

# Simple Neural Network based TB Classification

Anirudh Anand, Karthik Raja Anandan, Bhuvana Jayaraman and  
Miralinee Thanga Nadar Thanga Thai

*Sri Sivasubramaniya Nadar College of Engineering, Chennai, Tamil Nadu, India*

## Abstract

Analysis of images is a vitally important task in medical applications. It helps the prompt detection and categorization of diseases, among others. This paper depicts a intuitive and simple approach to classify the Tuberculosis found in the 3D CT-images of patients' chests as a part of the ImageCLEF2021 challenge. A simple shallow neural network is employed with three layers. The model is trained using augmented images of the dataset. The proposed model is tested for it's accuracy and kappa coefficient to obtain the degree to which the model correctly classifies the chest images.

## Keywords

Computed Tomography, Tuberculosis classification, Neural Network, Tensorflow, Image Classification

## 1. Introduction

Tuberculosis (TB) is an airborne disease caused by Mycobacterium tuberculosis (MTB) that usually affects the lungs leading to severe coughing, fever, and chest pains. Although current research in the past four years has provided valuable insight into TB transmission, diagnosis and treatment, much remains to be discovered to effectively decrease the incidence of and eventually eradicate TB [1]. According to a report in 2013, around 3 million cases of TB went undiagnosed, mainly because of under trained staff, inaccurate tests and lack of equipment [2].

Computed Tomography (CT) uses X-rays to create images of objects. It has a plethora of applications and is of importance in the medical field [3]. Analysis of CT-images can provide useful insight in the diagnosis of TB. Motivation of the above, JBTTM team participated in the ImageCLEF2021 [4] Tuberculosis challenge [5] to categorize images of patients' chests into one of 5 significant types.

Prior to the development of the model, simple shallow neural networks and convolutional neural networks were studied [6]. Basics of Python's NumPy and associated libraries were also studied. Image compression techniques were researched. The model was then developed culminating knowledge gained from the same.

The approach used for analysis of 3D CT-images involved using simple neural networks to map test images to one of five classes. The model is trained using augmented images of the

---

*CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania*

✉ anirudh19015@cse.ssn.edu.in (A. Anand); karthikraja19048@cse.ssn.edu.in (K. R. Anandan);

bhuvanaj@ssn.edu.in (B. Jayaraman); mirnalineett@ssn.edu.in (M. T. N. T. Thai)


🌐 <https://www.ssn.edu.in/staff-members/dr-j-bhuvana/> (B. Jayaraman);

<https://www.ssn.edu.in/staff-members/dr-t-t-miralinee/> (M. T. N. T. Thai)

🆔 0000-0002-9328-6989 (B. Jayaraman); 0000-0001-6403-3520 (M. T. N. T. Thai)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

data set. Salient features of Python's NumPy [7] are implemented to realize the same.

## 2. Task and Dataset

As mentioned above, the broad objective of ImageCLEF2021 is to classify 3D CT-images of patients' lungs into one of 5 TB categories, namely: (1) Infiltrative, (2) Focal, (3) Tuberculoma, (4) Miliary and (5) Fibro-cavernous. A dataset containing chest CT scans of 1338 TB patients is used. 917 images for the Training (development) data set and 421 for the Test set. Additionally, metadata is provided for some images.

### 2.1. Multi-dimensional neuroimaging data

For all patients a single 3D CT image with an image size per slice of  $512 \times 512$  pixels and number of slices being around 100 is provided. All the CT images are stored in NIFTI file format with .nii.gz file extension (g-zipped .nii files). This file format stores raw voxel intensities in Hounsfield units (HU) as well the corresponding image metadata such as image dimensions, voxel size in physical units, slice thickness, etc. Python's Nibabel package [8] is used to read the .nii files.

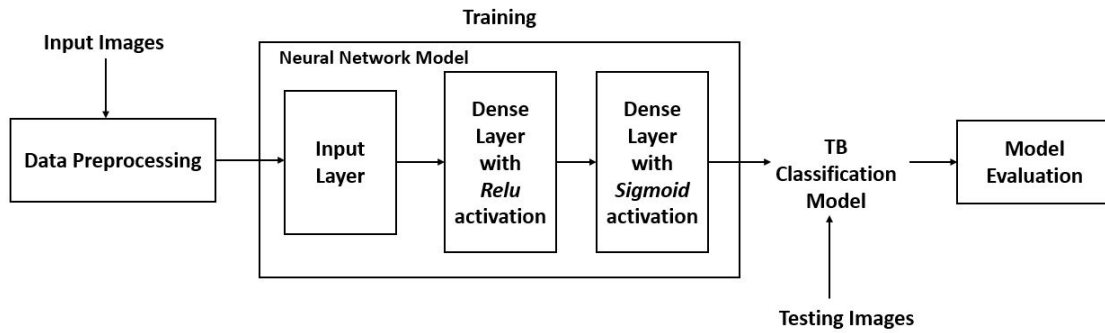
## 3. Methodologies

### 3.1. Data preprocessing

Nibabel library was used to load the zipped Nifti fileformat of CT-scan images and return them as NumPy arrays. The values of the NumPy array is normalized based on threshold values of hounsfield units. The given NumPy array contains raw voxel intensities in Hounsfield units (HU). The Hu for Air is -1000 and Hu for tissues is 500. And those intensities higher than 500 makes up the bones in the image. Here we are taking into those account only those between -1000 to 500 and used to normalize the voxel (Volume pixel) values of the NumPy array to the range [0 to 1]. This is scaled down to  $128 \times 128 \times 64$  image size from  $512 \times 512 \times 113$ . The resulting scaled down 3D-array is then rotated to randomize the orientation. Since the entire data set can't be read into memory in one go, it is read in batches of 20 and is then sent for training. A linear interpolation [9] operator from SciPy was used to scale down the image sizes. Since the CT-scan image was already given in higher resolution, it is assumed that the features and edges would be retained after a simple Linear interpolation. It also results in faster processing. Normalizing data is said to speed up the learning process and leads to faster convergence.

### 3.2. Image Augmentation

To increase the count of training set and to enhance variability into the training set, the obtained dataset is mapped to functions that rotate the images by degrees of 5 to create augmented data that is used for training. The training batch size is set as 20 (the maximum possible size without getting an out of memory error). Validation set contains equal number of all category dataset, to produce an unbiased accuracy. However, one additional instance each, was added to class 3 and class 4 to tune the model well during the training and hence the validation set size is 27.



**Figure 1:** Proposed Neural Network Architecture

### 3.3. Model Architecture

A simple shallow neural network model has been designed to classify the Tuberculosis found in the 3D CT-images. The proposed model has three layers in it as shown in Fig. 1 The first layer accepts the preprocessed images and are flattened before passing them to two fully connected layers.

The  $128 \times 128 \times 64$  3D image is flattened to  $(128 \times 128 \times 64)$  single dimensional vector and is passed through a dense layer of 600 (picked at random after working with lower dimensions) neurons and the output of this layer is given to the last layer which returns a one hot encoded vector.

The first fully connected layer uses 'relu' as activation function, since that activation function handles the problem of vanishing gradient. The last classification layer has 5 nodes corresponding to the classes of Tuberculosis and employs sigmoid activation function, that produces outcome similar to the probabilistic values pertaining to the classes.

The loss function used is Binary cross entropy which is optimized using the Stochastic Gradient Descent optimizer. The model performance during training is evaluated using the accuracy metric.

### 3.4. Convolutional Neural Network (CNN) Model

The proposed model for Tuberculosis classification is arrived after exploring another model using the Convolutional layers called as Convolutional Neural Network (CNN). The CNN model has been designed with 4 convolutional layers, each followed by max pooling layer to reduce the spatial dimension of the images. The convolutional layers extract the features from the input images and are fed to 2 fully connected layers. Batch normalization is done to avoid model over fitting. The model configuration and parameter details are shown in Fig. 2.

CNN model used the same pre-processing techniques followed by Neural Network model. This CNN model did not show any promising results while training when compared to the Neural Network model explained in section 3.3. The average validation accuracy measured was only 0.15. We suspect that lack of data can attributed to this poor accuracy. Hence we had to alter our model to a much simpler neural network that can work with smaller amount of data.

## 4. Experiments and Results

### 4.1. Hardware used

Google Colab notebook was used to train the model. A general purpose RAM size of 8GB was allotted with a 2.3GHz Intel Xenon CPU.

### 4.2. Code

Implementation is done using Python and the URL to the code is shared below. <https://colab.research.google.com/drive/1wbTPPON2AF72OMnCkchTcQl2Cd87KeFR?usp=sharing> [10].

### 4.3. Result

The two models are trained for 20 epochs, accuracy metric is used to study the performance of the model during training. The Metric for both the models are given in Table 1. Complex deep learning networks namely ResNet, GoogleNet have achieved around 0.4033 in 2017 imageclef. So keeping them into account, we have tried out to build a simpler Neural Network model for classifying and have achieved validation accuracy of 0.20. The Neural network model was alone out of the two experiments, submitted to the ImageCLEFmedical task for evaluation.

**Table 1**

Training metrics

Model	Training Accuracy	Validation Accuracy
Neural Network	0.647	0.20
CNN	0.566	0.15

The proposed model has obtained a testing accuracy of 0.221 and a kappa value of 0.038 as reported in Table 2. These metrics were used for ranking and placed us in the ninth place ImageClef 2021 TB classification [5] challenge.

**Table 2**

Evaluated results of ImageCLEF medical

Rank	Participant	Kappa	Accuracy
06	uaic2021	0.129	0.333
07	IALab_PUC	0.120	0.401
08	KDE-lab	0.117	0.382
<b>09</b>	<b>JBTTM</b>	0.038	0.221
10	Zhao_Shi_	0.015	0.380
11	YNUZHOU	-0.008	0.385

## 5. Conclusion

The crux of the JBTTM's submission is based on simple and shallow neural networks (with an input layer, single hidden layer and an output layer). Other model were such as the 3D CNN model were also experimented. They weren't selected due to their low accuracy. The team's submission placed it ninth out of a total of eleven participant teams. A rigorous assessment of the submission showed that the model can be improved by adding more meaningful layers and/or adding more neurons per layer in such a way that the model doesn't become intractable. When compared to previous year's results, the submission of JBTTM and other teams shows a steady improvement in the accuracy.

## References

- [1] N. Fogel, Tuberculosis: A disease without boundaries (2015).
- [2] Stop tb partnership - fact sheet: The missing 3 million (2019). URL: <http://www.stoptb.org/assets/documents/resources/factsheets/StopTBinfographicMissing3Million.pdf>.
- [3] S. Assili, A review of tomographic reconstruction techniques for computed tomography <https://arxiv.org/abs/1808.09172> (2018).
- [4] B. Ionescu, H. Muller, R. Peteri, A. Ben Abacha, M. Sarrouti, D. Demner-Fushman, S. A. Hasan, S. Kozlovski, V. Liauchuk, Y. Dicente, V. Kovalev, O. Pelka, A. G. S. de Herrera, J. Jacutprakart, C. M. Friedrich, R. Berari, A. Tauteanu, D. Fichou, P. Brie, M. Dogariu, L. D. Ștefan, M. G. Constantin, J. Chamberlain, A. Campello, A. Clark, T. A. Oliver, H. Moustahfid, A. Popescu, J. Deshayes-Chossart, Overview of the ImageCLEF 2021: Multimedia retrieval in medical, nature, internet and social media applications, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction, Proceedings of the 12th International Conference of the CLEF Association (CLEF 2021), LNCS Lecture Notes in Computer Science, Springer, Bucharest, Romania, 2021.
- [5] S. Kozlovski, V. Liauchuk, Y. Dicente Cid, V. Kovalev, H. Müller, Overview of ImageCLEFt-tuberculosis 2021 - CT-based tuberculosis type classification, in: CLEF2021 Working Notes, CEUR Workshop Proceedings, CEUR-WS.org <<http://ceur-ws.org>>, Bucharest, Romania, 2021.
- [6] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, T. Lillicrap, A simple neural network module for relational reasoning, arXiv preprint arXiv:1706.01427 (2017).
- [7] What is numpy? <https://numpy.org/doc/stable/user/whatisnumpy.html> (2021).
- [8] Nibabel access a cacophony of neuro-imaging file formats <https://nipy.org/nibabel/> (2016).
- [9] A. Amanatiadis, I. Andreadis, A survey on evaluation methods for image interpolation, Measurement Science and Technology 20 (2009) 104015.
- [10] H. Zunair, 3d image classification from ct scans [https://keras.io/examples/vision/3D\\_image\\_classification](https://keras.io/examples/vision/3D_image_classification) (2020).

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 128, 128, 64, 1)]	0
conv3d_4 (Conv3D)	(None, 126, 126, 62, 64)	1792
max_pooling3d_4 (MaxPooling3D)	(None, 63, 63, 31, 64)	0
batch_normalization_4 (Batch Normalization)	(None, 63, 63, 31, 64)	256
conv3d_5 (Conv3D)	(None, 61, 61, 29, 64)	110656
max_pooling3d_5 (MaxPooling3D)	(None, 30, 30, 14, 64)	0
batch_normalization_5 (Batch Normalization)	(None, 30, 30, 14, 64)	256
conv3d_6 (Conv3D)	(None, 28, 28, 12, 128)	221312
max_pooling3d_6 (MaxPooling3D)	(None, 14, 14, 6, 128)	0
batch_normalization_6 (Batch Normalization)	(None, 14, 14, 6, 128)	512
conv3d_7 (Conv3D)	(None, 12, 12, 4, 256)	884992
max_pooling3d_7 (MaxPooling3D)	(None, 6, 6, 2, 256)	0
batch_normalization_7 (Batch Normalization)	(None, 6, 6, 2, 256)	1024
global_average_pooling3d_1 (Global Average Pooling)	(None, 256)	0
dense_2 (Dense)	(None, 512)	131584
dropout_1 (Dropout)	(None, 512)	0
dense_3 (Dense)	(None, 5)	2565
Total params: 1,354,949		
Trainable params: 1,353,925		
Non-trainable params: 1,024		

**Figure 2:** CNN Architecture Summary