COVID-19 Pneumonia Detection using CNNs

In this project, we developed a Convolutional Neural Network (CNN) to classify chest X-ray images into three categories: normal, pneumonia, and COVID-19. The primary objective was to leverage deep learning techniques to accurately identify and distinguish between these conditions, providing a valuable tool for aiding medical diagnosis.

The dataset used for this project was the COVID-19 Radiography Database, which contains labeled chest X-ray images for normal cases, viral pneumonia cases, and COVID-19 cases. The images were preprocessed and augmented to enhance the model's robustness and generalization capabilities.

Our CNN model was designed with multiple convolutional, max-pooling. We employed TensorFlow and Keras for model building, training, and evaluation. Real-time data augmentation techniques were applied to further improve the model's performance.

Throughout the training process, we monitored the model's accuracy and loss using matplotlib, making iterative adjustments to optimize the results. The final model demonstrated high accuracy in classifying the X-ray images into the three target categories.

```
import os
import cv2
import numpy as np
from sklearn.preprocessing import LabelEncoder
import tensorflow
from tensorflow import keras
from keras.layers import Conv2D,MaxPool2D,Flatten,Dense
from keras.models import Sequential
from sklearn.model_selection import train_test_split
```

```
main_path='/content/drive/MyDrive/Datasets/Pneumonia'
sub_dir=os.listdir(main_path)
sub_dir
```

```
['COVID', 'NORMAL', 'PNEUMONIA']
```

```
x=[]
y=[]
for subdir in sub_dir:
  sub_path=os.path.join(main_path,subdir)
  img_names=os.listdir(sub_path)
  for img in img_names:
    img_path=os.path.join(sub_path,img)
    img_array=cv2.imread(img_path,0)
    # print(img_array.shape)
    # break
    if img_array is not None:
       img_resized = cv2.resize(img_array, (232,232))
       img_final=img_resized.reshape(232,232,1)
       img_final=img_final/255
       x.append(img_resized)
       y.append(subdir)
```

```
len(x)
```

```
5224
```

```
len(y)
```

```
5224
```

```
x=np.array(x)
```

```
x.shape
```

```
(5224, 232, 232)
```

```
label=LabelEncoder()
y=label.fit_transform(y)
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

CNN model building

We build a CNN model to classify the images into their respective categories. The architecture of our model is:

```
model=Sequential([Conv2D(32,(3,3),input_shape=(232,232,1),activation='relu'),
                  MaxPool2D(2,2),
                  Conv2D(16,(3,3),activation='relu'),
                  MaxPool2D(2,2),
                  Flatten(),
                  Dense(32,activation='relu'),
                  Dense(16,activation='relu'),
                  Dense(3,activation='softmax')])
```

```
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

```
acc=model.fit(x_train,y_train,epochs=10,validation_data=(x_test,y_test))
```

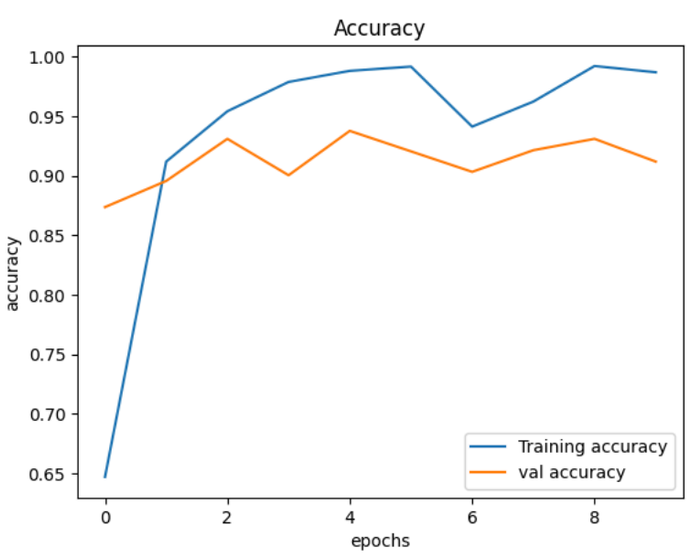
```
Epoch 1/10
131/131 [==============================] - 6s 32ms/step - loss: 16.3309 - accuracy: 0.6468 - val_loss: 0.4402 - val_accuracy: 0.8737
Epoch 2/10
131/131 [==============================] - 4s 28ms/step - loss: 0.3191 - accuracy: 0.9119 - val_loss: 0.3507 - val_accuracy: 0.8957
```

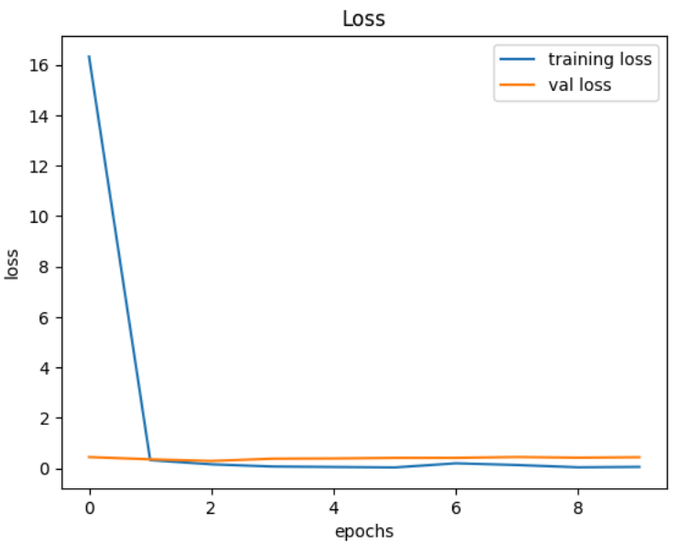
```
Epoch 3/10
131/131 [==============================] - 4s 30ms/step - loss: 0.1528 - accuracy: 0.9543 - val_loss: 0.2855 - val_accuracy: 0.9311
Epoch 4/10
131/131 [==============================] - 4s 29ms/step - loss: 0.0651 - accuracy: 0.9789 - val_loss: 0.3750 - val_accuracy: 0.9005
Epoch 5/10
131/131 [==============================] - 4s 29ms/step - loss: 0.0464 - accuracy: 0.9883 - val_loss: 0.3848 - val_accuracy: 0.9378
Epoch 6/10
131/131 [==============================] - 4s 28ms/step - loss: 0.0290 - accuracy: 0.9919 - val_loss: 0.4081 - val_accuracy: 0.9206
Epoch 7/10
131/131 [==============================] - 4s 28ms/step - loss: 0.1921 - accuracy: 0.9414 - val_loss: 0.4100 - val_accuracy: 0.9033
Epoch 8/10
131/131 [==============================] - 4s 30ms/step - loss: 0.1249 - accuracy: 0.9624 - val_loss: 0.4421 - val_accuracy: 0.9215
Epoch 9/10
131/131 [==============================] - 4s 28ms/step - loss: 0.0339 - accuracy: 0.9923 - val_loss: 0.4166 - val_accuracy: 0.9311
Epoch 10/10
131/131 [==============================] - 4s 28ms/step - loss: 0.0497 - accuracy: 0.9871 - val_loss: 0.4342 - val_accuracy: 0.9120
```

```
model.evaluate(x_test,y_test)
```

```
33/33 [==============================] - 0s 8ms/step - loss: 0.4342 - accuracy: 0.9120
[0.4341915249824524, 0.9119617342948914]
```

```
#plotting graphs for accuracy
import matplotlib.pyplot as plt
plt.figure(0)
plt.plot(acc.history['accuracy'],label='Training accuracy')
plt.plot(acc.history['val_accuracy'], label='val accuracy')
plt.title('Accuracy')
plt.xlabel('epochs')
plt.ylabel('accuracy')
plt.legend()
plt.show()
plt.figure(1)
plt.plot(acc.history['loss'], label='training loss')
plt.plot(acc.history['val_loss'], label='val loss')
plt.title('Loss')
plt.xlabel('epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```






```
def prediction(path):
  img_array=cv2.imread(path,0)
  img_resize=cv2.resize(img_array,(232,232))
  img_final=img_resize.reshape(1,232,232,1)
  img_final=img_final/255
  y_pred=model.predict(img_final)
  pred_label=np.argmax(y_pred)
  pred_label=label.inverse_transform([pred_label])
  return pred_label
```

```
prediction('/content/drive/MyDrive/Datasets/COVID-19_Radiography_Dataset/COVID/images/COVID-10.png')
```

```
1/1 [==============================] - 0s 67ms/step
array(['COVID'], dtype='<U9')
```

Conclusion The final model achieved an impressive accuracy of 91.20% and less loss on the validation dataset. These results demonstrate the model's capability to accurately distinguish between normal, pneumonia, and COVID-19 cases, providing a valuable tool for assisting in medical diagnosis.