**Machine Learning Project Documentation**

**Deployment Report: YOLO-Based Brain Tumor Detection Model**

**1. Overview**

The deployment phase focused on making the **brain tumor detection model** accessible through a web-based interface using **Gradio** and **Hugging Face**. The key steps included:

* **Model Training**: Training a **YOLO-based deep learning model** on labeled brain tumor images.
* **Model Serialization**: Saving the trained model in a compatible format.
* **Deployment Using Gradio**: Creating an interactive UI for image uploads and predictions.
* **Hosting on Hugging Face**: Deploying the model on Hugging Face Spaces for public access.

**2. Model Serialization**

Before deployment, the trained model was serialized for efficient storage and loading. The serialization process involved:

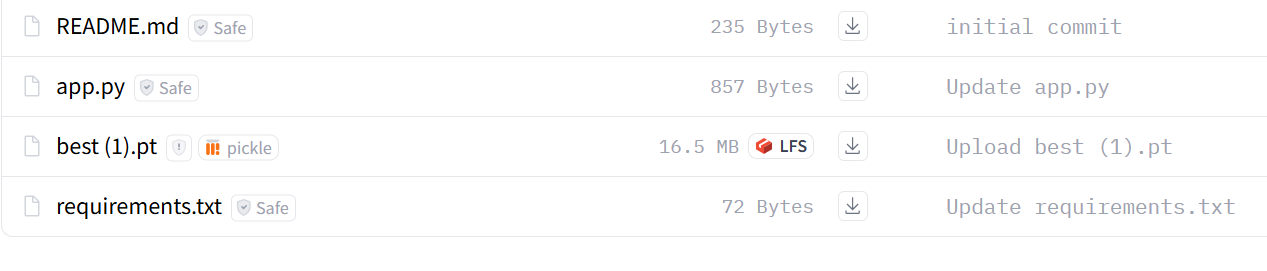
* **Saving the Model**: Using **TorchScript (.pt) or ONNX (.onnx)** format for efficient storage and inference.
* **Storage Considerations**: Optimizing file size and ensuring compatibility with deployment frameworks.
* **Preprocessing Pipelines**: Saving any preprocessing functions alongside the model.
* **Model Serialization Code**

from ultralytics import YOLO

# Load and save the trained model

model = YOLO("best (1).pt")

model.export(format="onnx")

****

**3. Model Serving**

The serialized model is **served using Gradio** to provide a user-friendly interface. The deployment options considered were:

* **On-Premises Deployment**: Using a local Gradio interface for testing.
* **Cloud Deployment**: Deploying on **Hugging Face Spaces**, which allows free hosting of AI models with GPU support.
* **Containerization (Optional)**: Using Docker for scalable deployment.
* **Model Serving Code**

import gradio as gr

from ultralytics import YOLO

from PIL import Image

# Load the YOLO model

model = YOLO("best (1).pt")

def predict(image):

results = model(image)

annotated\_image = results[0].plot()

return Image.fromarray(annotated\_image)

interface = gr.Interface(

fn=predict,

inputs=gr.Image(type="pil"),

outputs=gr.Image(type="pil"),

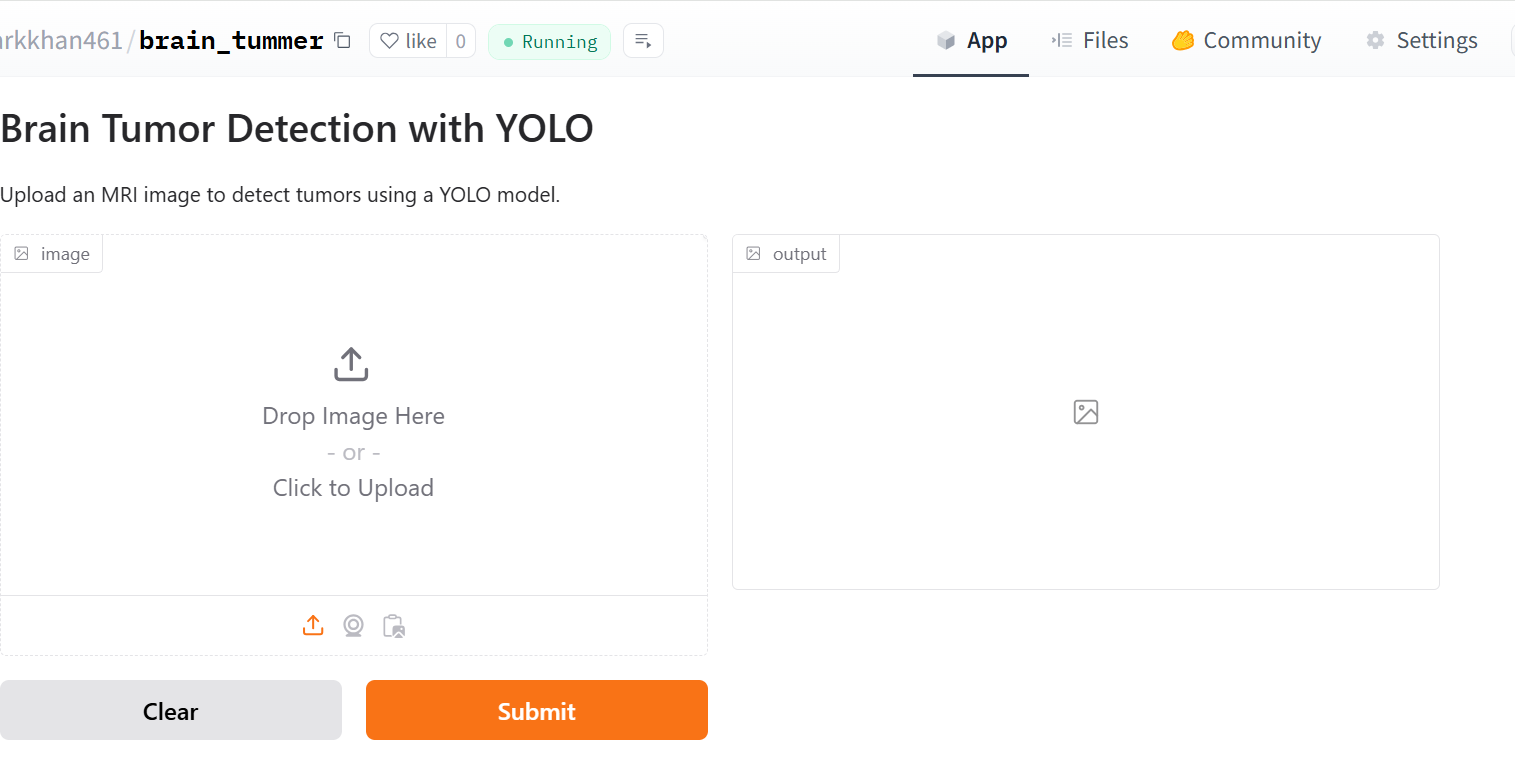
title="Brain Tumor Detection with YOLO",

description="Upload an MRI image to detect tumors using a YOLO model.",

)

interface.launch()

**Image of the Gradio UI running**

****

**4. API Integration**

To make the model easily accessible, it was wrapped in a Gradio interface and deployed as an API:

* **API Endpoints**: Hugging Face Spaces automatically generates an API endpoint when deploying a Gradio app.
* **Input Format**: Accepts image files (.jpg, .png, .jpeg) as input.
* **Response Format**: Returns predictions with bounding boxes (if applicable) and confidence scores.

****

**5. Security Considerations**

Security measures were implemented to ensure safe deployment:

* **Access Control**: Limiting API access to authorized users if needed.
* **Data Encryption**: Ensuring secure transmission of uploaded medical images.
* **Model Protection**: Avoiding exposure of model weights and internal architecture.

**6. Monitoring and Logging**

To ensure reliability, monitoring and logging mechanisms were set up:

* **Performance Metrics**: Tracking inference time, accuracy, and resource usage.
* **Logging**: Storing logs of user requests and model predictions.
* **Alerts**: Setting up notifications for model failures or unusual activity.
* **Requirements File**

Below are the required dependencies for running the application:

gradio

numpy

pillow

requests

ultralytics>=8.0.0

torch>=1.10.0