AI VIET NAM – COURSE 2024

Streamlit - Project

Ngày 1 tháng 7 năm 2024

Ngày thực hiện:	01/07/2024
Người thực hiện:	Đinh Thị Tâm
Nguồn:	AIO2024 - Week4
Nguồn dữ liệu (nếu có):	Link of Data Sources
Từ khóa:	Streamlit project
Người tóm tắt:	Đinh Thị Tâm

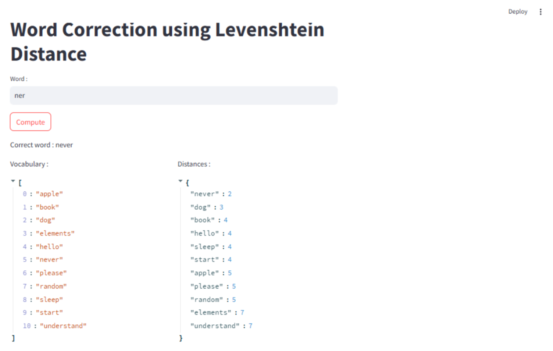
I. Câu hỏi tự luận

1. Câu 1:

   (a) Code

```python
import streamlit as st
import os


def load_vocab(file_path):
    with open(file_path, 'r') as f:
        lines = f.readlines()
    words = sorted(set([line.strip().lower() for line in lines]))
    return words


def levenshtein_distance(token1, token2):
    distances = [[0]*(len(token2)+1) for i in range(len(token1)+1)]

    for t1 in range(len(token1) + 1):
        distances[t1][0] = t1

    for t2 in range(len(token2) + 1):
        distances[0][t2] = t2

    a = 0
    b = 0
    c = 0

    for t1 in range(1, len(token1) + 1):
        for t2 in range(1, len(token2) + 1):
            if (token1[t1-1] == token2[t2-1]):
                distances[t1][t2] = distances[t1 - 1][t2 - 1]
            else:
                a = distances[t1][t2 - 1]
                b = distances[t1 - 1][t2]
                c = distances[t1 - 1][t2 - 1]

                if (a <= b and a <= c):
                    distances[t1][t2] = a + 1
                elif (b <= a and b <= c):
                    distances[t1][t2] = b + 1
                else:
                    distances[t1][t2] = c + 1

    return distances[len(token1)][len(token2)]


def main(vocabs):
    st.title("Word Correction using Levenshtein Distance")
    word = st.text_input('Word :')
    if st.button("Compute"):
        # compute levenshtein distance
        leven_distances = dict()
        for vocab in vocabs:
            leven_distances[vocab] = levenshtein_distance(word, vocab)
            # sorted by distance
        sorted_distences = dict(sorted(leven_distances.items(), key=lambda
    item:
                                       item[1]))
        correct_word = list(sorted_distences.keys())[0]
        st.write('Correct word : ', correct_word)
        col1, col2 = st.columns(2)
        col1.write('Vocabulary :')
        col1.write(vocabs)
        col2.write('Distances :')
        col2.write(sorted_distences)


if __name__ == '__main__':
    current_file_path = os.path.abspath(__file__)
    current_directory = os.path.dirname(current_file_path)
    path_file = current_directory+'\\data\\vocab.txt'
    vocabs = load_vocab(path_file)
    # print(vocabs)
    main(vocabs)
```

(b) Kết quả thực thi



Word Correction using Levenshtein Distance

Word :

ner

Correct word : never

Vocabulary :

```
[
  0 : "apple"
  1 : "book"
  2 : "dog"
  3 : "elements"
  4 : "hello"
  5 : "never"
  6 : "please"
  7 : "random"
  8 : "sleep"
  9 : "start"
  10 : "understand"
]
```

Distances :

```
{
  "never" : 2
  "dog" : 3
  "book" : 4
  "hello" : 4
  "sleep" : 4
  "start" : 4
  "apple" : 5
  "please" : 5
  "random" : 5
  "elements" : 7
  "understand" : 7
}
```

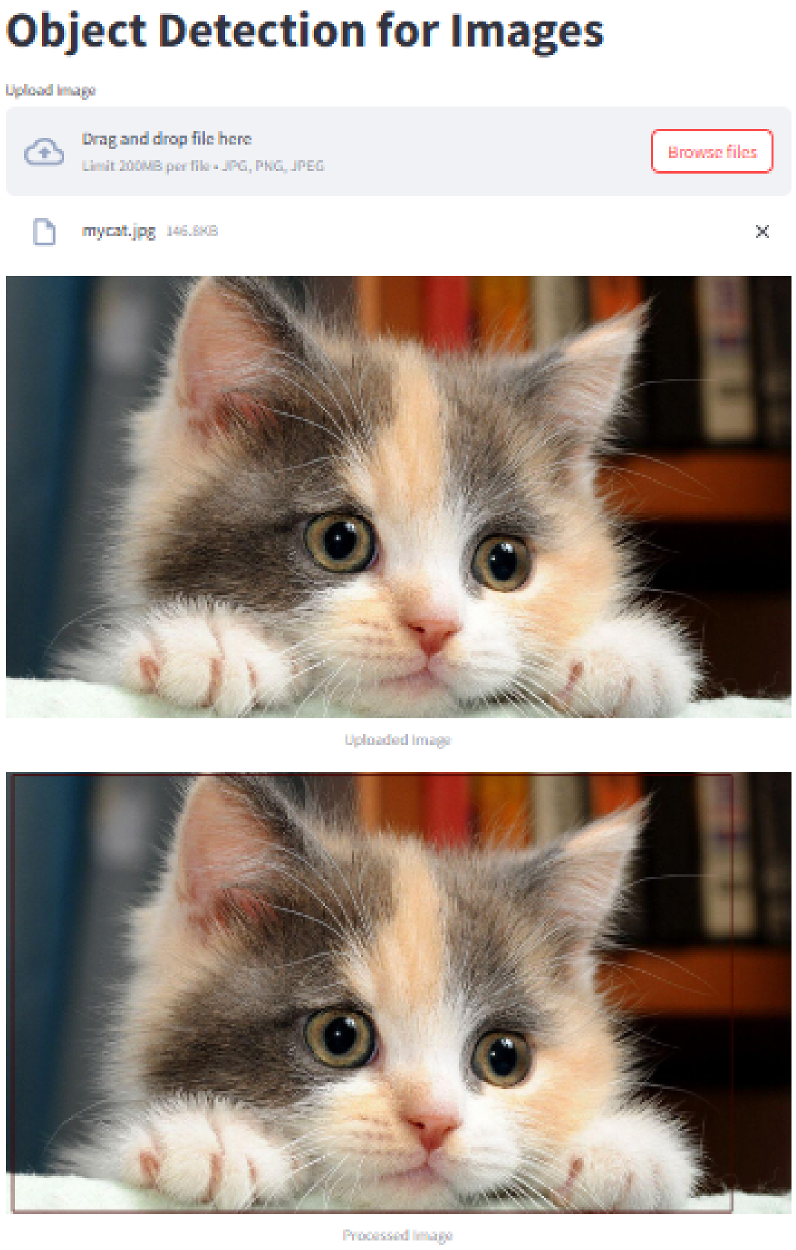
Hình 1: Levenshtein

2. Câu 2

   (a) Code

```python
import cv2
import numpy as np
from PIL import Image
import streamlit as st
import os
MODEL = "/model/MobileNetSSD_deploy.caffemodel"
CONFIG = "/model/MobileNetSSD_deploy.config.txt"


def get_location(Model=MODEL, prototxt=CONFIG):
    current_file_path = os.path.abspath(__file__)
    current_directory = os.path.dirname(current_file_path)
    model = current_directory+MODEL
    config = current_directory+CONFIG
    return model, config


def process_image(image):
    blob = cv2.dnn.blobFromImage(
        cv2.resize(image, (300, 300)), 0.007843, (300, 300), 127.5
    )
    model, config = get_location()
    # net = cv2.dnn.readNetFromCaffe(PROTOTXT, MODEL)
    net = cv2.dnn.readNetFromCaffe(config, model)
    net.setInput(blob)
    detections = net.forward()
    return detections


def annotate_image(
    image, detections, confidence_threshold=0.5
):
    # loop over the detections
    (h, w) = image.shape[:2]
    for i in np.arange(0, detections.shape[2]):
        confidence = detections[0, 0, i, 2]

        if confidence > confidence_threshold:
            # extract the index of the class label from the `detections`,
            # then compute the (x, y)-coordinates of the bounding box for
            # the object
            idx = int(detections[0, 0, i, 1])
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")
            cv2.rectangle(image, (startX, startY), (endX, endY), 70, 2)
    return image


def main():

    st.title('Object Detection for Images')
    file = st.file_uploader('Upload Image', type=['jpg', 'png', 'jpeg'])
    if file is not None:
        st.image(file, caption="Uploaded Image")

        image = Image.open(file)
        image = np.array(image)
        try:
            detections = process_image(image)
            processed_image = annotate_image(image, detections)
            st.image(processed_image, caption="Processed Image")
        except:
            st.write('Not detected object')


if __name__ == "__main__":
    main()
```

(b) Kết quả thực thi



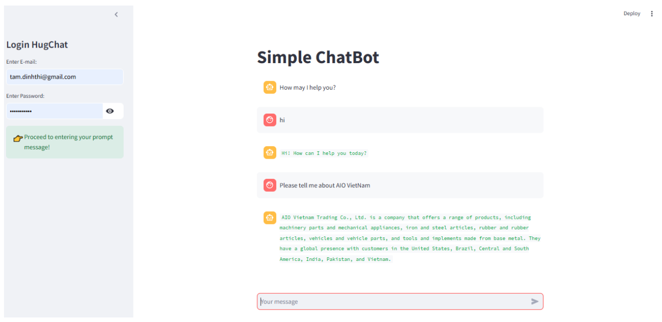
Hình 2: Detected object

3. Câu 3

   (a) Code

```python
import streamlit as st
from hugchat import hugchat
from hugchat.login import Login

# App title
st.title('Simple ChatBot')

# Hugging Face Credentials
with st.sidebar:
    st.title('Login HugChat')
    hf_email = st.text_input('Enter E-mail:')
    hf_pass = st.text_input('Enter Password:', type='password')
    if not (hf_email and hf_pass):
        st.warning('Please enter your account!', icon=\'      ')
    else:
        st.success('Proceed to entering your prompt message!', icon=\\'   í
    ')


# Store LLM generated responses
if "messages" not in st.session_state.keys():
    st.session_state.messages = [
        {"role": "assistant", "content": "How may I help you?"}]

# Display chat messages
for message in st.session_state.messages:
    with st.chat_message(message["role"]):
        st.write(message["content"])

# Function for generating LLM response


def generate_response(prompt_input, email, passwd):
    # Hugging Face Login
    sign = Login(email, passwd)
    cookies = sign.login()
    # Create ChatBot
    chatbot = hugchat.ChatBot(cookies=cookies.get_dict())
    return chatbot.chat(prompt_input)


# User-provided prompt
if prompt := st.chat_input(disabled=not (hf_email and hf_pass)):
    st.session_state.messages.append({"role": "user", "content": prompt})
    with st.chat_message("user"):
        st.write(prompt)

# Generate a new response if last message is not from assistant
if st.session_state.messages[-1]["role"] != "assistant":
    with st.chat_message("assistant"):
        with st.spinner("Thinking..."):
            response = generate_response(prompt, hf_email, hf_pass)
            st.write(response)
    message = {"role": "assistant", "content": response}
    st.session_state.messages.append(message)

```

(b) Kết quả thực thi



Login HugChat

Enter E-mail:

tam.dinhthi@gmail.com

Enter Password:

•••••••••••

Proceed to entering your prompt message!

Simple ChatBot

How may I help you?

hi

Hi! How can I help you today?

Please tell me about AIO VietNam

AIO Vietnam Trading Co., Ltd. is a company that offers a range of products, including machinery parts and mechanical appliances, iron and steel articles, rubber and rubber articles, vehicles and vehicle parts, and tools and implements made from base metal. They have a global presence with customers in the United States, Brazil, Central and South America, India, Pakistan, and Vietnam.

Your message

Hình 3: Chatbot

II. Câu hỏi trắc nghiệm

Điền trực tiếp trên google form