVID-19 cases and accelerate treatment of those who need it the most.

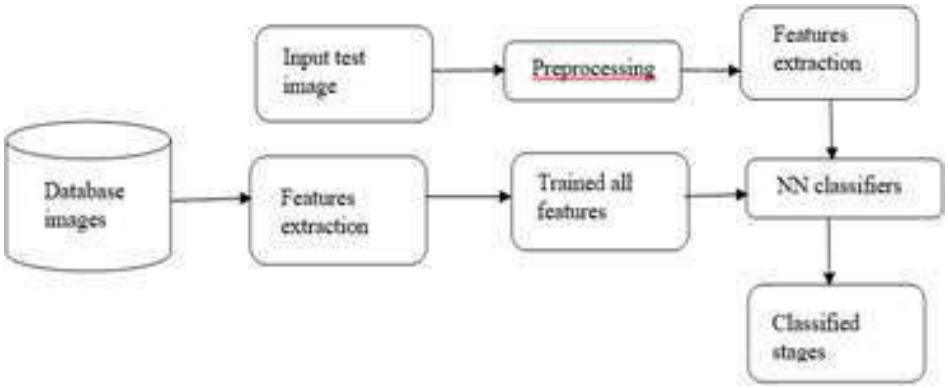

Figure 1 Block Diagram of Proposed System

B. COVID-19 DETECTTION IN CT IMAGES WITH DEEEP LEARNING:

Early detection and diagnosis are critical factors to control the COVID-19 spreading. A number of deep learning-based methodologies have been recently proposed for COVID-19 screening in CT scans as a tool to automate and help with the diagnosis. These approaches, however, suffer from at least one of the following problems: (i) they treat each CT scan slice independently and (ii) the methods are trained and tested with sets of images from the same dataset. Treating the slices independently means that the same patient may appear in the training and test sets at the same time which may produce misleading results. It also raises the question of whether the scans from the same patient should be evaluated as a group or not. Moreover, using a single dataset raises concerns about the generalization of the methods. Different datasets tend to present images of varying quality which may come from different types of CT machines reflecting the conditions of the countries and cities from where they come from. In order to address these two problems, in this work, we propose an Efficient Deep Learning Technique for the screening of COVID-19 with a voting-based approach. In this approach, the images from a given patient are classified as group in a voting system. The approach is tested in the two biggest datasets of COVID-19 CT analysis with a patient-based split. A cross dataset study is also presented to assess the robustness of the models in a more realistic scenario in which data comes from different distributions. The cross-dataset analysis has shown that the generalization power of deep learning models is far from acceptable for the task since accuracy drops from 87.68% to 56.16% on the best evaluation scenario. These results highlighted that the methods that aim at COVID-19 detection in CT-images have to improve significantly to be considered as a clinical option and larger and more diverse datasets are needed to evaluate the methods in a scenario.

III. PROPOSED SYSTEM

Consolidation of CXR images for healthy subjects, pa-tients having pneumonia or other bacterial infection and covid patients from different sources. Retaining only frontal CXR images. Resizing images to a uniform size and Divide the images into two portions one major portion to train the models and another portion for testing the efficacy of the trained model. While dividing the images into training and testing, en-sure that there is no patient overlap i.e. different imagesof the same patient is not present in both training and testing datasets. Train the models - DenseNet201, ResNet50v2 and Inceptionv3Run the trained models on the test images and selectclass label value based on weighted average ensemble of the 3 models.

A. PREPROCESSING

Image pre-processing may have dramatic positive effects on the quality of feature extraction and the results of image analysis. Image pre-processing is analogous to the mathematical normalization of a data set, which is a common step in many feature descriptor methods. Or to make a musical analogy, think of image pre-processing as a sound system with a range of controls, such as raw sound with no volume controls;volume control with a simple tone knob; volume control plus treble, bass, and mid; or volumecontrol plus a full graphics equalizer, effects processing, and great speakers in an acoustically superior room. In that way, this chapter promotes image pre-processing by describing a combination of corrections and enhancements that are an essential part of a computer vision pipeline.

In this section we suggest opportunities for image pre-processing that are guided according to the feature descriptor method you've chosen. Raw image data direct from a camera may have a variety of problems and therefore it is not likely to produce the best computer vision results. This is why careful consideration of image pre-processing is fundamental. For example, a local binary descriptor using gray scale data will require different pre-processing than will a color SIFT algorithm; additionally, some exploratory work is required to fine-tune the image pre-processing stage for best results. We explore image pre-processing by following the vision pipelines of four fundamental families of feature description methods.

B. FEATURE EXTRACTION USING GLCM

To create a GLCM, use the graycomatrix function. The graycomatrix function creates a gray-level co-occurrence matrix (GLCM) by calculating how often a pixel with the intensity (gray-level) value i occurs in a specific spatial relationship to a pixel with the value j. By default, the spatial relationship is defined as the pixel of interest and the pixel to its immediate right (horizontally adjacent), but you can specify other spatial relationships between the two pixels. Each element (i,j) in the resultant GLCM is simply the sum of the number of times that the pixel with value i occurred in the specified spatial relationship to a pixel with value j in the input image. Because the processing required to calculate a GLCM for the full dynamic range of an image is prohibitive, graycomatrix scales the input image. By default, graycomatrix uses scaling to reduce the number of intensity values in gray scale image from 256 to eight. The number of gray levels determines the size of the GLCM. To control the number of gray levels in the GLCM and the scaling of intensity values, using the Num Levels and the Gray Limits parameters of the graycomatrix function. See the graycomatrix reference page for more information.

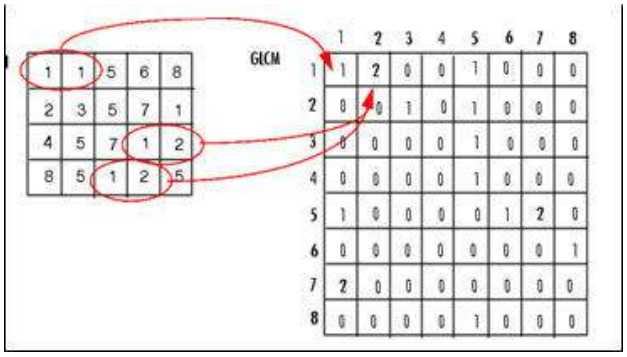

Figure 2 GLCM operations

The gray-level co-occurrence matrix can reveal certain properties about the spatial distribution of the gray levels in the texture image. For example, if most of the entries in the GLCM are concentrated along the diagonal, the texture is coarse with respect to the specified offset.illustrate, the following figure shows how graycomatrix calculates the first three values in a GLCM. In the output GLCM, element (1,1) contains the value 1 because there is only one instance in the input image where two horizontally adjacent pixels have the values 1 and 1, respectively.

GLCM (1,2) contains the value 2 because there are two instances where two horizontally adjacent pixels have the values 1 and 2. Element (1,3) in the GLCM has the value 0 because there are no instances of two horizontally adjacent pixels with the values 1 and 3. graycomatrix continues processing the input image, scanning the image for other pixel pairs (i,j) and recording the sums in the corresponding elements of the GLCM.

To create multiple GLCMs, specify an array of offsets to the graycomatrix function. These offsets define pixel relationships of varying direction and distance. For example, you can define an array of offsets that specify four directions (horizontal, vertical, and two diagonals) and four distances. In this case, the input image is represented by 16 GLCMs. When you calculate statistics from these GLCMs, you can take the average.

It specifies these offsets as a p-by-2 array of integers. Each row in the array is a two-element vector, [row_offset, col_offset], that specifies one offset. Row_offset is the number of rows between the pixel of interest and its neighbour. Col_offset is the number of columns between the pixel of interest and its neighbour. This example creates an offset that specifies four directions and 4 distances for each direction. After you create the GLCMs, you can derive several statistics from them using the graycoprops function. These statistics provide information about the texture of an image.Statistic such a as Contras, Correlation, Energy, Homogeneity gives information about image

C. NN CLASSIFIERS

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates.

In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria.The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems.Here, Supervised learning with non knowledge based classifier will be used for image classification.

The neural network model BPN is used here to act as a classifier with radial basis function for network activation function.The training samples features with assigned target vectors are fed into created BPN model for supervised training to get network parameters such as node biases and weighting factors. Finally, test image features are simulating with trained network to make decision of skin lesion stages like normal or abnormality.

The network classifies input vector into a specific class because that class has the maximum probability to be correct.

The BPN has three layers: the Input Layer, Radial Basis Layer and the Competitive layer. Radial Basis Layer evaluates vector distances between input vector and row weight vectors in weight matrix. These distances are scaled by Radial Basis Function nonlinearly.Competitive Layer finds the shortest distance among them, and thus finds the training pattern closest to the input pattern based on their distance.

D. ARCHITECTURE OF A NN:

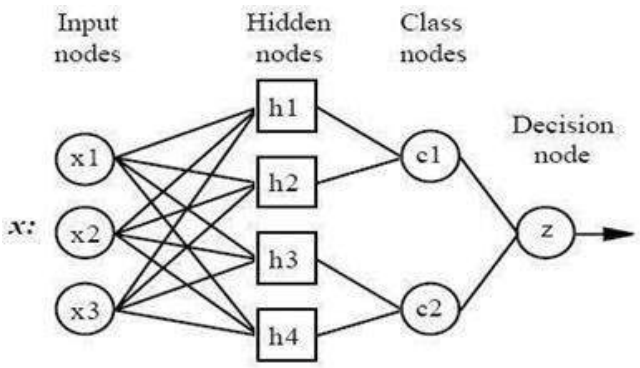

Figure 3 Architecture of NN

Input layer:

There is one neuron in the input layer for each predictor variable. In the case of categorical variables, *N*-1 neurons are used where *N* is the number of categories. The input neurons (or processing before the input layer) standardizes the range of the values by subtracting the median and dividing by the interquartile range. The input neurons then feed the values to each of the neurons in the hidden layer.

Hidden layer:

This layer has one neuron for each case in the training data set. The neuron stores the values of the predictor variables for the case along with the target value. When presented with the *x* vector of input values from the input layer, a hidden neuron computes the Euclidean distance of the test case from the neuron's center point and then applies the RBF kernel function using the sigma value(s). The resulting value is passed to the neurons in the pattern layer.

Pattern layer / Summation layer:

The next layer in the network is different for NN networks and for GRNN networks. For NN networks there is one pattern neuron for each category of the target variable. The actual target category of each training case is stored with each hidden neuron; the weighted value coming out of a hidden neuron is fed only to the pattern neuron that corresponds to the hidden neuron's category. The pattern neurons add the values for the class they represent (hence, it is a weighted vote for that category).

For GRNN networks, there are only two neurons in the pattern layer. One neuron is the denominator summation unit the other is the numerator summation unit. The denominator summation unit adds up the weight values coming from each of the hidden neurons. The numerator summation unit adds up the weight values multiplied by the actual target value for each hidden neuron.

Decision layer:

The decision layer is different for NN and GRNN networks. For NN networks, the decision layer compares the weighted votes for each target category accumulated in the pattern layer and uses the largest vote to predict the target category. For GRNN networks, the decision layer divides the value accumulated in the numerator summation unit by the value in the denominator summation unit and uses the result as the predicted target value.

E. ANACONDA SOFTWARE

Anaconda is a Python distribution that is particularly popular for data analysis and scientific computing

- Open source project developed by Continuum Analytics, Inc.
- Available for Windows, Mac OS X and Linux
- Includes many popular packages: NumPy, SciPy, Matplotlib, Pandas,
- IPython, Cython
- Includes Spyder, a Python development environment
- Includes conda, a platform-independent package manager

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012.As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free Package versions in Anaconda are managed by the package management system conda.This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only conda, Python, the packages they depend on, and a small number of other packages.

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command line interface (CLI).

The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason condaexists.When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages. It will install a package and any of its dependencies regardless of the state of the existing installation. Because of this, a user with a working installation of, for example, Google Tensorflow, can find that it stops working having used pip to install a different package that requires a different version of the dependent numpy library than the one used by Tensorflow. In some cases, the package may appear to work but produce different results in detail.In contrast, conda analyses the current

environment including everything currently installed, and, together with any version limitations specified (e.g. the user may wish to have Tensorflow version 2,0 or higher), works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Open source packages can be individually installed from the Anaconda repository ,Anaconda Cloud (anaconda.org), or the user's own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on PyPI may be installed into a conda environment using pip, and conda will keep track of what it has installed itself and what pip has installed. Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud, PyPI or other repositories.The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it is possible to create new environments that include any version of Python packaged with anaconda.

RESULT

Based on the proposed solution, a simple desktop tool for the detection of COVID-19 positive and negative cases has been developed. This allowed any medical personnel to browse a chest X-Ray image and feeding it to the application . The application in turn will execute the model proposed in this work and provide the label for the given chest X-Ray image. As a result, this will detect the Covid positive and negative cases along with their probabilities.

CONCLUSION

Fast and timely detection of Covid +ve patients are necessary to avoid spreading of the disease and keeping it in control. This research work has been done to detect the Covid+ve patients from Chest X-Ray images in a simple and inexpensive way. In the work proposed in this paper, three state-of-the-art deep learning models have been adopted and ensembled. The proposed model has achieved a classification accuracy of 95.7%. Even more important fact is it has given a sensitivity of 98% i.e. out of 100 Covid +ve patients,98 can correctly diagnosed by our proposed model. It is believed that this research work along with the GUI interface will help the doctors to detect the affected patients with the help of computer aided analysis, that too within a few seconds. We do believe that this will significantly add a value in the medical field.This work was supported in part by the Center of Excellence in Medical Imaging, Data Science Lab, University of Calcutta.

REFERENCES

[1]. Shah,Faisal&Joy,SajibKumarSaha& Ahmed, Farzad&Humaira, Mayeesha& Ami, Amit & Paul, Shimul& Jim, Md. (2020). A Comprehensive Survey of COVID-19 Detection Using Medical Images.10.31224/osf.io/9fdyp.

[2]. Mukherjee, H., Ghosh, S., Dhar, A. et al. Deep neural network to detect COVID- 19: one architecture for both CT Scans and Chest X-rays. ApplIntell (2020).https://doi.org/10.1007/s10489-020-01943-6.

[3]. Henderi,M.Maulana,H.L.H.S.Warnars,D.SetiyadiandT.Qurrohman,"Model Decision Support System For Diagnosis COVID-19 Using Forward Chaining: A Case in Indonesia," 2020 8th International Conference on Cyber and IT Service Management (CITSM), Pangkal, Indonesia, 2020, pp. 1-4, doi: 10.1109/CITSM50537.2020.9268853.

[4]. D. Haritha, C. Praneeth and M. K. Pranathi, "Covid Prediction from X-ray Images,"*2020 5th International Conference on Computing, Communication and Security (ICCCS)*, Patna, India, 2020, pp. 1-5, doi: 10.1109/ICCCS49678.2020.9276795.

[5]. Jain, R. Gupta, M., Taneja, S. et al. Deep learning based detection and analysis of COVID-19 on chest X-ray images. ApplIntell 51690–1700 (2021).https://doi.org/10.1007/s10489-020-01902-1.