ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

PHẠM THỊ HOÀ
NGÀNH LÝ THUYẾT XÁC SUẤT VÀ THỐNG KÊ TOÁN HỌC

ĐỒ ÁN CUỐI KÌ

XỬ LÝ NGÔN NGỮ TỰ NHIÊN

GIẢNG VIÊN HƯỚNG DẪN
PGS.TS. Đinh Điền
TS. Nguyễn Hồng Bửu Long
TS. Lương An Vinh

Tp. Hồ Chí Minh - 2024

Mục lục

1 Yêu cầu đồ án

Với đề tài này, các bạn hãy tìm hiểu và cài đặt các API xử lý văn bản tiếng Việt (tiền xử lý, chuẩn hóa encoding, dấu câu, dấu thanh, loại bỏ các ký tự đặc biệt/dư thừa/mã HTML..., tách câu). API có thể thực hiện riêng rẽ hoặc đồng thời nhiều thao tác cùng lúc.

2 Nội dung thực hiện

Loại bỏ nhiễu và dữ liệu không cần thiết: Văn bản thô thường chứa nhiều ký tự không cần thiết như dấu câu, ký tự đặc biệt, số, hoặc khoảng trắng thừa, mã html. Loại bỏ các thành phần này giúp giảm nhiễu trong dữ liệu và tập trung vào các thông tin có ý nghĩa.

Chuẩn hóa văn bản:

- Giảm thiểu sự phân biệt chữ hoa và chữ thường: chuyển hết về chữ thường
- Chuẩn hóa encoding: Thay thế cách gõ Unicode tổ hợp bằng cách gõ của Unicode dựng sẵn.
- Chuẩn hoá dấu câu, dấu thanh

Tách từ: Tiếng Việt là ngôn ngữ đa âm tiết, việc tách từ giúp nhận diện rõ ràng các từ trong câu. Điều này rất quan trọng vì nhiều mô hình NLP yêu cầu dữ liệu đầu vào là các từ hoặc cụm từ.

Tách câu: Tách câu là một bước tiền xử lý quan trọng trong việc chuẩn bị dữ liệu cho nhiều tác vụ NLP khác nhau. Dữ liệu được tách câu sẽ dễ dàng hơn để gán nhãn, huấn luyện mô hình, và đánh giá kết quả

Loại bỏ từ dừng (StopWords): Loại bỏ các từ dừng không mang nhiều ý nghĩa như "là", "và", "nhưng", "với",...

2.1 Code các hàm Tiền xử lý dữ liệu

File name: **pre_processing.py** là file code bao gồm toàn bộ các hàm dùng để tiền xử lý dữ liệu ngôn ngữ tiếng Việt.

2.1.1 Loại bỏ các mã HTML ra khỏi văn bản.

```
from bs4 import BeautifulSoup


def remove_html(text):
    soup = BeautifulSoup(text, "html.parser")
    return soup.get_text()
```

2.1.2 Loại bỏ tất cả các ký tự không phải là chữ cái hoặc số

```
import re
def remove_special_characters(text):
    # Loại bỏ tất cả các ký tự không phải là chữ cái hoặc số
    return re.sub(r'[^a-zA-Z0-9\sàáạảãâầấậẩẫăằắặẳẵèéẹẻẽêềếệểễìíịỉĩòóọỏõôồốộổỗơờớợ
```

2.1.3 Hàm chuẩn hóa về kiểu chữ thường

```
def to_lowercase(text):
    return text.lower()
```

2.1.4 Hàm loại bỏ khoảng trắng thừa

```
import re


def remove_extra_whitespace(text):
    text = re.sub(r'\s+', ' ', text).strip()
    text = text.replace(" .", ".")
    text = text.replace(" !", "!")
    text = text.replace(" ?", "?")
```

```
    text = text.replace(" ,", ",")
    return text
```

2.1.5 Hàm chuẩn hoá dấu thanh

```
def text_normalize_f(text):
    text = text_normalize(text)
    text = remove_extra_whitespace(text)
    return text
```

2.1.6 Hàm dùng để tách từ

```
def pyvi_word_tokenize(text):
    return ViTokenizer.tokenize(text)
```

2.1.7 Hàm chuẩn hoá unicode

```
# Chuẩn hoá unicode
def chuan_hoa_unicode(text):
    text = unicodedata.normalize('NFC', text)
    return text
```

2.1.8 Hàm loại bỏ từ dừng

```
def remove_stop_words(text):
    words = text.split()
    filtered_words = [word for word in words if word not in stop_words]
    return ' '.join(filtered_words)
```

2.1.9 Hàm chuẩn hoá dấu câu

```
def normalize_punctuation(text):
    # Loại bỏ khoảng trắng thừa trước dấu câu
    text = re.sub(r'\s+([,.!?])', r'\1', text)
    # Thêm khoảng trắng sau dấu câu nếu chưa có
```

```
    text = re.sub(r'([,.!?])([^\s])', r'\1 \2', text)
    # Loại bỏ khoảng trắng thừa
    text = re.sub(r'\s+', ' ', text).strip()
    # Loại khoảng trắng thừa
    text = remove_extra_whitespace(text)
    return text
```

2.1.10 Hàm tách câu

```
def sentence_tokenize(text):
    text = sent_tokenize(text)
    text_new_sent = []
    for _text in text:
        text_new = re.sub(
            r'[^a-zA-Z0-9\sàáạảãâầấậẩẫăằắặẳẵèéẹẻẽêềếệểễìíịỉĩòóọỏõôồốộổỗơờớợởỡùúụủ
        text_new = re.sub(r'\s+', ' ', text_new).strip()

        if len(text_new) == 0:
            continue
        if text_new[-1] == " ":
            text_new = text_new[:-1]
        text_new_sent.append(text_new)

    return text_new_sent
```

2.1.11 Hàm tổng hợp để tiền xử lý văn bản

```
def preprocess_text(text,
                    clean_html=True,
                    clean_special_char=True,
                    clean_extra_whitespace=True,
                    chuan_hoa_unicode_action=True,
```

```
                chuan_hoa_chu_thuong= True,
                chuan_hoa_dau_thanh=True,
                chuan_hoa_dau_cau= True,
                split_word_pyvi=False,
                split_word_vietokenizer=True,
                split_sent=True,
                remove_sw=True
                ):
if clean_html:
    text = remove_html(text)
if clean_special_char:
    text = remove_special_characters(text)
if clean_extra_whitespace:
    # Loại khoảng trắng thừa
    text = remove_extra_whitespace(text)
# Chuẩn hoá dữ liệu
if chuan_hoa_unicode_action:
    text = chuan_hoa_unicode(text)
if chuan_hoa_chu_thuong:
    text = to_lowercase(text)
if chuan_hoa_dau_thanh:
    text = text_normalize_f(text)
if chuan_hoa_dau_cau:
    text = normalize_punctuation(text)
if split_word_pyvi:
    text = pyvi_word_tokenize(text)
    # Chuẩn hoá dấu câu
    text = normalize_punctuation(text)
    # Loại khoảng trắng thừa
    text = remove_extra_whitespace(text)
```

```
elif split_word_vietokenizer:
    text = word_tokenize(text)
if remove_sw:
    text = remove_stop_words(text=text)
# Tách câu
if split_sent:
    text = sentence_tokenize(text)
    return text
elif clean_special_char:
    # Loại bỏ kí tự đặc biệt
    text = re.sub(
        r'[^a-zA-Z0-9\sàáạảãâầấậẩẫăằắặẳẵèéẹẻẽêềếệểễìíịỉĩòóọỏõôồốộổỗơờớợởỡùúụủ
    # Loại khoảng trắng thừa
    text = remove_extra_whitespace(text)

return [text]
```

2.2 API

Đoạn mã dưới đây định nghĩa một API sử dụng FastAPI để tiền xử lý văn bản với nhiều lựa chọn khác nhau. API này có một endpoint `/api/v1/pre-processing` dùng để tiền xử lý văn bản với nhiều tùy chọn như làm sạch HTML, chuẩn hóa ký tự đặc biệt, loại bỏ khoảng trắng thừa, chuẩn hóa Unicode, chữ thường, dấu thanh, dấu câu, tách từ, tách câu, và loại bỏ từ dừng.

2.2.1 Các tham số API:

- `text`: Chuỗi văn bản đầu vào.
- Các tham số boolean cho phép người dùng lựa chọn có thực hiện các bước tiền xử lý như:
  - Loại bỏ HTML (`clean_html`)
  - Loại bỏ ký tự đặc biệt (`clean_special_char`)
  - Loại bỏ khoảng trắng thừa (`clean_extra_whitespace`)
  - Chuẩn hóa Unicode (`chuan_hoa_unicode_action`)
  - Chuyển chữ thường (`chuan_hoa_chu_thuong`)
  - Chuẩn hóa dấu thanh (`chuan_hoa_dau_thanh`)
  - Chuẩn hóa dấu câu (`chuan_hoa_dau_cau`)
  - Tách từ (`split_word`)
  - Tách câu (`split_sent`)
  - Loại bỏ từ dừng (`remove_sw`)

2.2.2 Phần code cho hàm tạo API

Hàm `preprocess_text` được gọi để thực hiện các tác vụ này và trả về kết quả là văn bản đã được tiền xử lý.

```
from fastapi import FastAPI
import uvicorn
```

```
import asyncio
import sys
import os
sys.path.append(os.path.join(os.path.dirname(__file__), "../.."))
from src.api.pre_processing import *

app = FastAPI()

@app.get("/api/v1/pre-processing")
async def api_preprocess_text(text: str,
                              clean_html: bool = True,
                              clean_special_char: bool = True,
                              clean_extra_whitespace: bool = True,
                              chuan_hoa_unicode_action: bool = True,
                              chuan_hoa_chu_thuong: bool = True,
                              chuan_hoa_dau_thanh: bool = True,
                              chuan_hoa_dau_cau:bool = True,
                              split_word_pyvi: bool = True,
                                split_sent: bool = True,
                                remove_sw: bool=True
                                ):
    text = preprocess_text(text=text,
                              clean_html=clean_html,
                              clean_special_char=clean_special_char,
                              clean_extra_whitespace=clean_extra_whitespace,
                              chuan_hoa_unicode_action=chuan_hoa_unicode_action,
                              chuan_hoa_chu_thuong=chuan_hoa_chu_thuong,
                              chuan_hoa_dau_thanh=chuan_hoa_dau_thanh,
                              chuan_hoa_dau_cau=chuan_hoa_dau_cau,
                              split_word_pyvi=split_word_pyvi,
```

```
                            split_sent=split_sent,
                            remove_sw =remove_sw
                            )
    return {"data": text}


def API():
    uvicorn.run(app, host="0.0.0.0", port=8000)
```

2.3 GUI

2.3.1 Giải thích về giao diện xử lý văn bản bằng Streamlit

Đoạn mã này tạo ra một ứng dụng giao diện người dùng sử dụng **Streamlit** để thực hiện các tác vụ tiền xử lý văn bản tiếng Việt. Dưới đây là giải thích chi tiết cho từng phần của mã:

- **Thư viện sử dụng**:
  - `requests`: Được sử dụng để gửi các yêu cầu HTTP tới API.
  - `json`: Được sử dụng để xử lý và phân tích dữ liệu JSON từ API.
  - `streamlit`: Một thư viện phổ biến để xây dựng giao diện người dùng dễ dàng và nhanh chóng.
- **API Call**: Hàm `call_api` được sử dụng để gọi API từ backend. Hàm này gửi yêu cầu GET tới endpoint của API với các tham số đầu vào như văn bản cần xử lý và các lựa chọn cho các tác vụ tiền xử lý (như loại bỏ HTML, ký tự đặc biệt, khoảng trắng thừa, chuẩn hóa Unicode, dấu câu, v.v.). Kết quả của việc tiền xử lý được nhận lại dưới dạng JSON và trả về dưới dạng dữ liệu đã xử lý.
- **Giao diện người dùng**: Hàm `gui` xây dựng giao diện người dùng cho việc xử lý văn bản. Giao diện bao gồm:
  - Một ô nhập liệu để người dùng nhập đoạn văn bản cần xử lý.
  - Các tùy chọn checkbox cho phép người dùng lựa chọn các tác vụ tiền xử lý, được chia thành ba cột:
    - **Các tác vụ làm sạch dữ liệu**: Xóa HTML, ký tự đặc biệt, khoảng trắng thừa, và từ dừng (stopwords).
    - **Các tác vụ chuẩn hóa dữ liệu**: Chuẩn hóa Unicode, chuyển chữ thường, chuẩn hóa dấu thanh và dấu câu.
    - **Các tác vụ tách dữ liệu**: Tách từ và tách câu.
  - Nút `Xử lý`: Khi nhấn nút này, các tác vụ được thực hiện dựa trên các lựa chọn người dùng và kết quả văn bản sau khi xử lý được hiển thị bên dưới.

- **Kết quả**: Sau khi người dùng nhấn nút "Xử lý", văn bản đã được tiền xử lý sẽ hiển thị trong khung dưới dạng danh sách từng câu hoặc đoạn văn bản đã qua các bước xử lý mà người dùng đã lựa chọn.

2.3.2 Code giao diện xử lý văn bản sử dụng Streamlit

Đoạn mã dưới đây tạo ra giao diện người dùng cho việc tiền xử lý văn bản tiếng Việt sử dụng `Streamlit`. Giao diện cho phép người dùng nhập văn bản và lựa chọn các tác vụ xử lý như loại bỏ HTML, ký tự đặc biệt, khoảng trắng thừa, chuẩn hóa Unicode, chữ thường, dấu thanh, dấu câu, tách từ, tách câu và loại bỏ stopwords. Khi người dùng nhấn nút "Xử lý", kết quả sẽ được hiển thị trên giao diện.

```
import requests
import json
import streamlit as st

base_url = "http://0.0.0.0:8000/"
path_url = "api/v1/pre-processing"
headers = {"Content-Type": "application/json"}

def call_api(text,
             clean_html=True,
             clean_special_char=True,
             clean_extra_whitespace=True,
             chuan_hoa_unicode_action=True,
             chuan_hoa_chu_thuong= True,
             chuan_hoa_dau_thanh=True,
             chuan_hoa_dau_cau=True,
              split_word=True,
              split_sent=True,
              remove_sw=True
              ):
```

```
    param = {
        "text": text,
        "clean_html": clean_html,
        "clean_special_char": clean_special_char,
        "clean_extra_whitespace": clean_extra_whitespace,
        "chuan_hoa_unicode_action": chuan_hoa_unicode_action,
        "chuan_hoa_chu_thuong": chuan_hoa_chu_thuong,
        "chuan_hoa_dau_thanh": chuan_hoa_dau_thanh,
        "chuan_hoa_dau_cau": chuan_hoa_dau_cau,
        "split_word": split_word,
        "split_sent": split_sent,
        "remove_sw": remove_sw,
    }
    url = base_url + path_url
    rsp = requests.get(url=url, params=param, headers=headers)
    data = json.loads(rsp.text)["data"]
    return data

def gui():
    st.title("ĐỒ ÁN XỬ LÝ VĂN BẢN TIẾNG VIỆT")

    text = st.text_input("Nhập vào đoạn văn bản cần xử lý:")
    st.write("Lựa chọn các tác vụ bạn mong muốn:")
    col1, col2, col3 = st.columns(3)
    with col1:
        st.markdown("<p style='color: blue;'>Các tác vụ làm sạch dữ liệu</p>", un
        clean_html = st.checkbox("Xoá mã HTML")
        clean_special_char = st.checkbox("Xoá các kí tự đặc biệt")
        clean_extra_whitespace = st.checkbox("Xoá các khoảng trắng thừa")
        remove_sw = st.checkbox("Loại bỏ stopwords")
```

```
with col2:
    st.markdown("<p style='color: blue;'>Các tác vụ chuẩn hoá dữ liệu</p>", u
    chuan_hoa_unicode_action = st.checkbox("Chuẩn hoá unicode")
    chuan_hoa_chu_thuong = st.checkbox("Chuẩn hoá về chữ thường")
    chuan_hoa_dau_thanh = st.checkbox("Chuẩn hoá dấu thanh")
    chuan_hoa_dau_cau = st.checkbox("Chuẩn hoá dấu câu")
with col3:
    st.markdown("<p style='color: blue;'>Các tác vụ tách dữ liệu</p>", unsafe
    split_word = st.checkbox("Tách từ")
    split_sent = st.checkbox("Tách câu")
if st.button('Xử lý'):
    data = call_api(text=text,
                    clean_html=clean_html,
                    clean_special_char=clean_special_char,
                    clean_extra_whitespace=clean_extra_whitespace,
                    chuan_hoa_unicode_action=chuan_hoa_unicode_action,
                    chuan_hoa_chu_thuong=chuan_hoa_chu_thuong,
                    chuan_hoa_dau_thanh=chuan_hoa_dau_thanh,
                    chuan_hoa_dau_cau= chuan_hoa_dau_cau,
                    split_word=split_word,
                    split_sent=split_sent,
                    remove_sw=remove_sw)
    with st.container():
        st.header("Văn bản sau khi được xử lý: ")
        for i in data:
            st.write(i)
else:
    st.write('Hãy nhấn nút Xử lý để bắt đầu')
```