Machine Learning Case Study and Project
Predicting Fractures and Brain Tumors Using CNN

SY Computer Engineering - Batch C2
UCE2022632 - Vaidehi Sarag
UCE2022625 - Chinmayee Randive

Problem Statement:

Accurate diagnosis of medical conditions such as fractures and brain tumors is critical for effective treatment and patient care. However, the manual interpretation of medical images by healthcare professionals can be time-consuming and prone to errors. Therefore, there is a need to develop automated systems that can analyze medical images and assist in the early detection of fractures and brain tumors.

Introduction:

Medical imaging techniques, such as X-rays and MRI scans, provide valuable insights into the internal structures of the human body. Interpretation of these images traditionally relies on the expertise of radiologists and physicians. However, advancements in machine learning have opened up new possibilities for automating the analysis of medical images.

In this case study, we aim to leverage convolutional neural networks (CNNs), a powerful class of deep learning models, to develop predictive models for detecting fractures and brain tumors from medical images. By training CNN models on labeled datasets containing X-ray and MRI images, we seek to build robust classifiers capable of accurately identifying pathological conditions.

Tech Stack Used:

Platform : Google Colab, Streamlit (possible UI)
Language : Python

Libraries :

1. Kaggle API - datasets
2. Zipfile - extraction of dataset files
3. Os - file operations and directory management

4. Numpy - Mathematical and Array operations
5. Matplotlib.pyplot - plotting and visual representation
6. Matplotlib.image - reading and displaying images
7. Cv2 (OpenCV) - Computer Vision tasks
8. PIL - Python Image Library - imaging
9. Scikit - Machine Learning model
10. Tensorflow and Keras - Training ML models and Building Neural Networks
11. Streamlit - Possible Web Page Creation for User Interface

---

Dataset Information:

Source : Kaggle
Type : Images
Using a key instead of downloading entire datasets of images.

For X-ray Analysis and Fracture Detection:

```
!pip install kaggle

# configuring the path of Kaggle.json file
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json

#importing dataset
!kaggle datasets download -d
vuppalaadithyasairam/bone-fracture-detection-using-xrays
!kaggle datasets download -d bebofekry/bone-fracture-atlas-ds-compressed
```

For MRI Analysis and Brain Tumor Detection:

```
!pip install kaggle

# configuring the path of Kaggle.json file
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d
shreyag1103/brain-mri-scans-for-brain-tumor-classification
```

Code and Output:

For X-ray Analysis and Fracture Detection:
(Google Colab PDF is attached externally.)
Image Processing :

```
data=[]

fractured_path1='/content/archive (6)/train/fractured/'
fractured_path2='/content/FracAtlas/FracAtlas/images/Train/Fractured/'

for img_file in fractured_files1:
  image = Image.open(fractured_path1 + img_file)
  image = image.resize((128,128))
  image = image.convert('RGB')
  image = np.array(image)
  data.append(image)

for img_file in fractured_files2:
  image = Image.open(fractured_path2 + img_file)
  image = image.resize((128,128))
  image = image.convert('RGB')
  image = np.array(image)
  data.append(image)


not_fractured_path1='/content/archive (6)/train/not fractured/'
not_fractured_path2='/content/FracAtlas/FracAtlas/images/Train/Non_fractur
ed/'

for img_file in not_fractured_files1:
  image = Image.open(not_fractured_path1 + img_file)
  image = image.resize((128,128))
  image = image.convert('RGB')
  image = np.array(image)
```

```
  data.append(image)

for img_file in not_fractured_files2:
  image = Image.open(not_fractured_path2 + img_file)
  image = image.resize((128,128))
  image = image.convert('RGB')
  image = np.array(image)
  data.append(image)
```

CNN:

```
import tensorflow as tf
from tensorflow import keras

num_of_classes = 2

model = keras.Sequential()

model.add(keras.layers.Conv2D(32, kernel_size=(3,3), activation='relu',
input_shape=(128,128,3)))
model.add(keras.layers.MaxPooling2D(pool_size=(2,2)))


model.add(keras.layers.Conv2D(64, kernel_size=(3,3), activation='relu'))
model.add(keras.layers.MaxPooling2D(pool_size=(2,2)))

model.add(keras.layers.Flatten())

model.add(keras.layers.Dense(128, activation='relu'))
model.add(keras.layers.Dropout(0.5))

model.add(keras.layers.Dense(64, activation='relu'))
model.add(keras.layers.Dropout(0.5))


model.add(keras.layers.Dense(num_of_classes, activation='sigmoid'))
```

```
# compile the neural network
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['acc'])

# training the neural network
history = model.fit(X_train_scaled, Y_train, validation_split=0.1,
epochs=5)


# Input from User and Prediction

input_image_path = input('Path of the image to be predicted: ')

input_image = cv2.imread(input_image_path)

cv2_imshow(input_image)

input_image_resized = cv2.resize(input_image, (128,128))

input_image_scaled = input_image_resized/255

input_image_reshaped = np.reshape(input_image_scaled, [1,128,128,3])

input_prediction = model.predict(input_image_reshaped)

print(input_prediction)


input_pred_label = np.argmax(input_prediction)

print(input_pred_label)


if input_pred_label == 1:

  print('The bone is fractured')
```

```
else:

  print('The bone is not fractured')
```

For MRI Analysis and Brain Tumor Detection:
(Google Colab PDF is attached externally.)

CNN:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,
random_state=2)

# scaling the data

X_train_scaled = X_train/255

X_test_scaled = X_test/255

import tensorflow as tf
from tensorflow import keras

num_of_classes = 4

model = keras.Sequential()

model.add(keras.layers.Conv2D(32, kernel_size=(3,3), activation='relu',
input_shape=(128,128,3)))
model.add(keras.layers.MaxPooling2D(pool_size=(2,2)))


model.add(keras.layers.Conv2D(64, kernel_size=(3,3), activation='relu'))
model.add(keras.layers.MaxPooling2D(pool_size=(2,2)))

model.add(keras.layers.Flatten())

model.add(keras.layers.Dense(128, activation='relu'))
model.add(keras.layers.Dropout(0.5))

model.add(keras.layers.Dense(64, activation='relu'))
```

```
model.add(keras.layers.Dropout(0.5))


model.add(keras.layers.Dense(num_of_classes, activation='sigmoid'))

# compile the neural network
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['acc'])

# training the neural network
history = model.fit(X_train_scaled, Y_train, validation_split=0.1,
epochs=50)

input_image_path = input('Path of the image to be predicted: ')

input_image = cv2.imread(input_image_path)

cv2_imshow(input_image)

input_image_resized = cv2.resize(input_image, (128,128))

input_image_scaled = input_image_resized/255

input_image_reshaped = np.reshape(input_image_scaled, [1,128,128,3])

input_prediction = model.predict(input_image_reshaped)

print(input_prediction)


input_pred_label = np.argmax(input_prediction)

print(input_pred_label)


if input_pred_label == 1:

  print('The person has tumor')
```

```
elif input_pred_label == 2:
  print('The person has tumor')

elif input_pred_label == 3:
  print('The person has tumor')

elif input_pred_label == 4:
  print('The person does not have tumor')

else:
  print('The person does not have tumor.')
```

Conclusion:

The development of CNN-based models for predicting fractures and brain tumors from medical images represents a significant advancement in healthcare technology. These models have the potential to assist healthcare professionals in making accurate diagnoses and providing timely treatment to patients.

By leveraging machine learning techniques, we can improve the efficiency and accuracy of medical image analysis, leading to better patient outcomes. However, further research and validation are necessary to ensure the reliability and generalization of these models across different populations and imaging modalities.

In conclusion, CNN-based models offer promising opportunities for enhancing diagnostic capabilities in medical imaging and advancing the field of radiology. With continued research and innovation, we can harness the power of machine learning to improve healthcare delivery and patient care.