```
!pip install tensorflow fastapi uvicorn python-multipart streamlit

import numpy as np
import os
import cv2
import zipfile
import requests
import shutil
from sklearn.model_selection import train_test_split
from tensorflow.keras import layers, models
from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.optimizers import Adam
```

```
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.64.1)
Requirement already satisfied: tensorboard<2.18,>=2.17 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.17.0)
Requirement already satisfied: keras>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.4.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.26.4)
Requirement already satisfied: starlette<0.39.0,>=0.37.2 in /usr/local/lib/python3.10/dist-packages (from fastapi) (0.38.6)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,!=2.0.0,!=2.0.1,!=2.1.0,<3.0.0,>=1.7.4 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: click>=7.0 in /usr/local/lib/python3.10/dist-packages (from uvicorn) (8.1.7)
Requirement already satisfied: h11>=0.8 in /usr/local/lib/python3.10/dist-packages (from uvicorn) (0.14.0)
Requirement already satisfied: altair<6,>=4.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (4.2.2)
Requirement already satisfied: blinker<2,>=1.0.0 in /usr/lib/python3/dist-packages (from streamlit) (1.4)
Requirement already satisfied: cachetools<6,>=4.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (5.5.0)
Requirement already satisfied: pandas<3,>=1.3.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (2.1.4)
Requirement already satisfied: pillow<11,>=7.1.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (10.4.0)
Requirement already satisfied: pyarrow>=7.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (14.0.2)
Requirement already satisfied: rich<14,>=10.14.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (13.8.1)
Requirement already satisfied: tenacity<9,>=8.1.0 in /usr/local/lib/python3.10/dist-packages (from streamlit) (8.5.0)
Requirement already satisfied: toml<2,>=0.10.1 in /usr/local/lib/python3.10/dist-packages (from streamlit) (0.10.2)
Requirement already satisfied: gitpython!=3.1.19,<4,>=3.0.7 in /usr/local/lib/python3.10/dist-packages (from streamlit) (3.1.43)
Requirement already satisfied: pydeck<1,>=0.8.0b4 in /usr/local/lib/python3.10/dist-packages (from streamlit) (0.9.1)
Requirement already satisfied: tornado<7,>=6.0.3 in /usr/local/lib/python3.10/dist-packages (from streamlit) (6.3.3)
Requirement already satisfied: watchdog<5,>=2.1.5 in /usr/local/lib/python3.10/dist-packages (from streamlit) (4.0.2)
Requirement already satisfied: entrypoints in /usr/local/lib/python3.10/dist-packages (from altair<6,>=4.0->streamlit) (0.4)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from altair<6,>=4.0->streamlit) (3.1.4)
Requirement already satisfied: jsonschema>=3.0 in /usr/local/lib/python3.10/dist-packages (from altair<6,>=4.0->streamlit) (4.23.
Requirement already satisfied: toolz in /usr/local/lib/python3.10/dist-packages (from altair<6,>=4.0->streamlit) (0.12.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow)
Requirement already satisfied: gitdb<5,>=4.0.1 in /usr/local/lib/python3.10/dist-packages (from gitpython!=3.1.19,<4,>=3.0.7->str
Requirement already satisfied: namex in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->tensorflow) (0.12.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas<3,>=1.3.0->streamli
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas<3,>=1.3.0->streamlit) (2024.2
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas<3,>=1.3.0->streamlit) (2024
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,!=2.
Requirement already satisfied: pydantic-core==2.23.4 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,!=2.0
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->ten
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflow) (3.
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflo
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorflo
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from rich<14,>=10.14.0->streamli
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from rich<14,>=10.14.0->stream
Requirement already satisfied: anyio<5,>=3.4.0 in /usr/local/lib/python3.10/dist-packages (from starlette<0.39.0,>=0.37.2->fastap
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.18,>=2.17->tensorfl
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.18,>=2.17->tensorfl
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.4.0->starlette<0.39.0,>=
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.4.0->starlette<0.39.0,
Requirement already satisfied: smmap<6,>=3.0.1 in /usr/local/lib/python3.10/dist-packages (from gitdb<5,>=4.0.1->gitpython!=3.1.1
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->altair<6,>=4.0->streamlit
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0->altair<6,>=4.0->st
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0->altair<6,>=4
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=3.0->altair<6,>=4.0->s
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0->rich<14,>=10.14
Downloading python_multipart-0.0.12-py3-none-any.whl (23 kB)
Installing collected packages: python-multipart
Successfully installed python-multipart-0.0.12
```

Generate create a dataframe with 2 columns and 10 rows Close

```
# Download the dataset
!wget -O data.zip https://dicom5c.blob.core.windows.net/public/Data.zip

# Unzip the dataset
with zipfile.ZipFile('data.zip', 'r') as zip_ref:
    zip_ref.extractall('data')

# Check the contents
os.listdir('data')
```

```
--2024-10-01 05:49:03--  https://dicom5c.blob.core.windows.net/public/Data.zip
Resolving dicom5c.blob.core.windows.net (dicom5c.blob.core.windows.net)... 20.209.56.201
```

```
import os

# List the contents of the data directory
data_directory = "data"
print(os.listdir(data_directory))
```

```
['__MACOSX', 'Data']
```

```
def load_data(data_dir):
    images, masks = [], []

    for file in os.listdir(data_dir):
        if file.endswith('.tif'):
            file_path = os.path.join(data_dir, file)
            if '_mask' in file:
                mask = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
                masks.append((file.replace('_mask', ''), mask))
                print(f"Loaded mask: {file}")
            else:
                img = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
                images.append((file, img))
                print(f"Loaded image: {file}")

    valid_images, valid_masks = [], []
    for img_name, img in images:
        for mask_name, mask in masks:
            if img_name == mask_name:
                valid_images.append(img)
                valid_masks.append(mask)
                break

    print(f"Valid images: {len(valid_images)}, Valid masks: {len(valid_masks)}")
    return np.array(valid_images), np.array(valid_masks)



X, y = load_data(data_directory)
print(f"Images shape: {X.shape}, Masks shape: {y.shape}")
```

```
Valid images: 0, Valid masks: 0
Images shape: (0,), Masks shape: (0,)
```

```
if X.shape[0] > 0 and y.shape[0] > 0:
    X_train, X_test, y_train, y_test = preprocess_and_split_data(data_directory)
else:
    print("No valid images or masks found. Please check the dataset.")
```

```
No valid images or masks found. Please check the dataset.
```

```
def nested_unet(input_shape):
    inputs = layers.Input(shape=input_shape)

    conv1 = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(inputs)
    conv1 = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(conv1)
    pool1 = layers.MaxPooling2D(pool_size=(2, 2))(conv1)

    conv2 = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(pool1)
    conv2 = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(conv2)
    pool2 = layers.MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = layers.Conv2D(256, (3, 3), activation='relu', padding='same')(pool2)
    conv3 = layers.Conv2D(256, (3, 3), activation='relu', padding='same')(conv3)
    pool3 = layers.MaxPooling2D(pool_size=(2, 2))(conv3)

    conv4 = layers.Conv2D(512, (3, 3), activation='relu', padding='same')(pool3)
```

```
    conv4 = layers.Conv2D(512, (3, 3), activation='relu', padding='same')(conv4)

    conv5 = layers.Conv2D(512, (3, 3), activation='relu', padding='same')(conv4)
    conv5 = layers.Conv2D(512, (3, 3), activation='relu', padding='same')(conv5)

    outputs = layers.Conv2D(1, (1, 1), activation='sigmoid')(conv5)

    model = models.Model(inputs, outputs)
    return model



def attention_block(x, g, inter_channel):
    theta_x = layers.Conv2D(inter_channel, (2, 2), strides=(2, 2))(x)
    phi_g = layers.Conv2D(inter_channel, (1, 1))(g)
    add = layers.add([theta_x, phi_g])
    act = layers.Activation('relu')(add)
    psi = layers.Conv2D(1, (1, 1), activation='sigmoid')(act)
    return layers.multiply([x, psi])

def attention_unet(input_shape):
    inputs = layers.Input(shape=input_shape)

    conv1 = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(inputs)
    conv1 = layers.Conv2D(64, (3, 3), activation='relu', padding='same')(conv1)
    pool1 = layers.MaxPooling2D(pool_size=(2, 2))(conv1)

    conv2 = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(pool1)
    conv2 = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(conv2)
    pool2 = layers.MaxPooling2D(pool_size=(2, 2))(conv2)

    conv3 = layers.Conv2D(256, (3, 3), activation='relu', padding='same')(pool2)
    conv3 = layers.Conv2D(256, (3, 3), activation='relu', padding='same')(conv3)

    # Attention mechanism
    att3 = attention_block(conv3, conv2, 128)

    up2 = layers.UpSampling2D(size=(2, 2))(att3)
    concat2 = layers.concatenate([up2, conv2])
    conv2_out = layers.Conv2D(128, (3, 3), activation='relu', padding='same')(concat2)

    outputs = layers.Conv2D(1, (1, 1), activation='sigmoid')(conv2_out)

    model = models.Model(inputs, outputs)
    return model



input_shape = X_train.shape[1:]  # Assuming shape is (height, width, channels)

# Instantiate models
model_nest = nested_unet(input_shape)
model_attention = attention_unet(input_shape)

# Compile models
model_nest.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])
model_attention.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accuracy'])

# Define Model Checkpoint
checkpoint_nest = ModelCheckpoint('nested_unet_best.h5', save_best_only=True, monitor='val_loss', mode='min')
checkpoint_attention = ModelCheckpoint('attention_unet_best.h5', save_best_only=True, monitor='val_loss', mode='min')

# Train models
model_nest.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, callbacks=[checkpoint_nest])
model_attention.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, callbacks=[checkpoint_attention])


# FastAPI app
from fastapi import FastAPI, UploadFile, File
import numpy as np
from tensorflow.keras.models import load_model

app = FastAPI()

# Load your best model
model = load_model('nested_unet_best.h5', custom_objects={'dice_score': dice_score})

@app.post("/predict/")
async def predict(file: UploadFile = File(...)):
    img = await file.read()
    img = preprocess_image(img)  # Implement this function to preprocess input image
    prediction = model.predict(np.expand_dims(img, axis=0))
    return {"prediction": prediction.tolist()}
```

```
# Save this script as app.py and run with: !uvicorn app:app --reload


import streamlit as st
import requests
import numpy as np
import cv2

st.title("Brain MRI Metastasis Segmentation")

uploaded_file = st.file_uploader("Upload MRI Image", type=["tif", "png", "jpg"])

if uploaded_file is not None:
    img = cv2.imdecode(np.frombuffer(uploaded_file.read(), np.uint8), cv2.IMREAD_GRAYSCALE)
    st.image(img, caption="Uploaded MRI Image", use_column_width=True)

    if st.button("Segment"):
        response = requests.post("http://127.0.0.1:8000/predict/", files={"file": uploaded_file})
        st.image(response.json()["prediction"], caption="Segmentation Result")
```

```
2024-10-01 05:56:03.878 WARNING streamlit.runtime.scriptrunner_utils.script_run_context: Thread 'MainThread': missing ScriptRunConte
2024-10-01 05:56:04.045
  Warning: to view this Streamlit app on a browser, run it with the following
  command:

    streamlit run /usr/local/lib/python3.10/dist-packages/colab_kernel_launcher.py [ARGUMENTS]
2024-10-01 05:56:04.050 Thread 'MainThread': missing ScriptRunContext! This warning can be ignored when running in bare mode.
2024-10-01 05:56:04.053 Thread 'MainThread': missing ScriptRunContext! This warning can be ignored when running in bare mode.
2024-10-01 05:56:04.058 Thread 'MainThread': missing ScriptRunContext! This warning can be ignored when running in bare mode.
2024-10-01 05:56:04.060 Thread 'MainThread': missing ScriptRunContext! This warning can be ignored when running in bare mode.
2024-10-01 05:56:04.061 Thread 'MainThread': missing ScriptRunContext! This warning can be ignored when running in bare mode.
```