Created By : MOHAMAD ZIKRY BIN
MOHAMAD SHUKRY

Image classification of PNEUMONIA diseases

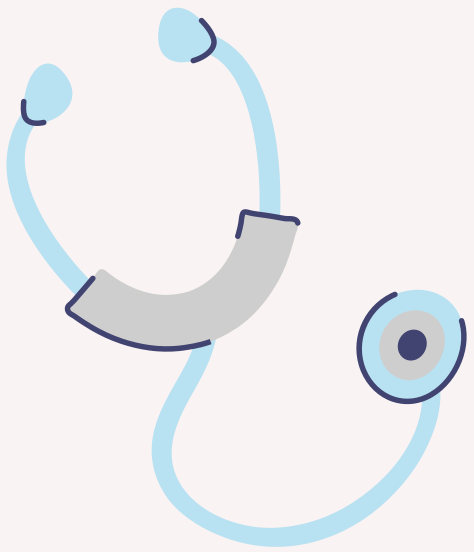
Table of content

1. **Use case summary**
2. **Data Understanding**
3. **Model Architecture**
4. **Model Code**
5. **Result**
6. **Demo**
7. **Reference**

Use case Summary

PNEOMONIA disease

PNEOMONIA is a type of lung infection that is caused by bacteria , fungi or viruses .Pneumonia can caused the swelling of lung tissue and it can cause the lungs to develop fluid or pus in the lungs .There are 2 type of PNUMENIA which is Bacterial Pneumonia and Viral Pneumonia. Bacterial Pneumonia is more severe compare to Viral Pneumonia as Viral Pneumonia tends to recover on its own.



Data Understanding

Pneumonia data set

Type Data Set

Pneumonia Image data set-Xray

Size of data set

- Training data set-5224
- Testing data set-624

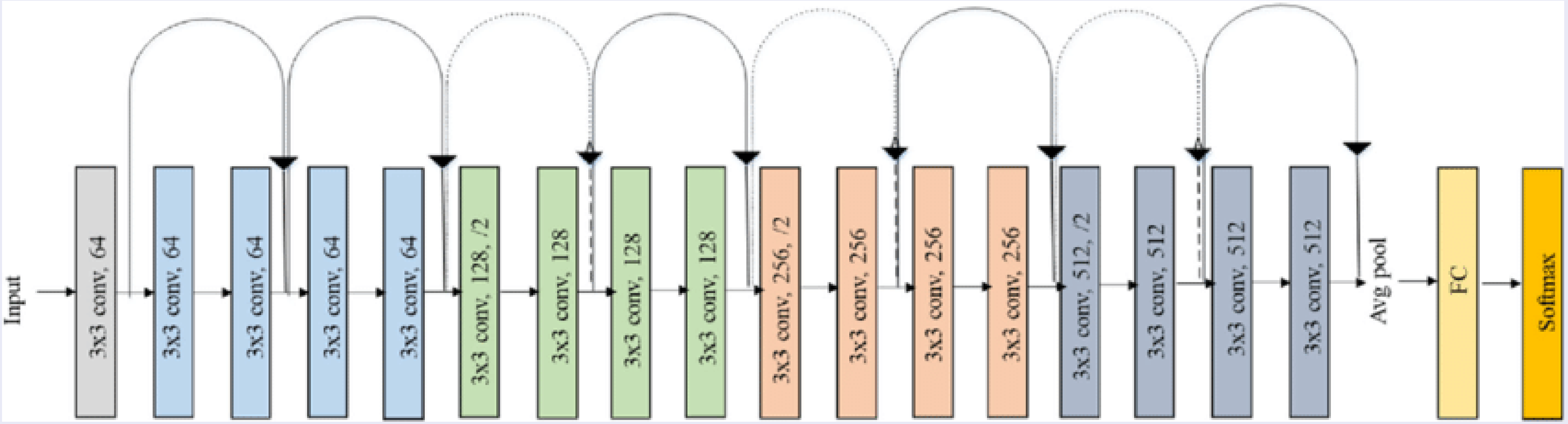
Data Source

Data Source: -

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

Model Architecture

Model Architecture



Code of the model

Code of the model

Image Transformation

```
#Define the Data Taransformation
transform = transforms.Compose([
    transforms.Resize((224, 224)), #Resiszing the image for Resnet-18 Algorithm implementation from 513 x 512 to 224x224
    transforms.RandomHorizontalFlip(), #Data Augmentation process where the image are randomly roting haorizontally
    transforms.RandomRotation(20),  # data Augmatation process where the image will rotate in a range of  +- 20 degrees
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

# Load the data Set
train_dataset = datasets.ImageFolder(root='/content/drive/MyDrive/train_set', transform=transform)
test_dataset = datasets.ImageFolder(root='/content/drive/MyDrive/test_set', transform=transform)

train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=32, shuffle=False)
```

Code of the model

```
[ ] model = models.resnet18(pretrained=True)
    model
```

Implement pre-trained model

```
[ ] #Freeze the layers
    for param in model.parameters():
        param.requires_grad = False
```

```
[ ] for param in model.layer4.parameters():
        param.requires_grad = True
```

Freeze the layer of the model and unfreeze layer 4.0 and 4.1

```
#Add fully connected layer to model development
model.fc = nn.Linear(num_features, 2)
model = model.to(device)
model
```

Fine tine the model by adding a fully connected layer

Code of the model

```
#define the loss function and Optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
```

Cross Entropy loss function and adam optimizer

```
#Set the random seed
torch.manual_seed(42)
torch.cuda.manual_seed(42)
```

Set Random seed to 42

Code of the model

```python
#### Train model
train_loss=[]
train_accuary=[]
test_loss=[]
test_accuary=[]

num_epochs = 10   #(set no of epochs)
start_time = time.time() #(for showing time)
# Start loop
for epoch in range(num_epochs): #(loop for every epoch)
    print("Epoch {} running".format(epoch)) #(printing message)
    """ Training Phase """
    model.train()    #(training model)
    running_loss = 0.   #(set loss 0)
    running_corrects = 0
    # load a batch data of images
    for i, (inputs, labels) in enumerate(train_loader):
        inputs = inputs.to(device)
        labels = labels.to(device)
        # forward inputs and get output
        optimizer.zero_grad()
        outputs = model(inputs)
        _, preds = torch.max(outputs, 1)
        loss = criterion(outputs, labels)
```

```python
        # get loss value and update the network weights
        loss.backward()
        optimizer.step()
        running_loss += loss.item()
        running_corrects += torch.sum(preds == labels.data).item()
    epoch_loss = running_loss / len(train_dataset)
    epoch_acc = running_corrects / len(train_dataset) * 100.
    # Append result
    train_loss.append(epoch_loss)
    train_accuary.append(epoch_acc)
    # Print progress
    print('[Train #{}] Loss: {:.4f} Acc: {:.4f}% Time: {:.4f}s'.format(epoch+1, epoch_loss, epoch_acc, time.time() -start_time))
```

Train Model

Code of the model

```
#Testing Phase
  model.eval()
  with torch.no_grad():
      running_loss = 0.
      running_corrects = 0
      for inputs, labels in test_loader:
          inputs = inputs.to(device)
          labels = labels.to(device)
          outputs = model(inputs)
          _, preds = torch.max(outputs, 1)
          loss = criterion(outputs, labels)
          running_loss += loss.item()
          running_corrects += torch.sum(preds == labels.data).item()
      epoch_loss = running_loss / len(test_dataset)
      epoch_acc = running_corrects / len(test_dataset) * 100.
      # Append result
      test_loss.append(epoch_loss)
      test_accuary.append(epoch_acc)
      # Print progress
      print('[Test #{}] Loss: {:.4f} Acc: {:.4f}% Time: {:.4f}s'.format(epoch+1, epoch_loss, epoch_acc, time.time()- start_time))
```

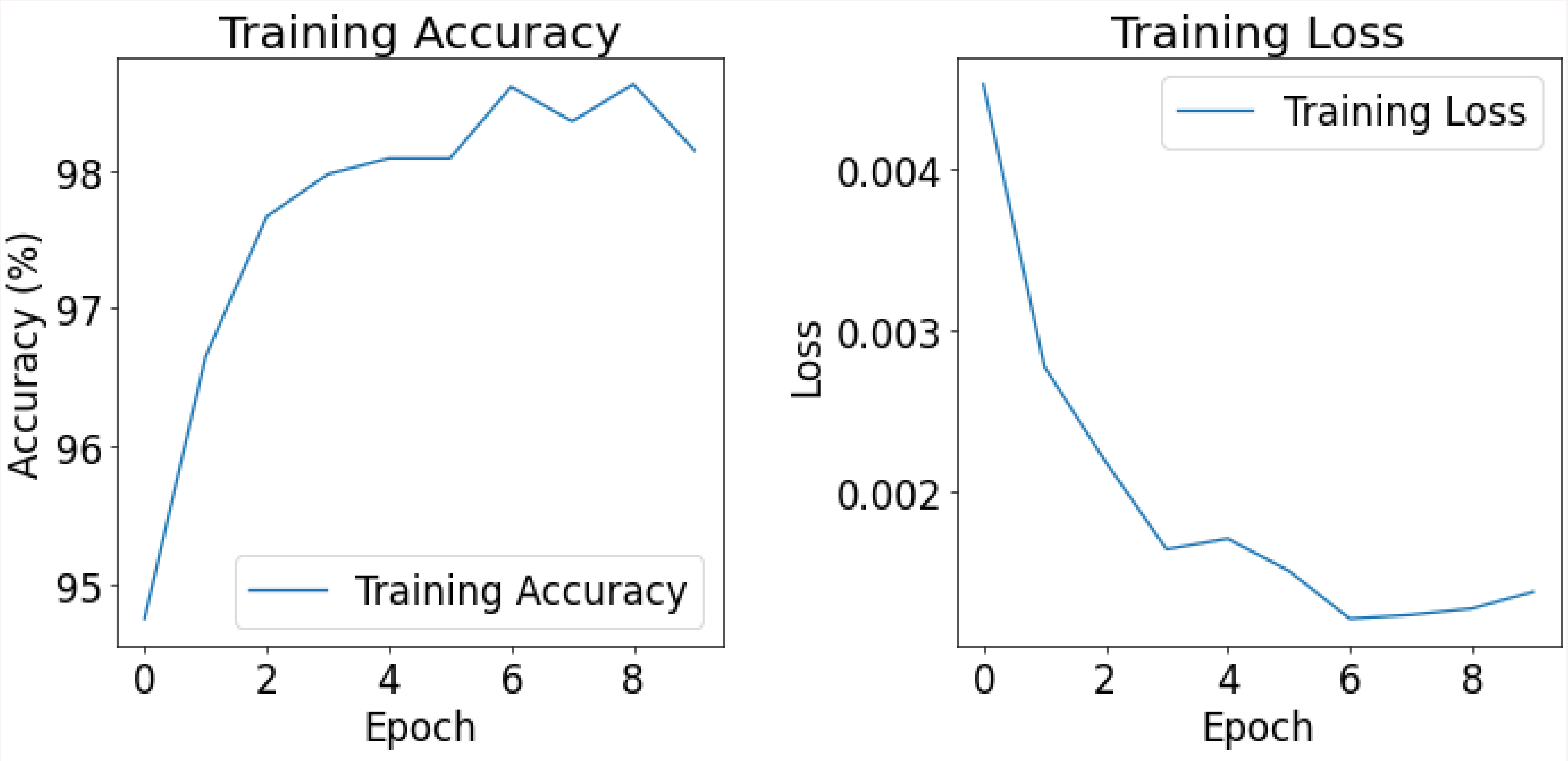
Test Model

Result

Training Accuracy and Loss

Accuracy

Loss



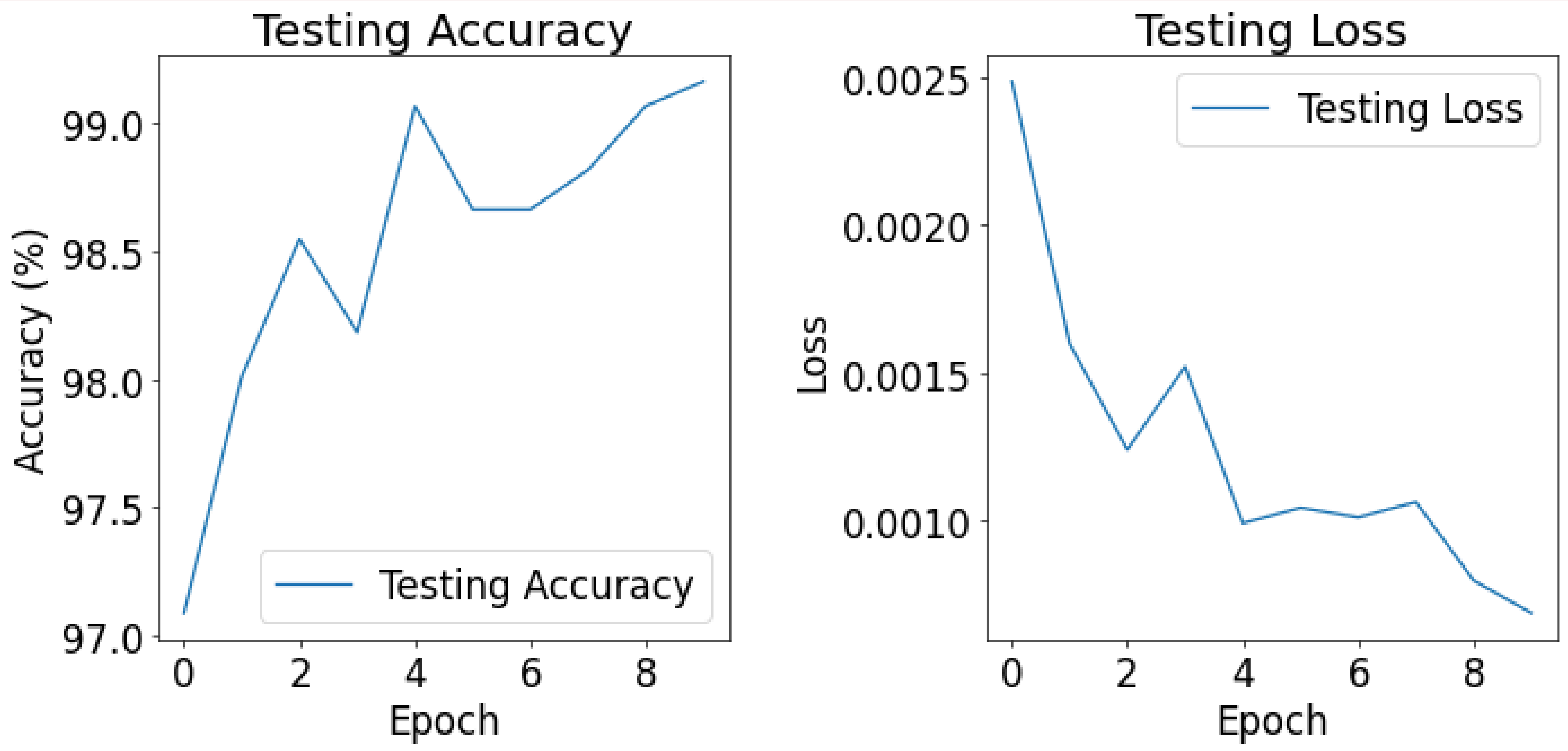
98.1432%

0.0014

Testing Accuracy and Loss

Accuracy

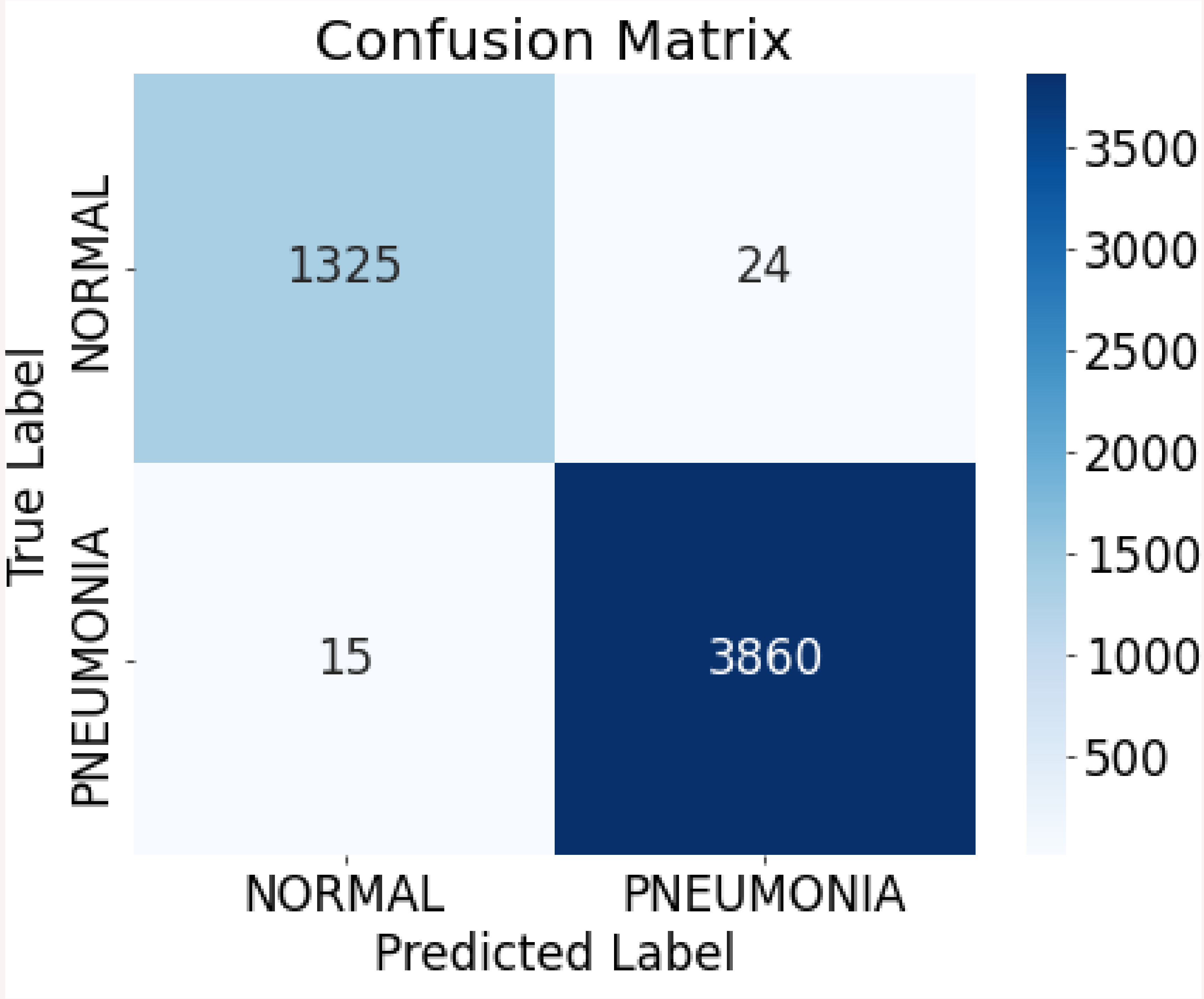
Loss



99.1577%

0.0007

Matrix confusion



Demo

Challanges

The Current Landscape of Medicine

Time of train and Testing the Model

Web Application

Conclusion

Conclusion

Encouraging discussion and questions

In conclusion the development of the PNEUMONIA Image classification has presented the huge potential of AI to society . It presents the technological advancement through various fields and the potential to grow more.

Reference

- Kei Dang, (27 January 2023), Deep learning — Computer vision (CV) using Transfer Learning (ResNet-18) in Pytorch — Skin cancer classification. https://medium.com/@chaouch.thameur.tc61/image-classification-transfer-learning-and-fine-tuning-using-tensorflow-8cd5ea84c707.

- Sansak Chilamkurthy, (nd), Transfer Learning for Computer Vision Tutorial https://medium.com/@chaouch.thameur.tc61/image-classification-transfer-learning-and-fine-tuning-using-tensorflow-8cd5ea84c707.

- Rohit Mundi, (2 December 2021), ResNet — Understand and Implement from scratchhttps://medium.com/analytics-vidhya/resnet-understand-and-implement-from-scratch-d0eb9725e0db.

- Computer Vision Engineer, (3 July 2023), Object Detection Web Application with Python, Streamlit and Detectron2 | Computer vision tutorial. https://youtu.be/n_eMARPqBZI?si=umBTVpPBM5QIv_Tv.

Thank you