Using CNNs and Transfer Learning to Detect Pneumonia in Chest X-rays

Jack Brosgol, Alex Fischmann

Agenda

01 Background

02 Methods

03 Experiments

04 Results

05 Ethical Implications

06 Future Work

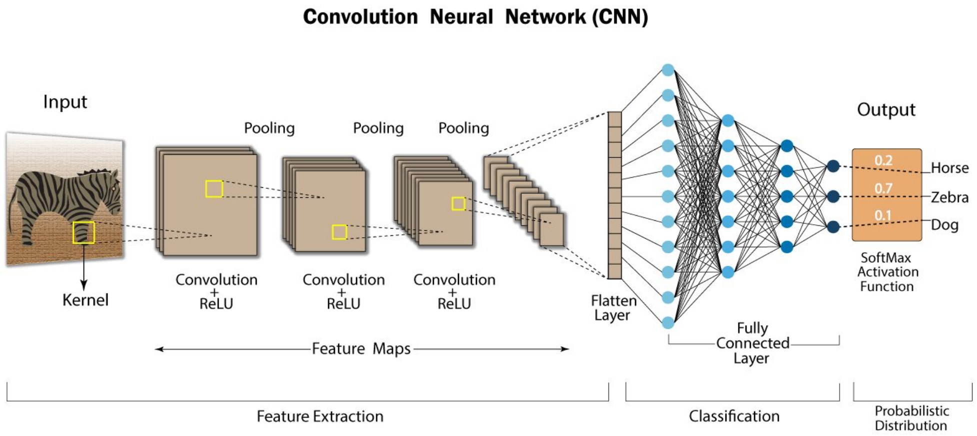
Background

Normal Bacterial Viral

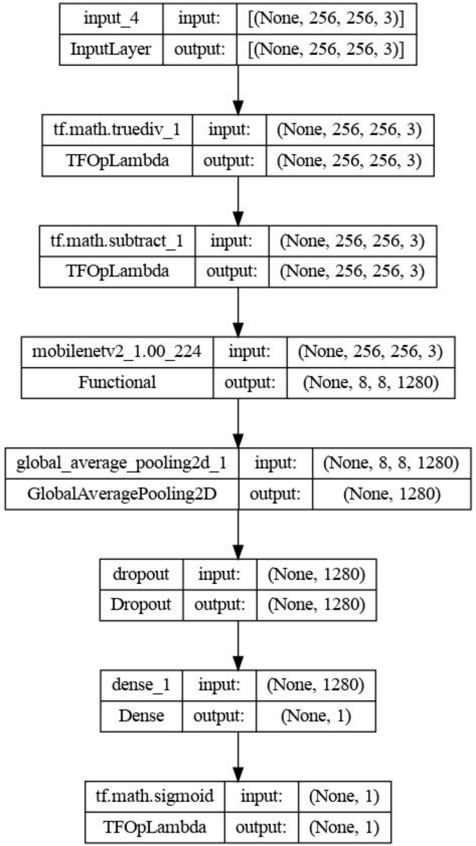
- Responsible for 14% of all deaths of children under 5 years old in 2019 (World Health Organization)
- Two different kinds, treatment can differ
- Doctors look for white spots in lung, called infiltrates

Methods: Convolutional Neural Networks (CNNs)

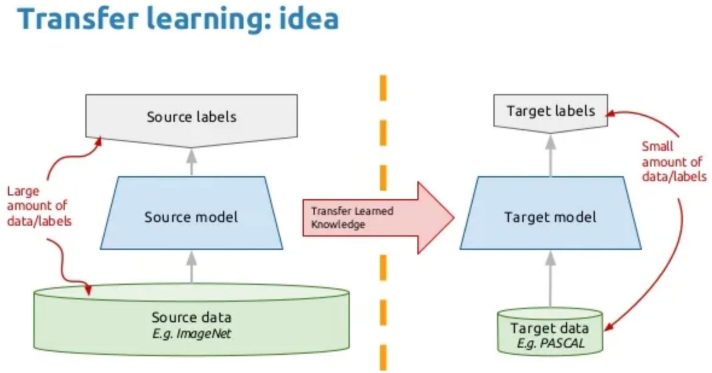
- Deep learning architecture designed for images
- Characteristics
  - Convolutional layer → feature maps
  - Pooling layer → downsampling
  - Fully connected layer for classification



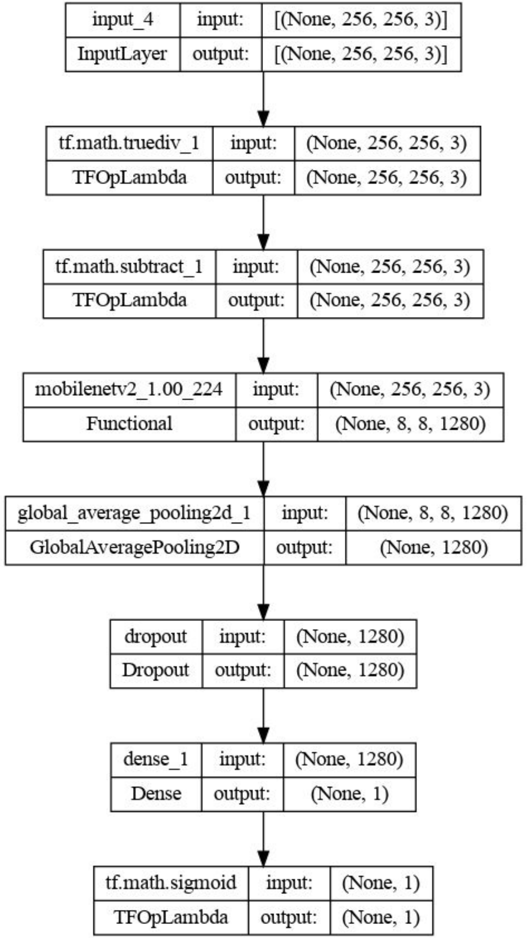
Methods: Transfer Learning

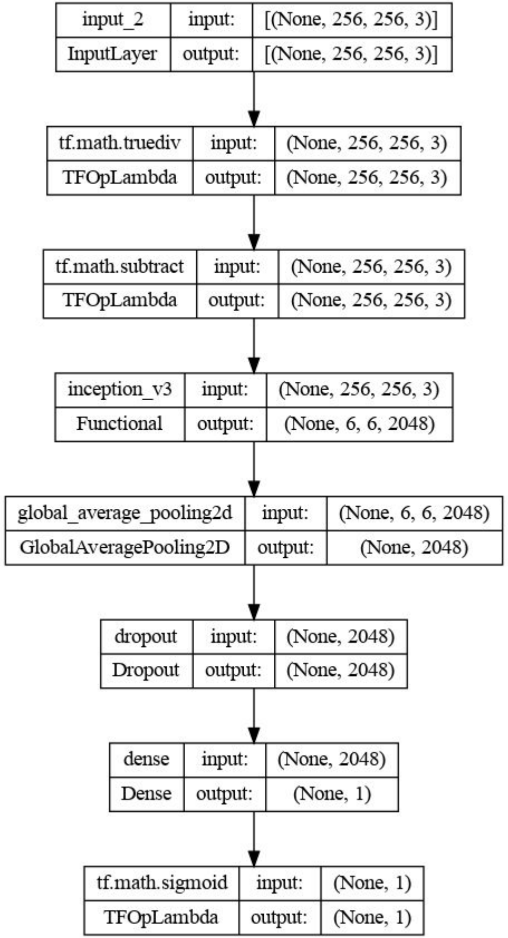


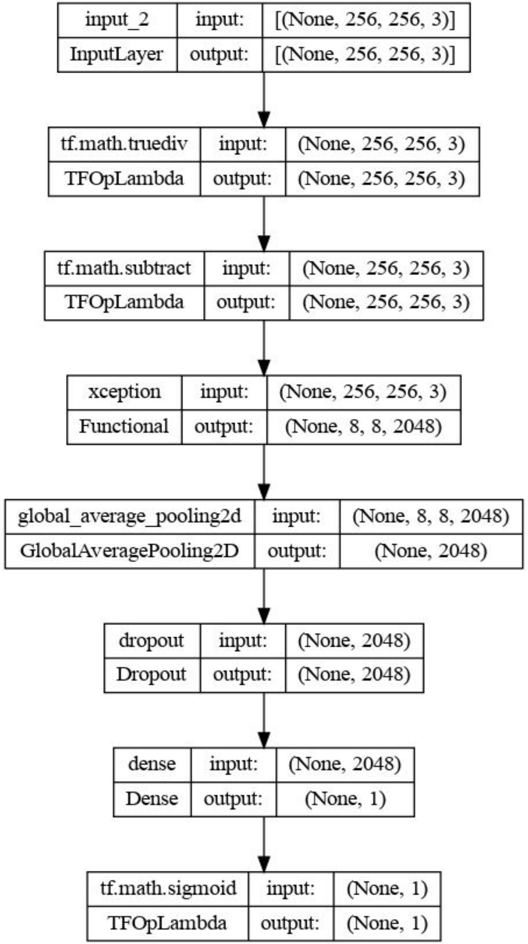
- Add new layers
- Freeze bottom layers -> Train
- Similar Low Level Patterns
- Disadvantages: Less Freedom over Architecture
- Advantages: Small data sets, Computational



Models

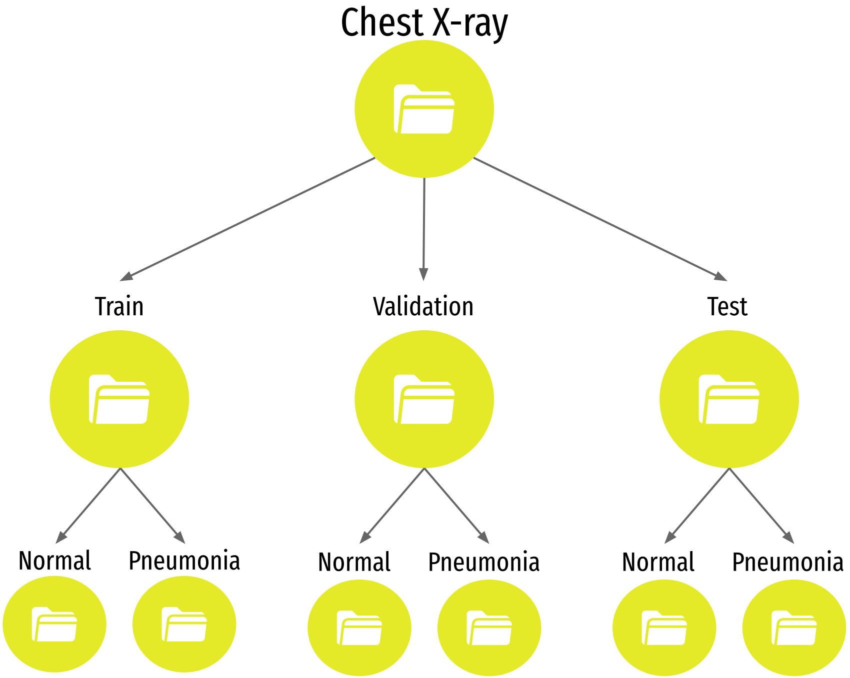

input_4
InputLayer
input:
output:
[(None, 256, 256, 3)]
[(None, 256, 256, 3)]
tf.math.truediv_1
TFOpLambda
input:
output:
(None, 256, 256, 3)
(None, 256, 256, 3)
tf.math.subtract_1
TFOpLambda
input:
output:
(None, 256, 256, 3)
(None, 256, 256, 3)
mobilenetv2_1.00_224
Functional
input:
output:
(None, 256, 256, 3)
(None, 8, 8, 1280)
global_average_pooling2d_1
GlobalAveragePooling2D
input:
output:
(None, 8, 8, 1280)
(None, 1280)
dropout
Dropout
input:
output:
(None, 1280)
(None, 1280)
dense_1
Dense
input:
output:
(None, 1280)
(None, 1)
tf.math.sigmoid
TFOpLambda
input:
output:
(None, 1)
(None, 1)


input_2
InputLayer
input:
output:
[(None, 256, 256, 3)]
[(None, 256, 256, 3)]
tf.math.truediv
TFOpLambda
input:
output:
(None, 256, 256, 3)
(None, 256, 256, 3)
tf.math.subtract
TFOpLambda
input:
output:
(None, 256, 256, 3)
(None, 256, 256, 3)
inception_v3
Functional
input:
output:
(None, 256, 256, 3)
(None, 6, 6, 2048)
global_average_pooling2d
GlobalAveragePooling2D
input:
output:
(None, 6, 6, 2048)
(None, 2048)
dropout
Dropout
input:
output:
(None, 2048)
(None, 2048)
dense
Dense
input:
output:
(None, 2048)
(None, 1)
tf.math.sigmoid
TFOpLambda
input:
output:
(None, 1)
(None, 1)

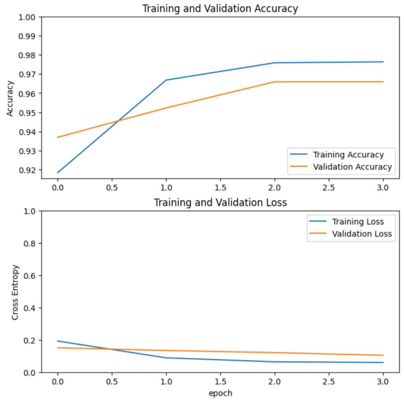

input_2
InputLayer
input:
output:
[(None, 256, 256, 3)]
[(None, 256, 256, 3)]
tf.math.truediv
TFOpLambda
input:
output:
(None, 256, 256, 3)
(None, 256, 256, 3)
tf.math.subtract
TFOpLambda
input:
output:
(None, 256, 256, 3)
(None, 256, 256, 3)
xception
Functional
input:
output:
(None, 256, 256, 3)
(None, 8, 8, 2048)
global_average_pooling2d
GlobalAveragePooling2D
input:
output:
(None, 8, 8, 2048)
(None, 2048)
dropout
Dropout
input:
output:
(None, 2048)
(None, 2048)
dense
Dense
input:
output:
(None, 2048)
(None, 1)
tf.math.sigmoid
TFOpLambda
input:
output:
(None, 1)
(None, 1)

Data

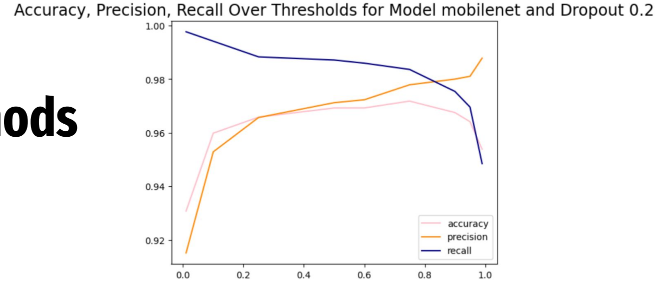
- 5,836 images
- 72% infected | 28% not infected
- Raw data split (89/1/10)
- One to five years old



Workflow and Analysis Methods

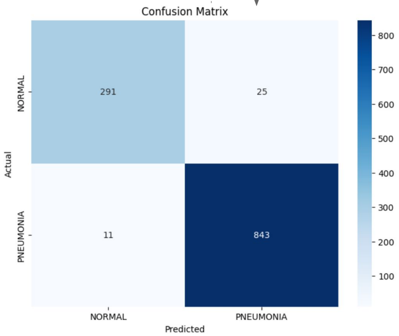


MobilenetV2, Dropout = 0.2

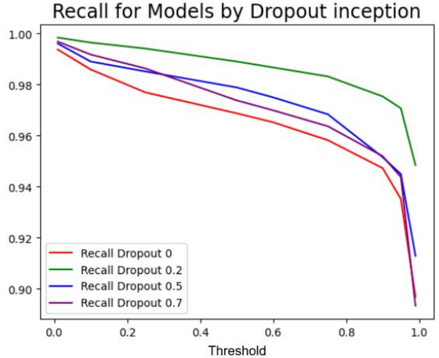


MobilenetV2, Dropout = 0.2

```
For each dataset: #raw data, shuffled data
        For each base model:
                Pre-process data
                Build Model # e.g. add pooling layer
                For each dropout value:
                        Repeat 3 times:
                                Train model
                                Test model
                                        Output CMs, accuracy, precision, recall
                        Average the performance metrics from 3 trials
                Graph comparison of dropouts over all thresholds
```



MobilenetV2, Dropout = 0., Threshold = 0.5

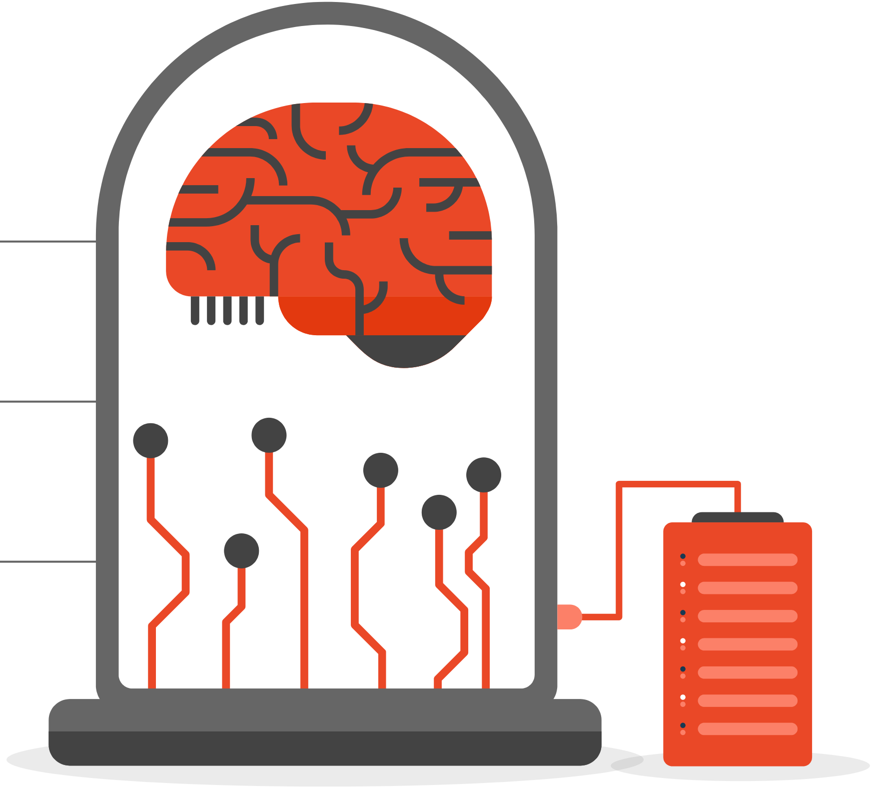


Results

01 **Shuffled vs Unshuffled**

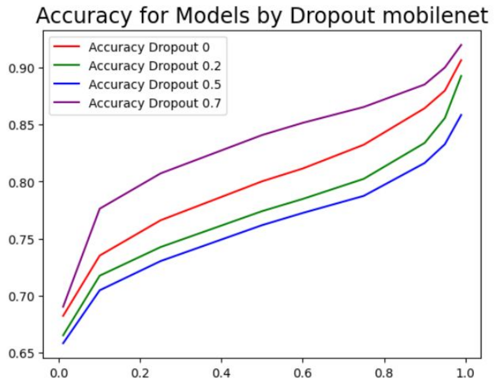
02 **Impact of Different Dropouts**

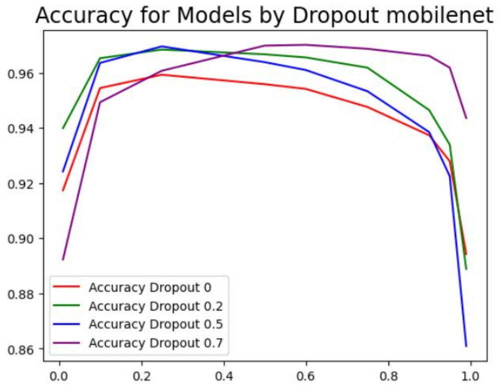
03 **Impact of Different Base Models**



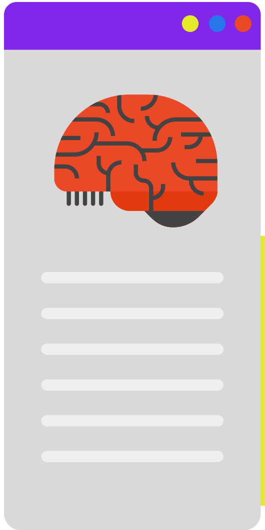
Results: Shuffled vs Unshuffled

- Unshuffled 89/1/10 vs. Shuffled 70/10/20
- Unshuffled → threshold and overfitting
- Plausible explanations:
  - Different sensors → brightness
  - Differences in train/test
  - Impact of train/val/test split
  - Not class imbalance
- Precision and Recall at threshold = 0.5
  - Shuffled: ~97% for both
  - Unshuffled: 75% vs 99%





Results: Impact of Different Dropouts



Discussion

01 Analysis

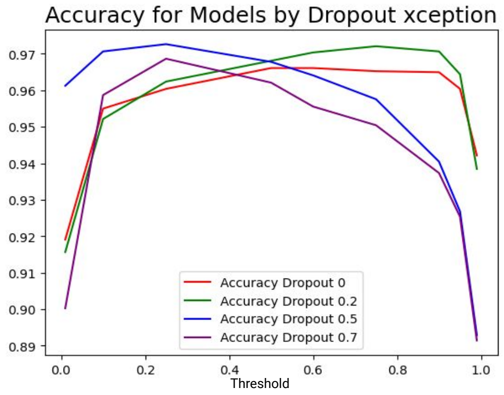
1. Performance over different thresholds
2. Paired t-tests to compare dropouts

02 Trade-off between Precision and Recall

How to weigh the importance of both

Results: Impact of Different Dropouts

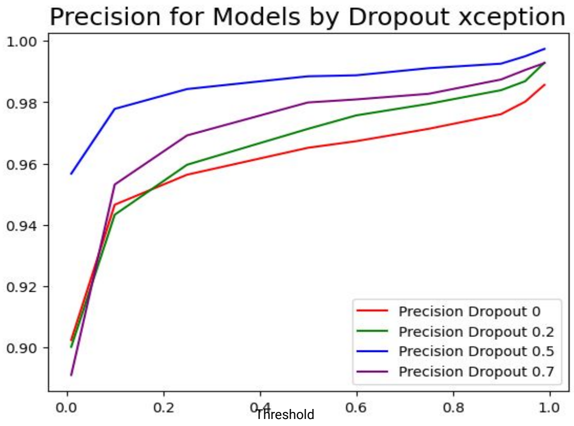
Xception

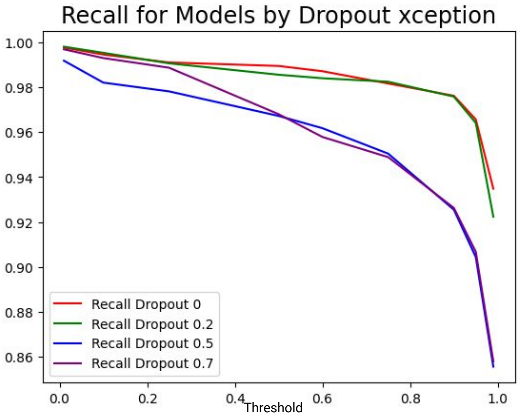


- Accuracy: 0.2, 0.5, 0.7

- Recall: 0.2
    - Precision trade-off justified

- → 0.2

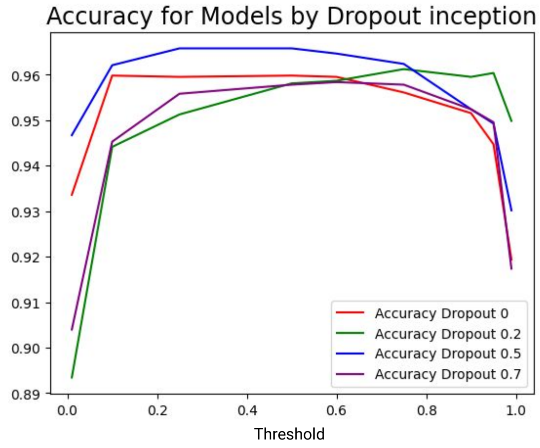


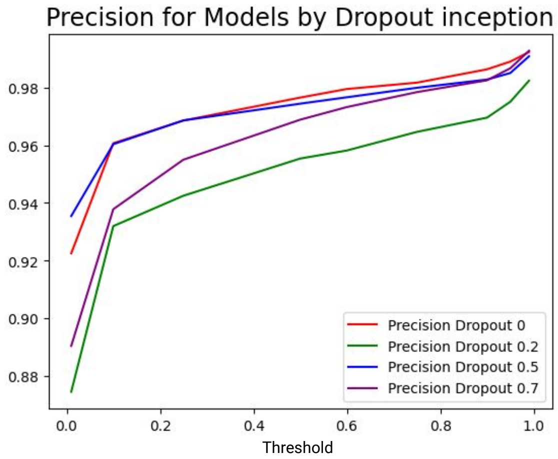


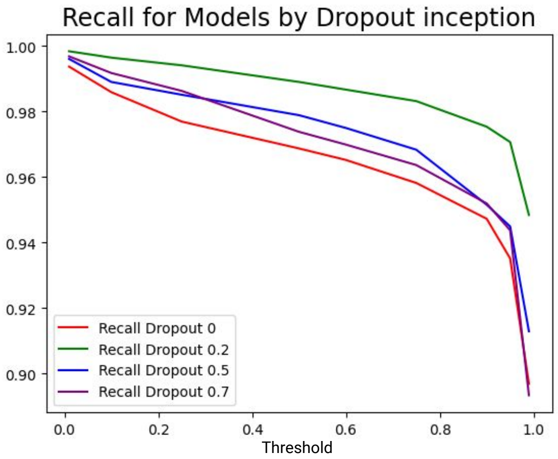
Results: Impact of Different Dropouts

Inception

- Accuracy: 0.5 has highest (statistically)
- 0.2 has higher recall (by ~1%), but much lower precision (by ~4%)
- → 0.5



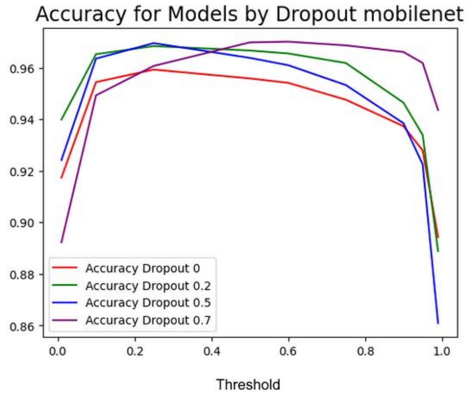


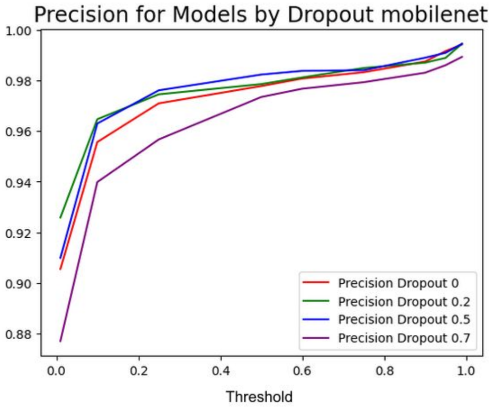


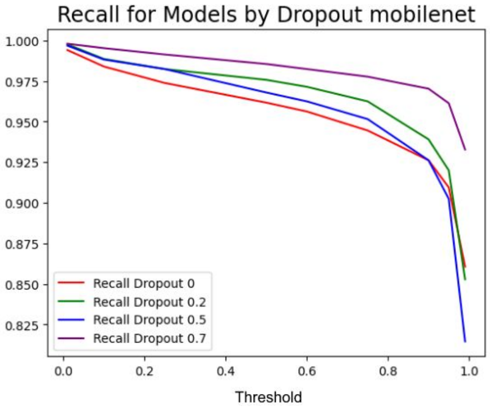
Results: Impact of Different Dropouts

MobilenetV2

- Accuracy: 0.2, 0.5, 0.7 vs 0.0
- Is the trade-off too big for 0.7?
- → 0.2 and 0.5







Results: Impact of Different Base Models

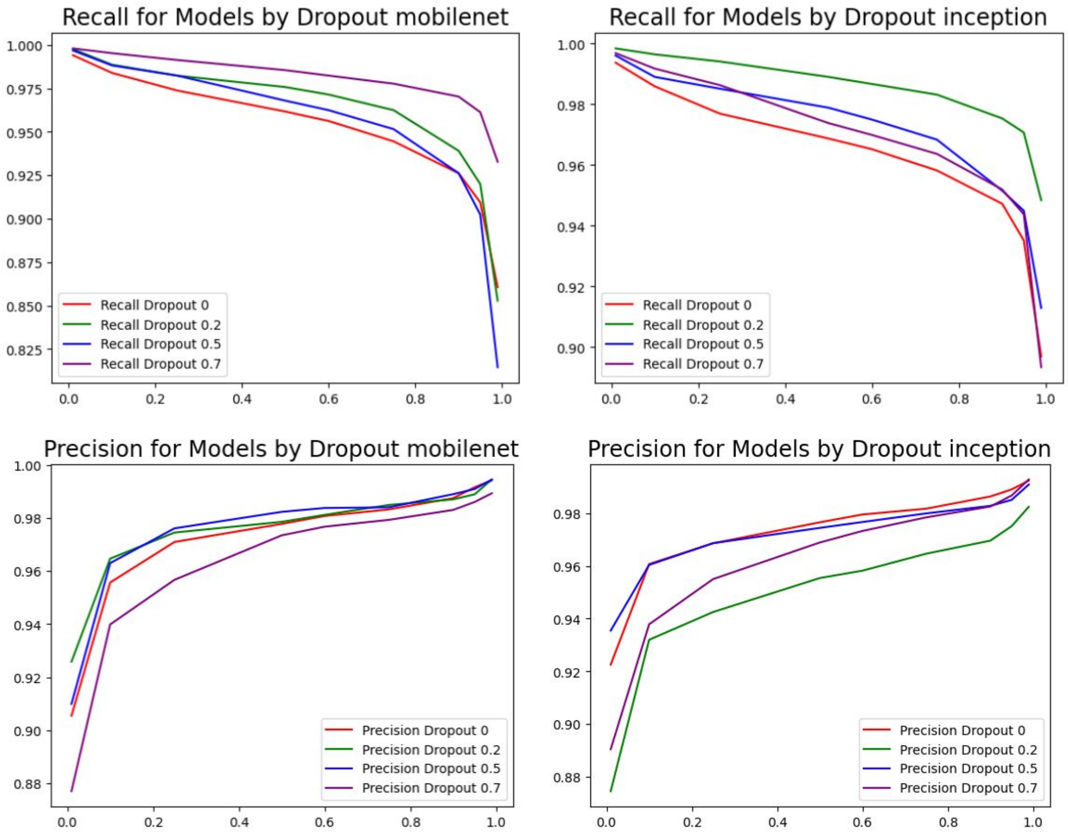
Table 1. Comparison of Base Models with Dropout = 0.2 and Threshold = 0.5

	MobileNet	Inception	Xception
Accuracy	0.9667	0.9581	**0.9681**
Precision	**0.9785**	0.9553	0.9713
Recall	0.9780	**0.9891**	0.9856

Table 2. Comparison of Base Models with Dropout = 0.5 and Threshold = 0.5

	MobileNet	Inception	Xception
Accuracy	0.9638	0.9658	**0.9678**
Precision	0.9823	0.9743	**0.9884**
Recall	0.9680	**0.9789**	0.9672

Results: Impact of Different Base Models



Ethical Implications

- False Positives, False Negatives -> Precision Recall Tradeoff
- Scope of our data
- Additional method for diagnosis, not replacement
- Advantages:

  (1) Patients diagnosed faster

  (2) Doctors can spend time more efficiently
- Potential Harmful Effects:

  (1) Lack of Personalized Care

  (2) Corporate Structure

Future Work

- Experiment with more dropout values between 0.2 and 0.5
- K-fold Validation
- Different base models

Thank you!

Sources

https://www.who.int/news-room/fact-sheets/detail/pneumonia

https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

https://www.radiologyinfo.org/en/info/pneumonia

https://developersbreach.com/convolution-neural-network-deep-learning/

https://towardsdatascience.com/transfer-learning-with-convolutional-neural-networks-in-pytorch-dd09190245ce