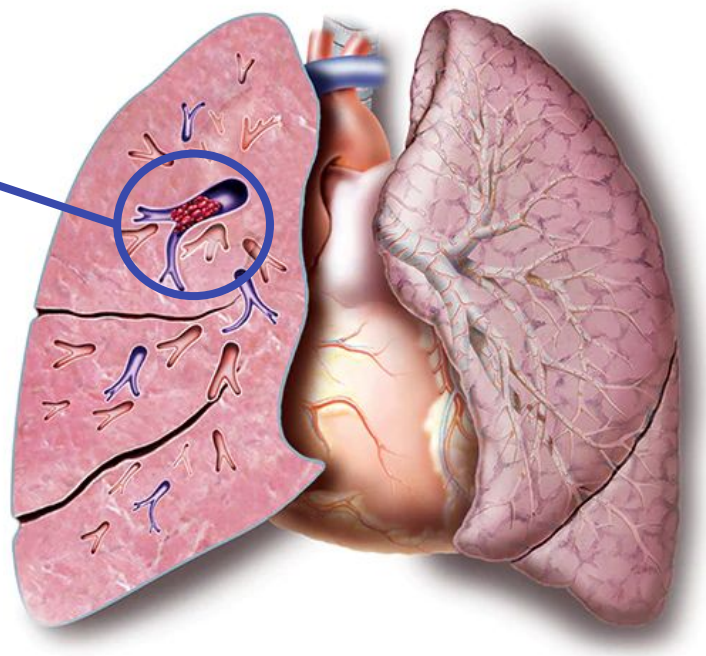
Pulmonary Embolism Classification Using CNN

Ben Inoyatov, Jamil Mirabito, Kyle Kieser

https://github.com/Kkies/PE_Classifier

What is Pulmonary Embolism?

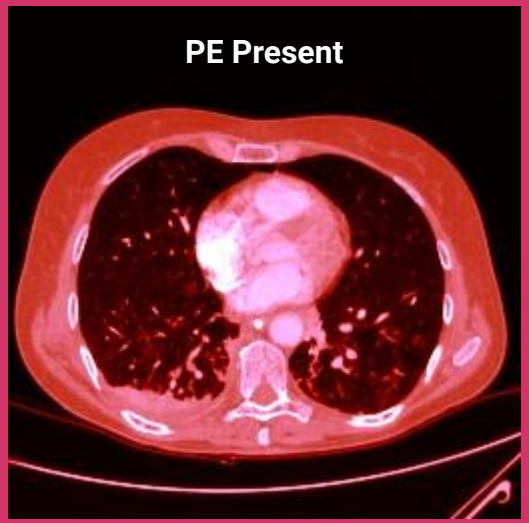
Pulmonary embolism (PE) is a blockage in one of the pulmonary arteries in your lungs. In most cases, pulmonary embolism is caused by blood clots that travel to the lungs from deep veins in the legs or, rarely, from veins in other parts of the body (deep vein thrombosis).



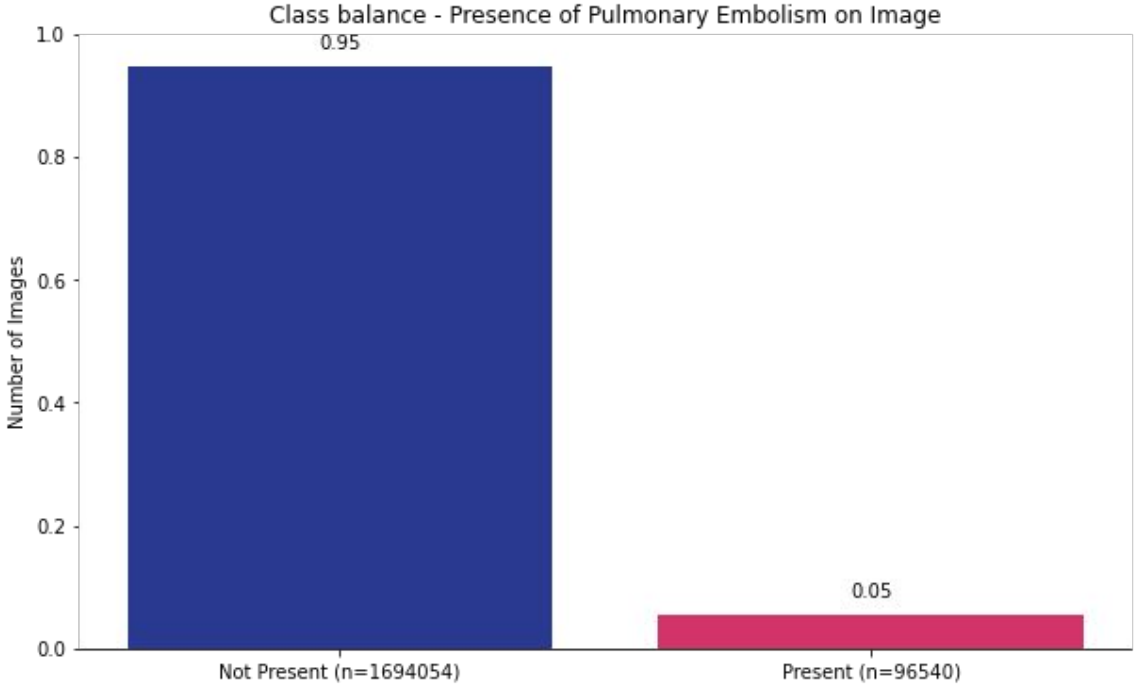
Sources: MayoClinic (definition); MedPage Today (image)

The Challenge

PEs can be hard to spot even for experienced and specialized practitioners. PEs are time consuming to confirm and are prone to overdiagnosis. With the advent of Deep Learning, we have the opportunity to help practitioners more quickly identify PE using Convolutional Neural Networks (CNN).



Pulmonary Embolism Data



Data: RSNA STR Pulmonary Embolism Detection Dataset

Sample: 1,790,594 images; 7,297 participants

Only 5 percent of images in the dataset presented a Pulmonary Embolism

Modeling Process

Random Sampling

Since the dataset was so large and classes were unbalanced, **we randomly selected 5,000 images from each class** to train, validate, and test our model.

Train: 4,200 images
Validation: 1,800 images
Test: 5,000 images

Modeling

We tested a t**otal of 13 models in Jupyter Notebooks and Google Colab**.

10 used multiple optimizers, batch sizes, epochs, & numbers of layers.

3 were Transfer Learning models - **Inception & ResNet 50**

Model Evaluation

We optimized our models on the Recall score so as to limit the number of PE misses (false negatives).

Model results

- Baseline: 95% accuracy if only predicting no PE*
- Adam was the best optimizer
- Transfer Learners (Inception & ResNet) increased complexity for marginal improvement
- Final model had 3 convolutional layers with a max pooling layer after each one**
- Each convolutional layer had half the amount of neurons as the previous convolutional layer

	Vanilla	Adam Optimizer	Increasing Neurons	Decreasing Neurons	Final Model
Accuracy	0.797	0.848	0.842	0.848	**0.916**
Recall	0.624	0.776	0.802	0.792	**0.945**
F1	0.751	0.833	0.833	0.836	**0.919**

* Before balanced sampling
See **Appendix Slide 13 for Final Model Parameters

Conclusion

A **Convolutional Neural Network Model (CNN)** can effectively classify Pulmonary Embolism with a **recall rate of 94.5 percent** and an **accuracy rate of 91.6 percent**

This model can help to **reduce the time it takes to confirm PEs** by indicating to practitioners which slices most likely have a PE present

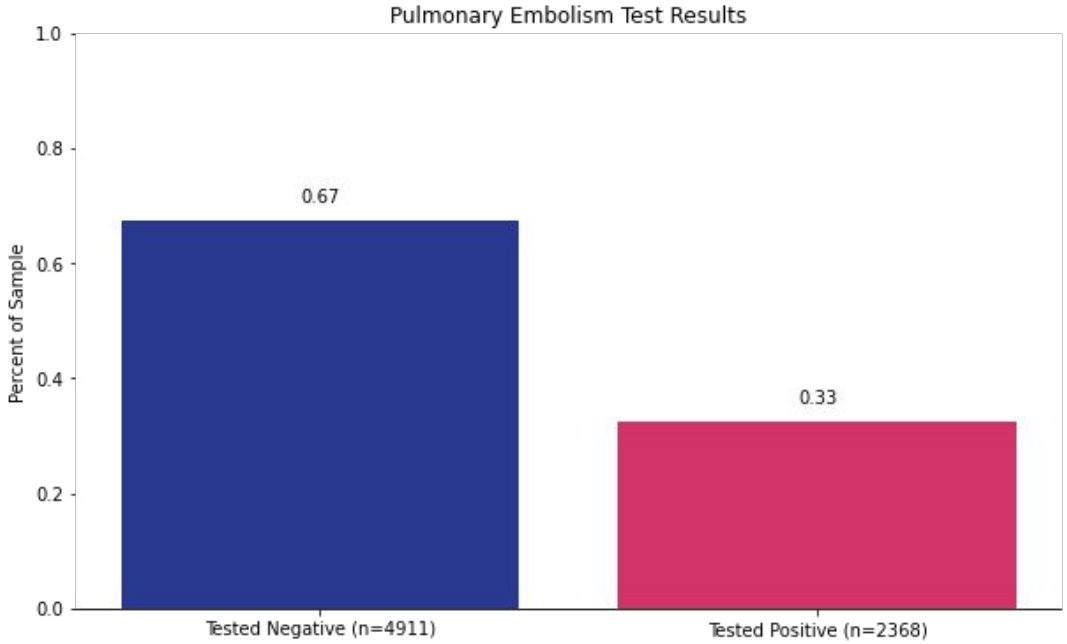
Limitation & Recommendations

- Given the large size of the dataset and computational limitations, we were unable to train the model on the full dataset thereby limiting the performance of the model. With more time and computational power, **future models could train/validate/test on more than just 5,000 images.**
- We were unable to train the model to identify *where* the PE was located in each image (i.e., right lung, left lung, pulmonary artery). **Future models could create multi-class outputs to identify the location of the PE.**
- We were unable to incorporate RV/LV into our analyses. **Future models could compare the size of the Left and Right Ventricles (RV/LV ratio*) to predict probability of morbidity.**

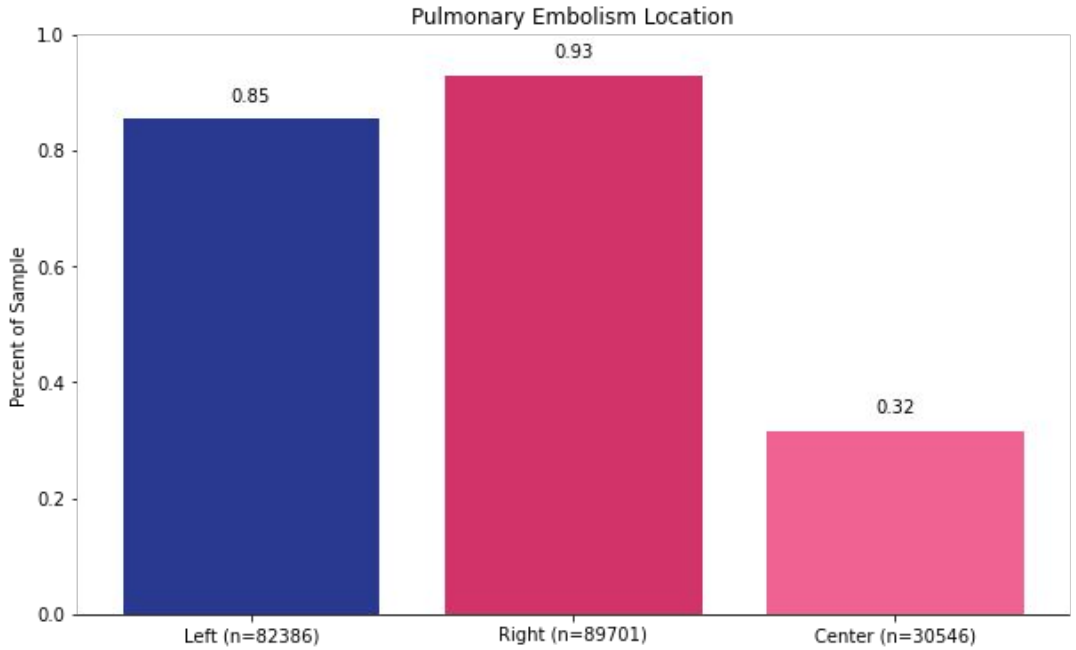
*See **Appendix Slide 12** for more information.

Appendix

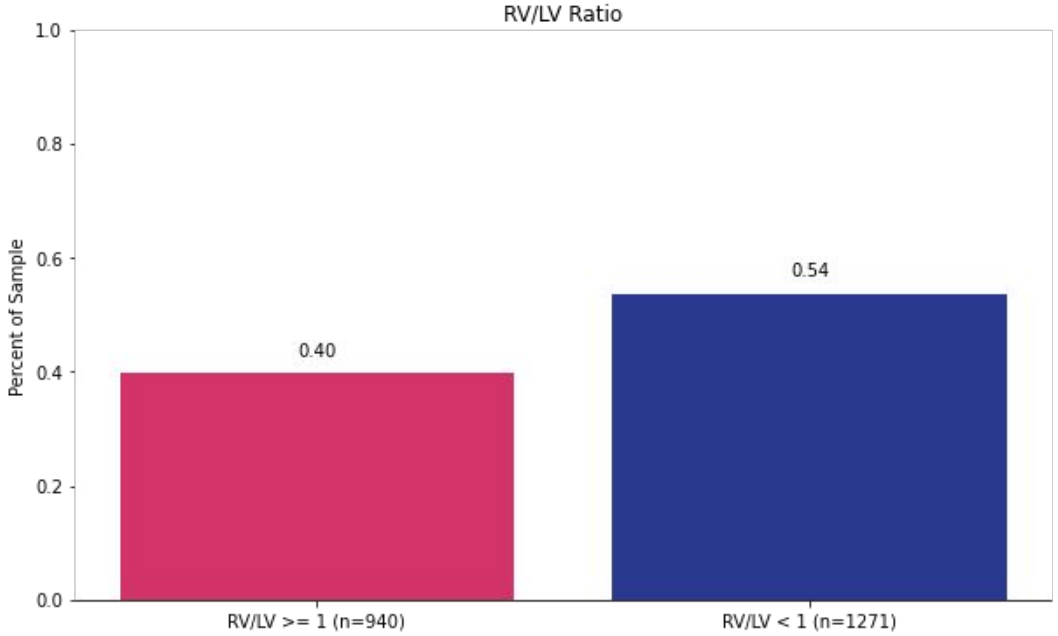
Among patients in the study, 33 percent tested positive for a pulmonary embolism



Most images show PEs in multiple locations in patients' lungs



Among individuals who test positive for PE, 54 percent have an RV/LV ratio greater than 1



Final Model Parameters

```
cnn10 = models.Sequential()
cnn10.add(layers.Conv2D(512, (3, 3), activation='relu', input_shape=(256, 256,  3)))
cnn10.add(layers.MaxPooling2D(pool_size = (2, 2), strides = (2,2)))
cnn10.add(layers.Conv2D(256, (3, 3), activation='relu'))
cnn10.add(layers.MaxPooling2D(pool_size = (2, 2), strides = (2,2)))
cnn10.add(layers.Conv2D(128, (3, 3), activation='relu'))
cnn10.add(layers.MaxPooling2D(pool_size = (2, 2)))
cnn10.add(layers.Flatten())
cnn10.add(layers.Dense(256, activation='relu'))
cnn10.add(layers.Dense(1, activation='sigmoid'))

cnn10.compile(loss='binary_crossentropy',
              optimizer="adam",
              metrics=['acc', tf.keras.metrics.Recall()])
```

```
cnn10.fit(X_train2,
        y_train2,
        epochs=10,
        batch_size=50,
        validation_data=(X_val2, y_val2),
        verbose = 1,
        workers = 7)
```

When to call for help

- Shortness of breath
- Chest pain
- Pain on deep breaths
- Rapid heart rate
- Extreme sweating