BRAIN TUMOR MRI IMAGES

Ofri Oren

Data Science Cohort

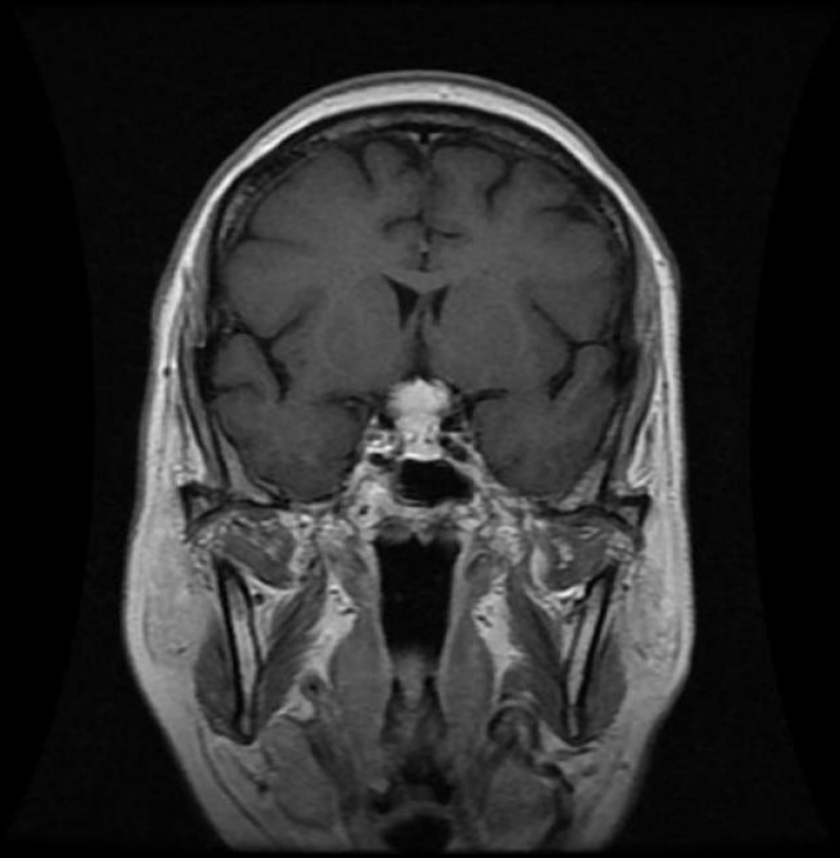
Capstone III

Machine Learning

table of contents

summary

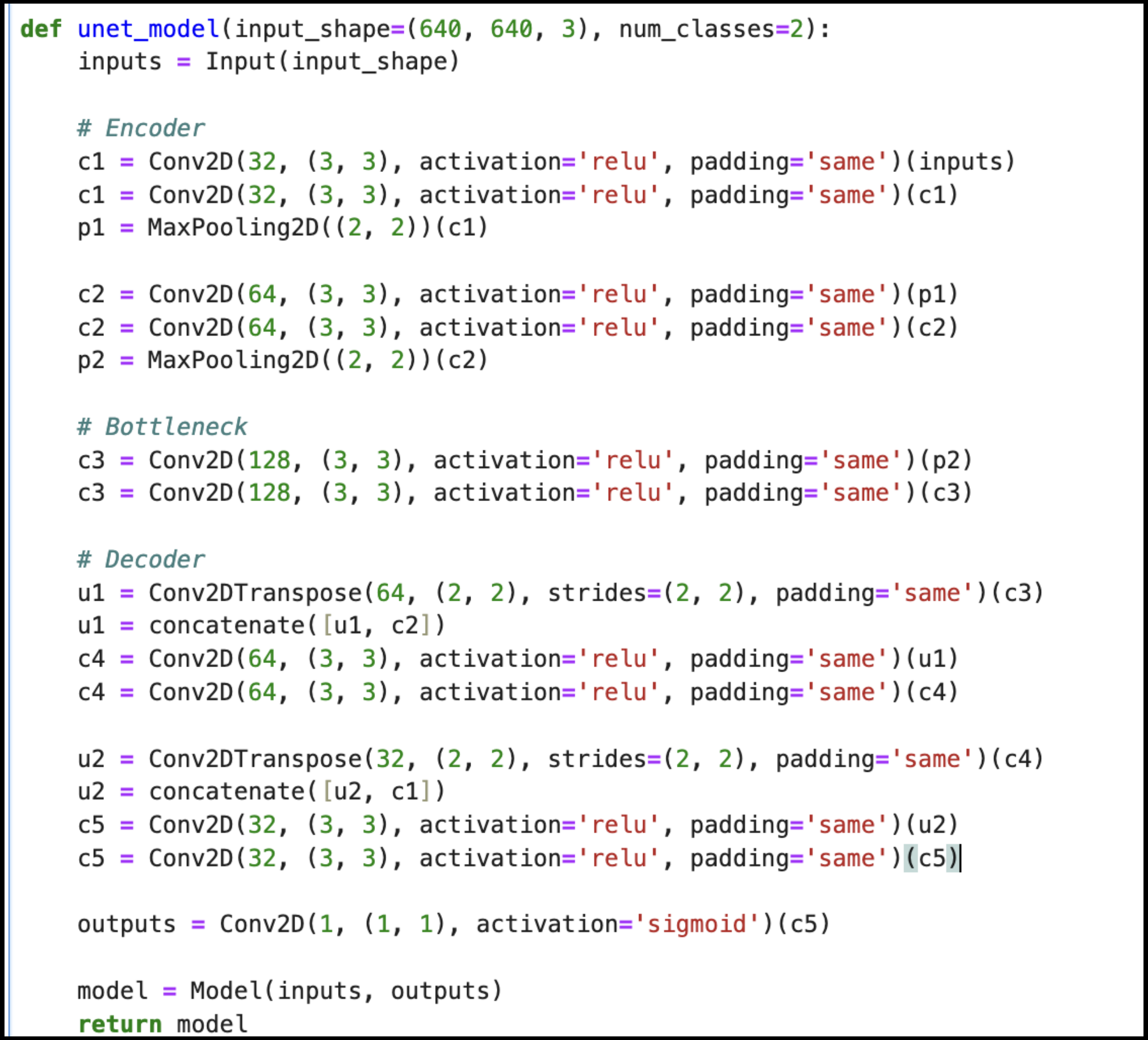
- MRI scans on Kaggle
- Pre-labeled segmentation of brain tumors
- Classification of the type of tumor based on location



exploratory data analysis

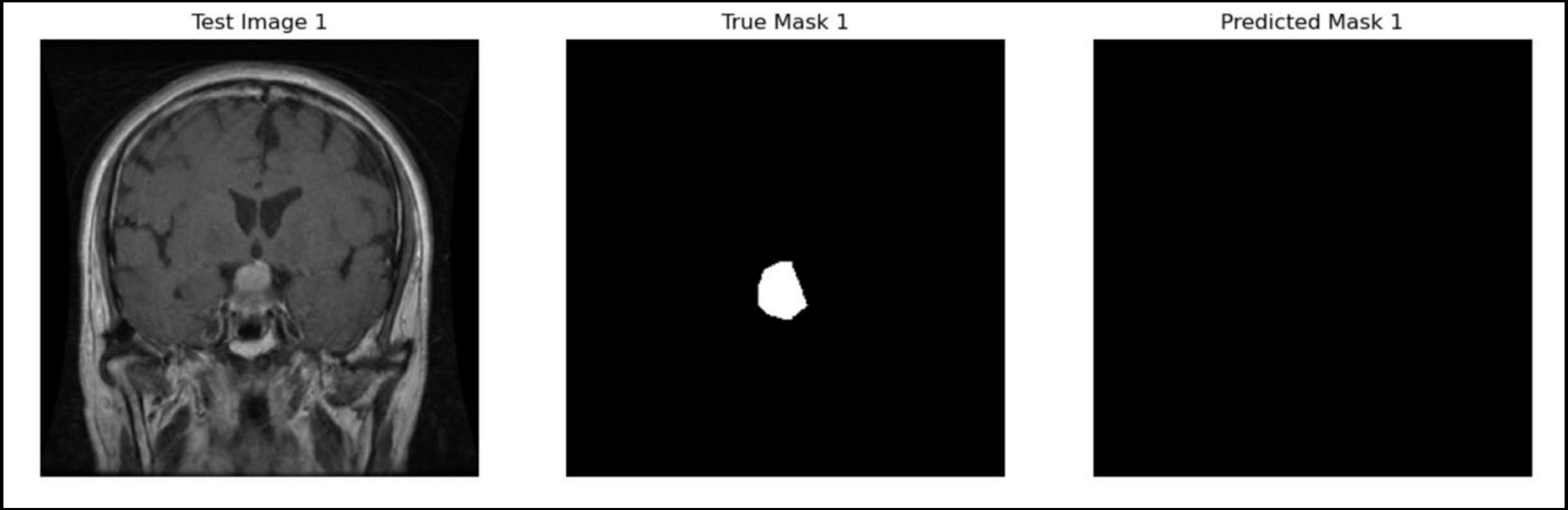
- Three types of tumors including pituitary, glioma, meningioma, and images of brains without tumors
- There were 827 images of pituitary tumors, 826 glioma, 822 meningioma, and 395 no-tumor images.

Processing & Modelling

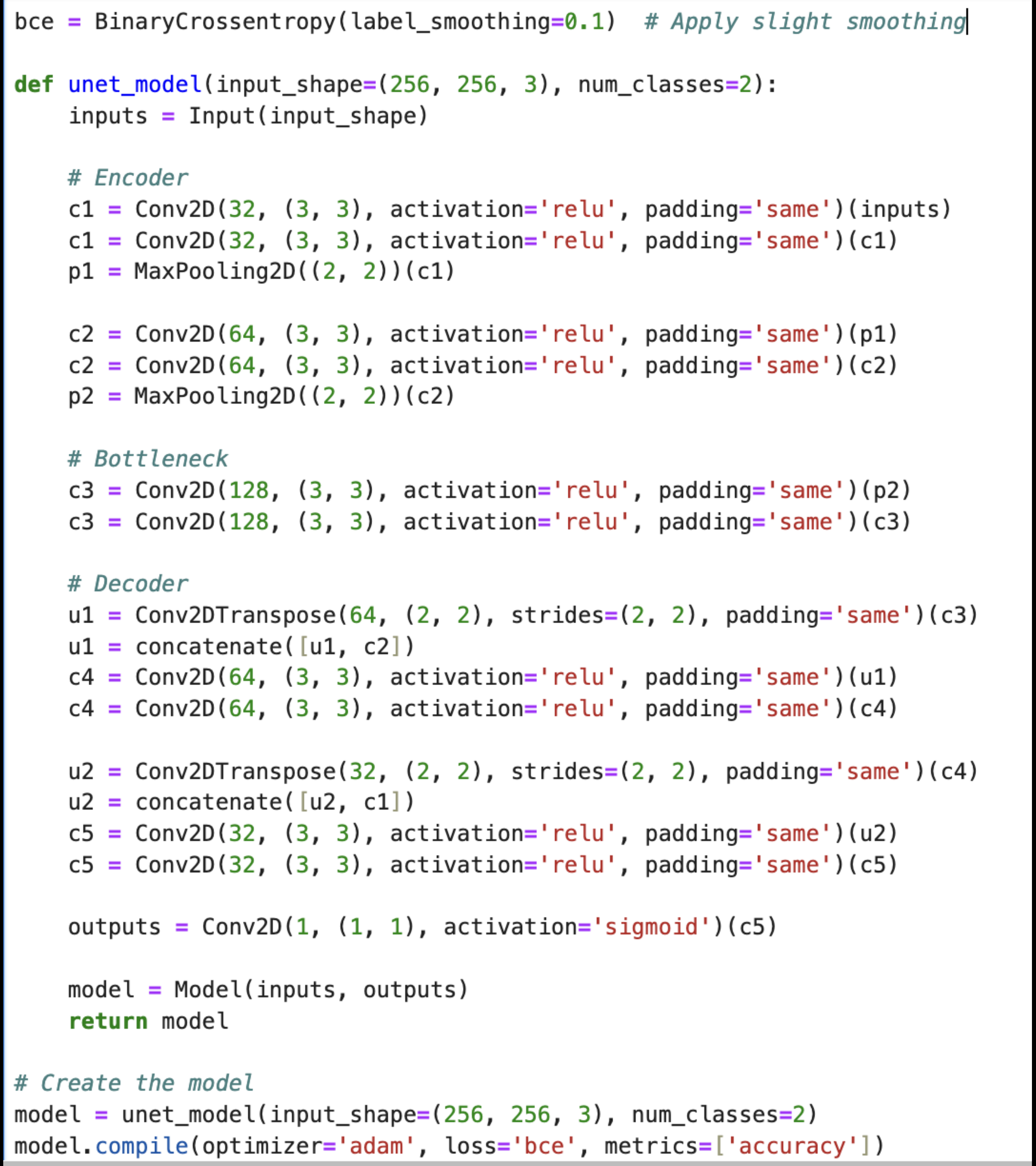


```
def unet_model(input_shape=(640, 640, 3), num_classes=2):
    inputs = Input(input_shape)

    # Encoder
    c1 = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)
    c1 = Conv2D(32, (3, 3), activation='relu', padding='same')(c1)
    p1 = MaxPooling2D((2, 2))(c1)

    c2 = Conv2D(64, (3, 3), activation='relu', padding='same')(p1)
    c2 = Conv2D(64, (3, 3), activation='relu', padding='same')(c2)
    p2 = MaxPooling2D((2, 2))(c2)

    # Bottleneck
    c3 = Conv2D(128, (3, 3), activation='relu', padding='same')(p2)
    c3 = Conv2D(128, (3, 3), activation='relu', padding='same')(c3)

    # Decoder
    u1 = Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same')(c3)
    u1 = concatenate([u1, c2])
    c4 = Conv2D(64, (3, 3), activation='relu', padding='same')(u1)
    c4 = Conv2D(64, (3, 3), activation='relu', padding='same')(c4)

    u2 = Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same')(c4)
    u2 = concatenate([u2, c1])
    c5 = Conv2D(32, (3, 3), activation='relu', padding='same')(u2)
    c5 = Conv2D(32, (3, 3), activation='relu', padding='same')(c5)

    outputs = Conv2D(1, (1, 1), activation='sigmoid')(c5)

    model = Model(inputs, outputs)
    return model
```

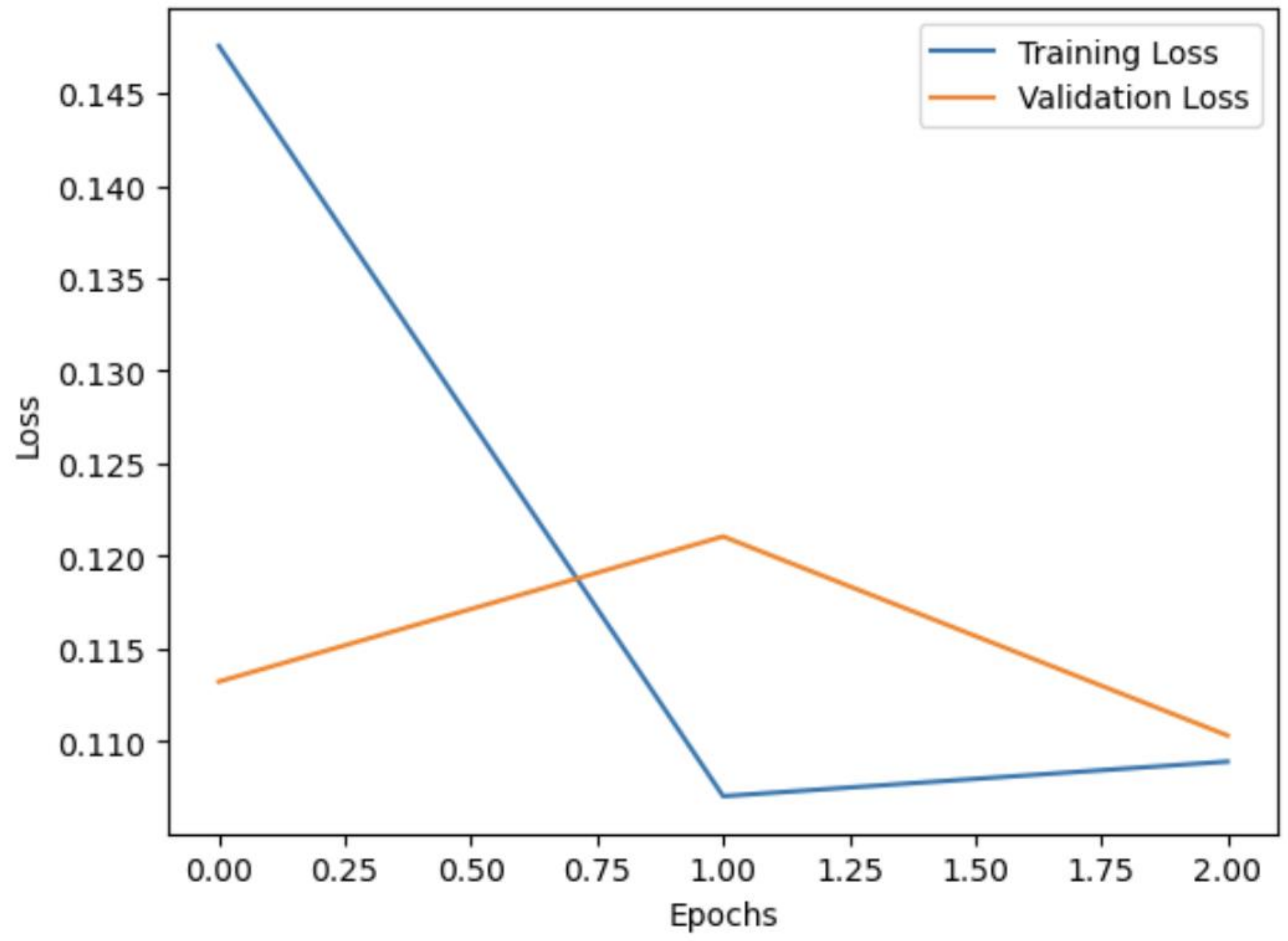
Processing & Modelling



Processing & Modelling



```python
bce = BinaryCrossentropy(label_smoothing=0.1)  # Apply slight smoothing

def unet_model(input_shape=(256, 256, 3), num_classes=2):
    inputs = Input(input_shape)

    # Encoder
    c1 = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)
    c1 = Conv2D(32, (3, 3), activation='relu', padding='same')(c1)
    p1 = MaxPooling2D((2, 2))(c1)

    c2 = Conv2D(64, (3, 3), activation='relu', padding='same')(p1)
    c2 = Conv2D(64, (3, 3), activation='relu', padding='same')(c2)
    p2 = MaxPooling2D((2, 2))(c2)

    # Bottleneck
    c3 = Conv2D(128, (3, 3), activation='relu', padding='same')(p2)
    c3 = Conv2D(128, (3, 3), activation='relu', padding='same')(c3)

    # Decoder
    u1 = Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same')(c3)
    u1 = concatenate([u1, c2])
    c4 = Conv2D(64, (3, 3), activation='relu', padding='same')(u1)
    c4 = Conv2D(64, (3, 3), activation='relu', padding='same')(c4)

    u2 = Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same')(c4)
    u2 = concatenate([u2, c1])
    c5 = Conv2D(32, (3, 3), activation='relu', padding='same')(u2)
    c5 = Conv2D(32, (3, 3), activation='relu', padding='same')(c5)

    outputs = Conv2D(1, (1, 1), activation='sigmoid')(c5)

    model = Model(inputs, outputs)
    return model

# Create the model
model = unet_model(input_shape=(256, 256, 3), num_classes=2)
model.compile(optimizer='adam', loss='bce', metrics=['accuracy'])
```



Processing & Modelling

evaluation and recommendations

- Anomaly detection
- Color mask tumor instead of polygon
- Stained MRI scans