

Abstract

The article presents a preprocessing and modeling approach to predict lung cancer using CT scans. The preprocessing steps involve resampling, interpolation, and separation of air regions from lung regions using morphological techniques. The proposed model uses a 3D version of the ResNet model with modifications to address the problem of vanishing gradients. The model was trained and tested using a 5-fold cross-validation approach and achieved high accuracy. However, the model exhibits high variance, which may be addressed by increasing the number of layers and output channels. Overall, the article provides a comprehensive approach to predict lung cancer using CT scans.

Preprocessing

Since the thickness of each image slice is different, it is necessary to make the images uniform by performing resampling and using third-degree polynomial interpolation for the third dimension for all images. For this purpose, the thickness of each image slice, which is available in the DICOM file header, has been used, and the internal algorithm of third-degree polynomial interpolation from the scipy library has been used to perform the resizing operation.

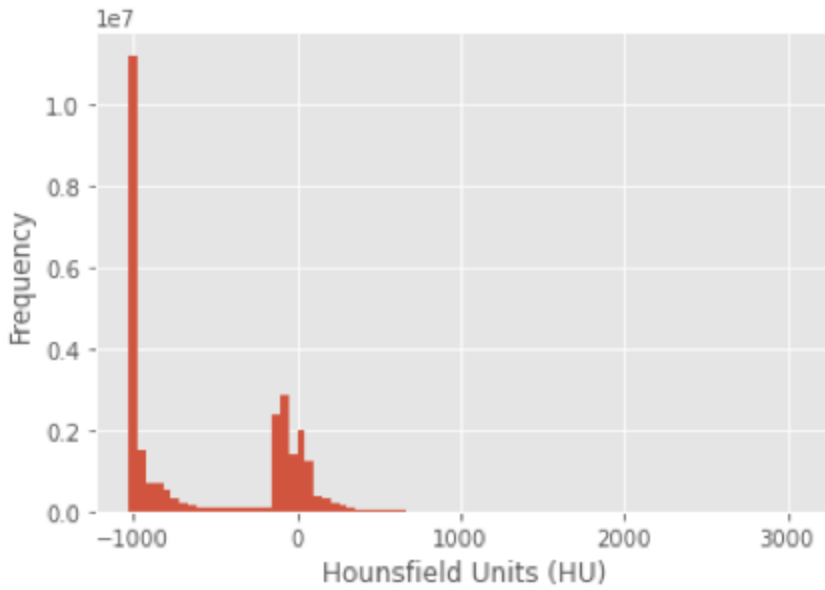
The images in the dataset are in the Hounsfield Unit (HU) scale. This scale is used by CT-Scan devices to create contrast between different tissues in the body. Some of the values of this scale are described in Table 1.

Tissue	HU
Air	1000-
Fat	120-
Water	0
Muscle	40+
Bone	400+

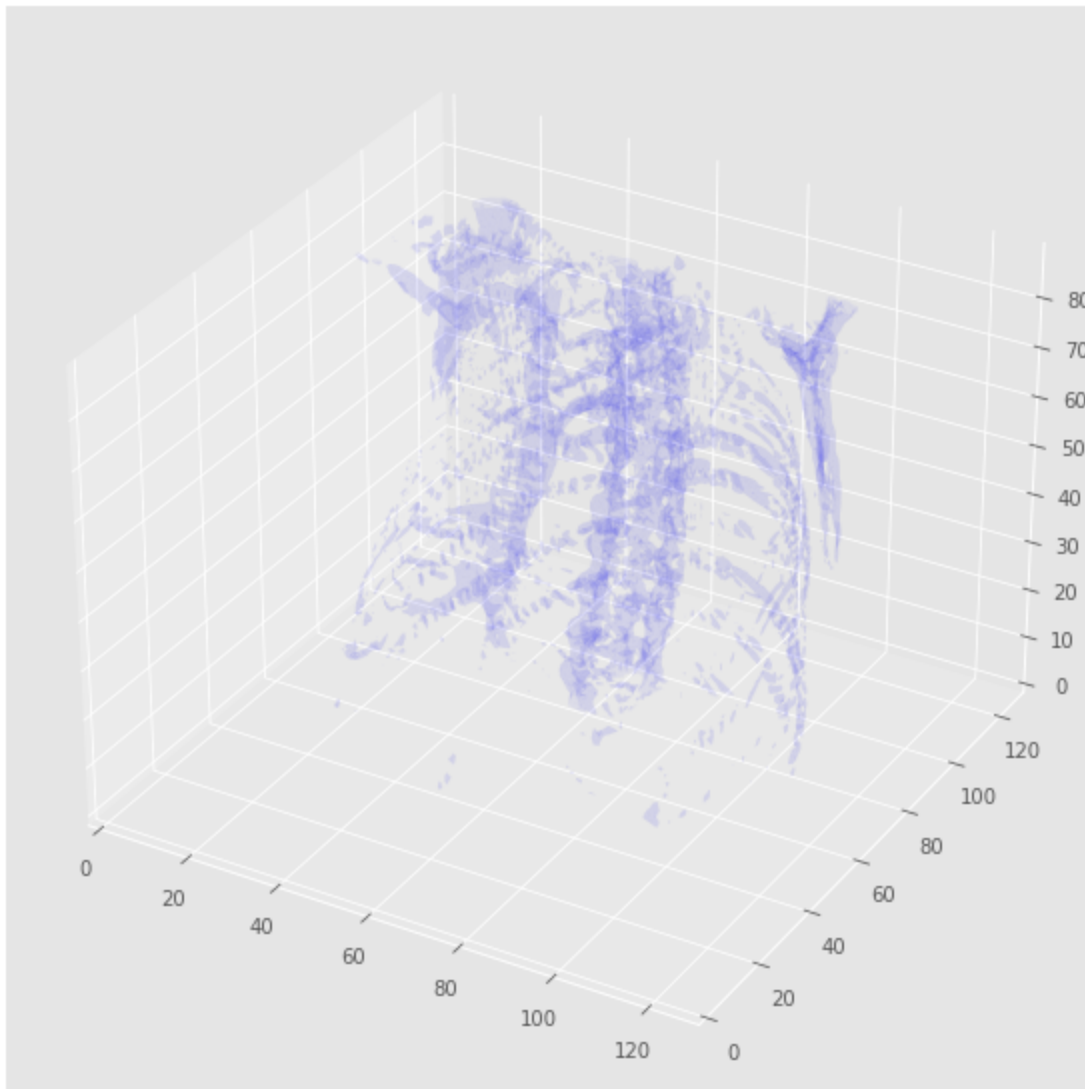
Since the images are obtained by imaging devices from different manufacturers, their output values are different from each other. However, the values can be obtained in the HU scale according to the formula using two parameters, slope and intercept, which are determined by the manufacturer company:

$$hu = \text{pixel_value} * \text{slope} + \text{intercept}$$

Histogram of HU values for patient053 in image 1 shows the presence of many regions of air and also fat.



Using the `marching_cubes` function, a three-dimensional mesh can be extracted from the image and a view of it can be displayed. By separating values greater than 400 and applying the function, a three-dimensional image of the patient's bone structure can be observed. Figure 2 shows the bone structure of sample patient053.

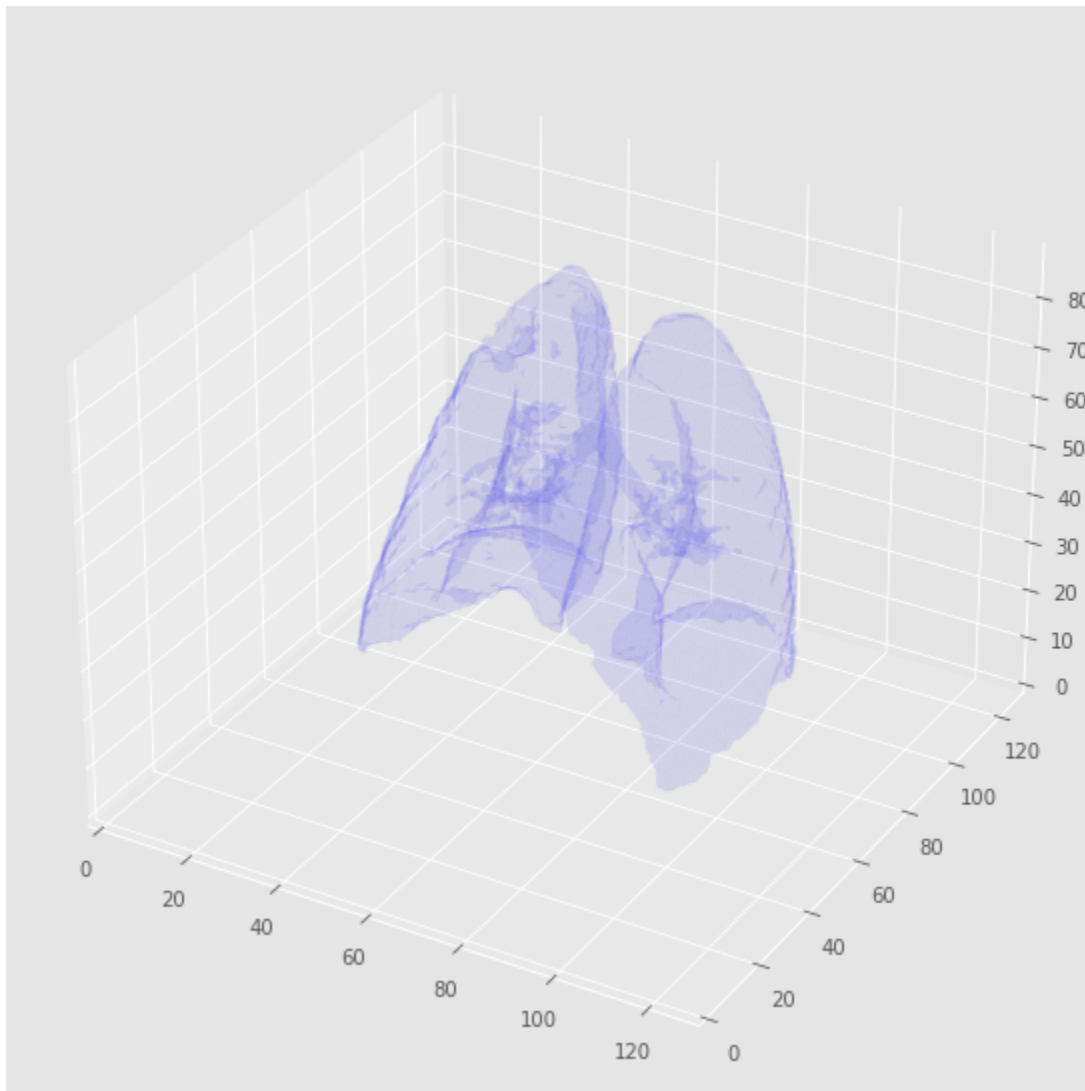


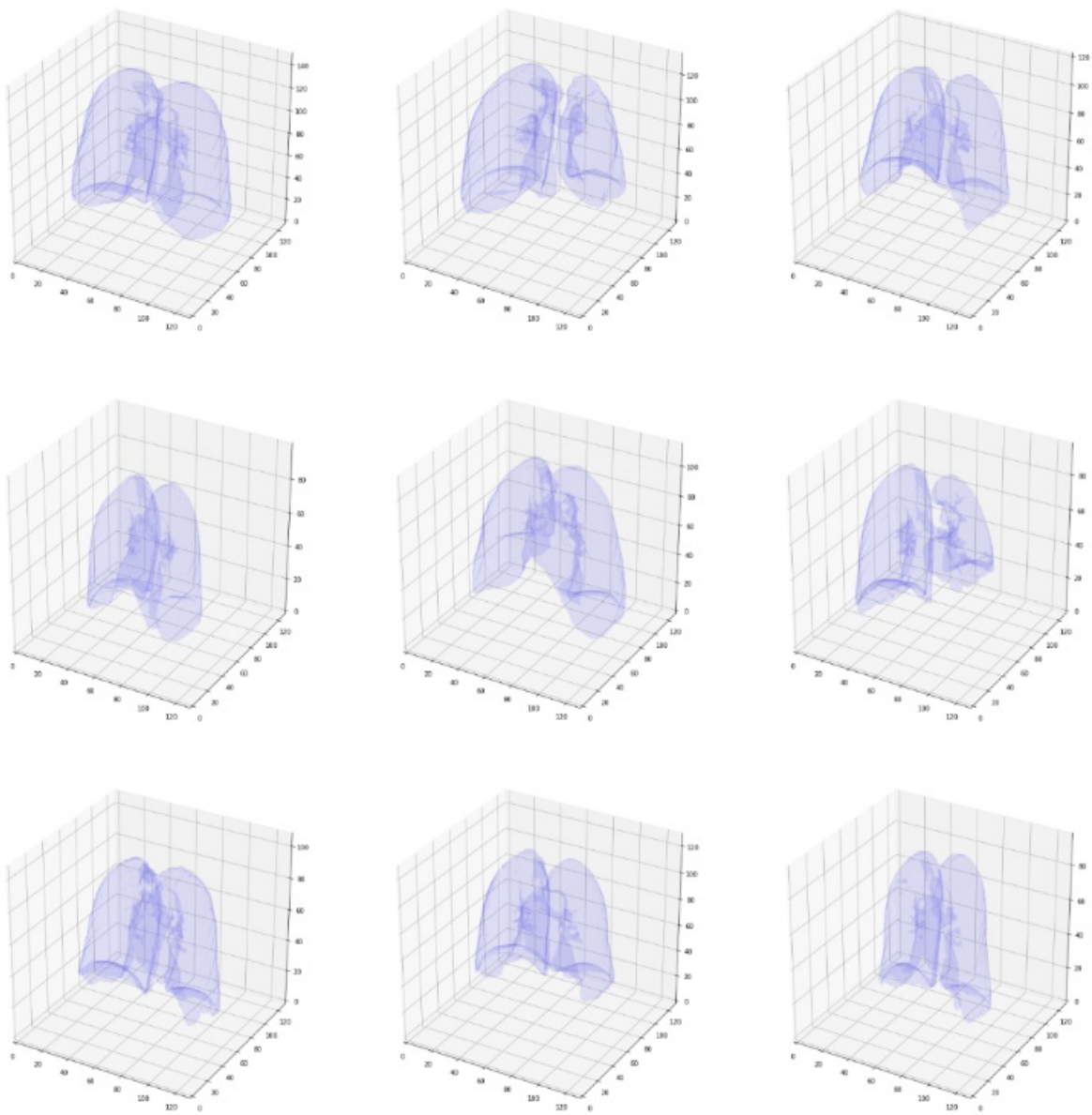
Given that HU values provide valuable information about different tissues, it is possible to separate the lungs and surrounding tissues from the rest of the image using morphological techniques or graph theory. This process can reduce the problem space and greatly facilitate the learning process. The following steps are performed on each image for this purpose:

1. Non-air regions are separated by applying a filter with a threshold value of (greater than HU 320), resulting in a binary image.
2. Connected component analysis is applied to determine the labels of the connected regions.
3. The label of the air surrounding the patient can be obtained from the first pixel in the corner of the three-dimensional image.
4. The air regions surrounding the patient are changed to a value of 0 in the binary image.
5. Now, only the air inside the lungs has a value of 1 in the binary image.

6. In each axial slice, the largest connected region is identified (including tissue and air surrounding the patient), and the remaining regions are set to 0.
7. Only the largest region with a value of 0 is identified and separated from the image.

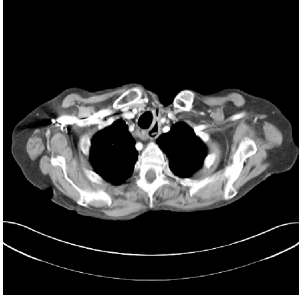
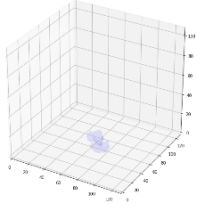
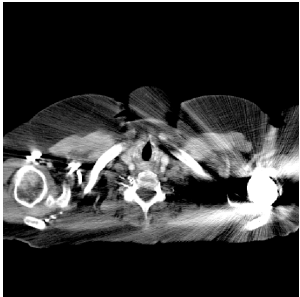
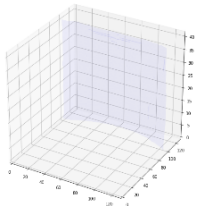
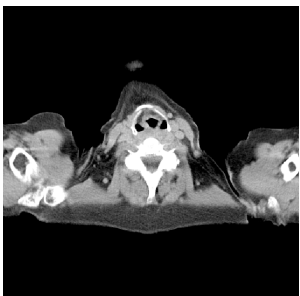
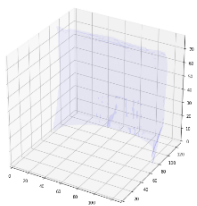
After performing the above steps, the internal regions of the lung are separated and the remaining regions are removed from the image. Image 3 shows this process for the patient053 sample. In this image, it can be seen that the algorithm has effectively separated the lung and its internal space from the rest of the image. Image 4 shows the result of running this algorithm on other samples.





Although this algorithm did not yield satisfactory results for three samples (019_Patient_172, Patient, and 267_Patient) due to positioning issues in the images or excessive noise in the slices. These three samples were removed from the dataset used. The results of running this algorithm for these three samples can be seen in Table 2.

Sample	Slice	Segmentation result
--------	-------	---------------------

Patient_019		
Patient_172		
Patient_267		

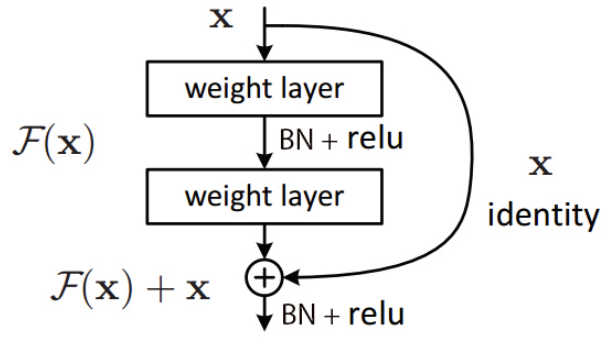
Finally, by using padding, all images will be resized to $128 \times 128 \times 128$.

Model

The model used in this project for prediction is the ResNet model, which won the ImageNet competition in 2015, with the difference that 3D convolutional layers are used in this project. Due to the use of residual blocks, batch normalization layer, and ReLu activation function, despite using a very large number of layers, the network will not face vanishing gradient problem. The details of the layers used in the proposed model can be seen in Table 3. Also, to use the class activation map algorithm, it is necessary to use a global average pooling layer directly after the cnn layers, which will convert the data dimensions to the number of output channels. The input to the network is 3D images with dimensions of $128 \times 128 \times 128$ and a single channel, which will be reduced to dimensions of $8 \times 8 \times 8$ and 128 channels throughout the network. However, due to time and hardware constraints, it is not feasible to increase the number of layers and channels too much.

Layer	Output size	Filter size	Followed by
Conv3d	16 x 64 x 64 x 64	7 x 7 stride 2	Batch_norm + ReLu
ResBlock3d	16 x 64 x 64 x 64	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	16 x 64 x 64 x 64	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	16 x 64 x 64 x 64	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	32 x 32 x 32 x 32	3 x 3 stride 2	Batch_norm + ReLu
ResBlock3d	32 x 32 x 32 x 32	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	32 x 32 x 32 x 32	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	32 x 32 x 32 x 32	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	64 x 16 x 16 x 16	3 x 3 stride 2	Batch_norm + ReLu
ResBlock3d	64 x 16 x 16 x 16	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	64 x 16 x 16 x 16	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	64 x 16 x 16 x 16	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	128 x 8 x 8 x 8	3 x 3 stride 2	Batch_norm + ReLu
ResBlock3d	128 x 8 x 8 x 8	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	128 x 8 x 8 x 8	3 x 3 stride 1	Batch_norm + ReLu
ResBlock3d	128 x 8 x 8 x 8	3 x 3 stride 1	Batch_norm + ReLu
Average Pooling	128	8 x 8 stride 8	
Fully connected	2		

Each ResBlock3d layer consists of two conv3d layers, and their final outputs are added to the input of the block. The operation of this block can be seen in Figure 5 of the ResNet3D block.



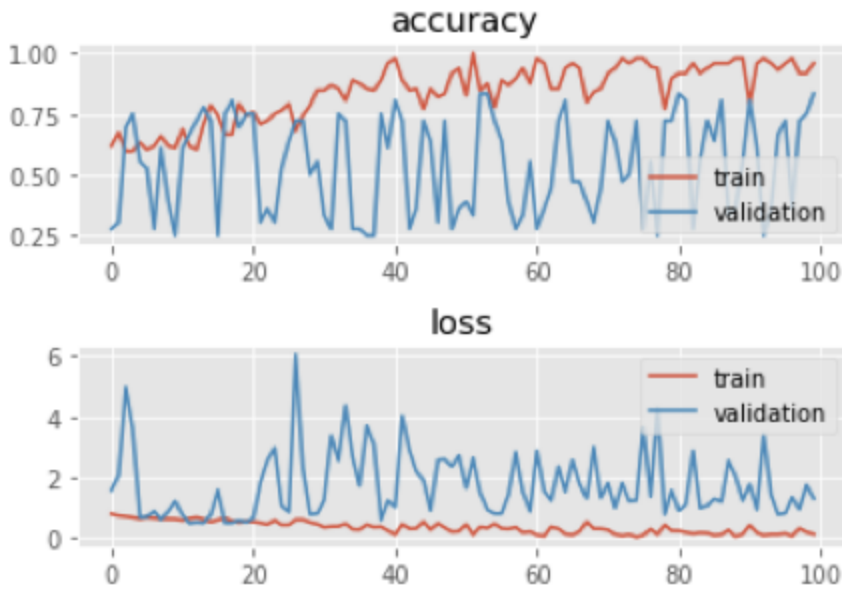
In each ResBlock, a dropout layer with a rate of 0.2 is used after the ReLu activation function.

Results

The proposed model was implemented using the PyTorch library and trained on a system with an NVIDIA GeForce RTX 2060 GPU. To investigate overfitting, a 5-fold cross-validation approach was used, and the maximum accuracy for each fold is shown in Table 4. The model was trained for 100 epochs using the cross-entropy loss function and the Adam optimizer.

.Fold no	Maximum validation accuracy
1	91.6
2	83.3
3	83.3
4	83.3
5	91.6
average	86.7

20% of the data was separated as test data and finally, the model was used to make predictions on this test data, which was trained with the remaining 80% of the data. The error and accuracy graphs of the predictions during training and evaluation of the test data can be seen in Figure 6.



The final performance report of the model on the test data is shown in Figure 7.

	precision	recall	f1-score	support
0	1.00	0.33	0.50	3
1	0.80	1.00	0.89	8
accuracy			0.82	11
macro avg	0.90	0.67	0.69	11
weighted avg	0.85	0.82	0.78	11

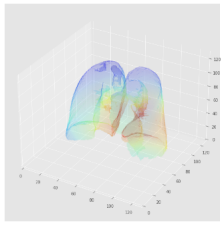
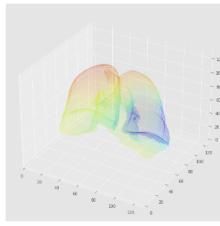
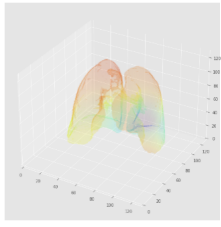
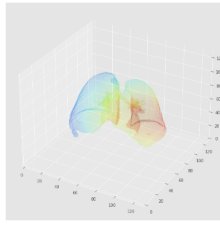
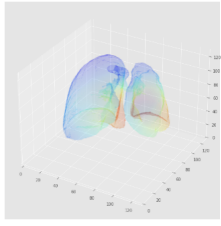
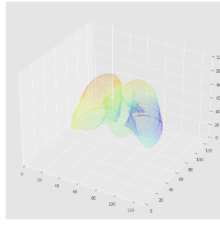
By examining the accuracy and loss plots, it can be observed that the proposed model has a very high variance. Addressing this issue requires further investigation and work on the network. Increasing the number of residual layers and output channels can lead to a more stable model. Furthermore, not using fully connected layers for non-linear classification (due to the use of the CAM algorithm) has been effective in creating this variance. However, in general, it can be seen that the network has performed well in reducing bias.

Class activation map

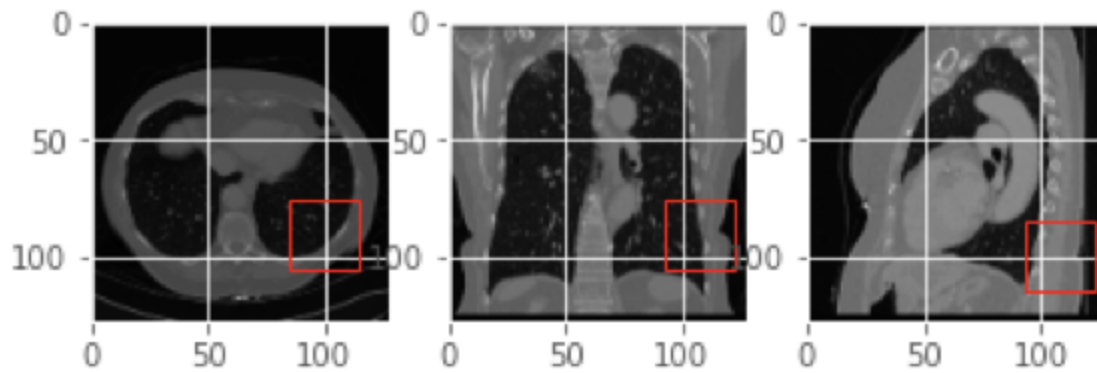
By executing the CAM algorithm, we can obtain a general view of which parts of the image the network pays more attention to in order to assign samples to each class. This algorithm first retrieves the weights of the fc layer for each class of the model. When a sample enters the model and its class is determined, the weights of the last layer corresponding to that class are multiplied in the output of the average pool layer. The resulting vector is used to calculate the

weighted average of the convolutional layers. This creates an image in the size of the output of the convolutional layers. By resizing this image to the size of the original image and converting it to a heatmap, we can see which parts of the image the network has given more weight to in determining the class of the sample.

Given that the images used in this model are three-dimensional, it was necessary to use a three-dimensional coloring technique. The three-dimensional shapes are rendered into small triangles and their arrangement in space results in the final volume. The general idea is to determine the center of each triangle and extract its color from the image obtained from the CAM algorithm. The output of this method is visible for 5 random samples in Table 5.

Patient	Predicted	CAM model	Patient	Predicted	CAM model
Patient_043	1		Patient_208	0	
Patient_175	0		Patient_075	1	
Patient_013	1		Patient_191	0	

Based on the obtained results, it can be seen that the network pays more attention to the lung wall and in some cases its upper part for detecting each class. By finding the maximum value in the heatmap and determining its location, more accurate information can be obtained about the area of interest by creating sagittal, axial, and coronal cuts. Image 8 displays these three cuts for sample 043_Patient.



Future work

To improve the performance and reduce the variance of the model, newer architectures such as DensNet can be used, as well as increasing the depth of the network and the number of channels. It is also possible to improve the model accuracy by removing the dictated architecture for running the CAM algorithm. Another potential method that may have a significant impact on model performance is to use attention layers between residual blocks.