

COVID-19 Detection

By
Hooman Shiraninehr

The Problem:

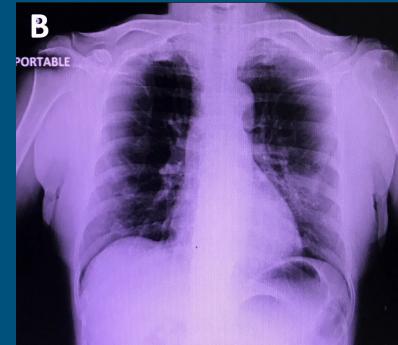
COVID-19 tests can be costly and slow.

The Solution:

Use convolutional neural networks and computer vision to detect COVID-19 based on chest X-ray.

Data columns (total 30 columns):			
#	Column	Non-Null Count	Dtype
0	patientid	950	non-null
1	offset	697	non-null
2	sex	870	non-null
3	age	713	non-null
4	finding	950	non-null
5	RT_PCR_positive	593	non-null
6	survival	361	non-null
7	intubated	248	non-null
8	intubation_present	250	non-null
9	went_icu	397	non-null
10	in_icu	335	non-null
11	needed_supplemental_o2	90	non-null
12	extubated	37	non-null
13	temperature	78	non-null
14	pO2_saturation	119	non-null
15	leukocyte_count	16	non-null
16	neutrophil_count	28	non-null
17	lymphocyte_count	40	non-null
18	view	950	non-null
19	modality	950	non-null
20	date	661	non-null
21	location	894	non-null
22	folder	950	non-null
23	filename	950	non-null
24	doi	382	non-null
25	url	950	non-null
26	license	705	non-null
27	clinical_notes	768	non-null
28	other_notes	436	non-null
29	Unnamed: 29	5	non-null

The Data



Data Source: <https://github.com/ieee8023/covid-chestxray-dataset>.

Data Wrangling

- ❖ Dropped columns not needed.
 - ❖ Replaced nulls with ‘Unknown’/Avg.
 - ❖ Removed 21 records with no image.
 - ❖ Images are considered “Positive” if “finding” column in meta data has “COVID-19”. Otherwise, “Negative”.
 - ❖ 75/15/15 split for training/validation/test
-

Exploratory Data Analysis

- ❖ Data is relatively balanced (563 positive cases and 366 negative)
 - ❖ Images have different sizes. So we changed all to 256x256.
 - ❖ As training set is small (650 images total), it is augmented by generating 11 new images per image.
-

Model Selection

- ❖ MobileNet is used as base model after discarding the last layer of 1000 neurons.
- ❖ MobileNet has lightweight architecture. It uses depthwise separable convolutions.
- ❖ Weights in the base model are pre-trained on ImageNet.

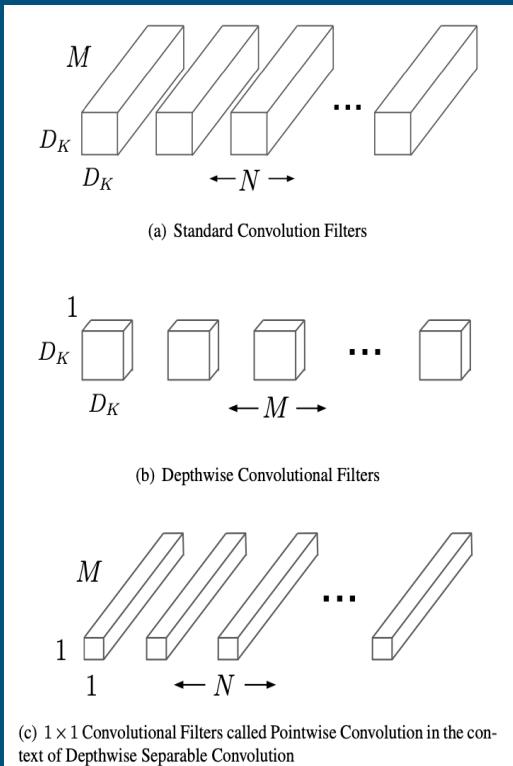


Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32 \text{ dw}$	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64 \text{ dw}$	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128 \text{ dw}$	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128 \text{ dw}$	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256 \text{ dw}$	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256 \text{ dw}$	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5 \times$ Conv dw / s1	$3 \times 3 \times 512 \text{ dw}$	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512 \text{ dw}$	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024 \text{ dw}$	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Model Selection

- ❖ Base model is frozen.
- ❖ 2 models are considered with 1 layer added to the base model and different dropout values.
- ❖ Notice that dropout values are relatively large as lower values lead to overfitting.

```
base_model = MobileNet(weights='imagenet',include_top=False)

# Freeze the base_model
base_model.trainable = False
```

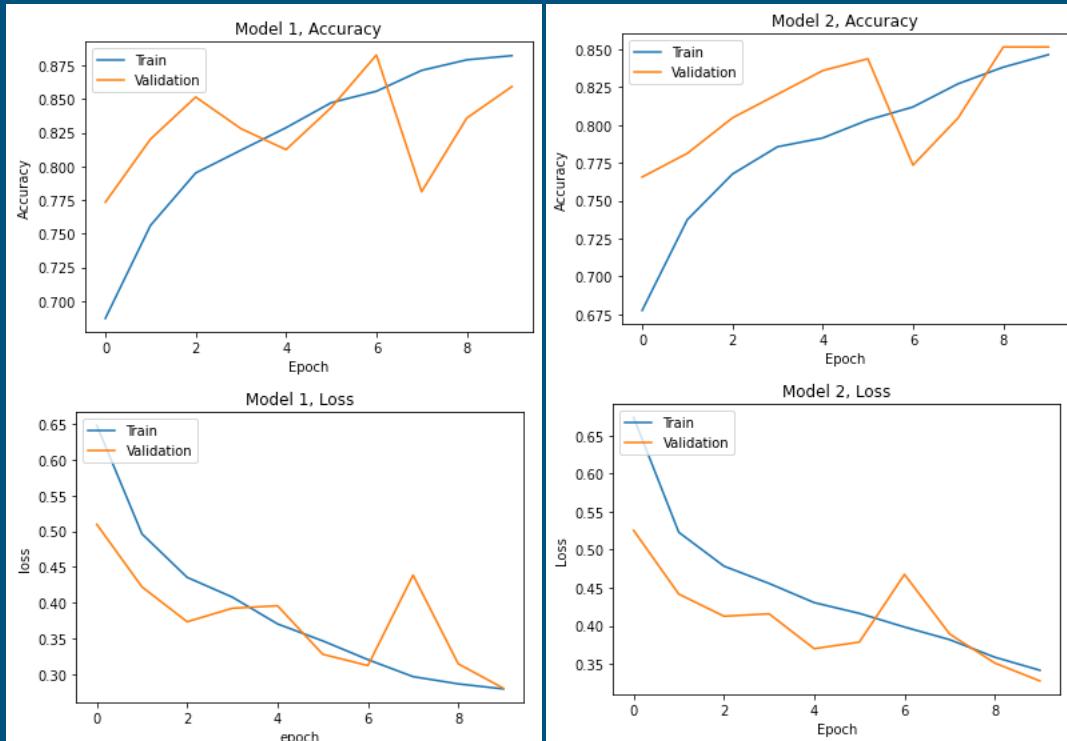
```
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024,activation='relu')(x)
x = keras.layers.Dropout(0.6)(x)
preds = Dense(2,activation='softmax')(x)
model1 = Model(inputs=base_model.input,outputs=preds)

x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024,activation='relu')(x)
x = keras.layers.Dropout(0.7)(x)
preds = Dense(2,activation='softmax')(x)
model2 = Model(inputs=base_model.input,outputs=preds)
```

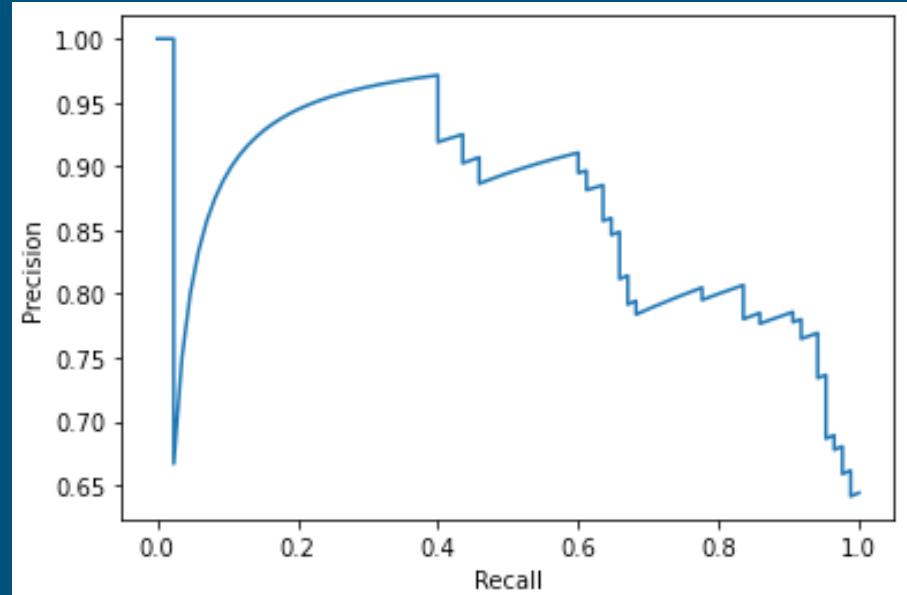
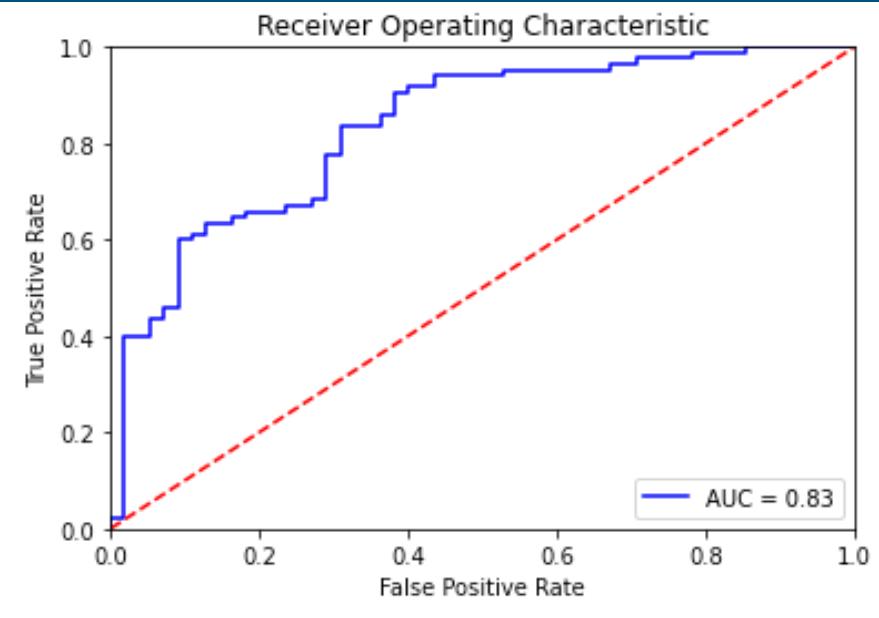
```
datagen = ImageDataGenerator(preprocessing_function=preprocess_input)
train_generator = datagen.flow_from_directory(
    directory=root_dir+"/train/",
    target_size=(256,256),
    color_mode="rgb",
    batch_size=32,
    class_mode="categorical",
    shuffle=True,
    seed=42
)
valid_generator = datagen.flow_from_directory(
    directory=root_dir+"/val/",
    target_size=(256,256),
    color_mode="rgb",
    batch_size=32,
    class_mode="categorical",
    shuffle=True,
    seed=42
)
test_generator = datagen.flow_from_directory(
    directory=root_dir+"/test/",
    target_size=(256,256),
    color_mode="rgb",
    batch_size=1,
    class_mode='categorical',
    shuffle=False,
    seed=42
)
```

Loss and Accuracy

- ❖ In both models, accuracy and loss for training set are improved as value of epoch is increased.
- ❖ On validation set, we see some fluctuations but both accuracy and loss seem relatively stable.
- ❖ If we increase number of epochs to 20 or 30, we see gaps between the two curves for each model which means the model will overfit.
- ❖ We choose Model 1 as it has slightly higher accuracy and lower loss.



ROC and Precision-Recall Curves on Test Set



Accuracy and Confusion Matrix on Test Set

- ❖ Set threshold to 50% for probability.

- ❖ Accuracy: 79 %

- ❖ Confusion matrix = $\begin{bmatrix} 33 & 22 \\ 8 & 77 \end{bmatrix}$

Takeaways

- ❖ As training set is small, augmenting images seem to be important.
- ❖ To avoid overfitting, we had to use relatively large drop out values.
- ❖ It seems that accuracy levels over 80% are not easy to achieve with this set of data.
- ❖ Given that the ultimate goal of our model is to detect a disease, it makes sense to focus on reducing false negatives rather than increasing accuracy.

Future Research

- ❖ Study further optimizing the models.
 - ❖ Investigate using additional patients' information to further enhance models.
-

Thank You!

- ❖ Kaggle and [Dr. Joseph Paul Cohen](#) (University of Montreal) for providing data
 - ❖ Lucas Allen for guidance during project
-

Questions?
