# Covid -19 diagnosis with 3D CNN and Lung -CT segmentation

Fardeen Hasib Mozumder

ID : 1018062288

## Introduction

- CT can play a vital role in diagnosis of covid-19

Advantages:

- Rt-PCR has low accuracy (60-70)% and requires some time, whereas diagnosis from CT scan is faster, which is crucial to mange the pandemic.

Cons:

- Features are not well distinguished from CAP (community acquired pneumonia)

Features:

- Bilateral ground glass opacities and consolidation

## Data

- Using the default dataset SPGC covid dataset
- The dataset is highly non-uniform
- Very high number of Covid positive samples (170 samples)in opposite to non covid samples (CAP : 60 samples or Healthy : 76 samples)
- Structural difference of images for different patients ( ie : difference in number of slices, Hounsfield Unit range, Pixel spacing, slice thickness )

## Data Preprocessing

Data Splitting :

- 55 number of samples from each category for training process( 40 for training & 15 for validation)
- 5 from each category for testing
- (So, total 60 from each category, as CAP has minimum number of samples of 60)

Tackling non-uniformity in Data:

- Hu range was set to -1000 to 400 ( as, air hu =~ -1000 and lung tissue hu =~ 300 addition 100 accounts for infection and blood clots ), considering , values outside the range are not of interest diagnosing covid.

## Data Preprocessing

Data Normalization:

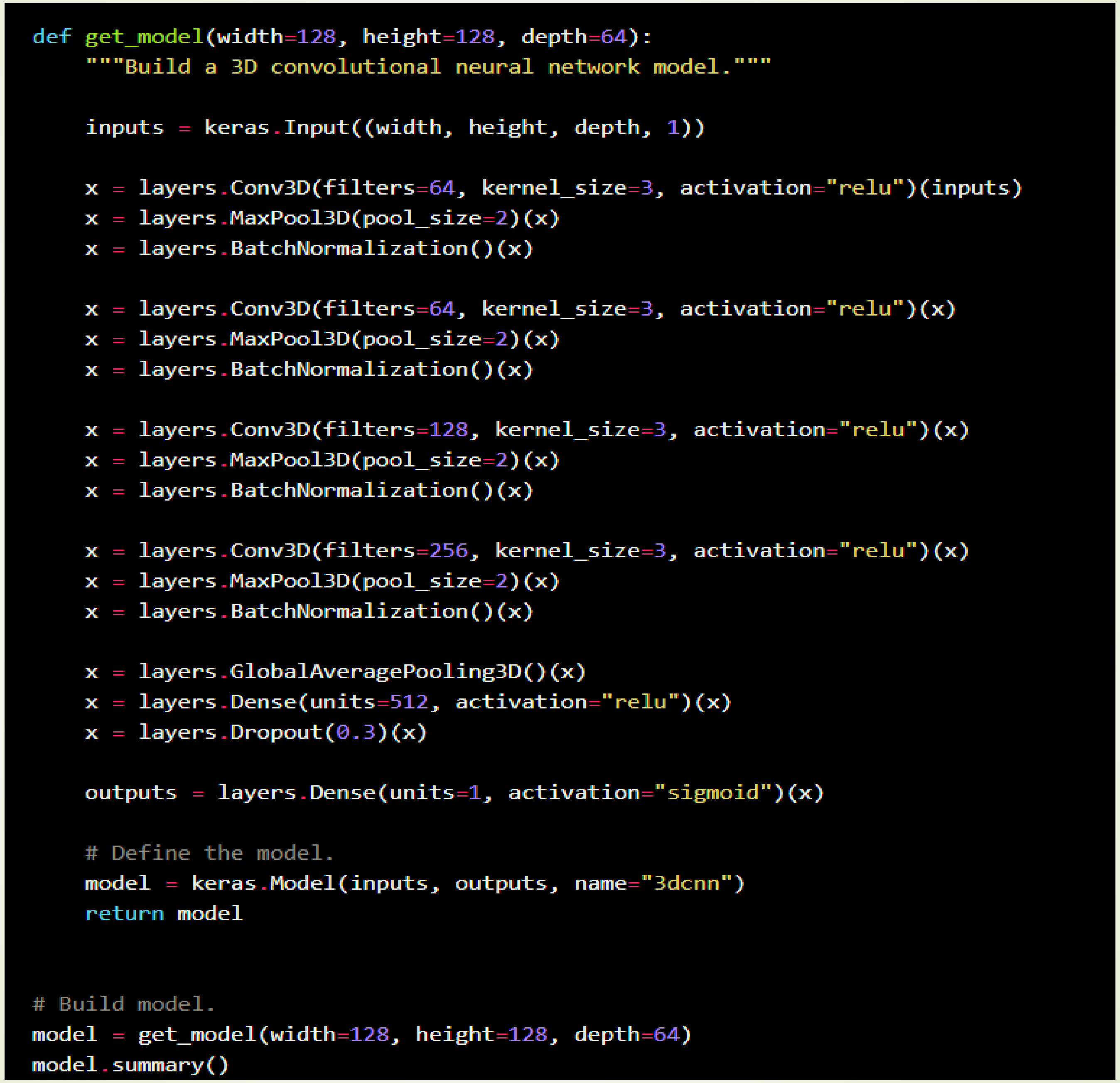
- Hounsfield Unit normalized to 0 to 1.

 [ values closer to 0 represent air or non infected region, values closer to 1 represent infected region]

- Lungs were  segmented and cropped to reduce loss (details in later slides)
- 3D images was resized to (50,128,128) before passing to training

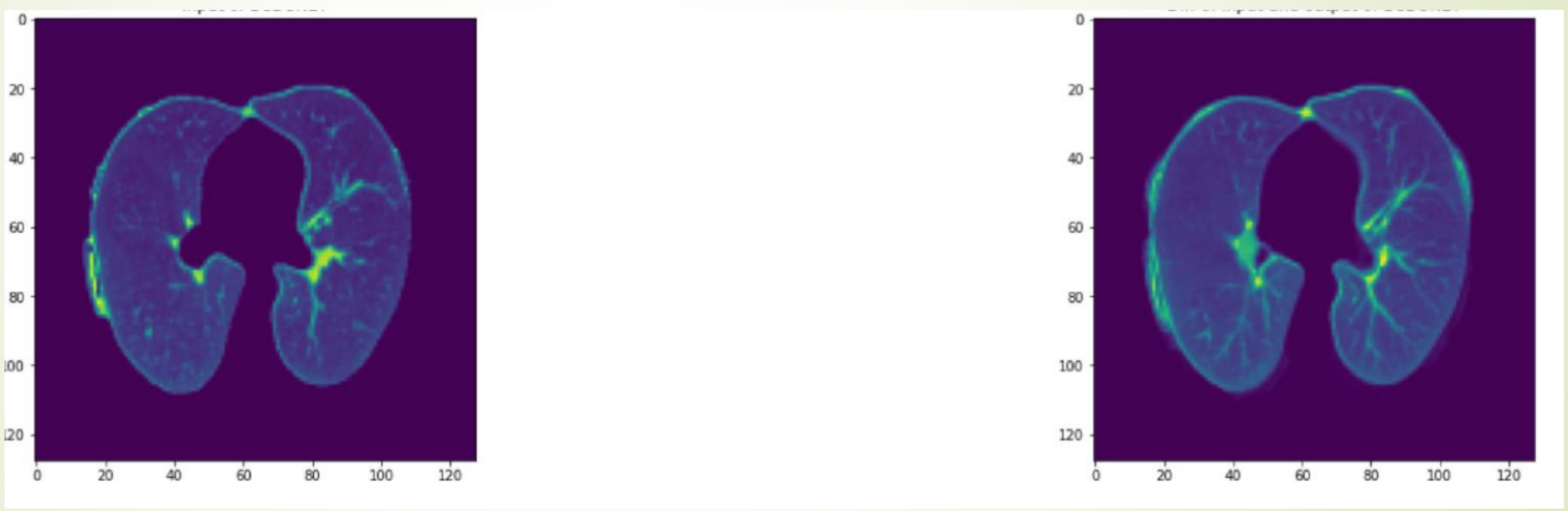
## Methodology:
## 3D CNN

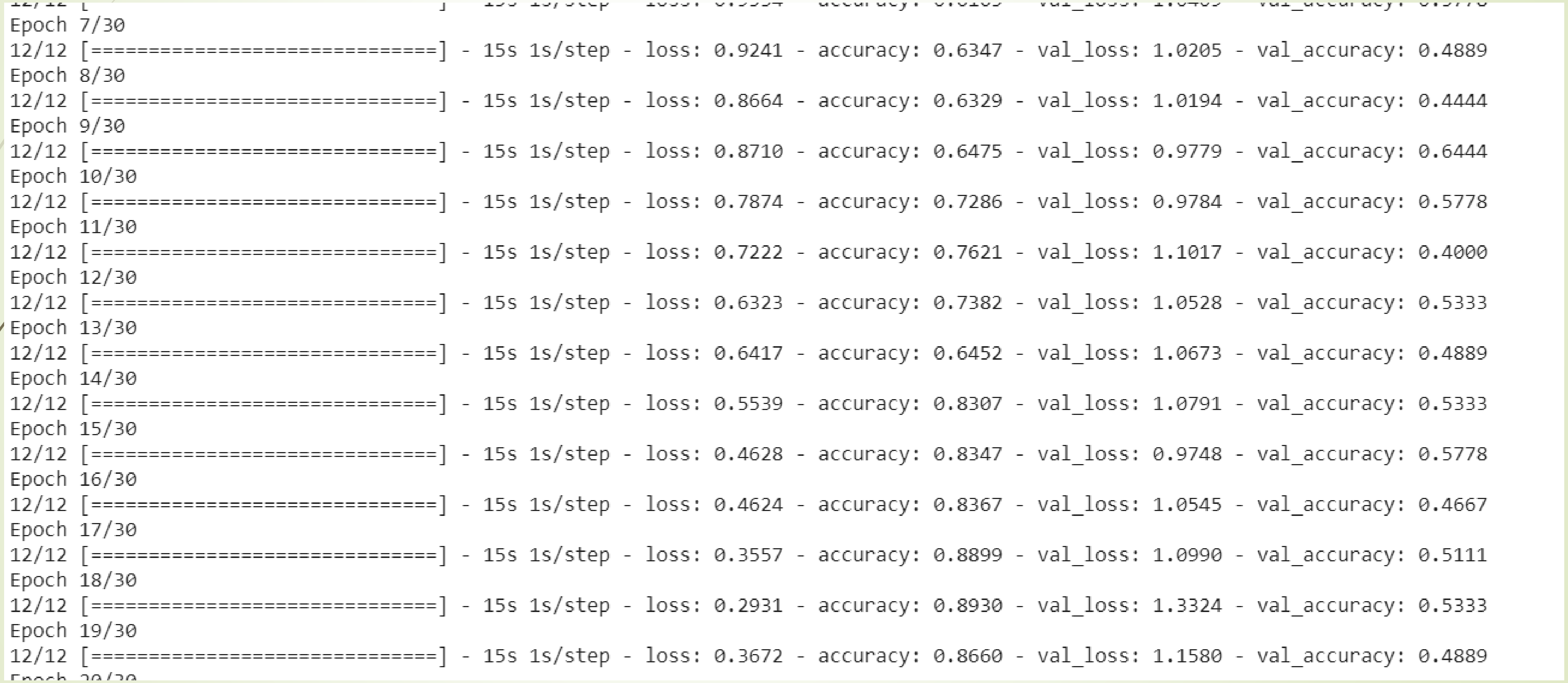
- Preprocessed data is passed to a 3D CNN model with ( 4 segments of of 2 3D CNN models followed by a 3D maxpooling layer)
- The kernel size is of 3d CNN layer is (3,3,3) and the activation function is relu
- The pooling size is (2,2,2,) for the maxpooling layer.
- After these, there are 2 dense layers with node of 32 and 3 .
- Activation  function of last dense layer is softmax as there are 3 classes.
- A dropout layer is used between the dense layer to avoid overfitting



```python
def get_model(width=128, height=128, depth=64):
    """Build a 3D convolutional neural network model."""

    inputs = keras.Input((width, height, depth, 1))

    x = layers.Conv3D(filters=64, kernel_size=3, activation="relu")(inputs)
    x = layers.MaxPool3D(pool_size=2)(x)
    x = layers.BatchNormalization()(x)

    x = layers.Conv3D(filters=64, kernel_size=3, activation="relu")(x)
    x = layers.MaxPool3D(pool_size=2)(x)
    x = layers.BatchNormalization()(x)

    x = layers.Conv3D(filters=128, kernel_size=3, activation="relu")(x)
    x = layers.MaxPool3D(pool_size=2)(x)
    x = layers.BatchNormalization()(x)

    x = layers.Conv3D(filters=256, kernel_size=3, activation="relu")(x)
    x = layers.MaxPool3D(pool_size=2)(x)
    x = layers.BatchNormalization()(x)

    x = layers.GlobalAveragePooling3D()(x)
    x = layers.Dense(units=512, activation="relu")(x)
    x = layers.Dropout(0.3)(x)

    outputs = layers.Dense(units=1, activation="sigmoid")(x)

    # Define the model.
    model = keras.Model(inputs, outputs, name="3dcnn")
    return model


# Build model.
model = get_model(width=128, height=128, depth=64)
model.summary()
```

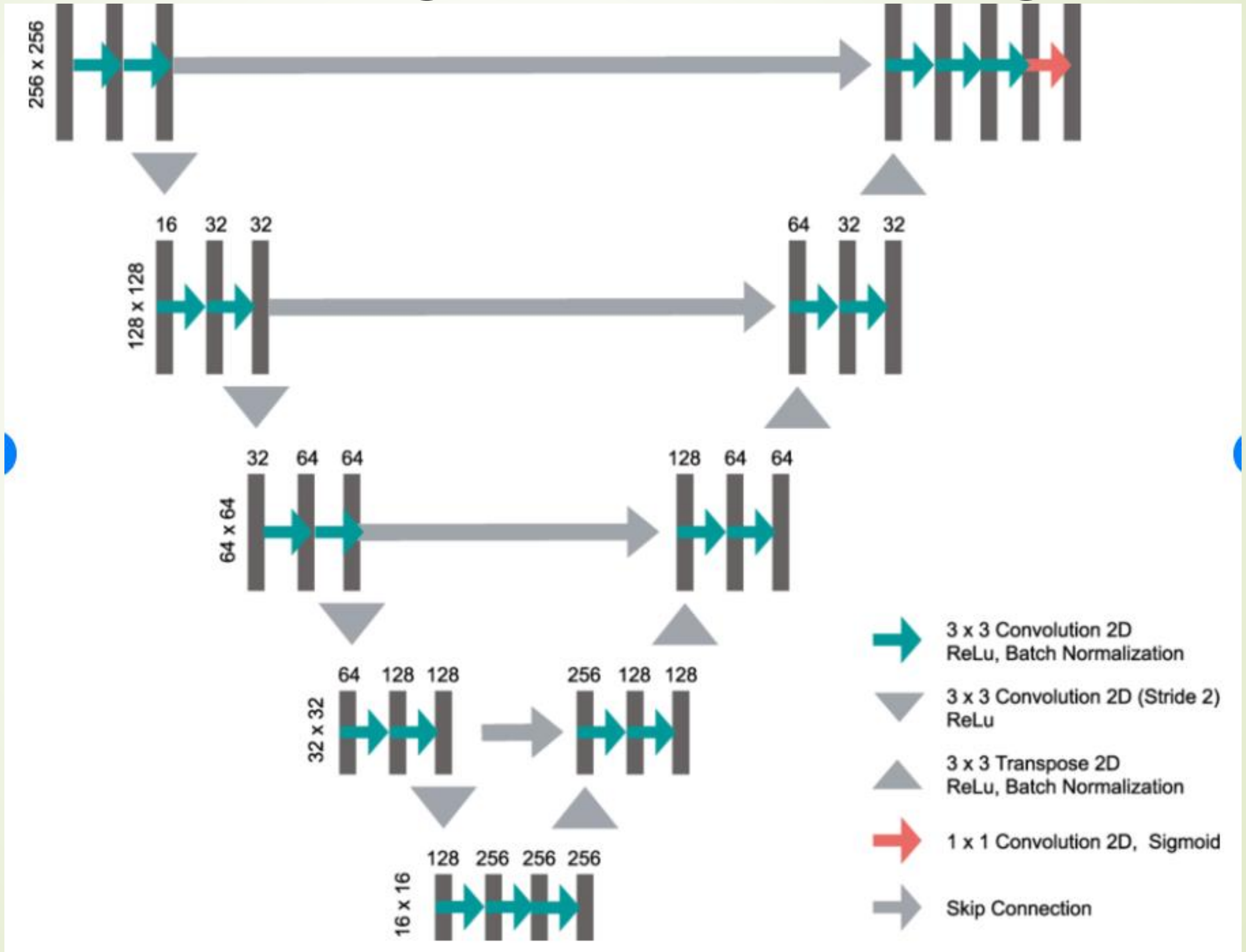
# Methodology:
3D CNN



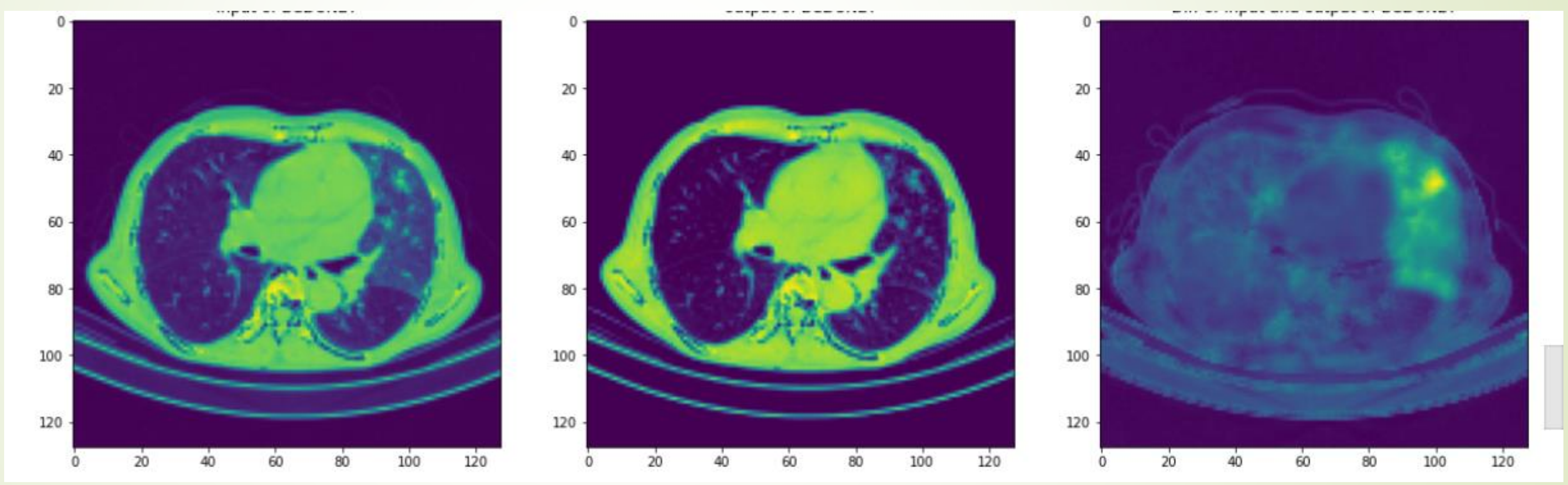


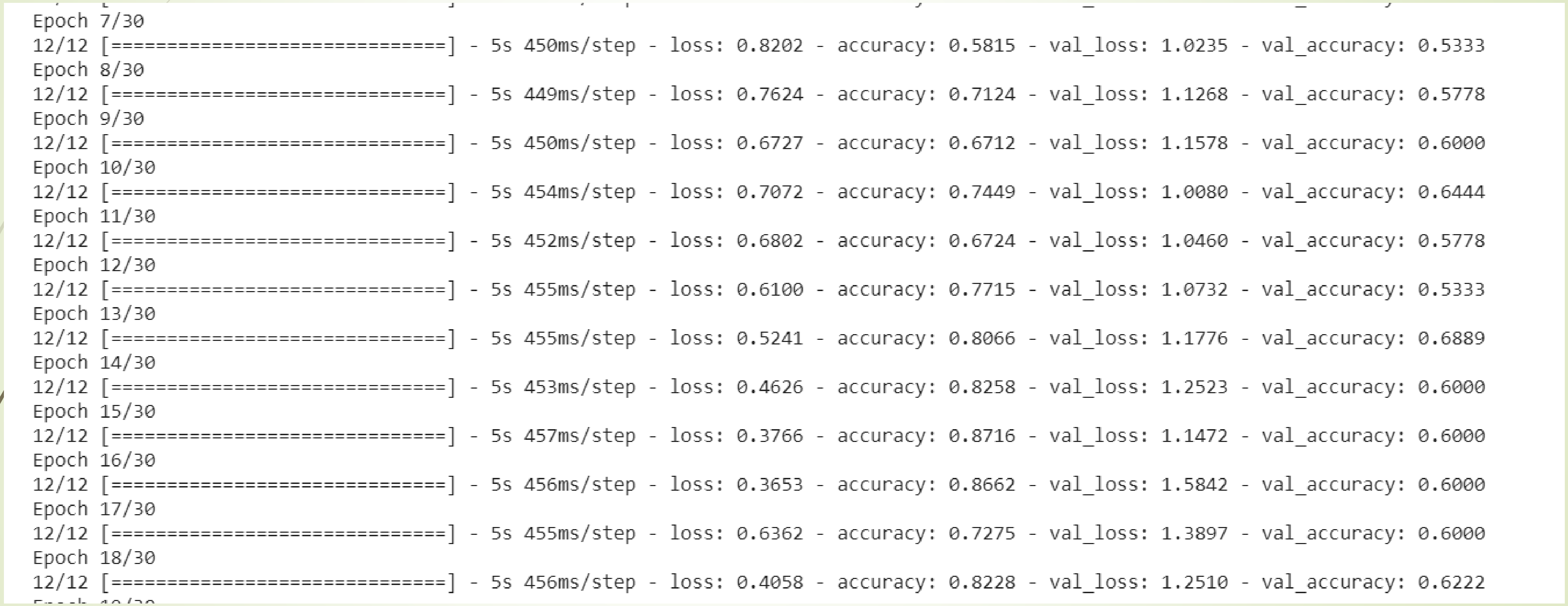
```
Epoch 7/30
12/12 [==============================] - 15s 1s/step - loss: 0.9241 - accuracy: 0.6347 - val_loss: 1.0205 - val_accuracy: 0.4889
Epoch 8/30
12/12 [==============================] - 15s 1s/step - loss: 0.8664 - accuracy: 0.6329 - val_loss: 1.0194 - val_accuracy: 0.4444
Epoch 9/30
12/12 [==============================] - 15s 1s/step - loss: 0.8710 - accuracy: 0.6475 - val_loss: 0.9779 - val_accuracy: 0.6444
Epoch 10/30
12/12 [==============================] - 15s 1s/step - loss: 0.7874 - accuracy: 0.7286 - val_loss: 0.9784 - val_accuracy: 0.5778
Epoch 11/30
12/12 [==============================] - 15s 1s/step - loss: 0.7222 - accuracy: 0.7621 - val_loss: 1.1017 - val_accuracy: 0.4000
Epoch 12/30
12/12 [==============================] - 15s 1s/step - loss: 0.6323 - accuracy: 0.7382 - val_loss: 1.0528 - val_accuracy: 0.5333
Epoch 13/30
12/12 [==============================] - 15s 1s/step - loss: 0.6417 - accuracy: 0.6452 - val_loss: 1.0673 - val_accuracy: 0.4889
Epoch 14/30
12/12 [==============================] - 15s 1s/step - loss: 0.5539 - accuracy: 0.8307 - val_loss: 1.0791 - val_accuracy: 0.5333
Epoch 15/30
12/12 [==============================] - 15s 1s/step - loss: 0.4628 - accuracy: 0.8347 - val_loss: 0.9748 - val_accuracy: 0.5778
Epoch 16/30
12/12 [==============================] - 15s 1s/step - loss: 0.4624 - accuracy: 0.8367 - val_loss: 1.0545 - val_accuracy: 0.4667
Epoch 17/30
12/12 [==============================] - 15s 1s/step - loss: 0.3557 - accuracy: 0.8899 - val_loss: 1.0990 - val_accuracy: 0.5111
Epoch 18/30
12/12 [==============================] - 15s 1s/step - loss: 0.2931 - accuracy: 0.8930 - val_loss: 1.3324 - val_accuracy: 0.5333
Epoch 19/30
12/12 [==============================] - 15s 1s/step - loss: 0.3672 - accuracy: 0.8660 - val_loss: 1.1580 - val_accuracy: 0.4889
Epoch 20/30
```

# Automatic segmentation using BCDU


256 x 256
16
32
32
64
32
32
128 x 128
32
64
64
128
64
64
64 x 64
64
128
128
256
128
128
32 x 32
128
256
256
256
16 x 16
3 x 3 Convolution 2D ReLu, Batch Normalization
3 x 3 Convolution 2D (Stride 2) ReLu
3 x 3 Transpose 2D ReLu, Batch Normalization
1 x 1 Convolution 2D, Sigmoid
Skip Connection

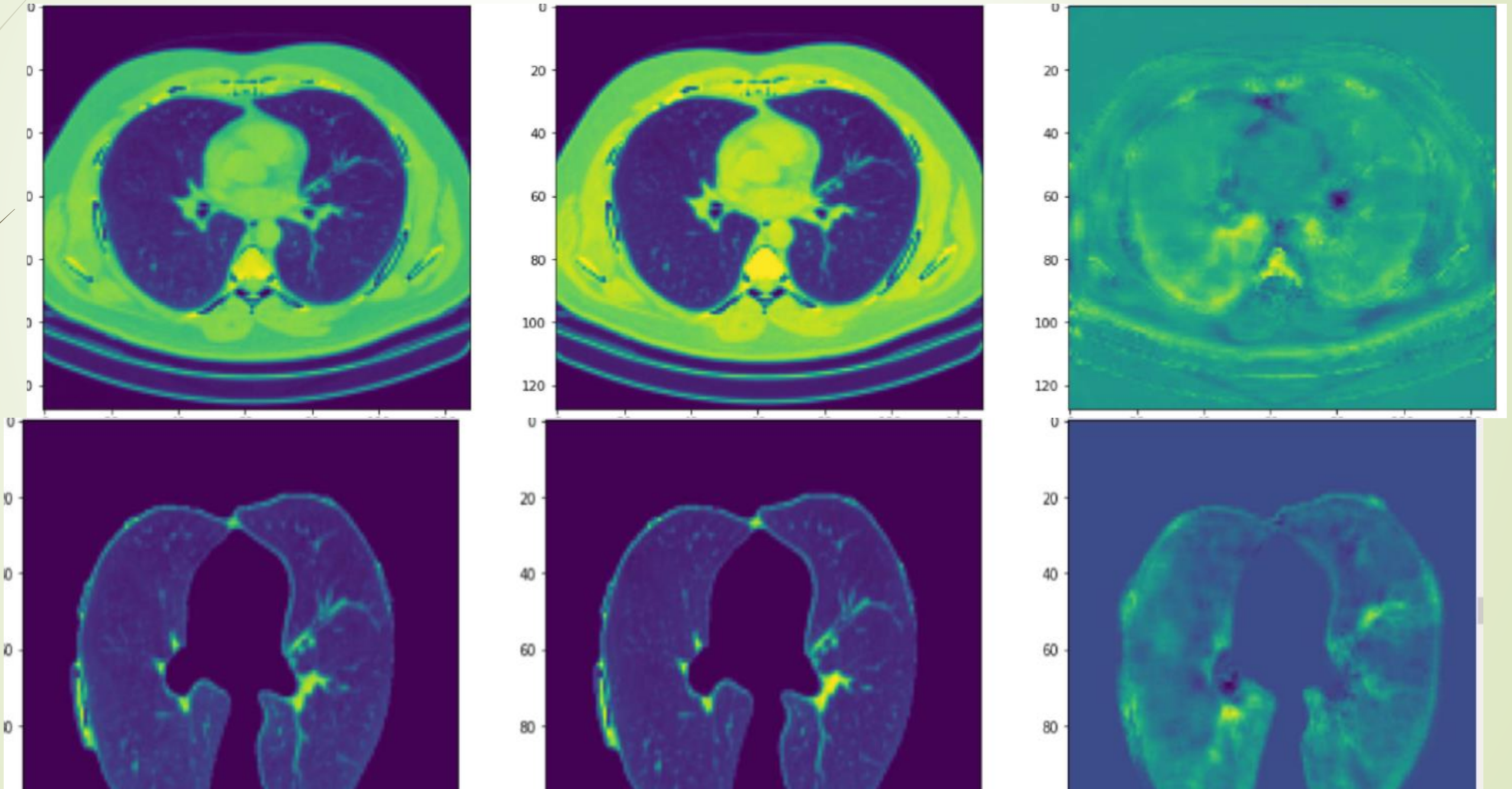
# Effects of BCDU



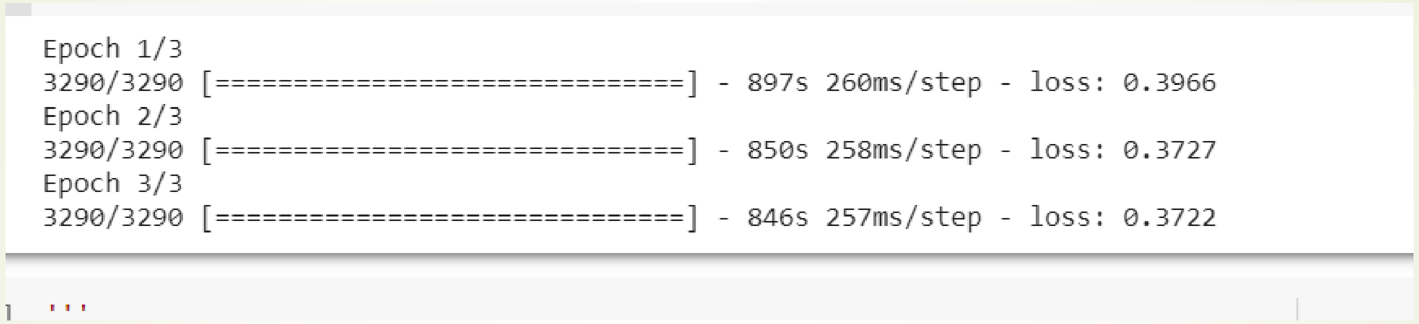


```
Epoch 7/30
12/12 [==============================] - 5s 450ms/step - loss: 0.8202 - accuracy: 0.5815 - val_loss: 1.0235 - val_accuracy: 0.5333
Epoch 8/30
12/12 [==============================] - 5s 449ms/step - loss: 0.7624 - accuracy: 0.7124 - val_loss: 1.1268 - val_accuracy: 0.5778
Epoch 9/30
12/12 [==============================] - 5s 450ms/step - loss: 0.6727 - accuracy: 0.6712 - val_loss: 1.1578 - val_accuracy: 0.6000
Epoch 10/30
12/12 [==============================] - 5s 454ms/step - loss: 0.7072 - accuracy: 0.7449 - val_loss: 1.0080 - val_accuracy: 0.6444
Epoch 11/30
12/12 [==============================] - 5s 452ms/step - loss: 0.6802 - accuracy: 0.6724 - val_loss: 1.0460 - val_accuracy: 0.5778
Epoch 12/30
12/12 [==============================] - 5s 455ms/step - loss: 0.6100 - accuracy: 0.7715 - val_loss: 1.0732 - val_accuracy: 0.5333
Epoch 13/30
12/12 [==============================] - 5s 455ms/step - loss: 0.5241 - accuracy: 0.8066 - val_loss: 1.1776 - val_accuracy: 0.6889
Epoch 14/30
12/12 [==============================] - 5s 453ms/step - loss: 0.4626 - accuracy: 0.8258 - val_loss: 1.2523 - val_accuracy: 0.6000
Epoch 15/30
12/12 [==============================] - 5s 457ms/step - loss: 0.3766 - accuracy: 0.8716 - val_loss: 1.1472 - val_accuracy: 0.6000
Epoch 16/30
12/12 [==============================] - 5s 456ms/step - loss: 0.3653 - accuracy: 0.8662 - val_loss: 1.5842 - val_accuracy: 0.6000
Epoch 17/30
12/12 [==============================] - 5s 455ms/step - loss: 0.6362 - accuracy: 0.7275 - val_loss: 1.3897 - val_accuracy: 0.6000
Epoch 18/30
12/12 [==============================] - 5s 456ms/step - loss: 0.4058 - accuracy: 0.8228 - val_loss: 1.2510 - val_accuracy: 0.6222
```

# Effects of preprocessed lung cropping and segmentation

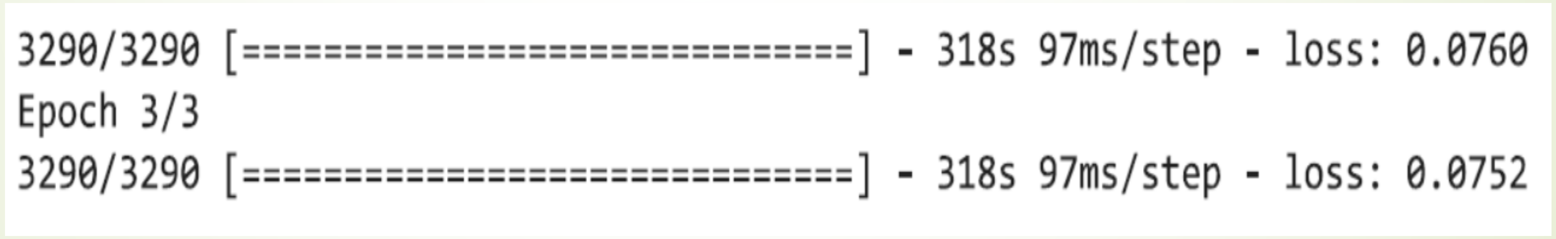


## Effects of preprocessed lung cropping and segmentation



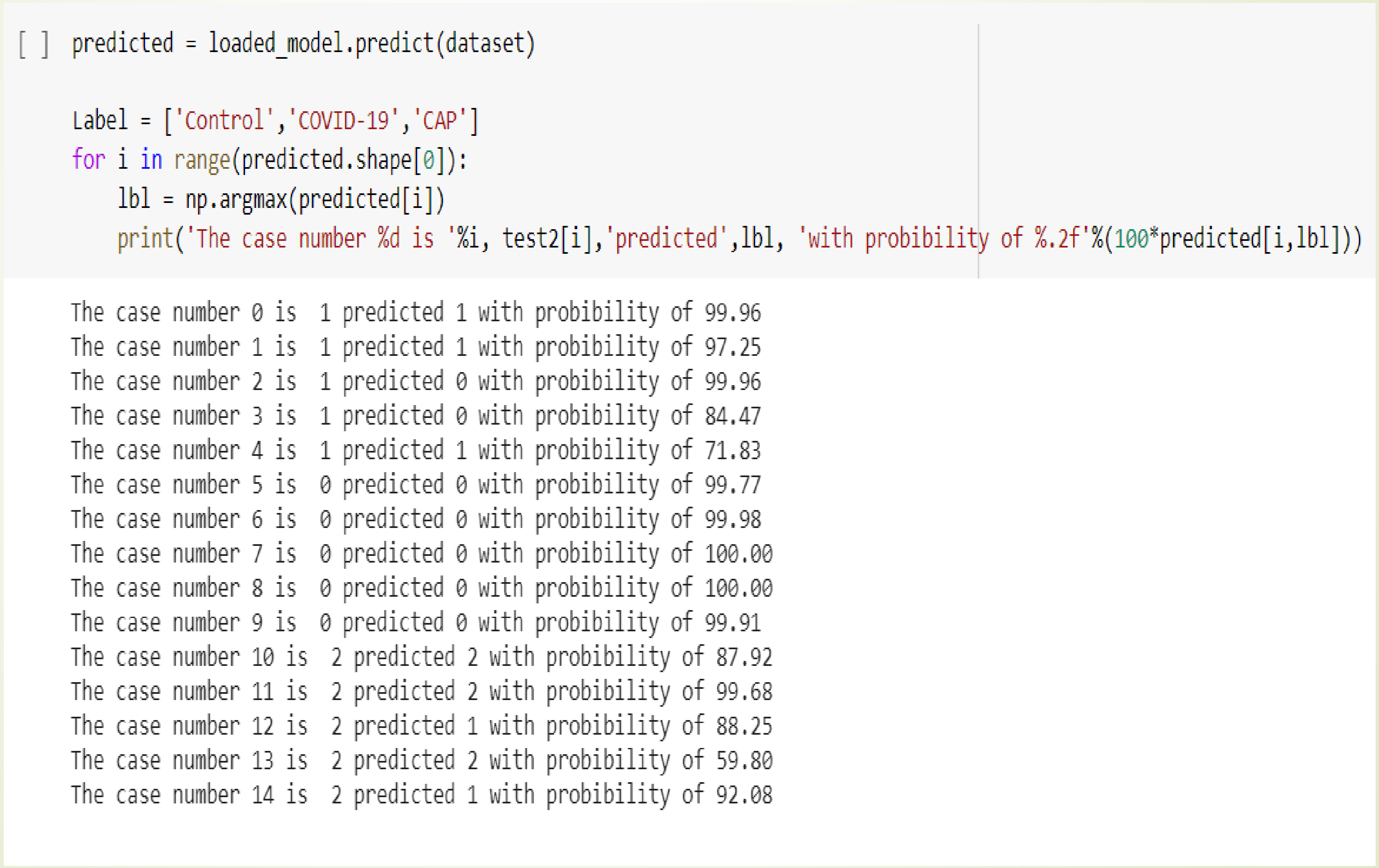
```
Epoch 1/3
3290/3290 [==============================] - 897s 260ms/step - loss: 0.3966
Epoch 2/3
3290/3290 [==============================] - 850s 258ms/step - loss: 0.3727
Epoch 3/3
3290/3290 [==============================] - 846s 257ms/step - loss: 0.3722
```

```
'''
```



```
3290/3290 [==============================] - 318s 97ms/step - loss: 0.0760
Epoch 3/3
3290/3290 [==============================] - 318s 97ms/step - loss: 0.0752
```

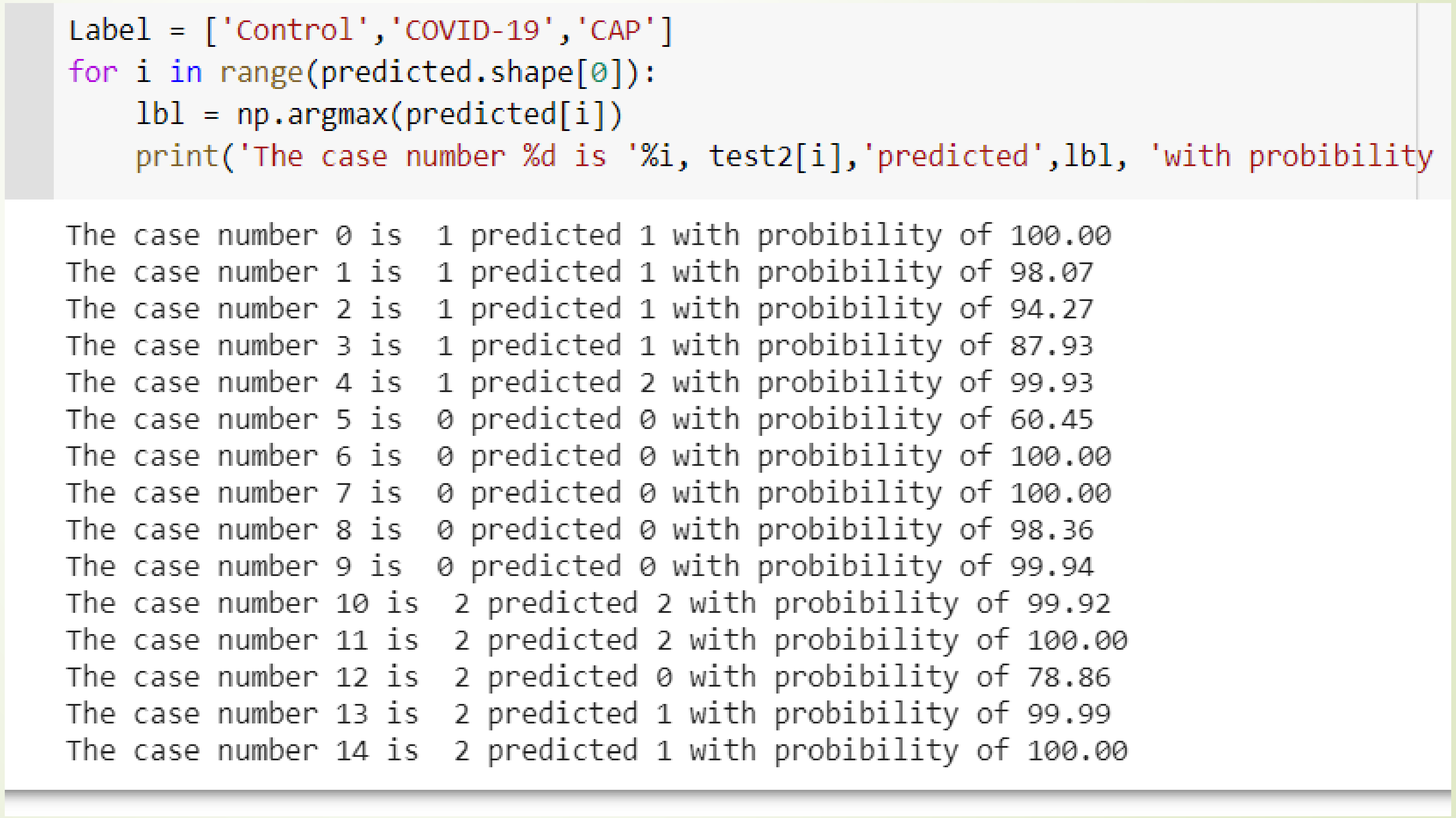
## Testing on uncropped Lungs with BCDU net



```
[ ] predicted = loaded_model.predict(dataset)

    Label = ['Control','COVID-19','CAP']
    for i in range(predicted.shape[0]):
        lbl = np.argmax(predicted[i])
        print('The case number %d is '%i, test2[i],'predicted',lbl, 'with probibility of %.2f'%(100*predicted[i,lbl]))
```

```
    The case number 0 is  1 predicted 1 with probibility of 99.96
    The case number 1 is  1 predicted 1 with probibility of 97.25
    The case number 2 is  1 predicted 0 with probibility of 99.96
    The case number 3 is  1 predicted 0 with probibility of 84.47
    The case number 4 is  1 predicted 1 with probibility of 71.83
    The case number 5 is  0 predicted 0 with probibility of 99.77
    The case number 6 is  0 predicted 0 with probibility of 99.98
    The case number 7 is  0 predicted 0 with probibility of 100.00
    The case number 8 is  0 predicted 0 with probibility of 100.00
    The case number 9 is  0 predicted 0 with probibility of 99.91
    The case number 10 is  2 predicted 2 with probibility of 87.92
    The case number 11 is  2 predicted 2 with probibility of 99.68
    The case number 12 is  2 predicted 1 with probibility of 88.25
    The case number 13 is  2 predicted 2 with probibility of 59.80
    The case number 14 is  2 predicted 1 with probibility of 92.08
```

# Testing on cropped Lungs with BCDU net



```
Label = ['Control','COVID-19','CAP']
for i in range(predicted.shape[0]):
    lbl = np.argmax(predicted[i])
    print('The case number %d is '%i, test2[i],'predicted',lbl, 'with probibility
```

```
The case number 0 is  1 predicted 1 with probibility of 100.00
The case number 1 is  1 predicted 1 with probibility of 98.07
The case number 2 is  1 predicted 1 with probibility of 94.27
The case number 3 is  1 predicted 1 with probibility of 87.93
The case number 4 is  1 predicted 2 with probibility of 99.93
The case number 5 is  0 predicted 0 with probibility of 60.45
The case number 6 is  0 predicted 0 with probibility of 100.00
The case number 7 is  0 predicted 0 with probibility of 100.00
The case number 8 is  0 predicted 0 with probibility of 98.36
The case number 9 is  0 predicted 0 with probibility of 99.94
The case number 10 is  2 predicted 2 with probibility of 99.92
The case number 11 is  2 predicted 2 with probibility of 100.00
The case number 12 is  2 predicted 0 with probibility of 78.86
The case number 13 is  2 predicted 1 with probibility of 99.99
The case number 14 is  2 predicted 1 with probibility of 100.00
```

## Findings

- Segmentation with BCDU net on cropped lungs results in better and more effective diagnosis as it gives fewer false negatives.
- Needs better and more effective segmentation process to differentiate between CAP vs Covid to increase accuracy

## Future work

- Exploring more segmentation networks ( Unet++, Segnet, Linknet, PSPnet etc)
- Exploring more 3D CNN networks suitable for our data.
- Optimizing the threshold levels of HU to increase efficiency
- Resizing our data to bigger unit before passing to networks

## Resource Limitations

- Hardware limitations:

  (GPU, RAM)
- Data Limitation :

  Small Dataset,

  Absence of manually segmented/masked samples to use as  ground truth

Any Questions?