PNEUMONIA DETECTION

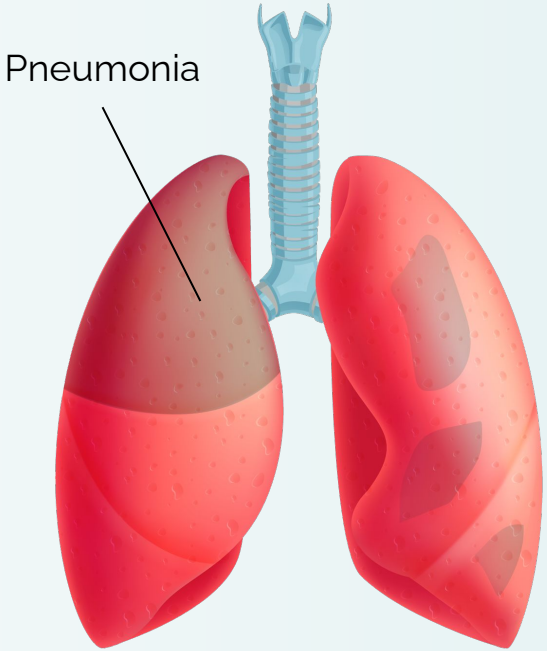
USING MACHINE LEARNING

Jay Raval

TABLE OF CONTENTS

01

Introduction to Pneumonia Detection



Introduction to Pneumonia Detection

What is Pneumonia?

A respiratory disease that affects the lungs, which could ultimately result in death.

How Do We Identify It?

One major sign of pneumonia is white spots in the lungs, which can be identified through image detection of chest X-rays.

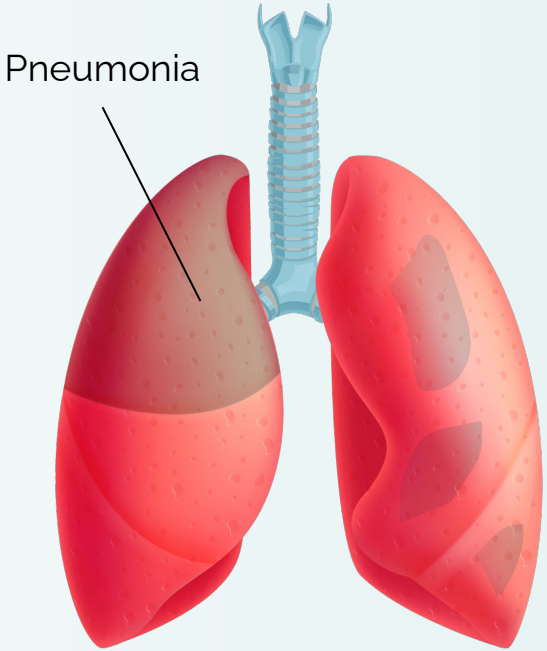
Further Details

X-rays work by passing radiation through the body, which can create an accurate image because the radiation passes through some materials more easily than others. The waves will pass right through air and water; in comparison, they will not penetrate through fat, tissue, or bone well.

Diagnosing pneumonia early is crucial, as it could significantly worsen if not detected.

02

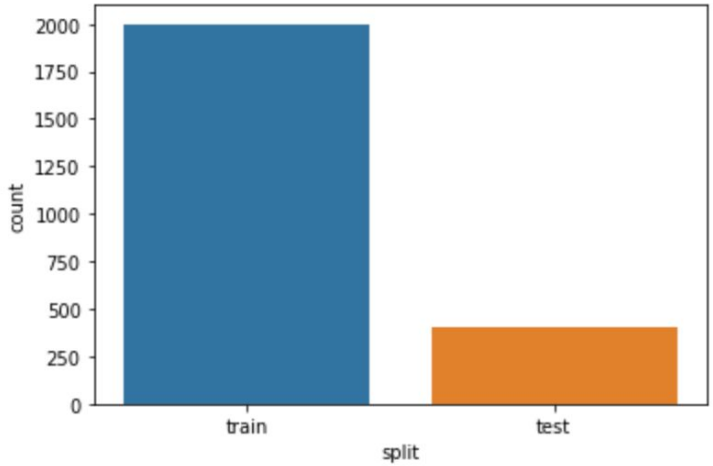
Modeling



Train vs. Test Data, Healthy vs. Pneumonia Images

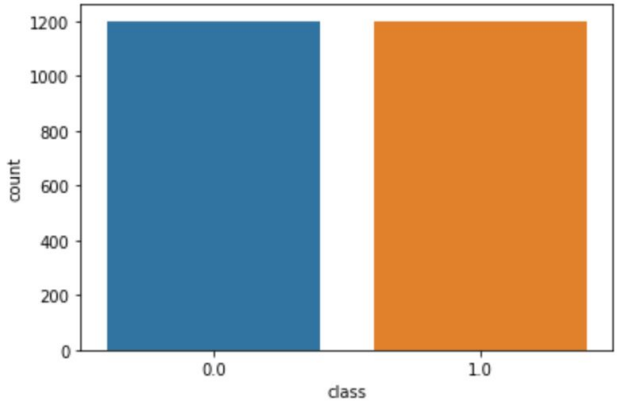
```
### YOUR CODE HERE
metadata.groupby(['split']).count()
sns.countplot(x = 'split', data = metadata)
### END CODE
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff3f46430d0>
```




```
[16] # grab our seaborn visualization toolbox!
     import seaborn as sns

     ### YOUR CODE HERE
     metadata.groupby(['class']).count()
     sns.countplot(x = 'class', data = metadata)
     ### END CODE
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff3f46e3650>
```

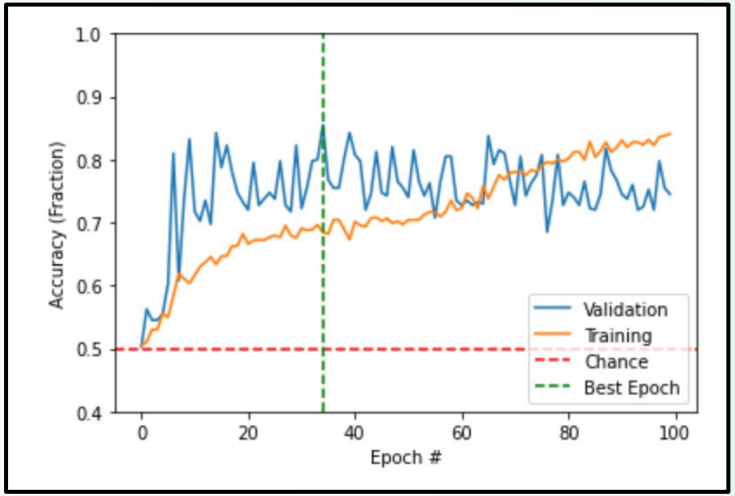


MLP & Dense Neural Networks

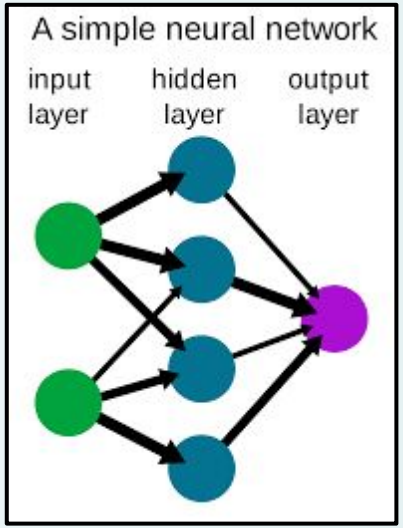
```python
#YOUR CODE HERE
from sklearn.neural_network import MLPClassifier

(train_data, train_labels) = get_train_data(flatten = True)
(test_data, test_labels) = get_test_data(flatten = True)

model = MLPClassifier(hidden_layer_sizes=(5))

model.fit(train_data, train_labels)

pred = model.predict(test_data)

accuracy = accuracy_score(test_labels, pred)
print(accuracy)

#YOUR CODE HERE
```

```
0.5
```

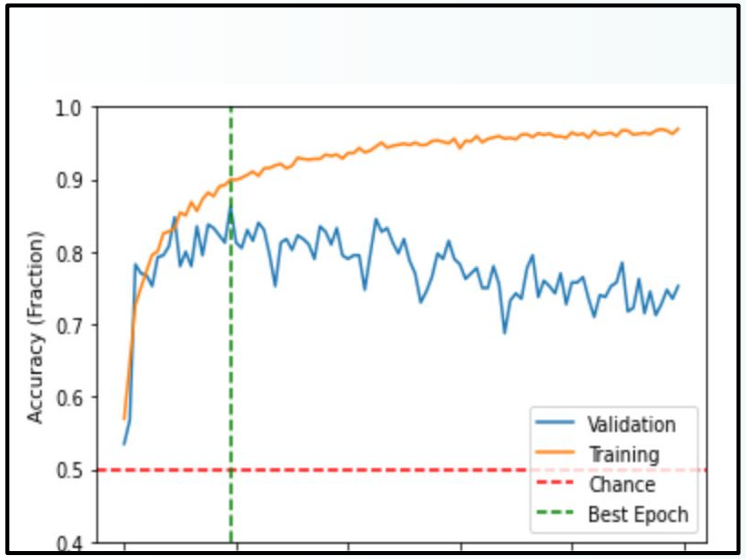
MLP Classifier Neural Network Code

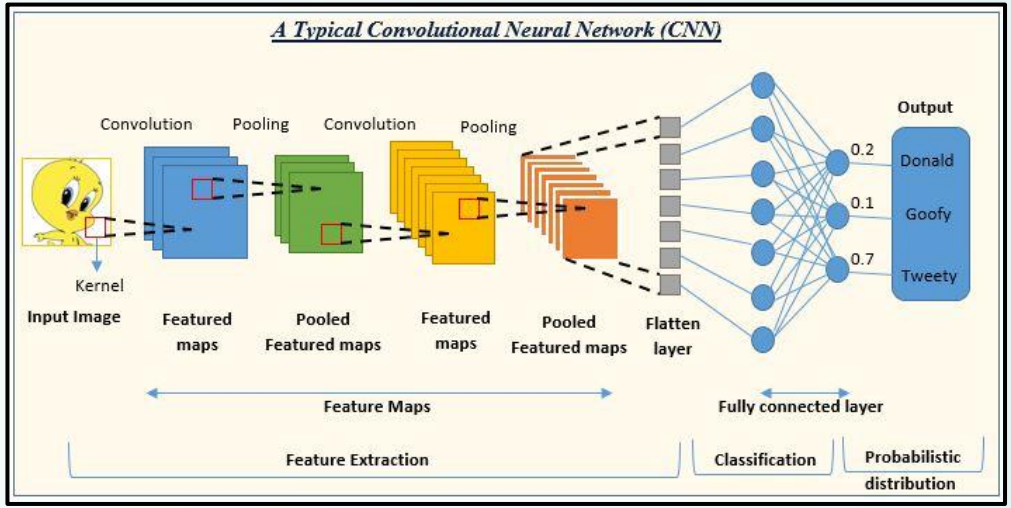


Dense Neural Network Accuracy Graph



CNNs (Convolutional Neural Networks)

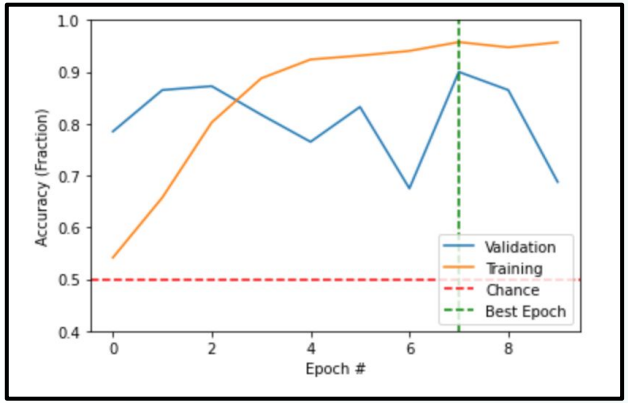


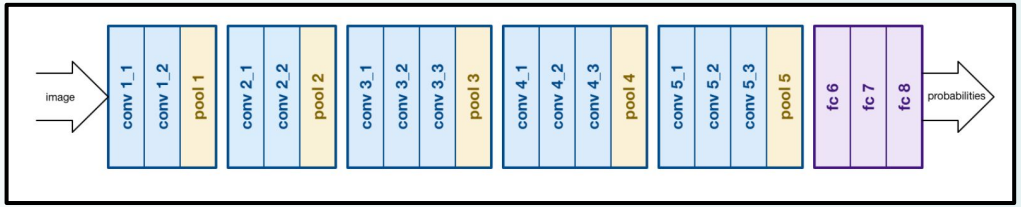


Input Layer: Images of lungs
Convolutional Layer: 64 x 64 layers (3 layers)

Transfer Learning Model (VGG16)

```
[14] ### YOUR CODE HERE
     train_data, train_labels = get_train_data()
     test_data, test_labels = get_test_data()

     transfer = TransferClassifier(name = 'VGG16')

     transfer.fit(train_data, train_labels, epochs = 10, validation_data = (test_data, test_labels), shuffle = True, callbacks

     plot_acc(transfer.history)

     ### END CODE
```

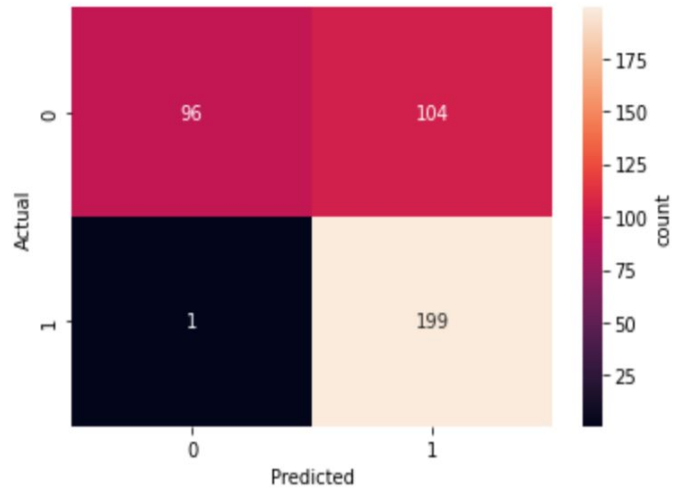


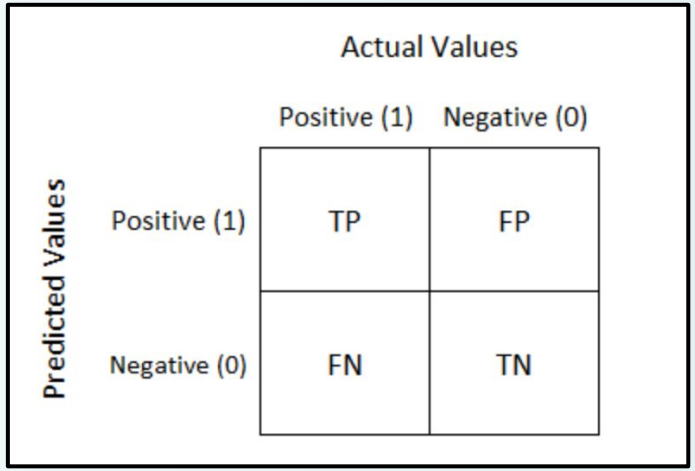


I decided to train the data using a new, expert CNN model (VGG16). This particular model has experience with over 14 million images, so it should be useful when distinguishing between healthy and pneumonia x-rays.

Confusion Matrix

```
sns.heatmap(confusion, annot = True, fmt = 'd', cbar_kws={'label':'count'});
plt.ylabel('Actual');
plt.xlabel('Predicted');
```



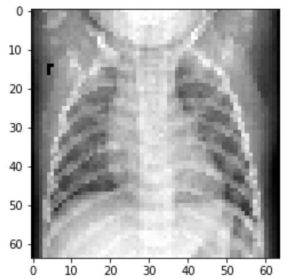


The confusion matrix is a way to evaluate the performance of the model by analyzing the number of true and false positives and negatives. In this matrix, I analyzed the Transfer Learning VGG16 Model.

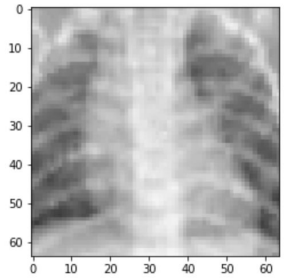
Augmentation

```
#YOUR CODE HERE
image = train_data[0]
plot_one_image(image)
new_image = scale(image, scale = 1.5)
plot_one_image(new_image)
```

Label:

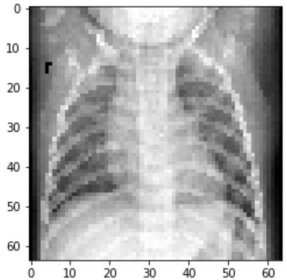


Label:

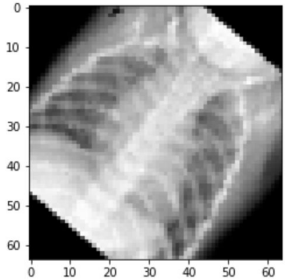


```
image = train_data[0]
plot_one_image(image)
new_image = rotate(image, rotate = 40)
plot_one_image(new_image)
```

Label:

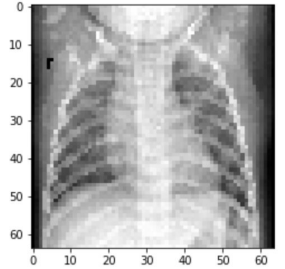


Label:

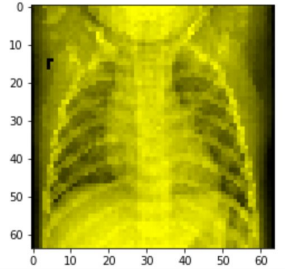


```
#YOUR CODE HERE
image = train_data[0]
plot_one_image(image)
new_image = remove_color(image, channel = 2)
plot_one_image(new_image)
```

Label:



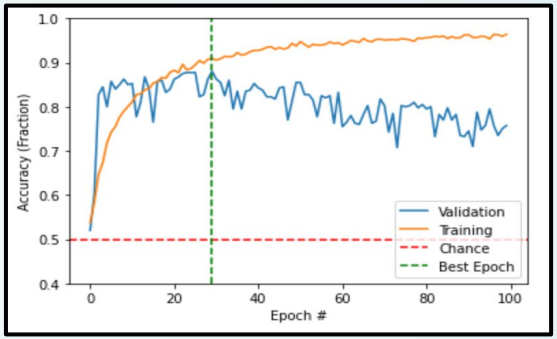
Label:



In order to allow the CNN model to handle various forms of scans. I trained it using augmented data. This augmentation improves the model and allows it to create more accurate predictions by exposing the model to “imperfect” scans. Imperfect scans mean tilted, rotated, or discolored.

Application (Field Data in CNNs)

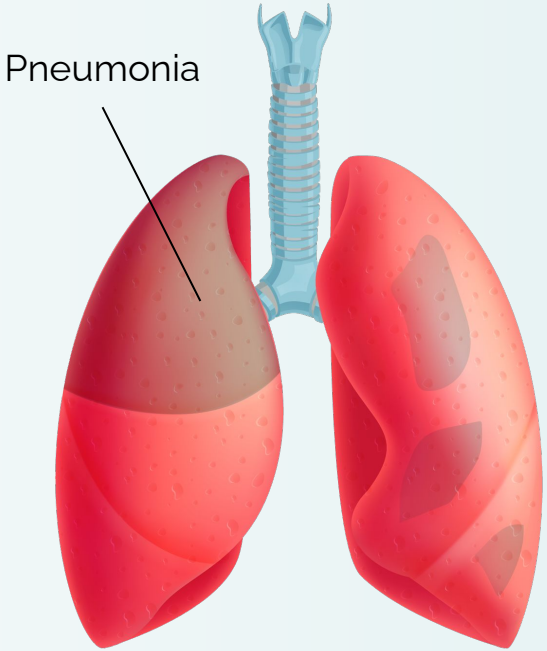
```
train_data, train_labels = get_train_data()
test_data, test_labels   = get_test_data()
field_data, field_labels   = get_field_data()

### YOUR CODE HERE

cnnnew = CNNClassifier(num_hidden_layers = 2)

cnnnew_history = cnnnew.fit(train_data, train_labels, epochs = 100, validation_data = (test_data, test_labels), shuffle = True, callbacks = [monitor])

plot_acc(cnnnew_history)

y_pred = (cnnnew.predict(field_data) > 0.5).astype("int32")
print(accuracy_score(field_labels, y_pred))

### END CODE
```



Here, I incorporated field data into the new CNN model. Though it didn't impact the model's accuracy greatly, the field data introduced it model to a larger dataset. This made the model more applicable to the real world.

03

ML App Deployment



Model Deployment

- Website using local Streamlit server (in development)
- Coded with Python
- Takes picture input and outputs “Healthy” or “Pneumonia”
- Possible to update to include TB

```
#@title Run this to host our website! { display-mode: "form" }

app = Flask(__name__)
run_with_ngrok(app)
CORS(app)

simple_webpage = "simple-webpage.html"
simple_html = codecs.open(simple_webpage, 'r').read()

@app.route("/")
def home():
    return simple_html

app.run()
```

Streamlit - Model Code

Old HTML Server Demo