import re

import torch

from transformers import AutoTokenizer, AutoModelForSequenceClassification

from torch.nn.functional import sigmoid

MODEL\_NAME = "unitary/toxic-bert"

DEVICE = "cuda" if torch.cuda.is\_available() else "cpu"

tokenizer = AutoTokenizer.from\_pretrained(MODEL\_NAME)

model = AutoModelForSequenceClassification.from\_pretrained(MODEL\_NAME)

model.to(DEVICE)

model.eval()

LABELS = [

    "toxic",

    "severe\_toxic",

    "obscene",

    "threat",

    "insult",

    "identity\_hate"

]

def preprocess(text):

    return re.sub(r"http\S+", "", text).strip()

def predict(text, threshold=0.7):

    text = preprocess(text)

    inputs = tokenizer(

        text,

        return\_tensors="pt",

        truncation=True,

        padding=True,

        max\_length=128

    ).to(DEVICE)

    with torch.no\_grad():

        outputs = model(\*\*inputs)

        probs = sigmoid(outputs.logits)[0]

    toxic\_score = probs[LABELS.index("toxic")].item()

    insult\_score = probs[LABELS.index("insult")].item()

    if toxic\_score > threshold or insult\_score > threshold:

        return "Toxic"

    return "Not Toxic"

if \_\_name\_\_ == "\_\_main\_\_":

    print("Toxicity Checker Ready! Type 'exit' to stop.\n")

    while True:

        user\_input = input("Enter comment: ")

        if user\_input.lower() == "exit":

            break

        result = predict(user\_input)

        print(f"Prediction: {result}\n")