Journal of Theoretical and Applied Information Technology
15th December 2021. Vol.99. No 23

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DIAGNOSIS OF COVID-19 USING 3D CONVOLUTIONAL NEURAL NETWORKS

[1]JYOTHI VISHNU VARDHAN, [2]POORNA CHANDRA VEMULA, [3]SRINIVAS G, [4]GUNASEKHAR CHOWDARY, [5]SUNIL KUMAR B

[1]Student, GITAM University, Department of Computer Science, Visakhapatnam, India

[2]Student, Vellore Institute of Technology, Department of Computer Science, Vellore, India

[3]Associate Professor, GITAM University, Department of Computer Science, Visakhapatnam, India

[4]Student, GITAM University, Department of Computer Science, Visakhapatnam, India

[5]Student, Sree Vidyanikethan, Department of Computer Science, Tirupati, India

E-mail: [1]kjvvnat@gmail.com, [2]developer.poornachandra@gmail.com, [3]sgorla@gitam.edu, [4]gunachowdary2203@gmail.com, [5]27sunilkumar27@gmail.com

ABSTRACT

To stop the fast-spreading of covid19, there needs to be a significant improvement in the speed with which the diagnosis is performed. Many studies have been done on using deep learning algorithms like convolutional neural networks and many of its variants available in the industry to make the diagnosis faster. However, most of these approaches involve using datasets that are not that compatible with the real world. In this paper, we will be using efficient techniques to address this problem by using CT scans and leveraging most of the features available in CT-scan images to build a model that can classify whether covid19 infects a person or not, given his CT scan as input to the model. As CT scan images are more reliable and can represent the condition of a person in a more detailed way than any other images like X-rays, these can be used for obtaining faster and precise results.

Keywords: *Data Augmentation, Image Processing, Voxnet, 3D CNN, Volumetric*

1.INTRODUCTION

The outbreak of covid19 was declared a pandemic officially by the World Health Organization in the first quarter of 2020. The disease is highly contagious and can develop as a more lethal acute respiratory distress syndrome(ARDS). Currently, for faster detection, RT-PCR tests are used, but these are very labor-intensive and not so reliable. Many previous studies have focused on solving this problem by using X-ray scans as data for deep learning methodologies [[1][2][3][4]]. Although obtaining the X-ray scans is cheap and faster, they lack detailed information because a conventional x-ray uses a fixed tube that sends x-rays only in one direction.

In contrast, CT scans are more detailed than conventional X-ray images and can reveal bones and soft tissues, and organs. However, obtaining CT scan images is not an easy task. The two most enormous publicly available datasets are the Covid-CT dataset [5], consisting of 349 covid samples and 397 normal samples. The other was prepared by Soares et al. [6], which contains 2482 CT scans collected from various hospitals in são Paulo in Brazil.

To provide an optimal model, we are building a 3D CNN called Voxnet[7]; this enables us to use all the features provided by 3D images without losing any vital information. Primarily, 3D CNN empowers to completely utilize the depth feature of the 3D CT-Scan images. Thus the main objective of this paper is to provide an efficient

solution for the detection of covid-19 through CT-scan images.

The remaining work is structured as follows. The details of the two datasets used in this Paper, COVID-CT [5] and SARS-COV-2 CT-Scan[6], are described in section 2. Methodology, Model Architectures, Environment setup, and results are described in sections 3,4,5, and 6, respectively. In the end, Section 7 concludes this work.

2.DATASETS

The current section discusses the two datasets considered as part of this work. These are two of the largest publicly available datasets to date known to us.

2.1COVID-CT Dataset

To assemble the COVID-CT dataset [5], CT-Scan images of COVID-19 infected patients were gathered from research articles deposited in repositories such as medRxiv and biRxiv. A total of 349 images from 216 COVID-19 infected patients were collected. The

CT-Scan images of healthy individuals are gathered from two other datasets (LUNA dataset, MedPix dataset). A total of 463 images from 55 non-covid patients were collected.

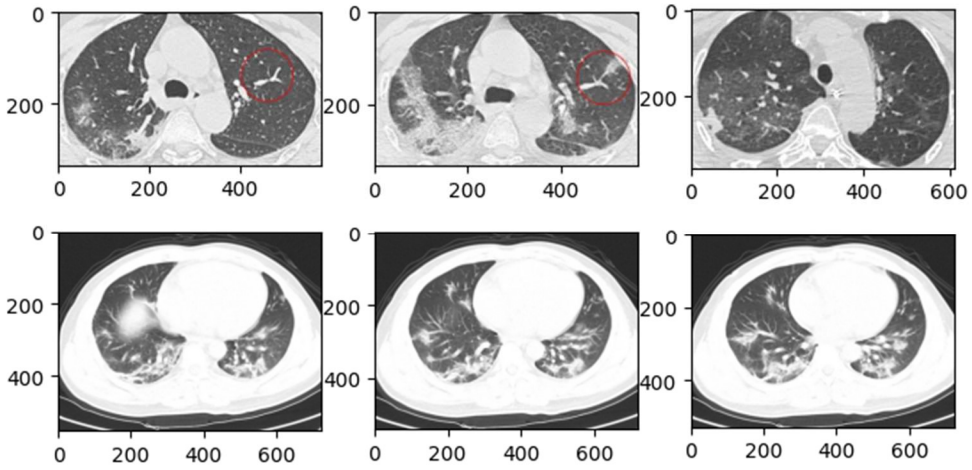

Figure 1: Sample Images from COVID-CT Dataset

2.2SARS-Cov-2 CT-Scan Dataset

The SARS-Cov-2 CT-Scan dataset[6] consists of 2842 CT-Scan images collected from 120 patients, out of which 1252 CT Scans are of 60 covid-19 infected patients, including males(32) and females(28). 1230 CT-Scan images of 60 non-covid infected individuals, including males(30) and females(30), are collected. This data is gathered from various hospitals across São Paulo, Brazil.

3.METHODOLOGY

3.1Image Description

Generally, the 3D data representations are classified into two categories: Euclidean-structured data and Non-Euclidean structured data. The CT scans that we have used in our work are known as volumetric data, in which images can be characterized as a regular grid in a three-dimensional space. The distribution and modeling of the 3D object in three-dimension space are done by using Voxels. Voxel-based representation is very simple, and it can encode details about the 3D figure. However, it has some problems, as described by[8]. In terms of memory, there will be some wastage because it contains the information of non-occupied parts of the scene. Still, this problem can be eliminated by proper pre-processing and is negligible compared to its value to the medical world.

3.2Image pre-processing

Many complications occur when dealing with 3D images. The major ones are the variability in the depth dimension of the Images and the requirement of high GPU memory to train the images. Due to this, many studies[[9,10,11,12,13]] have used 2D CNN architectures where the three-dimensional image is treated as a group of independent slices. But these approaches discard some of the information which is present along the depth axis, that is, the Z-axis and the

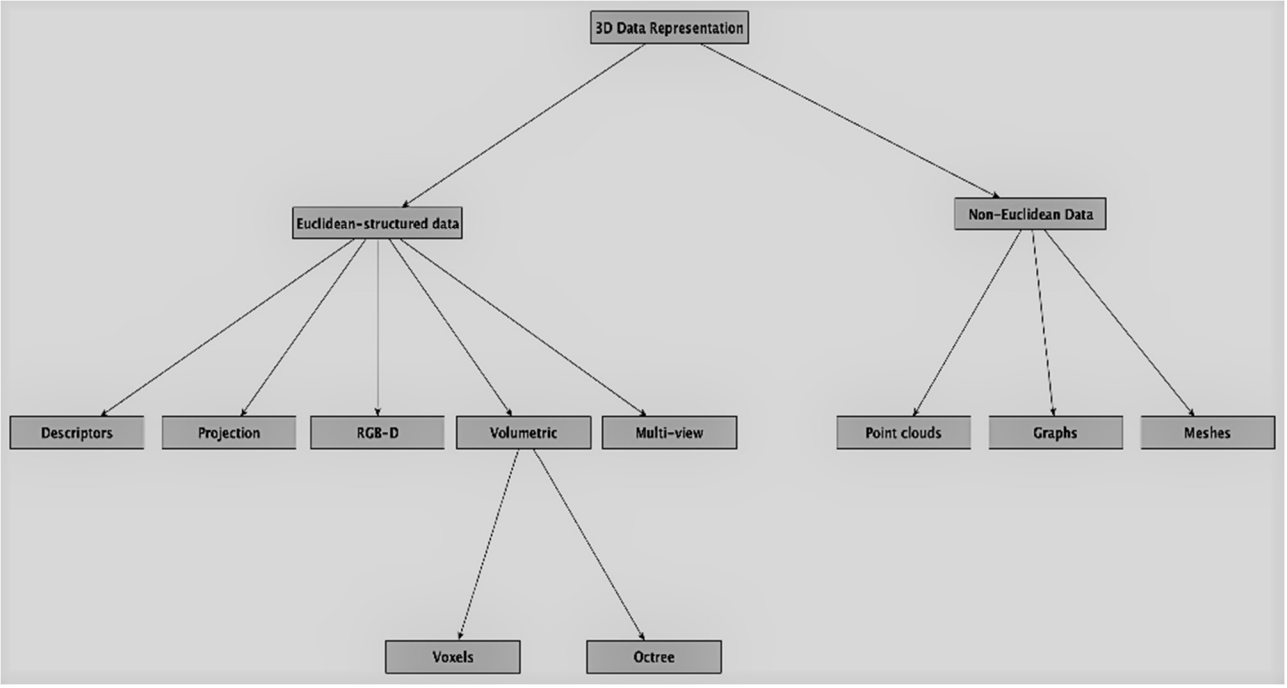

Figure 2: Classification of 3D Data Representations

Preservation of 3D context is also prevented. However, some studies[[14,15,16,17]] have proved that better results can be achieved by using complete volumetric data. In this paper, we do the following pre-processing steps before sending the batches of images into the model.

1. Firstly, the image is normalized by setting min and max values to -1000 and 400 because above 400 Voxels are bones that consist of different radio intensities. In practice, a threshold in the range of -1000 to 400 is generally used for normalizing CT scans.
2. To fix the data orientation, we first rotate the volumes by 90 degrees, and then we scale the Hounsfield values to be between 0 and 1.
3. Then to resize the image. First, we have set our desired depth, the desired width, desired height to 64, 128, 128, and then we compute the depth, width, height factors as follows:

$$Depth\ Factor = \frac{desired\ depth}{current\ depth}$$

$$Height\ Factor = \frac{desired\ height}{current\ height}$$

$$Width\ Factor = \frac{desired\ width}{current\ width}$$

4. Then the images were resized across the z-axis using the computed width, height, depth factors.

3.3 Data Augmentation

It is one of the most common ways of reducing overfitting in Image classification tasks, especially while using neural networks. Alexnet[19] was the first to use this technique. It is not only used for this purpose in the majority of the cases; it is used to tackle the problem of the availability of fewer data. Many types of Data augmentation will be used to create more copies of existing images by enlarging some of the areas, rotating the images in various angles, and flipping the image horizontally or vertically based on the type and nature of the image. The image instances generated must be as realistic as possible. A human must not guess that the augmented image is obtained from an existing image, while simply adding white noise to the image will not be helpful; whatever the modifications made, they must be learnable.

In this paper, we are dealing with three-dimensional images, and as we have already normalized and resized the images across the z-axis. The training set was augmented while the validation set was not touched. As we are using 3D convolutions to extract the features, we have added a dimension of size 1 to every image which was stored in the form of rank three tensors of shape [samples, height, width, depth] to be converted into the form which is like [samples, height, width, depth, 1]. Then we have rotated the images at random angles to generate a variance in the dataset.

4.MODEL ARCHITECTURES

The main difference between 2D CNNs and 3D CNNs is that 2D CNNs fail to exploit the depth dimension that is available in 3D images, there are various architectures proposed and used by many studies[[7, 8, 14, 15, 16, 17]] and it is proved that they have been successful in taking advantage of all the rich features provided by 3D images. The architectures that we have used in our work are discussed one by one as follows:

4.1Voxnet

This architecture has shown promising results in the works of [[7, 17, 18, 15]]. The structure is almost similar to that of VGG when it comes to the number of filters and kernel size, except that this is extended for 3d images. The model that we used in our work consists of 17 layers. In the first layer, the batches of images are taken as input, and they are forwarded to the first convolution layers, which contain a total of

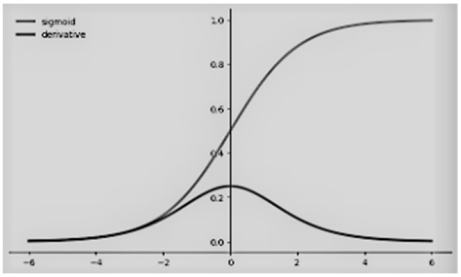

Figure 3: Sigmoid Function

64 filters whose kernel size is 3X3X3 along with RELU[20][21][figure 5], which was used as an activation function to avoid the problem of vanishing gradients which was a common problem in activation functions like sigmoid[21][figure 3] and tanh[figure 4], considering the fact we are dealing with 3D CNNs which will have lots of parameters.

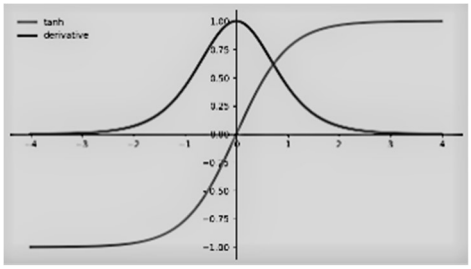

Figure 4: Tanh Function

Following the convolution layer, we added and a maxpool layer with poolsize 2X2X2. This was followed by a Batch-Normalization layer which consists of some non-trainable parameters in the

model, and it is mathematically represented as in the [eq-no 1].

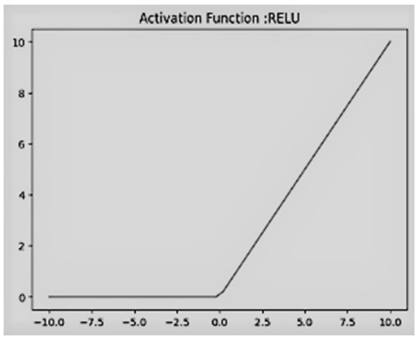

Figure 5: RELU Function

$$\boldsymbol{\mu}_B = \frac{1}{m_B}\sum_{i=1}^{m_B} \mathbf{x}^{(i)}$$

$$\boldsymbol{\sigma}_B^2 = \frac{1}{m_B}\sum_{i=1}^{m_B} \left(\mathbf{x}^{(i)} - \boldsymbol{\mu}_B\right)^2$$

$$\hat{\mathbf{x}}^{(i)} = \frac{\mathbf{x}^{(i)} - \boldsymbol{\mu}_B}{\sqrt{\boldsymbol{\sigma}_B^2 + \epsilon}}$$

$$\mathbf{z}^{(i)} = \boldsymbol{\gamma} \otimes \hat{\mathbf{x}}^{(i)} + \boldsymbol{\beta}$$

Equation 1 Batch Normalization Equation

The next three layers are the same as the above three with a Convolution, Maxpool, Batch-Normalization. The next three layers also follow the same architecture except that the number of filters increased to 128 in the conv layer. The patterns repeat the same even in the coming layers with the number of filters set to 256 in the convolution layer; then, at final, the feature maps are subjects to a Global-average-pooling 3D layer and a Dense layer with 512 neurons with a Dropout rate of 0.3 is added to the model. Finally, an output layer with 1 neuron is added to the model with sigmoid[21][eq-no 2] as an activation function as we deal with binary classification. The pictorial architecture of the model is shown in figure[6]

$$S(x) = \frac{1}{1 + e^{-x}}$$

Equation 2 Sigmoid Function

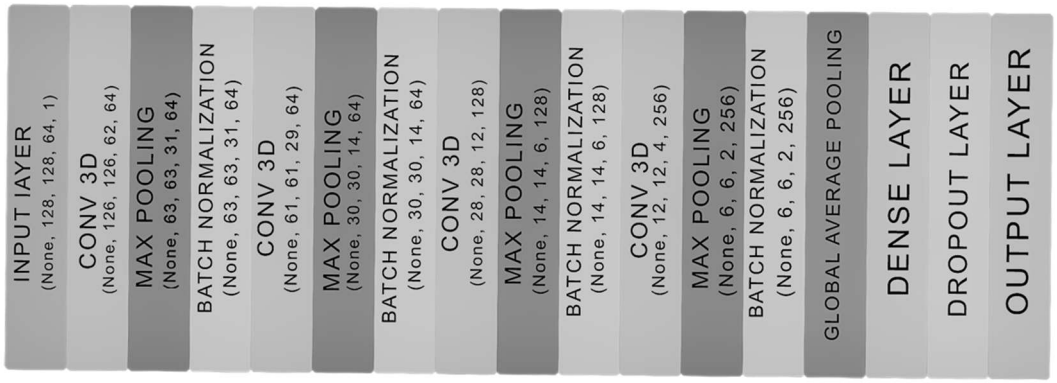

Figure 6: Architecture 4.1

4.2 Architecture

Then we experimented with the architecture used by [15] for human action recognition with some slight modifications. Firstly, we added an Input layer that takes a tensor of shapes [samples, height, width, depth, 1]. Following it, a Convolution layer of 32 filters was applied with a kernel size of 7X7X3(7X7 denotes the spatial dimension and 3 lies in the temporal dimension) with an activation function of RELU[20][21] followed by a pooling layer of poolsize 2. Then a Convolution layer of 64 filters of kernel size 7X6X3 along with 3X3 subsampling. Finally, a Convolution of 7X4 is added and connected to a Dense layer of 256 neurons and an output layer with a sigmoid[21] as the activation function. The architecture of the network is represented in the figure[7].

4.3V-net

As stated by [17], we built an Architecture almost similar to V-net except that we have used the network for classification. To begin with, we have added an Input layer to the model, which accepts an image input tensor of shape (128, 128, 64, 1) then a convolutional layer consisting of 16 filters with a kernel size of 5X5X3 with RELU[20][21] as the activation function followed by and Max Pooling layer of pool_size 2X2X2. Then the feature maps obtained from the first convolutional layer are subjected to another convolutional layer containing 32 filters with a kernel size of 5X5x3 with RELU[20][21] as the activation function followed by a max pooling layer similar to the above layer. Again the same structure is followed for the next three blocks except that the filters are changed to 64, 128, 256 in each of the blocks. The first convolutional layer captures the majority of the important features. To fasten our process, we have used a Max Pooling layer at each stage and captured the most important features in each of the feature maps. Due to the action of the Max pooling layer, the images were reduced through each of the layers at the beginning it resized to (64, 64, 32) and then to (32, 32, 16) following this to (16, 16, 8) and finally to (8, 8, 4). Then Global average pooling3D was performed, and we connected the obtained vector to a Dense layer consisting of 512 neurons with RELU[20][21][eq-no 3] as the activation function. At last, we have added an output layer with sigmoid[21] as the activation function as we are dealing with Binary classification; this architecture is represented in figure[8].

$$y = max(0, x)$$

Equation 3 RELU Function

Figure 7: Architecture 4.2

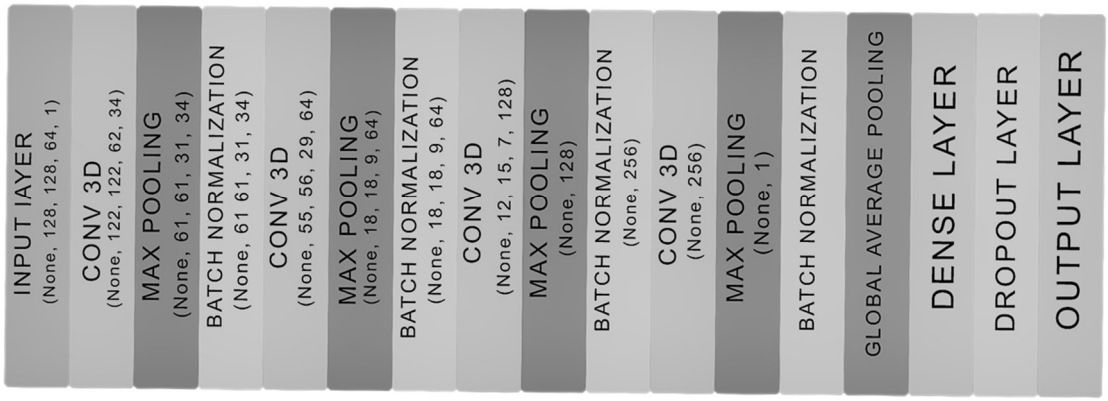

Figure 8: Architecture 4.3

5.ENVIRONMENT SETUP

We first processed and prepared the CT scan[5] dataset and divided it into training(70%) and validation(30%) sets randomly, then we split the SARS-Cov-2 CT-Scan dataset[6] into training(80%) and testing(20%). Our main objective was to make our model robust to all types of populations from various locations. We trained the model by combining the datasets [[5, 6]] collected from different locations to achieve this. The training set of the SARS-Cov-2 CT-scan dataset was distributed randomly to the training and validation sets of the CT scan[5] dataset. The architecture 4.1 was trained using an Intel(R) Core(TM) i7-5820K CPU 3.30 GHz, 32 GB Ram along with a 4GB NVIDIA GEFORCE 1650 Ti GPU. We used Tensorflow 2 Library to build the network. The model was initially set to train for 100 epochs with an initial learning rate of 0.0001 with an exponential decay rate of 0.96 for every 100000 steps. Binary cross entropy was used as our loss function, and we have used Adam[23] as our optimizer, whose formulation is represented in the equation[no 4]

$$\mathrm{m} \leftarrow \beta_1 \mathrm{m} - (1-\beta_1)\nabla_\theta J(\theta)$$
$$s \leftarrow \beta_2 \mathrm{s} + (1-\beta_2)\nabla_\theta J(\theta) \otimes \nabla_\theta J(\theta)$$
$$\hat{m} \leftarrow \frac{m}{1-\beta_1^{\mathrm{T}}}$$
$$\hat{s} \leftarrow \frac{s}{1-\beta_2^{\mathrm{T}}}$$
$$\theta \leftarrow \theta + \eta \hat{m} \oslash \sqrt{\hat{s}+\varepsilon}$$

Equation 4 Adams Equation

The architecture 4.2 was trained using an Intel(R) Core(TM) i7-5820K CPU 3.30 GHz, 32 GB Ram, along with an 8GB NVIDIA RTX 2080 Ti GPU. We have followed the same approach as architecture 4.1, like using Tensorflow 2 library to build the model. It was set to train for 100 epochs where we used a learning rate of 0.0001 with an exponential decay rate of 0.96 for every 10000 steps. In terms of loss function and optimizer, we have chosen Binary-crossentropy and Adam.

Architecture 4.3; Same as architecture 4.2, Architecture 4.3 was trained using an Intel(R) Core(TM) i7-5820K CPU 3.30 GHz, 32 GB Ram, and an 8GB NVIDIA rtx 2080 Ti GPU. We have followed the same approach as architecture 4.1, like using Tensorflow 2 library to build the model. It was set to train for 100 epochs where we used a learning rate of 0.0001 with an exponential decay rate of 0.96 for every 10000 steps. In terms of loss function and optimizer, we have chosen Binary-crossentropy and Adam.

6.RESULTS

The results obtained from our model are very promising as our end goal is to achieve high recall. Recall is nothing but out of the total number of positive labels, how many are predicted to be positive. As we are dealing with medical diagnosis, our goal is to get the type2 error as low as possible. We have tested our model with 294 covid test samples, out of which our model correctly classified 286. We have achieved an accuracy of 75% on the testing data. The various metrics that we have evaluated are shown in the tables[1,2,3].

Of all the models, architecture 4.3 gave the best results with a "recall" of 0.98 on covid images, whereas architecture 4.2's performance is the worst among the three architectures. Architecture 4.1 also has a good "recall" of 0.97 on covid images, close to the recall obtained using architecture 4.3. Still, the number of trainable parameters for architecture 4.3 is 3,451,585, and architecture 4.1 is 1,351,873. The number of trainable parameters in architecture 4.3 is high relative to architecture 4.1, even though their recall values are almost identical. Architecture 4.1 reduces the cost of resources and is a better choice in the case of low latency applications.

Table 1: Metrics for Architecture 4.1

	Precision	Recall	F1 Score
normal	0.96	0.56	0.71
covid	0.64	0.97	0.77

Table 2: Metrics for Architecture 4.2

	Precision	Recall	F1 Score
normal	0.83	0.60	0.69
covid	0.62	0.84	0.72

Table 3: Metrics for Architecture 4.3

	Precision	Recall	F1 Score
normal	0.97	0.48	0.64
covid	0.60	0.98	0.74

In the case of precision, the values are low in all three architectures. This is because the model predicted some of the non-covid samples as covid, but the main goal of this paper is to get very low Type 2 error, and we are successful in achieving this objective. However, we started investigating why this problem is occurring in the case of precision.

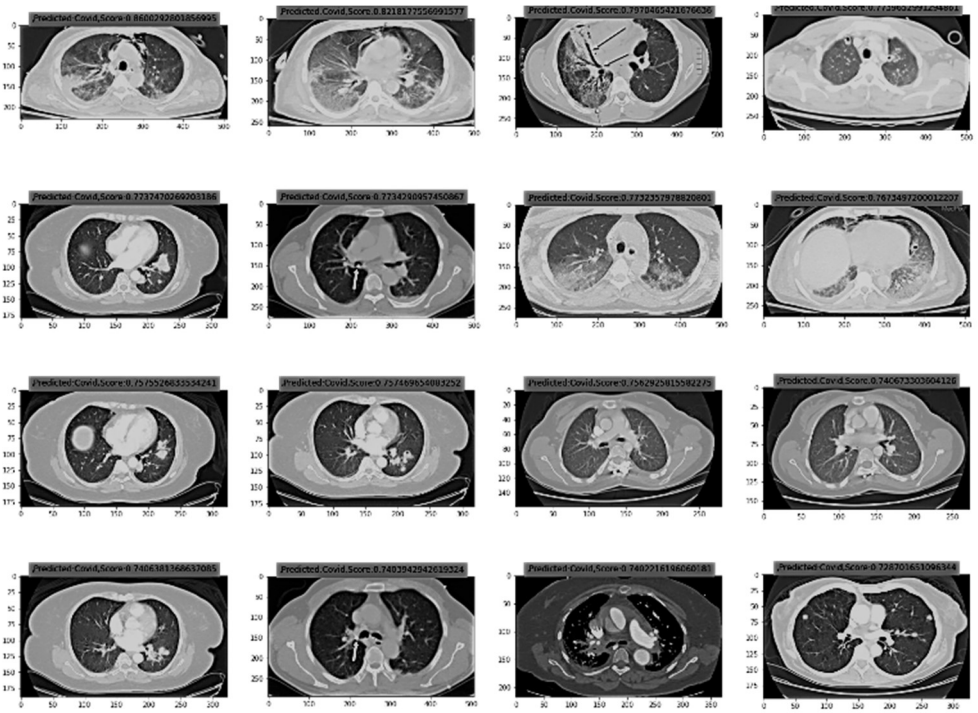

Figure 9: Samples Labeled Incorrect

So, we have analyzed the samples which have predicted non-covid samples as covid with a high probability score. Some of these samples seemed like covid. This may be due to manual errors while labeling the data by the creators. Some of these samples are shown in the figure[9]. As we do not possess much knowledge about radiology, we did not try to relabel the data. So, any further study regarding this problem will be appreciated.

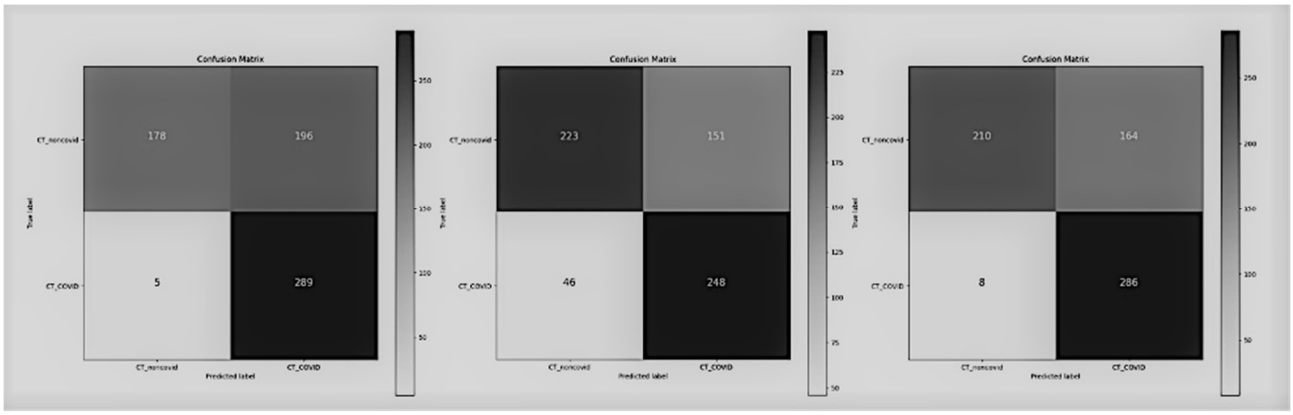

Figure 10: Confusion matrices of three Architectures

The confusion matrices of the three architectures are shown in the figure[10]. The confusion matrix represents different combinations of the actual and predicted values. In the confusion matrices obtained, we can see that eight covid test samples are classified incorrectly when architecture 4.1 is used, and only five images are classified incorrectly if architecture 4.3 is used. However, 46 images are incorrectly classified using architecture 4.2, which is relatively higher.

7.CONCLUSION AND FUTURE WORK

The architectures we have used gave good results, and the trainable parameters in the architectures 4.1 and 4.2 are very few compared to many state-of-the-art algorithms. The number of parameters in architecture 4.1 is 1.3 million, 3.4 million in architecture 4.2, and 1 million in architecture 4.3. The best performing model is architecture 4.3 with a recall of 0.98, but architecture 4.1 with a recall of 0.97 will be most preferable when it comes to latency. We hope that our work can be used for real-time diagnosis, due to which results of diagnosis can be obtained faster, and many people's lives can be saved.

Through this study, we can understand the power of deep learning, which can be leveraged to solve many complex issues like COVID-19. To build a robust system, many datasets of CT-Scan images from various geographical locations are required. Our next goal is to build a dataset consisting of CT-Scan images from patients in many states of India. As India is a densely populated and diverse country, CT-Scan images of patients from this diverse population can improve the model.

REFERENCES

[1] Albahli S, Albattah W. Detection of coronavirus disease from X-ray images using deep learning and transfer learning algorithms. J Xray Sci Technol. 2020;28(5):841-850. doi: 10.3233/XST-200720. PMID: 32804113; PMCID: PMC7592683.

[2] Zhang R, Guo Z, Sun Y, Lu Q, Xu Z, Yao Z, Duan M, Liu S, Ren Y, Huang L, Zhou F. COVID19XrayNet: A Two-Step Transfer Learning Model for the COVID-19 Detecting Problem Based on a Limited Number of Chest X-Ray Images. Interdiscip Sci. 2020 Dec;12(4):555-565. doi: 10.1007/s12539-020-00393-5. Epub 2020 Sep 21. PMID: 32959234; PMCID: PMC7505483.

[3] Mahmud T, Rahman MA, Fattah SA. CovXNet: A multi-dilation convolutional neural network for automatic COVID-19 and other pneumonia detection from chest X-ray images with transferable multi-receptive feature optimization. Comput Biol Med. 2020 Jul;122:103869. doi: 10.1016/j.compbiomed.2020.103869. Epub 2020 Jun 20. PMID: 32658740; PMCID: PMC7305745.

[4] Apostolopoulos ID, Mpesiana TA. Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks. Phys Eng Sci Med. 2020

Jun;43(2):635-640. doi: 10.1007/s13246-020-00865-4. Epub 2020 Apr 3. PMID: 32524445; PMCID: PMC7118364.

[5] Yang, X., Zhao, J., Zhang, Y., He, X., & Xie, P. (2020). COVID-CT-Dataset: A CT Scan Dataset about COVID-19. ArXiv, abs/2003.13865.

[6] Soares, Eduardo & Angelov, Plamen & Biaso, Sarah & Froes, Michele & Abe, Daniel. (2020). SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification. 10.1101/2020.04.24.20078584.

[7] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition," 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 922-928, doi: 10.1109/IROS.2015.7353481.

[8] Y. Xiang, Wongun Choi, Y. Lin and S. Savarese, "Data-driven 3D Voxel Patterns for object category recognition," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1903-1911, doi: 10.1109/CVPR.2015.7298800.

[9] Gao XW, Hui R, Tian Z. Classification of CT brain images based on deep learning networks. Comput Methods Programs Biomed. 2017 Jan;138:49-56. doi: 10.1016/j.cmpb.2016.10.007. Epub 2016 Oct 20. PMID: 27886714.

[10] M. Grewal, M. M. Srivastava, P. Kumar and S. Varadarajan, "RADnet: Radiologist level accuracy using deep learning for hemorrhage detection in CT scans," 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), 2018, pp. 281-284, doi: 10.1109/ISBI.2018.8363574.

[11] Kavitha, S., Nandhinee, P., Harshana, S., Jahnavi Srividya S., Harrinei, K.: Imageclef 2019: A 2d convolutional neural network approach for severity scoring of lung tuberculosis using ct images. CLEF2019 Working Notes 2380, 9–12 (2019)

[12] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *MICCAI*.

[13] Tabarcea, Augustus et al. "ImageCLEFmed Tuberculosis 2019: Predicting CT Scans Severity Scores using Stage-Wise Boosting in Low-Resource Environments." *CLEF* (2019).

[14] Huang, Xiaojie et al. "Lung nodule detection in CT using 3D convolutional neural networks." *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)* (2017): 379-383.

[15] S. Ji, W. Xu, M. Yang and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 1, pp. 221-231, Jan. 2013, doi: 10.1109/TPAMI.2012.59.

[16] Li, B. et al. "Vehicle Detection from 3D Lidar Using Fully Convolutional Network." *ArXiv* abs/1608.07916 (2016): n. Pag.

[17] F. Milletari, N. Navab and S. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," 2016 Fourth International Conference on 3D Vision (3DV), 2016, pp. 565-571, doi: 10.1109/3DV.2016.79.

[18] Zunair, Hasib et al. "Uniformizing Techniques to Process CT scans with 3D CNNs for Tuberculosis Prediction." *PRIME@MICCAI* (2020).

[19] Krizhevsky, Alex & Sutskever, Ilya & Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

[20] Agarap, Abien Fred. (2018). Deep Learning using Rectified Linear Units (ReLU).

[21] Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation Functions: Comparison of trends in Practice and Research for Deep Learning. ArXiv, abs/1811.03378.

[22] Quan, Tran Minh & Hildebrand, David & Jeong, Won-Ki. (2016). FusionNet: A Deep Fully Residual Convolutional Neural Network for Image Segmentation in Connectomics. Frontiers in Computer Science. 3. 10.3389/fcomp.2021.613981.

[23] Kingma, Diederik & Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.