الأكواد الأساسية لمشروع مسار – اللون الأحمر اسم الملف

gitattributes

\*.7z filter=lfs diff=lfs merge=lfs -text

\*.arrow filter=lfs diff=lfs merge=lfs -text

\*.bin filter=lfs diff=lfs merge=lfs -text

\*.bz2 filter=lfs diff=lfs merge=lfs -text

\*.ckpt filter=lfs diff=lfs merge=lfs -text

\*.ftz filter=lfs diff=lfs merge=lfs -text

\*.gz filter=lfs diff=lfs merge=lfs -text

\*.h5 filter=lfs diff=lfs merge=lfs -text

\*.joblib filter=lfs diff=lfs merge=lfs -text

.lfs. filter=lfs diff=lfs merge=lfs -text

\*.mlmodel filter=lfs diff=lfs merge=lfs -text

\*.model filter=lfs diff=lfs merge=lfs -text

\*.msgpack filter=lfs diff=lfs merge=lfs -text

\*.npy filter=lfs diff=lfs merge=lfs -text

\*.npz filter=lfs diff=lfs merge=lfs -text

\*.onnx filter=lfs diff=lfs merge=lfs -text

\*.ot filter=lfs diff=lfs merge=lfs -text

\*.parquet filter=lfs diff=lfs merge=lfs -text

\*.pb filter=lfs diff=lfs merge=lfs -text

\*.pickle filter=lfs diff=lfs merge=lfs -text

\*.pkl filter=lfs diff=lfs merge=lfs -text

\*.pt filter=lfs diff=lfs merge=lfs -text

\*.pth filter=lfs diff=lfs merge=lfs -text

\*.rar filter=lfs diff=lfs merge=lfs -text

\*.safetensors filter=lfs diff=lfs merge=lfs -text

saved\_model//\* filter=lfs diff=lfs merge=lfs -text

.tar. filter=lfs diff=lfs merge=lfs -text

\*.tar filter=lfs diff=lfs merge=lfs -text

\*.tflite filter=lfs diff=lfs merge=lfs -text

\*.tgz filter=lfs diff=lfs merge=lfs -text

\*.wasm filter=lfs diff=lfs merge=lfs -text

\*.xz filter=lfs diff=lfs merge=lfs -text

\*.zip filter=lfs diff=lfs merge=lfs -text

\*.zst filter=lfs diff=lfs merge=lfs -text

tfevents filter=lfs diff=lfs merge=lfs -text

saving\_info

import csv

import os

DB\_FILE = "fan\_opinions\_log.csv"

def initialize\_db():

if not os.path.exists(DB\_FILE):

with open(DB\_FILE, "w", newline='', encoding='utf-8') as f:

writer = csv.writer(f)

writer.writerow(["input\_text", "output\_analysis", "is\_correct", "correction"])

def save\_opinion(input\_text, output\_analysis, is\_correct, correction):

with open(DB\_FILE, "a", newline='', encoding='utf-8') as f:

writer = csv.writer(f)

writer.writerow([input\_text, output\_analysis, is\_correct, correction])

from db import create\_opinions\_table, insert\_opinion

def initialize\_db():

create\_opinions\_table()

def save\_opinion(input\_text, output\_analysis, is\_correct, correction):

insert\_opinion(input\_text, output\_analysis, is\_correct, correction)

requirements.txt

transformers

torch

app.py

import gradio as gr

from transformers import pipeline

import re

import csv

import os

sentiment\_analyzer = pipeline("sentiment-analysis", model="distilbert-base-uncased-finetuned-sst-2-english")

LOG\_FILE = "fan\_opinions\_log.csv"

if not os.path.exists(LOG\_FILE):

with open(LOG\_FILE, "w", newline='', encoding='utf-8') as f:

writer = csv.writer(f)

writer.writerow(["input\_text", "output\_analysis", "is\_correct", "correction"])

def extract\_info(text, pattern):

match = re.search(pattern, text, re.IGNORECASE)

return match.group(2).strip() if match else "-"

def analyze\_opinion(opinion\_text):

result = sentiment\_analyzer(opinion\_text)[0]

label = result["label"]

score = round(result["score"] \* 100, 2)

sentiment = "🟣 رأي محايد"

if label == "POSITIVE":

sentiment = f" رأي إيجابي ({score}%)"

elif label == "NEGATIVE":

sentiment = f" رأي سلبي ({score}%)"

team = extract\_info(opinion\_text, r"(support(?:ing)?|cheering for|I cheer for)\s+([A-Za-z\s]+)")

name = extract\_info(opinion\_text, r"(My name is|اسمي)\s+([A-Za-z\s]+)")

age = extract\_info(opinion\_text, r"(?:I am|عمري)\s+(\d{1,2})\s+years? old|عمري\s+(\d{1,2})")

nationality = extract\_info(opinion\_text, r"(?:I am|أنا)\s+(Saudi|French|Egyptian|Indian|American|British|[A-Za-z]+)")

stadium = extract\_info(opinion\_text, r"(stadium|ملعب)\s+([A-Za-z\s]+)")

day = extract\_info(opinion\_text, r"(on|في)\s+([A-Za-z]+|Monday|Tuesday|Wednesday|Thursday|Friday|Saturday|Sunday)")

result\_text = f""" التصنيف: {sentiment}

🟣 الفريق المشجَّع: {team}

🟣 الاسم: {name}

🟣 العمر: {age}

🟣 الجنسية: {nationality}

🟣 الملعب: {stadium}

🟣 اليوم: {day}"""

return result\_text

def full\_pipeline(opinion\_text, is\_correct, correction):

output = analyze\_opinion(opinion\_text)

with open(LOG\_FILE, "a", newline='', encoding='utf-8') as f:

writer = csv.writer(f)

writer.writerow([opinion\_text, output, is\_correct, correction])

return output

custom\_css = """

body {

background-color: #5e17eb;

color: white;

font-weight: bold;

font-size: 18px;

}

textarea, input, select {

background-color: #4b0dc7 !important;

color: white !important;

border: 2px solid white !important;

font-weight: bold !important;

font-size: 18px !important;

}

textarea::placeholder {

color: #ddd !important;

}

.gradio-container {

font-family: 'Segoe UI', sans-serif;

}

"""

demo = gr.Interface(

fn=full\_pipeline,

inputs=[

gr.Textbox(lines=6, label="رأي المشجع"),

gr.Radio(choices=["نعم", "لا"], label="هل التحليل صحيح؟"),

gr.Textbox(lines=2, label="إذا لا، ما هو التصحيح؟ (اختياري)")

],

outputs="text",

title="تحليل آراء المشجعين",

description="أدخل رأي المشجع، وسنقوم بتحليله. إذا كان التحليل غير دقيق، صححه لنا!",

theme="default",

css=custom\_css

)

demo.launch()

README.md

metadata

title: Op Masar Ai

emoji: 🚀

colorFrom: gray

colorTo: green

sdk: gradio

sdk\_version: 5.23.3

app\_file: app.py

pinned: false

license: apache-2.0

DB!

import sqlite3

DB\_NAME = "fan\_opinions.db"

def connect\_db():

return sqlite3.connect(DB\_NAME)

def create\_opinions\_table():

conn = connect\_db()

cursor = conn.cursor()

cursor.execute("""

CREATE TABLE IF NOT EXISTS opinions (

id INTEGER PRIMARY KEY AUTOINCREMENT,

input\_text TEXT NOT NULL,

output\_analysis TEXT NOT NULL,

is\_correct TEXT,

correction TEXT

)

""")

conn.commit()

conn.close()

def insert\_opinion(input\_text, output\_analysis, is\_correct, correction):

conn = connect\_db()

cursor = conn.cursor()

cursor.execute("""

INSERT INTO opinions (input\_text, output\_analysis, is\_correct, correction)

VALUES (?, ?, ?, ?)

""", (input\_text, output\_analysis, is\_correct, correction))

conn.commit()

conn.close()