Classification of Lung Cancer nodules from CT scans using Neural Networks

09.01.2024, Coimbra

Munteanu Vadim

Adrian Walczak

01 Dataset

15

Seven academic centers and eight medical imaging companies collaborated to create this data

125gb

1,018

Cases

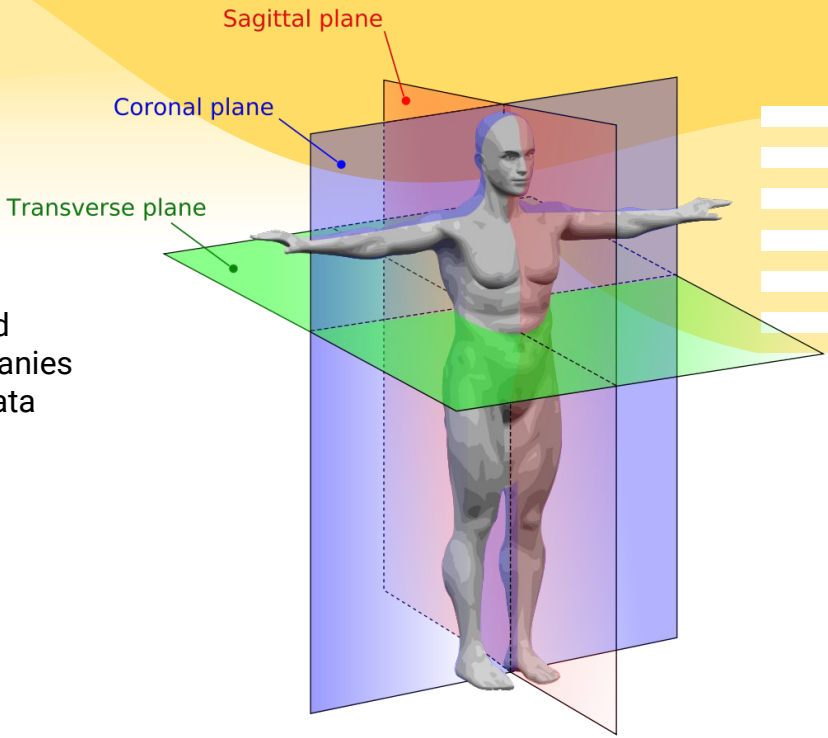


Image source

The Lung Image Database Consortium image collection (LIDC-IDRI) consists of diagnostic and lung cancer screening thoracic computed tomography (CT) scans with marked-up annotated lesions

For more info, click here

Objective

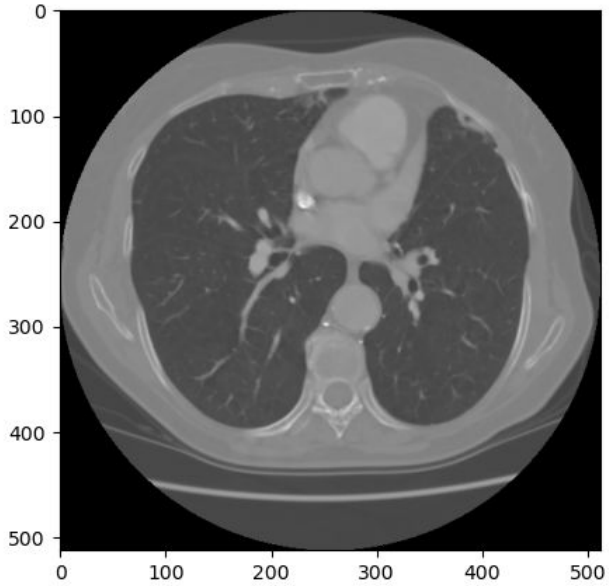
The objective is to use a dataset with CT images and develop a learning model to classify lung nodules as malignant or benign

And to let you visualize it better ...

Initial approach

Creating some kind of “probability” to have cancer .
Threshold for malignancy took 1.5.

$$l = \sigma\left(n_n\left(\frac{< score >}{thrsh} - 1\right)\right)$$



Problems with this approach:

1. **Few data points: 1018 (1 slice per patient)**
2. **Too big images to work on our local machines.**

02 Alternative Approach

1. Extract volumes with nodules
2. Condense annotations In 1 nodule
3. Remove background
4. Extract middle slice Of the nodule
5. Pad images until Desired shape
6. Normalize pixel values

Mean nodule

Each nodule has few annotations. Single data point in our dataset is "mean" annotation which corresponds to a nodule. Associated label is mean malignancy

Details about compressing annotations

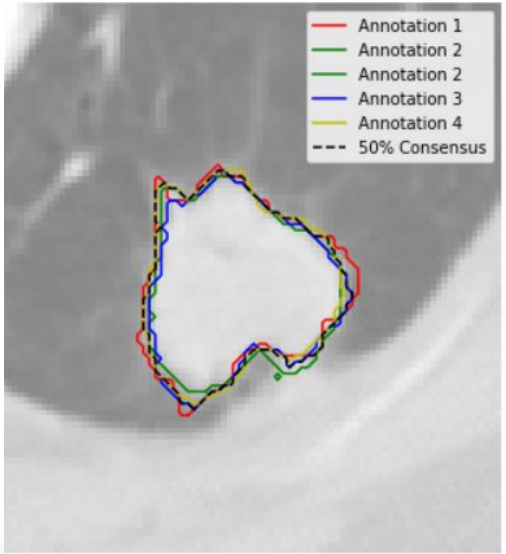
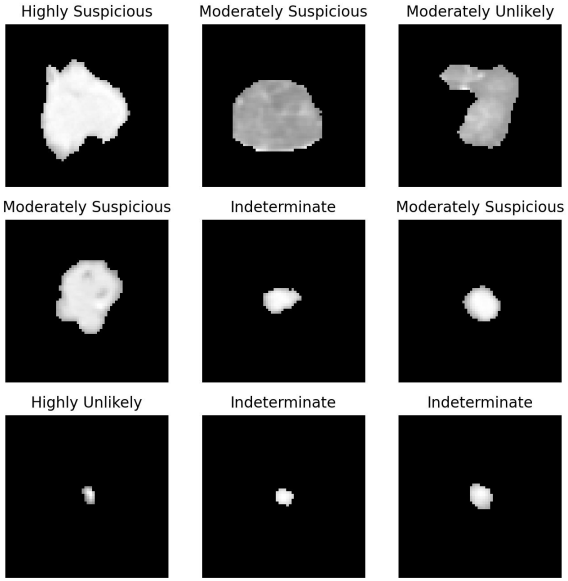


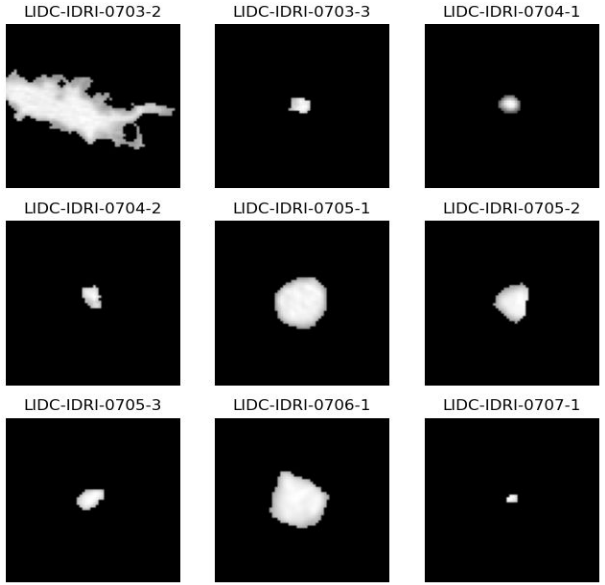
Image source

This approach solves the above mentioned problems:

1. We downgrade the images from 512x512 to 64x64
2. Considering that each patient has >=1 nodules => More data points!

End form of dataset





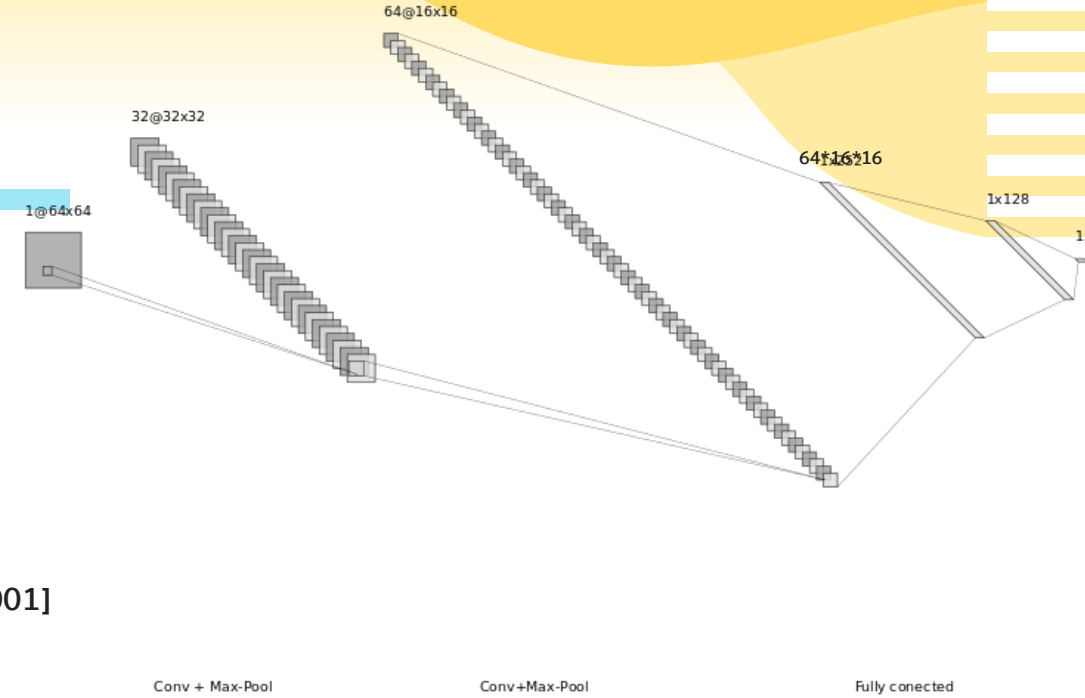

```
malignancy_map = {
        1: 'Highly Unlikely',
        2: 'Moderately Unlikely',
        3: 'Indeterminate',
        4: 'Moderately Suspicious',
        5: 'Highly Suspicious'
        }
```

~2625 images

03 Neural Net structure

Cross Entropy Loss function:

$$H(\vec{p}, \vec{q}) = \sum_i p_i \ln q_i$$

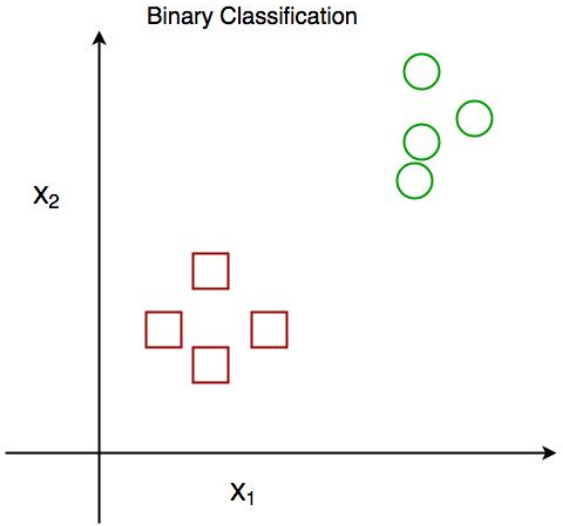


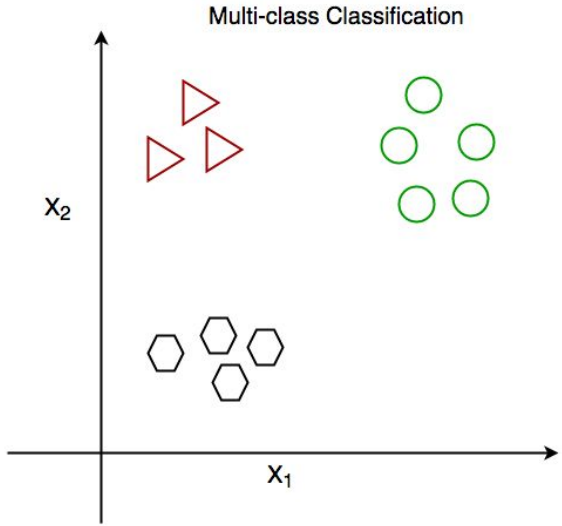
- Learning rates = [0.1, 0.01, 0.001,0.0001,0.00001]
- BATCH_SIZE=64
- optimizer=Adam

Check the great explanation of how cnn works below:

https://www.youtube.com/watch?v=JB8T_zN7ZC0&t=3036s&ab_channel=BrandonRohrer

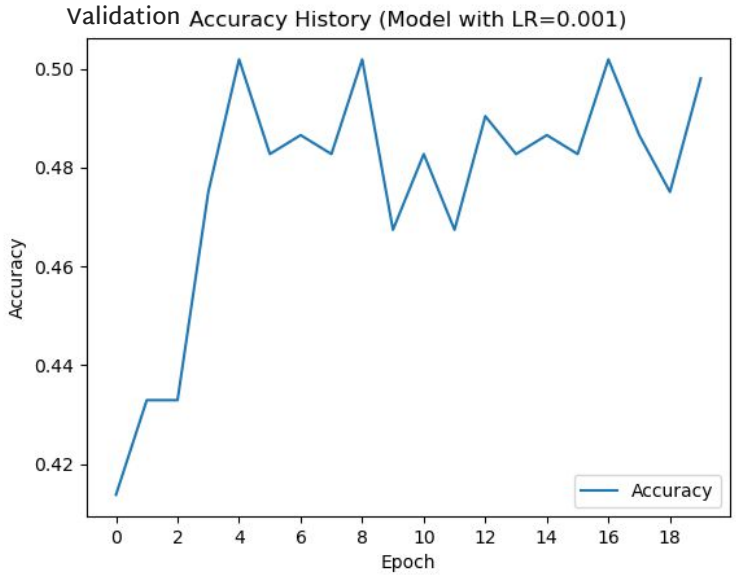
What we tried






```
malignancy_map = {
        1: 'Highly Unlikely',
        2: 'Moderately Unlikely',
        3: 'Indeterminate',
        4: 'Moderately Suspicious',
        5: 'Highly Suspicious'
        }
```

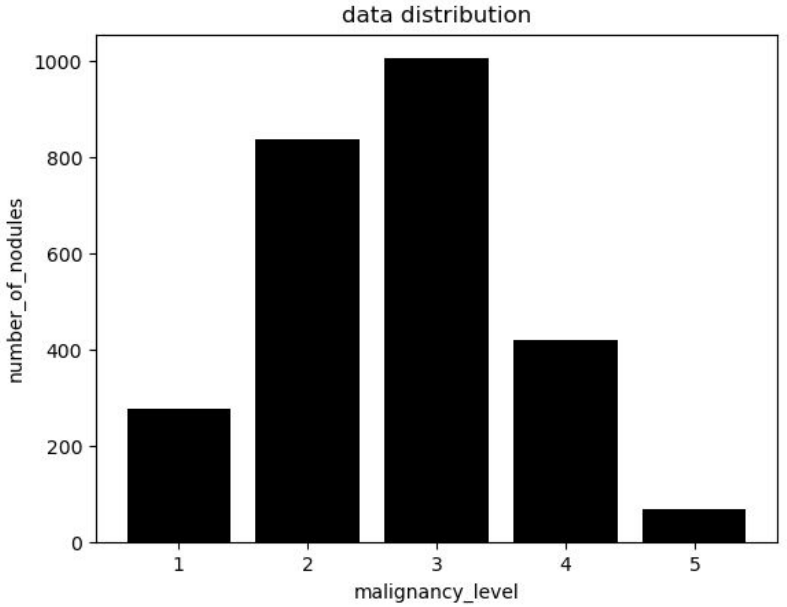
First result with multi-class classification




```
Test Accuracy: 56.11%
```

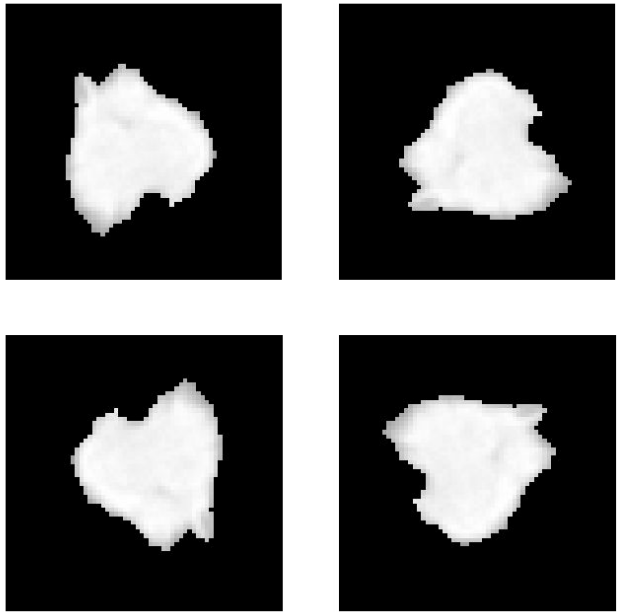
Why?

1. Dataset specification: The labels in the dataset are not perfect. They were created by a group of radiologists, and their assessments may vary slightly
2. Small # of datapoints
3. Imbalanced classes: -->
   Solution for 2 and 3: oversampling minority labels

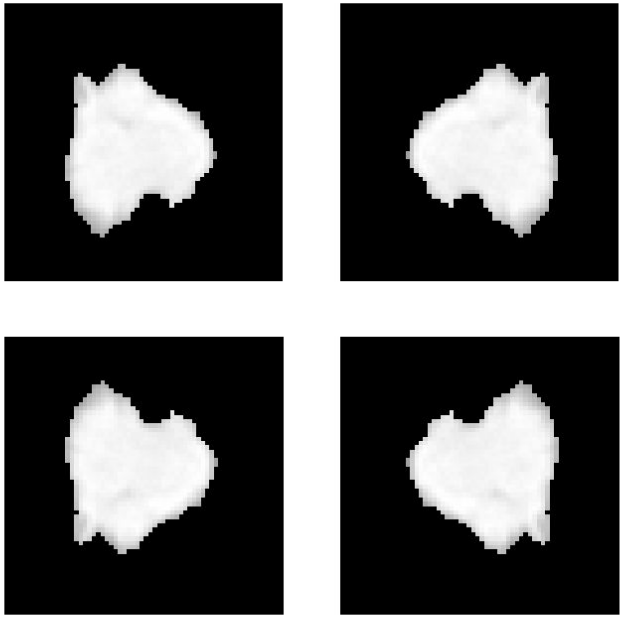


Data augmentation

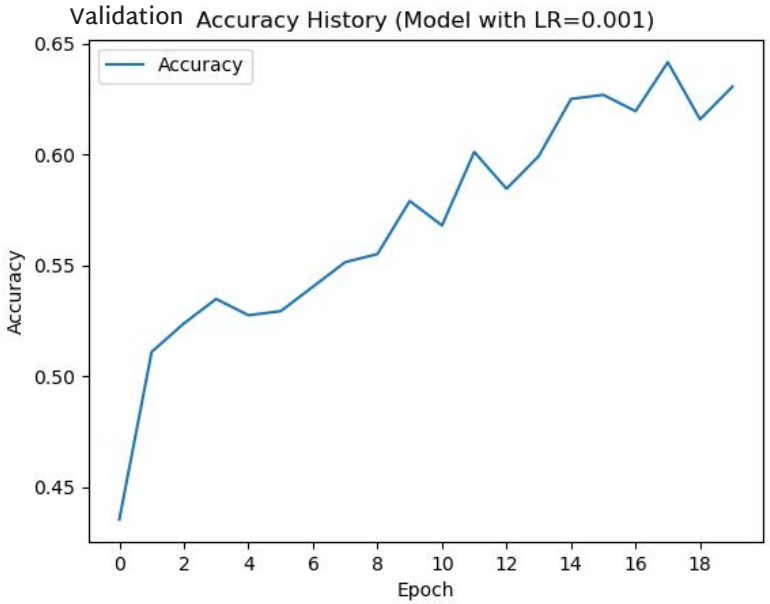
Synthetic data generated by rotations



Synthetic data generated by flips



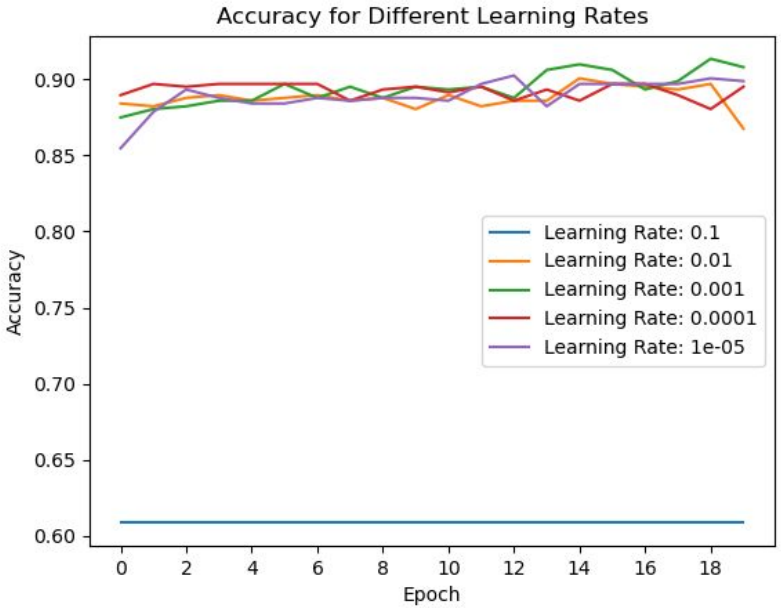
>10% improvement!

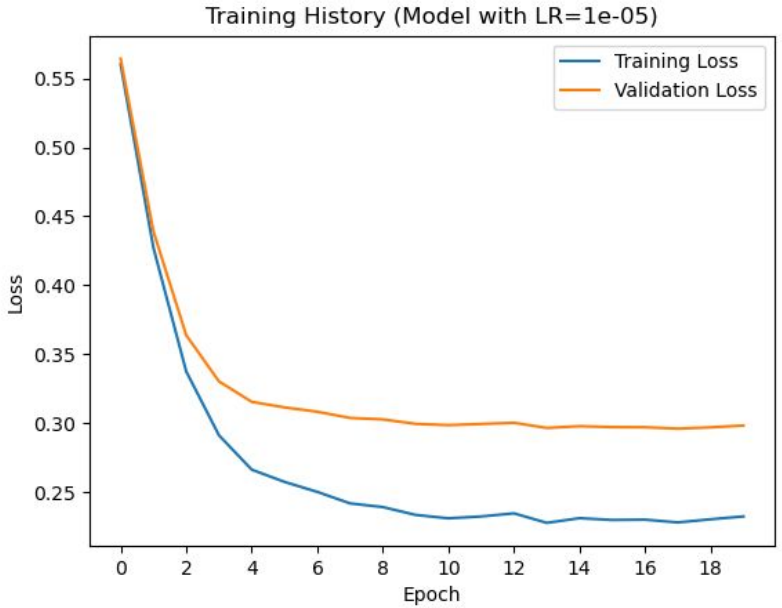


<- Results of Multi-class classification (binary classification next slide)

```
Test Accuracy: 67.80%
```

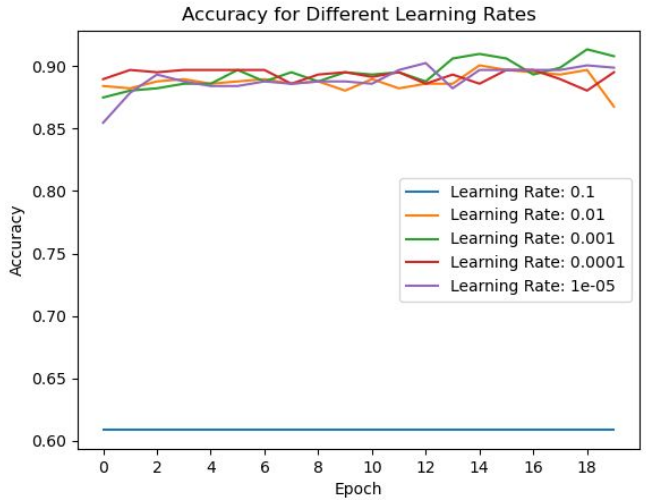
Binary approach



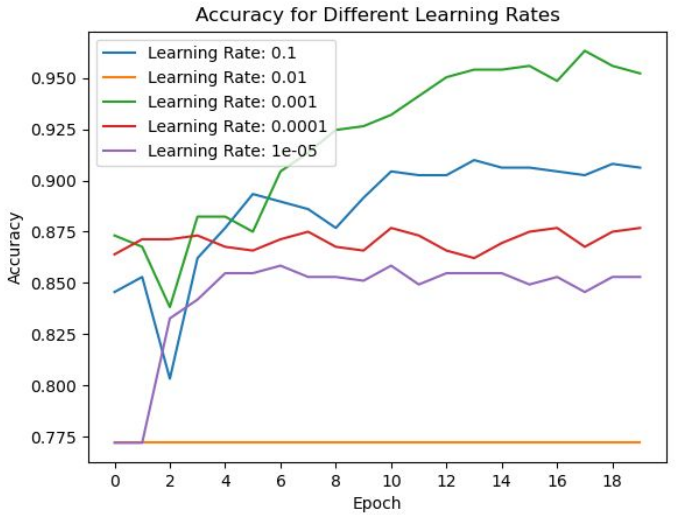


What if malignancy ==3?

The label equal to 3 indicates that the radiologists are uncertain whether the nodule indicates malignancy

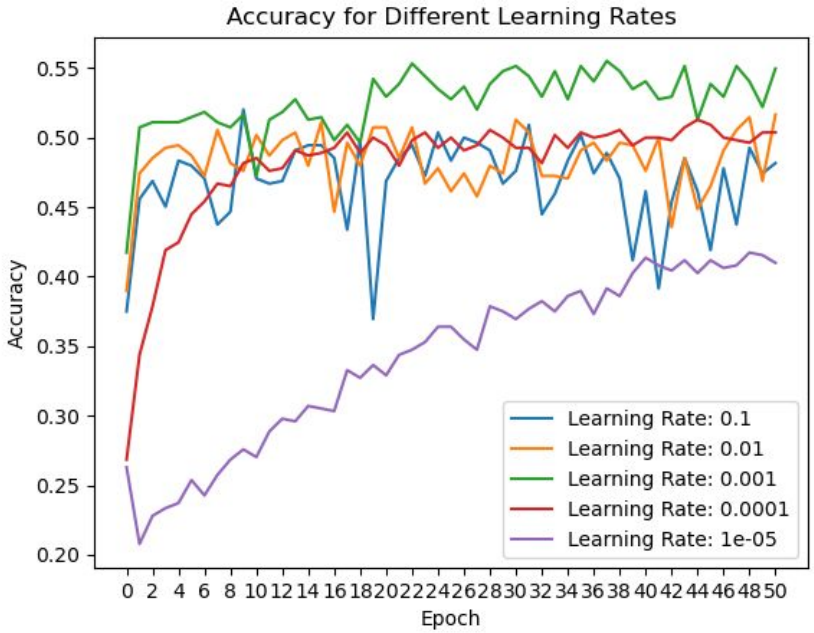


Binary classification where 3 == no cancer == label 0



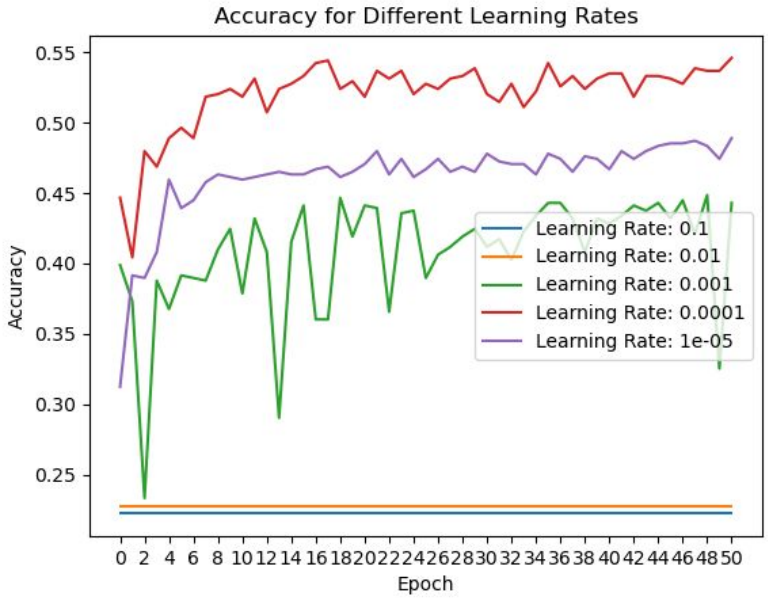
Binary classification where 3 == malignancy == label 1

Resnet18


Accuracy for Different Learning Rates
Accuracy
Epoch
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50
Learning Rate: 0.1
Learning Rate: 0.01
Learning Rate: 0.001
Learning Rate: 0.0001
Learning Rate: 1e-05

Vadim-nn

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.features = nn.Sequential(
            nn.Conv2d(1, 10, kernel_size=7, stride=1, padding=0),
            nn.MaxPool2d(kernel_size=3, stride=2, padding=0),
            nn.ReLU(),
            nn.Conv2d(10, 20, kernel_size=3, stride=1, padding=0),
            nn.MaxPool2d(kernel_size=2, stride=2, padding=0),
            nn.ReLU(),
            nn.Conv2d(20, 40, kernel_size=3, stride=1, padding=0),
            nn.MaxPool2d(kernel_size=2, stride=2, padding=0),
            nn.ReLU(),
        )
        self.classifier = nn.Sequential(
            nn.Flatten(),
            nn.Linear(2560, 1280),
            nn.ReLU(),
            nn.Linear(1280, 1280),
            nn.ReLU(),
            nn.Linear(1280, 1280),
            nn.ReLU(),
            nn.Linear(1280, 160),
            nn.ReLU(),
            nn.Linear(160, 5)
        )
```

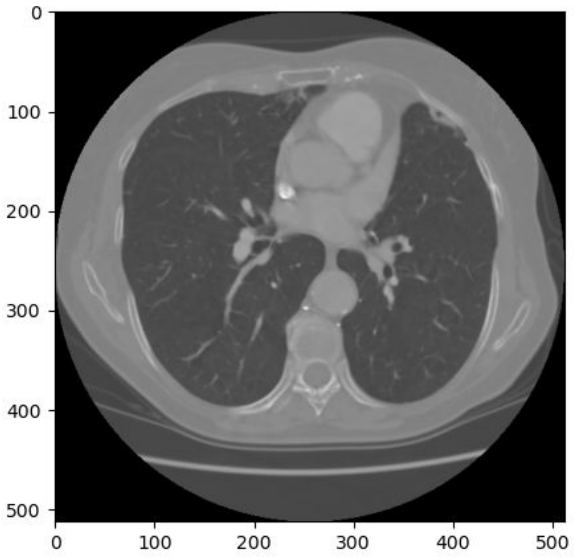


04

Summary and areas to explore

Future objectives

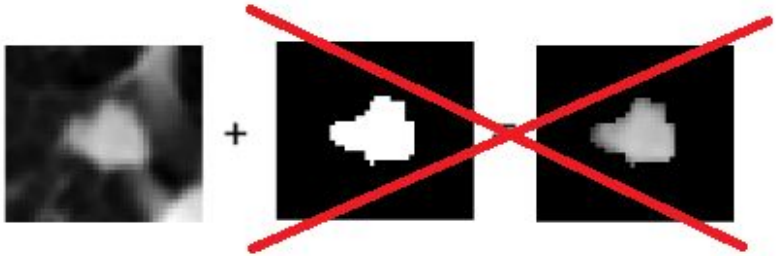
- Include volumetric information
- Use domain knowledge to select number of features
- Optimize hyperparameters using Ax
- with entire CT images (image segmentation) * a whole new project



Future objectives

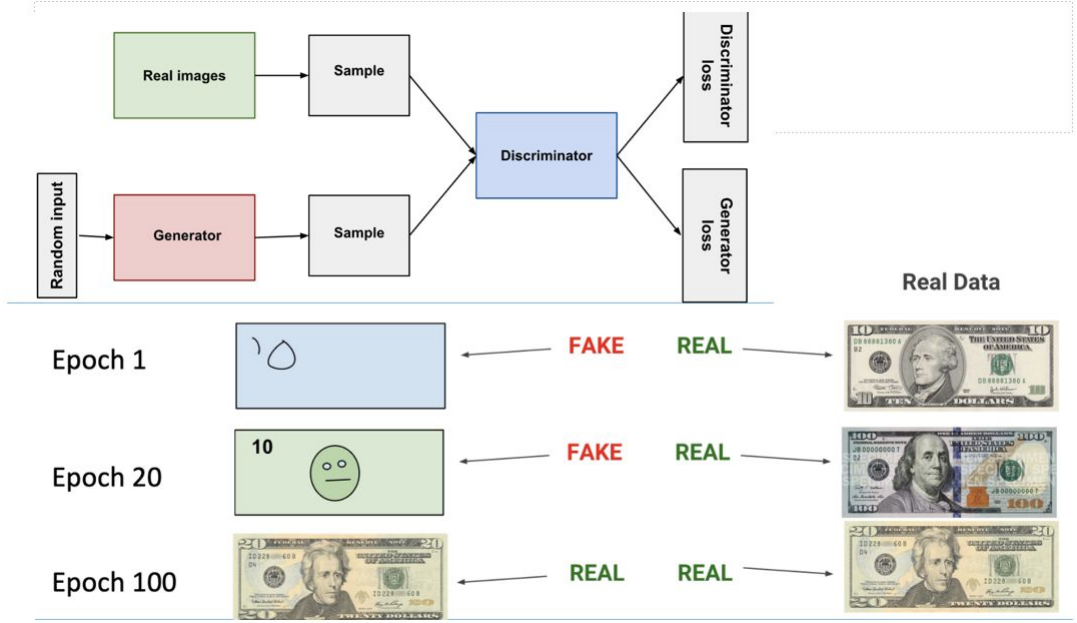
- <u>Using images with noise (bones arround e.g.)</u>

  Instead of using nodules after applying a Boolean mask, as we did, it's worth trying to train the model with original nodules — potentially new patterns to learn



Future objectives

- Generating synthetic data using more advanced techniques
  E.g.: GAN(Generative Adversarial Networks)



Thank you for attention!

You want to explore more? Check the code below:
https://github.com/VadimBim/DeepL-LIDC

References:

-**Dataset:** Data from The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): A completed reference database of lung nodules on CT scans (LIDC-IDRI) - The Cancer Imaging Archive (TCIA) Public Access - Cancer Imaging Archive Wiki

-**Ax: https://ax.dev/tutorials/tune_cnn_service.htm**
- picture with binary and multi-class classification: Getting started with Classification - GeeksforGeeks