|  |  |  |
| --- | --- | --- |
| **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC KINH TẾ KỸ THUẬT CÔNG NGHIỆP**  **KHOA: CÔNG NGHỆ THÔNG TIN** |  |  |



**KHÓA LUẬN TỐT NGHIỆP**

**Đề tài: Xây dựng ứng dụng tách từ và gán nhãn từ loại văn bản Tiếng Việt**

**Giảng viên hướng dẫn:** ThS. Đỗ Hồng Lĩnh

**Khoá :** 14

**Sinh viên thực hiện :**

1. Phạm Thị Tâm - 20103100960
2. Đặng Thị Minh Trang - 20103100173

**Hà Nội, 4/2024**

|  |  |  |
| --- | --- | --- |
| **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC KINH TẾ KỸ THUẬT CÔNG NGHIỆP**  **KHOA: CÔNG NGHỆ THÔNG TIN** |  |  |



**KHÓA LUẬN TỐT NGHIỆP**

**Đề tài: Xây dựng ứng dụng tách từ và gán nhãn từ loại văn bản Tiếng Việt**

**Giảng viên hướng dẫn:** ThS. Đỗ Hồng Lĩnh

**Khoá :** 14

**Sinh viên thực hiện :**

1. Phạm Thị Tâm - 20103100960
2. Đặng Thị Minh Trang - 20103100173

**Hà Nội, 4/2024**

# LỜI CAM ĐOAN

Chúng em xin cam đoan:

- Kết quả đạt được trong đề tài: “*Xây dựng ứng dụng tách từ và gán nhãn từ loại văn bản Tiếng Việt* “ là một sản phẩm nghiên cứu độc lập không có sự sao chép lại của người khác. Đây là sản phẩm mà chúng em đã đạt được trong quá trình nghiên cứu khi học tập tại trường. Các số liệu và kết quả thu được trong quá trình nghiên cứu của chúng em là của cá nhân hoặc là được tổng hợp từ nhiều nguồn tài liệu với thái độ khách quan, trung thực dưới sự hướng dẫn của giảng viên Ths. Đỗ Hồng Lĩnh.

- Các nguồn tài liệu của các tác giả được tham khảo đều được liệt kê và trích dẫn đầy đủ và rõ ràng trong khóa luận, chúng em không sử dụng bất cứ tài liệu nào mà không có trích dẫn. Tất cả các sao chép không hợp lệ, vi phạm quy chế đào tạo của nhà trường chúng em xin chịu hoàn toàn trách nhiệm theo quy định.

Hà Nội, ngày tháng năm 2024

Sinh viên thực hiện

***Phạm Thị Tâm***

***Đặng Thị Minh Trang***

# LỜI CẢM ƠN

Nhận được sự phân công của khoa Công nghệ Thông tin, trường Đại học Kinh tế - Kỹ thuật Công nghiệp và dưới sự đồng ý của giảng viên hướng dẫn ThS. Đỗ Hồng Lĩnh, chúng em đã được giao đề tài “Xây dựng ứng dụng tách từ và gán nhãn từ loại văn bản Tiếng Việt “.

Khóa luận có thể được coi là một dự án nho nhỏ. Do vậy, để có thể hoàn thành dự án này là một quá trình không phải dễ dàng đối với sinh viên chúng em. Chúng em xin gửi lời cảm ơn đến khoa Công Nghệ Thông Tin đã phê duyệt, tạo điều kiện tốt cho chúng em có thời gian để nghiên cứu và hoàn thành khoá luận. Đặc biệt, chúng em xin cảm ơn sâu sắc tới giảng viên Ths.Đỗ Hồng Lĩnh đã trực tiếp hướng dẫn, đồng hành cùng em trong cả quá trình nghiên cứu để chúng em có thể hoàn thành đề tài đúng nội dung và tiến độ được giao.

Đồng thời, cho phép chúng em xin được gửi lời cảm ơn tới toàn thể các thầy cô khoa Công Nghệ Thông Tin đã dạy dỗ, truyền đạt không chỉ những kiến thức chuyên môn mà còn là kinh nghiệm sống, kinh nghiệm làm việc trong quãng thời gian 4 năm qua.

Chúng em đã nỗ lực cố gắng hoàn thành khóa luận bằng chính năng lực của bản thân, tuy nhiên do còn ít kinh nghiệm thực tiễn, kiến thức có hạn nên chắc chắn sẽ không tránh được việc tồn tại những thiếu sót trong quá trình nghiên cứu đề tài và trình bày báo cáo. Chúng em rất mong nhận được những ý kiến đóng góp giá trị của các thầy cô để giúp đề tài này được hoàn thiện hơn nữa.

Nhân đây chúng em xin trân trọng cảm ơn sự quan tâm, giúp đỡ và những sự tham gia góp ý quý báu của các thầy cô giảng viên trong khoa giành cho khóa luận này. Em gửi lời chúc khoa Công nghệ Thông tin mình ngày càng phát triển và lớn mạnh, chúc các thầy cô sức khoẻ, tươi trẻ và nhiệt huyết trong sự nghiệp trồng người.

Hà Nội, ngày tháng năm 2024Sinh viên thực hiện

***Phạm Thị Tâm***

***Đặng Thị Minh Trang***

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Giảng viên hướng dẫn

*ThS. Đỗ Hồng Lĩnh*

# NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**MỤC LỤC**

[**LỜI CAM ĐOAN** i](#_Toc164170505)

[**LỜI CẢM ƠN** ii](#_Toc164170506)

[**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN** iii](#_Toc164170507)

[**NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN** iv](#_Toc164170508)

[**LỜI NÓI ĐẦU** ix](#_Toc164170509)

[**1. LÝ DO CHỌN ĐỀ TÀI** ix](#_Toc164170510)

[**2. MỤC TIÊU NGHIÊN CỨU** ix](#_Toc164170511)

[**3. ĐỐI TƯỢNG VÀ NHIỆM VỤ CỦA LUẬN VĂN** x](#_Toc164170512)

[**4. PHƯƠNG PHÁP VÀ NỘI DUNG NGHIÊN CỨU** x](#_Toc164170513)

[**5. CẤU TRÚC LUẬN VĂN** x](#_Toc164170514)

[**CHƯƠNG 1: TỔNG QUAN VỀ XỬ LÝ NGÔN NGỮ TỰ NHIÊN (NLP)** 1](#_Toc164170515)

[1.1 GIỚI THIỆU VỀ XỬ LÝ NGÔN NGỮ TỰ NHIÊN (NLP) 1](#_Toc164170516)

[**1.1.1 Lịch sử của NLP và các giai đoạn phát triển** 1](#_Toc164170517)

[**1.1.2 Ứng dụng của NLP** 2](#_Toc164170518)

[**1.1.3 Ưu điểm và nhược điểm của NLP** 4](#_Toc164170519)

[**1.1.4 Các bài toán cơ bản trong NLP** 6](#_Toc164170520)

[**1.1.5 Các kỹ thuật trong NLP** 7](#_Toc164170521)

[1.2 NGÔN NGỮ VÀ THƯ VIỆN SỬ DỤNG 14](#_Toc164170522)

[**1.2.1. Giới thiệu về ngôn ngữ lập trình Python** 14](#_Toc164170523)

[**1.2.2. Lịch sử phát triển** 15](#_Toc164170524)

[**1.2.3. Cách cài đặt Python** 15](#_Toc164170525)

[**1.2.4. Đặc điểm** 16](#_Toc164170526)

[**1.2.5. Thư viện** 17](#_Toc164170527)

[**1.2.6. Python có lợi ích gì?** 20](#_Toc164170528)

[**1.2.7. Các khái niệm và lệnh Python cơ bản** 21](#_Toc164170529)

[**CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG TÁCH TỪ VÀ GÁN NHÃN TỪ LOẠI** 25](#_Toc164170530)

[2.1. KHÁI QUÁT VỀ TỪ VỰNG TIẾNG VIỆT 25](#_Toc164170531)

[2.2. BÀI TOÁN TÁCH TỪ VÀ GÁN NHÃN TỪ LOẠI 27](#_Toc164170532)

[2.3. TỔNG QUAN VỀ CÁCH TIẾP CẬN GIẢI BÀI TOÁN 27](#_Toc164170533)

[2.4. HỆ THỐNG TÁCH TỪ 28](#_Toc164170534)

[**2.4.1 Hướng tiếp cận bài toán tách từ** 29](#_Toc164170535)

[**2.4.2 Các phương pháp sử dụng trong bài toán tách từ** 30](#_Toc164170536)

[2.4.2.1. Phương pháp so khớp từ dài nhất (Longest Matching) 30](#_Toc164170537)

[2.4.2.2 Phương pháp so khớp cực đại (Maximum Matching) 31](#_Toc164170538)

[2.4.2.3 Phương pháp tách từ sử dụng mô hình N-gram 34](#_Toc164170539)

[2.4.2.4 Phương pháp tách từ sử dụng mô hình Markov ẩn 36](#_Toc164170540)

[2.5 HỆ THỐNG GÁN NHÃN TỪ LOẠI 38](#_Toc164170541)

[**2.5.1 Các nhãn từ loại** 38](#_Toc164170542)

[**2.5.2 Mô hình markov ẩn (hidden markov model – hmm) kết hợp thuật toán Viterbi** 39](#_Toc164170543)

[**CHƯƠNG 3: CHƯƠNG TRÌNH THỰC NGHIỆM** 47](#_Toc164170544)

[3.1. XÂY DỰNG CHỨC NĂNG, NHIỆM VỤ CỦA HỆ THỐNG 47](#_Toc164170545)

[**3.1.1. Xây dựng chức năng hệ thống** 47](#_Toc164170546)

[**3.1.2. Mô tả chức năng hệ thống** 47](#_Toc164170547)

[3.2. MÔI TRƯỜNG CÀI ĐẶT 49](#_Toc164170548)

[3.3. CÁC THƯ VIỆN PYTHON SỬ DỤNG TRONG CHƯƠNG TRÌNH 49](#_Toc164170549)

[**3.3.1 Thư viện Flask** 49](#_Toc164170550)

[**3.3.2 Thư viện Pyvi** 50](#_Toc164170551)

[**3.3.3 Thư viện Underthesea** 51](#_Toc164170552)

[**3.3.4 Phương thức sử dụng các thư viện trong Python:** 51](#_Toc164170553)

[3.4 GIẢI THUẬT DÙNG CHO TÁCH TỪ VÀ GÁN NHÃN TỪ 51](#_Toc164170554)

[**3.4.1 Hàm ViTokenizer** 51](#_Toc164170555)

[**3.4.2 Hàm pos\_tag** 52](#_Toc164170556)

[3.5 CHƯƠNG TRÌNH CÀI ĐẶT 53](#_Toc164170557)

[3.6 GIAO DIỆN CHƯƠNG TRÌNH 58](#_Toc164170558)

[**3.6.1 Giao diện trang chủ** 58](#_Toc164170559)

[**3.6.2 Menu** 60](#_Toc164170560)

[**3.6.3 Chức năng tách từ** 61](#_Toc164170561)

[**3.6.4 Chức năng gán nhãn từ loại** 61](#_Toc164170562)

[**3.6.5 Giao diện phần footer** 62](#_Toc164170563)

[**CHƯƠNG 4: TỔNG KẾT** 63](#_Toc164170564)

[4.1. KẾT LUẬN 63](#_Toc164170565)

[4.2. HƯỚNG PHÁT TRIỂN 63](#_Toc164170566)

[**TÀI LIỆU THAM KHẢO** 64](#_Toc164170567)

**DANH MỤC HÌNH ẢNH**

[Hình 1.1 Kết quả ví dụ ma trận TF – IDF 8](#_Toc164170568)

[Hình 1.2 Chương trình chạy demo mô hình TF – IDF 9](#_Toc164170569)

[Hình 1.3 Chương trình chạy demo mô hình Consine Similarity 11](#_Toc164170570)

[Hình 1.4 Chương trình chạy demo mô hình N-grams 12](#_Toc164170571)

[Hình 1.5 Chương trình chạy demo mô hình Skip-grams 14](#_Toc164170572)

[Hình 1.6 Giao diện trang chủ download Python 16](#_Toc164170573)

[Hình 1.7 Chương trình chạy demo lệnh print() 22](#_Toc164170574)

[Hình 1.8 Chương trình chạy demo lệnh len() 22](#_Toc164170575)

[Hình 1.9 Chương trình chạy demo lệnh range() 23](#_Toc164170576)

[Hình 1.10 Chương trình chạy demo lệnh if - else() 23](#_Toc164170577)

[Hình 1.11 Chương trình chạy demo vòng lặp while() 23](#_Toc164170578)

[Hình 1.12 Chương trình chạy demo sử dụng hàm 24](#_Toc164170579)

[Hình 1.13 Chương trình chạy demo sử dụng thư viện và module 24](#_Toc164170580)

[Hình 1.14 Chương trình chạy demo lệnh nhập và xuất dữ liệu 24](#_Toc164170581)

[Hình 2.1 Sơ đồ hướng tiếp cận bài toán tách từ 29](file:///F:\thinh_giang\Báo%20cáo%20tiến%20độ\25_KLTN_K14_PhạmThịTâm_ĐặngThịMinhTrang_ThSĐỗHồngLĩnh_Côngnghệthôngtin_16.4.2024.docx#_Toc164170582)

[Hình 2.2 Mô hình Markov ẩn 36](file:///F:\thinh_giang\Báo%20cáo%20tiến%20độ\25_KLTN_K14_PhạmThịTâm_ĐặngThịMinhTrang_ThSĐỗHồngLĩnh_Côngnghệthôngtin_16.4.2024.docx#_Toc164170583)

[Hình 2.3 Biểu đồ chuyển tiếp trạng thái của mô hình Markov ẩn 37](file:///F:\thinh_giang\Báo%20cáo%20tiến%20độ\25_KLTN_K14_PhạmThịTâm_ĐặngThịMinhTrang_ThSĐỗHồngLĩnh_Côngnghệthôngtin_16.4.2024.docx#_Toc164170584)

[Hình 2.4 Bước đệ quy của thuật toán Viterbi 42](#_Toc164170585)

[Hình 2.5 Bước quay lui của thuật toán Viterbi 43](#_Toc164170586)

[Hình 3.1. Biểu đồ phân cấp chức năng hệ thống 48](#_Toc164170587)

[Hình 3.2. Biểu đồ luồng dữ liệu mức khung cảnh 49](#_Toc164170588)

[Hình 3.3. Giao diện màn hình trang chủ (1) 58](#_Toc164170589)

[Hình 3.4. Giao diện màn hình trang chủ (2) 59](#_Toc164170590)

[Hình 3.5. Giao diện giới thiệu 60](#_Toc164170591)

[Hình 3.6. Giao diện nhãn dán Tiếng Việt 60](#_Toc164170592)

[Hình 3.7. Giao diện phần liên hệ 61](#_Toc164170593)

[Hình 3.9. Kết quả gán nhãn từ tiếng Việt 62](#_Toc164170594)

[Hình 3.10. Giao diện phần footer 62](#_Toc164170595)

**DANH MỤC BẢNG BIỂU**

[Bảng 2.1 Độ dài của từ tính theo âm tiết 25](#_Toc164170596)

[Bảng 2.2 Kết quả ví dụ giải thuật phương pháp so khớp từ dài nhất 31](#_Toc164170597)

[Bảng 2.3 Danh sách nhãn từ loại 39](#_Toc164170598)

[Bảng 2.4 Xác suất chuyển từ loại 45](#_Toc164170599)

[Bảng 2.5 Xác suất nhận nhãn loại 45](#_Toc164170600)

[Bảng 3.1 Các đối tượng trên trang chủ 60](#_Toc164170601)

# **LỜI NÓI ĐẦU**

## **1. LÝ DO CHỌN ĐỀ TÀI**

Trong thời đại công nghệ 4.0 ngày nay, với sự phát triển của trí tuệ nhân tạo thì lĩnh vực xử lý ngôn ngữ tự nhiên đã mở ra những cánh cửa mới cho nhiều ứng dụng tiện ích trong ngôn ngữ học và xử lý văn bản. Việc xây dựng ứng dụng tách từ và gán nhãn từ loại cho văn bản Tiếng Việt không chỉ là một thách thức mà còn là cơ hội để tận dụng những tiềm năng của công nghệ nhằm cải thiện chất lượng xử lý thông tin ngôn ngữ trong thực tế.

Ngôn ngữ là cầu nối quan trọng giữa con người với con người cũng như giữa con người với máy tính và việc hiểu rõ cấu trúc ngôn ngữ, tách từ và gán nhãn từ loại đóng vai trò quan trọng trong việc tiền xử lý thông tin ngôn ngữ tự nhiên. Đối với Tiếng Việt, một ngôn ngữ đa dạng và phong phú, thách thức đặt ra là sự phức tạp của cú pháp và ngữ nghĩa. Đồng thời, những tiến bộ trong lĩnh vực học máy và trí tuệ nhân tạo đang mở ra những cơ hội mới để xây dựng các mô hình thông minh có khả năng hiểu và xử lý ngôn ngữ một cách chính xác và linh hoạt.

Làm thế nào chúng ta có thể thuận lợi từ những tiềm năng này để xây dựng một ứng dụng tách từ và gán nhãn từ loại cho văn bản Tiếng Việt một cách hiệu quả? Đó sẽ là một hành trình đầy thách thức nhưng cũng là cơ hội để đóng góp vào sự phát triển của công nghệ và ứng dụng vào nhiều lĩnh vực như xử lý ngôn ngữ tự nhiên, dịch thuật, và tìm kiếm thông tin.  
 Từ xu hướng phát triển công nghệ cũng như nhu cầu thực tiễn nêu trên, chúng em đã quyết định thực hiện đề tài ***“Xây dựng ứng dụng tách từ và gán nhãn từ loại văn bản Tiếng Việt”.***

## **2. MỤC TIÊU NGHIÊN CỨU**

- Tìm hiểu khái niện cơ bản và nghiên cứu tổng quan về xử lý ngôn ngữ tự nhiên, các thuật toán giải quyết bài toán tách từ và gán nhãn từ loại tiếng Việt.  
 - Tìm hiểu và nghiên cứu về mô hình markov ẩn và các thư viện hỗ trợ xử lý tiếng Việt như ViTokenizer, underthesea,….  
 - Nghiên cứu và tìm hiểu về ngôn ngữ lập trình Python.  
 - Xây dựng và cài đặt chương trình thực nghiệm.

## **3. ĐỐI TƯỢNG VÀ NHIỆM VỤ CỦA LUẬN VĂN**

*Đối tượng:* Đối tượng nghiên cứu của luận văn này là ứng dụng tách từ và gán nhãn từ loại Tiếng Việt dựa trên mô hình Markov ẩn và cách sử dụng các thư viện hỗ trợ như ViTokenizer, underthesea,….   
 *Nhiệm vụ:*

Luận văn tập trung vào cách tách từ và gán nhãn từ loạicho Tiếng Việt.

## **4. PHƯƠNG PHÁP VÀ NỘI DUNG NGHIÊN CỨU**

*Phương pháp nghiên cứu:* - Phương pháp nghiên cứu tài liệu.

- Phương pháp thực hành

*Nội dung nghiên cứu:*

- Nghiên cứu các đặc trưng của ngôn ngữ, các khái niệm, mục đích và phương pháp xử lý ngôn ngữ tự nhiên, các ứng dụng của lĩnh vực xử lý ngôn ngữ tự nhiên trong khoa học và đời sống.  
 - Nghiên cứu về mô hình markov ẩn và cách sử dụng các thư viện hỗ trợ tiếng Việt như ViTokenizer, underthesea,….  
 - Xây dựng chương trình thực nghiệm.  
 - Đưa ra nhận xét và đánh giá của bản thân.

## **5. CẤU TRÚC LUẬN VĂN**

Luận văn gồm 4 chương:  
 Chương 1: Tổng quan về xử lý ngôn ngữ tự nhiên (NLP) Chương 2: Phân tích và thiết kế hệ thống tách từ và gán nhãn từ loại

Chương 3: Chương trình thực nghiệm

Chương 4: Tổng kết

# **CHƯƠNG 1: TỔNG QUAN VỀ XỬ LÝ NGÔN NGỮ TỰ NHIÊN (NLP)**

## 1.1 GIỚI THIỆU VỀ XỬ LÝ NGÔN NGỮ TỰ NHIÊN (NLP)

### **1.1.1 Lịch sử của NLP và các giai đoạn phát triển**

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực của trí tuệ nhân tạo (AI) liên quan đến việc làm cho máy tính có khả năng hiểu, tạo ra và tương tác bằng ngôn ngữ tự nhiên với con người một cách dễ dàng và hiệu quả. Mục đích của những nghiên cứu trong lĩnh vực xử lý ngôn ngữ tự nhiên là hỗ trợ máy tính hiểu và thực thi hiệu quả những công việc liên quan đến ngôn ngữ như: giao tiếp giữa người và máy, tăng cường hiệu quả giao tiếp giữa người với người, hoặc tối thiểu nhất là nâng cao hiệu quả xử lý văn bản và tiếng nói. Sự ra đời của lĩnh vực xử lý ngôn ngữ tự nhiên diễn ra vào những năm đầu của thế kỷ 20 và đã trải qua nhiều giai đoạn phát triển và đã đạt được những thành tựu nhất định.

* Giai đoạn 1: **Những Bước Đầu Tiên (trước thế kỷ 20):** Những nghiên cứu ban đầu về xử lý ngôn ngữ tự nhiên hầu như tập trung vào việc phân tích và mô hình hóa ngôn ngữ từ cú pháp đến ngữ nghĩa. Những công việc này thường được thực hiện bằng tay.
* Giai đoạn 2: **Thập niên 1940-1950:** Trong giai đoạn này, máy tính đã trở thành một công cụ sẵn sàng cho nghiên cứu NLP. Claude Shannon, một nhà toán học và kỹ sư điện tử, đã xuất bản bài báo nổi tiếng về "Thông tin và Ngôn Ngữ" vào năm 1948, đưa ra các ý tưởng về mã hóa thông tin và xử lý ngôn ngữ tự nhiên.
* Giai đoạn 3: **Thập Niên 1950-1960:** Trong giai đoạn này, công nghệ NLP bắt đầu sử dụng máy tính để phân tích và xử lý ngôn ngữ. Một số dự án đáng chú ý bao gồm Georgetown-IBM experiment họ đã phát triển một hệ thống dịch máy đầu tiên.
* Giai đoạn 4: **Thập Niên 1960-1970:** Trong thời kỳ này, các mô hình thống kê và ngôn ngữ học xuất hiện, mở ra các phương pháp mới để xử lý ngôn ngữ tự nhiên. Một trong những ví dụ nổi bật là nghiên cứu về ngữ liệu và cấu trúc câu của Noam Chomsky.
* Giai đoạn 5: **Thập Niên 1980-1990:** Công nghệ NLP bắt đầu áp dụng nhiều hơn trong thực tế với việc phát triển các ứng dụng như hệ thống tìm kiếm thông tin và các hệ thống chatbot đơn giản.
* Giai đoạn 6: **Thập Niên 2000 - 2010:** Sự phát triển mạnh mẽ của web 2.0 và dữ liệu lớn đã thúc đẩy sự tiến bộ trong xử lý ngôn ngữ tự nhiên, với sự xuất hiện của các công nghệ như máy học, học sâu, và các phương pháp xử lý ngôn ngữ tự nhiên dựa trên dữ liệu.
* Giai đoạn 7: **Thập Niên 2010 - Đến Nay:** Sự phổ biến của trí tuệ nhân tạo, đặc biệt là học sâu và mạng nơ-ron hồi tiếp, đã tạo ra một cuộc cách mạng trong NLP. Các mô hình ngôn ngữ như Transformer đã giúp cải thiện nhiều nhiệm vụ trong NLP như dịch máy, xây dựng ngôn ngữ tự nhiên và nhận dạng ý định.

Tóm lại, sự phát triển của xử lý ngôn ngữ tự nhiên đã trải qua nhiều giai đoạn lịch sử với sự đóng góp của các ngành khoa học khác nhau như ngôn ngữ học, toán học, tương tác người máy, khoa học máy tính và trí tuệ nhân tạo và đã đạt được những thành công, kết quả nghiên cứu cụ thể ứng dụng rất nhiều vào sự phát triển của đời sống.

### **1.1.2 Ứng dụng của NLP**

Có nhiều ứng dụng của xử lý ngôn ngữ tự nhiên (NLP) trong nhiều lĩnh vực khác nhau. Dưới đây là một số ứng dụng phổ biến của NLP:

* **Dịch Máy:** Dịch máy là một trong những ứng dụng NLP phổ biến nhất. Công nghệ này cho phép dịch văn bản từ một ngôn ngữ sang ngôn ngữ khác một cách tự động, giúp giảm rào cản ngôn ngữ và tăng cơ hội giao tiếp và hợp tác toàn cầu. Ví dụ:
* Google Translate, Microsoft Translator : Dịch văn bản, trang web và văn bản tự động giữa nhiều ngôn ngữ.
* DeepL : Dịch máy sử dụng học sâu để cung cấp các bản dịch chất lượng cao.
* Hệ Thống Hỏi Đáp (Question Answering Systems) :
* Siri, Google Assistant, Amazon Alexa: Hệ thống trả lời câu hỏi và cung cấp thông tin từ dữ liệu trực tuyến.
* Quora, Stack Overflow: Các cộng đồng trả lời câu hỏi trực tuyến dựa trên cộng đồng người dùng.
* **Chatbot và Trò Chuyện Tự Động:** Chatbot sử dụng NLP để hiểu và phản hồi tự động cho người dùng trong các nhiệm vụ như hỗ trợ khách hàng, đặt hàng trực tuyến, hoặc cung cấp thông tin. Ví dụ:
* Facebook Messenger Bot, Slack Bot: Hệ thống trò chuyện tự động để hỗ trợ khách hàng và cung cấp thông tin sản phẩm hoặc dịch vụ.
* Xiaoice (Microsoft), Replika: Chatbots được thiết kế để tương tác xã hội và cung cấp hỗ trợ tâm lý.
* **Phân Tích Ý Kiến:** NLP được sử dụng để phân tích ý kiến của người dùng từ dữ liệu trên mạng xã hội, bài đánh giá sản phẩm, hoặc các bài viết trên blog để hiểu ý kiến, cảm xúc và xu hướng của công chúng. Ví dụ:
* Brandwatch, Talkwalker: Phân tích ý kiến và cảm xúc từ dữ liệu trên mạng xã hội và các nguồn tin tức để đo lường quan điểm của công chúng đối với thương hiệu hoặc sự kiện.
* Amazon Review Analysis: Phân tích ý kiến từ đánh giá sản phẩm trên Amazon để đánh giá sự hài lòng của khách hàng.
* **Phân Loại Văn Bản (Text Classification):**
* Gmail, Outlook: Phân loại email vào các hộp thư đến quan trọng, thư rác, hoặc các danh mục khác dựa trên nội dung.
* Twitter, Instagram: Phân loại nội dung dựa trên thẻ (hashtag), chủ đề hoặc nội dung để hiển thị trong các danh sách dựa trên sở thích của người dùng.
* **Tạo Văn Bản Tự Động (Text Generation):**
* GPT-3 (OpenAI), BERT (Google): Mô hình sinh văn bản tự động có khả năng tạo ra văn bản tự nhiên và logic dựa trên dữ liệu đầu vào.
* Kiểm Tra Lỗi Chính Tả Tự Động:
* Sử dụng phần mềm tin học để tự động kiểm tra và phát hiện lỗi chính tả trong văn bản (lỗi từ vựng, lỗi ngữ pháp, lỗi ngữ nghĩa) và đưa ra gợi ý cách sửa lỗi giúp cải thiện chất lượng và độ chính xác của văn bản.
* **Xử lý ngôn ngữ tự nhiên trong y tế:** Trong lĩnh vực chăm sóc sức khỏe con người, NLP dùng vào việc phân tích dữ liệu y khoa, các lưu ý của bác sĩ, và thông tin bệnh nhân để hỗ trợ chẩn đoán, điều trị, và nghiên cứu y khoa. Ví dụ:
* IBM Watson for Oncology: Hệ thống hỗ trợ quyết định trong chẩn đoán và điều trị ung thư dựa trên việc phân tích các tài liệu y khoa.
* Phân tích bản ghi y tế: Phân tích và trích xuất thông tin từ bản ghi y tế điện tử để hỗ trợ quản lý bệnh nhân và nghiên cứu y học.
* **Xử Lý Ngôn Ngữ Tự Nhiên Trong Tài Chính:** Trong tài chính, NLP được sử dụng để phân tích tin tức thị trường, bản tin tài chính, và báo cáo tài chính để dự đoán xu hướng thị trường, đánh giá rủi ro, và đưa ra quyết định đầu tư.
* **Hỗ Trợ Giáo Dục và Đào Tạo:** NLP được dùng để phát hiện năng lực học tập và tiến bộ của học sinh, cá nhân hóa các bài học cho mỗi học viên, và đưa ra cho giáo viên và đợn vị giáo dục những phản hồi tức thì của người học trong quá trình học tập.

### **1.1.3 Ưu điểm và nhược điểm của NLP**

**a, Ưu điểm**

* Giao Tiếp Tự Nhiên:
  + Ưu điểm : NLP cho phép con người dễ dàng hơn trong việc giao tiếp với máy tính thông qua ngôn ngữ tự nhiên, tạo ra trải nghiệm giao tiếp một cách tự nhiên hơn, trực quan và dễ sử dụng.
  + Ví dụ: Chatbot và hệ thống tự động nhắn tin cho phép người dùng tương tác với máy tính có cảm giác như đang nói chuyện với một con người thật.
* Tích Hợp Dữ Liệu Lớn:
* Ưu điểm : NLP có khả năng xử lý và phân tích dữ liệu lớn từ nhiều nguồn khác nhau như văn bản, email, trang web, và mạng xã hội.

Ví dụ : Công cụ phân tích ý kiến sử dụng NLP để đánh giá cảm xúc của người dùng từ hàng triệu bài đánh giá sản phẩm trên mạng.

* Tăng Hiệu Suất Làm Việc:
  + Ưu điểm : NLP giúp tự động hóa nhiều nhiệm vụ như phân loại văn bản, trả lời câu hỏi, và tạo ra nội dung, giúp tăng hiệu suất và tiết kiệm thời gian cho các doanh nghiệp và cá nhân.

Ví dụ : Hệ thống phân loại email tự động có thể giúp giảm thời gian xử lý email cho các tổ chức và doanh nghiệp.

* Hỗ Trợ Quyết Định:
  + Ưu điểm : NLP có thể phân tích và tóm tắt thông tin từ dữ liệu phức tạp, giúp người dùng ra quyết định dựa trên thông tin chính xác và tổng hợp.

Ví dụ: Công cụ phân tích văn bản trong tài chính có thể giúp nhà đầu tư đưa ra quyết định đầu tư thông minh dựa trên phân tích dữ liệu thị trường và tin tức tài chính.

**b, Nhược điểm:**

* Hiểu Biết Không Chính Xác:
  + Nhược điểm : Trong một số trường hợp, các hệ thống NLP vẫn gặp khó khăn trong việc hiểu ý nghĩa và ngữ cảnh của văn bản dẫn đến việc hiểu biết không chính xác hoặc đôi khi là hiểu biết sai lầm.
  + Ví dụ : Các hệ thống chatbot có thể hiểu sai ý định của người dùng và cung cấp phản hồi không đúng.
* Phụ Thuộc vào Dữ Liệu:
  + Nhược điểm: Hiệu suất của các hệ thống NLP phụ thuộc lớn vào chất lượng và độ phong phú của dữ liệu huấn luyện. Nếu thông tin thu thập chưa đầy đủ hoặc không tiêu biểu có thể dẫn đến kết quả sai hoặc không tốt.

Ví dụ: Dịch máy có thể đưa ra kết quả không chính xác nếu thiếu dữ liệu huấn luyện cho các ngôn ngữ hoặc lĩnh vực cụ thể.

* Bảo Mật và Quyền Riêng Tư:
  + Nhược điểm: NLP có thể tiết lộ thông tin nhạy cảm hoặc đưa ra suy luận sai lầm về cá nhân nếu không được xử lý và bảo mật cẩn thận.

Ví dụ : Hệ thống phân tích ý kiến có thể tiết lộ ý kiến cá nhân của người dùng nếu không được quản lý cẩn thận.

* Giải Quyết Hiểu Biết Ngôn Ngữ Phức Tạp:
  + Nhược điểm: Hiểu biết ngôn ngữ phức tạp như tiếng lóng, ngôn ngữ hài hước, hoặc ngôn ngữ hỗn hợp vẫn là một thách thức lớn cho các hệ thống NLP hiện nay.
  + Ví dụ: Các hệ thống NLP có thể gặp khó khăn trong việc hiểu và xử lý các biểu hiện ngôn ngữ không tiêu chuẩn.

### **1.1.4 Các bài toán cơ bản trong NLP**

* Mô hình hóa ngôn ngữ (Language modelling):

Mô hình hóa ngôn ngữ (LM) là mô hình cho phép ước lượng xác suất xuất hiện của từ hay chuỗi từ nào. Trong bài toán này, ta cần dựa trên lịch sử của các từ đã xuất hiện trước đó để dự đoán từ tiếp theo xuất hiện theo trình tự. Đây là một mô hình quan trọng trong NLP giúp cho máy móc có thể hiểu được thông tin định tính. Một số ứng dụng của Mô hình hóa ngôn ngữ bao gồm: dịch máy, tạo văn bản tự động, nhận dạng giọng nói, nhận dạng chữ viết tay, gợi ý từ và sửa lỗi chính tả.

* Phân loại văn bản (Text classification):

Phân loại văn bản là quá trình phân tích và gán nhãn cho các văn bản dựa trên nội dung của nó. Đây được coi là nền tảng cho hầu hết các học máy liên quan đến NLP, được sử dụng để xây dựng các công cụ khác nhau như trình phát hiện thư rác và chương trình phân tích cảm xúc.

* Trích xuất thông tin (Information extraction):

Trích xuất thông tin (IE) là bài toán phân tích và xác định các thông tin cần thiết đã được định nghĩa từ trước, sau đó tìm kiếm và trích xuất thông tin có liên quan từ các tài liệu văn bản không có cấu trúc hoặc bán cấu trúc như sự kiện, các mối quan hệ,....

* Truy xuất thông tin (Information retrieval):

Truy xuất thông tin là bài toán tìm kiếm thông tin từ các tài liệu văn bản không có cấu trúc nhằm thoả mãn nhu cầu tìm kiếm thông tin của người dùng. Google là một hệ thống điển hình của bài toán này. IR làm nhiệm vụ tìm kiếm các tài liệu có liên quan từ một bộ dữ liệu lớn các văn bản liên quan đến truy vấn do người dùng tìm kiếm.

* Tác tử phần mềm hội thoại (Conversational agent):

Tác tử phần mềm hội thoại là phần mềm thuộc AI hội thoại, được dùng để xây dựng các cuộc đối thoại mô phỏng các tương tác giữa người với người. Ví dụ các phần mềm phổ biến như Alexa, Siri, Google Home, … Các công nghệ như chatbot cũng được hỗ trợ bởi tác tử phần mềm hội thoại và ngày càng phổ biến trong các doanh nghiệp.

* Tóm tắt văn bản (Text summarization):

Tóm tắt văn bản là một trong những bài toán quan trọng trong xử lý ngôn ngữ tự nhiên, có chức năng tóm tắt văn bản nguồn từ người dùng thành văn bản rút gọn và vẫn giữ được những ý chính từ nội dung.

* Hỏi đáp (Question answering):

Hỏi đáp là bài toán khó được xây dựng để trả lời câu hỏi do người dùng đưa ra bằng ngôn ngữ tự nhiên của con người, dựa trên một kho dữ liệu nào đó.

* Dịch máy (Machine translation):

Dịch máy là một nhánh của xử lý ngôn ngữ tự nhiên thuộc ngôn ngữ học tính toán có chức năng dịch ngôn ngữ văn bản nguồn từ người dùng sang một ngôn ngữ khác theo yêu cầu, mà không cần sự tham gia của con người, điển hình là Google Dịch.

* Mô hình hóa chủ đề (Topic modelling):

Mô hình hóa chủ đề là một dạng mô hình thống kê giúp khai thác các “chủ đề” ẩn như cấu trúc ngữ nghĩa, những từ đặc trưng trong tập tài liệu văn bản.

**1.1.5 Các kỹ thuật trong NLP**

A. Kỹ thuật TF – IDF:

***1, Định nghĩa***

Term Frequency (TF): Tính toán tần số xuất hiện của mỗi từ trong văn bản bằng cách chia số lần xuất hiện trong mỗi văn bản cho tổng số từ trong văn bản.

Inverse Document Frequency (IDF): Tính toán giá trị thực của một từ khoá dựa trên tần số nghịch đảo của cụm từ đó trong toàn bộ các tập dữ liệu.

TF – IDF: Kết hợp giữa TF và IDF để đánh giá mức độ quan trọng của cụm từ trong một văn bản bằng cách nhân TF với IDF. Các giá trị TF – IDF cao thường phản ánh mức độ quan trọng của từ khoá đó trong một trang hoặc cả một văn bản nhưng ít xuất hiện trong nhiều văn bản.

***2, Công thức tính TF – IDF***

* **Công thức tính TF:**
* **Công thức tính IDF:**

Trong đó:

* Idf (t, D): là giá trị nghịch đảo tần suất của từ t trong tập văn bản.
* |D|: Tổng số văn bản trong tập D.
* |{|: thể hiện số văn bản trong tập D có chứa từ t.  
  **Công thức tính TF – IDF:**

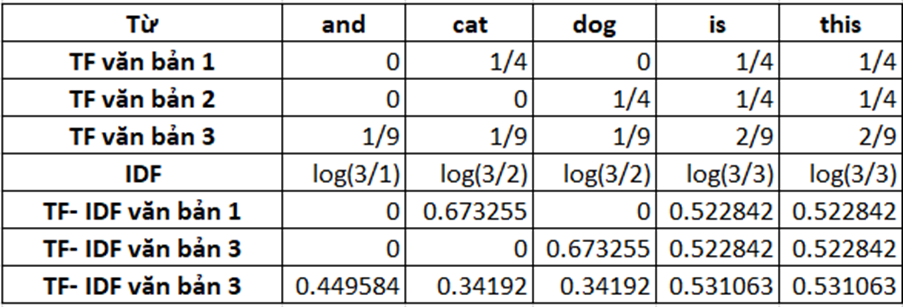
Khi đó:

Những từ có giá trị TF – IDF cao là từ khoá đó xuất hiện trong một trang hoặc cả một văn bản nhưng ít xuất hiện trong nhiều văn bản. Việc này có tác dụng giúp lọc ra những từ ngữ phổ biến và giữ lại từ khóa của văn bản đó.

Ví dụ cách tính TF – IDF:

Cho văn bản:

* Văn bản 1: “This is a cat.”
* Văn bản 2: “This is a dog.”
* Văn bản 3: “This is a cat and this is a dog.”

****

Hình 1. Kết quả ví dụ ma trận TF – IDF

Kết quả cuối cùng là ma trận TF – IDF cho từng từ trong mỗi văn bản.

***3, Cách xây dựng mô hình TF – IDF từ ví dụ***

1. Tiền xử lý văn bản

* Tokenization: Chia văn bản thành các từ hoặc token.
* Loại bỏ stop words: Loại bỏ các từ phổ biến không cần thiết như “is”, “the”, …

1. Tính toán TF

* Đếm số lần xuất hiện của mỗi từ trong mỗi văn bản.
* Tính giá trị của TF như sau: lấy số lần xuất hiện của mỗi từ chia cho cho tổng số từ trong văn bản.

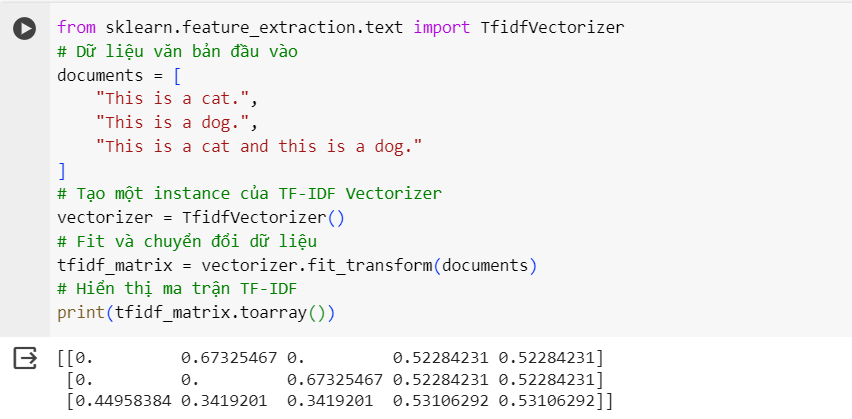
1. Tính toán IDF

* Đếm số văn bản mà mỗi từ xuất hiện.
* Tính toán IDF bằng logarit của tổng số văn bản chia cho số văn bản mà từ đó xuất hiện.

1. Tính toán TF – IDF

* Thực hiện phép nhân TF với IDF để tính toán kết quả giá trị TF – IDF cho mỗi từ trong văn bản.

***4, Cài đặt mô hình***



Hình 1. Chương trình chạy demo mô hình TF – IDF

B. Kỹ thuật Consine Similarity

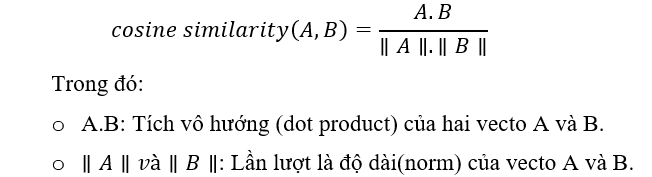
***1,*** ***Định nghĩa***

Kỹ thuật cosine similarity được sử dụng để đo độ tương đồng giữa hai vectơ trong không gian nhiều chiều.

Mỗi văn bản được biểu diễn bằng một vectơ trong không gian đa chiều, với mỗi chiều biểu diễn tần suất của các từ hoặc các đặc trưng trong văn bản.

***2,*** ***Công thức tính Cosine similarity***

* **Công thức tính Cosine similarity giữa hai vecto A và B:**

******

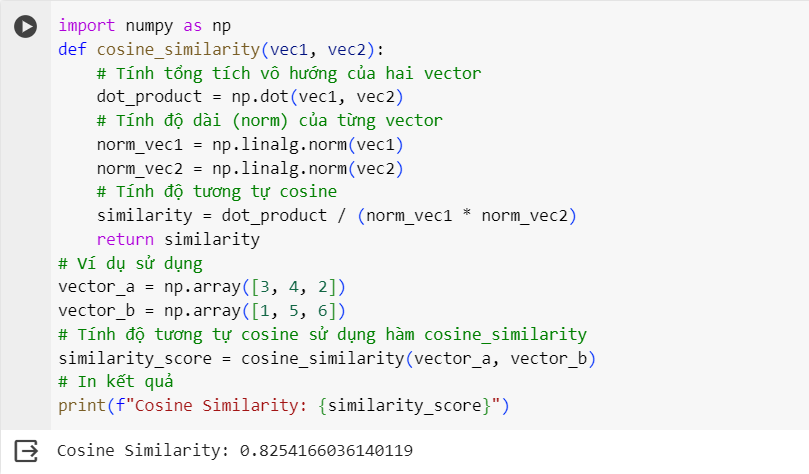
Để tính cosine similarity giữa hai vecto trong Python, chúng ta có thể sử dụng thư viện NumPy để thực hiện các phép toán đại số tuyến tính.

Kết quả sẽ là giá trị của cosine similarity giữa hai vectơ A và B. Giá trị này nằm trong khoảng từ -1 đến 1, trong đó 1 thể hiện hai vectơ hoàn toàn giống nhau, 0 thể hiện hai vectơ không có mối quan hệ tương đồng và -1 thể hiện hai vectơ hoàn toàn đối ngẫu nhau.

***3, Cách xây dựng mô hình Cosine similarity từ ví dụ***

* Tính tổng của tích vô hướng giữa hai vector, đây là phần quan trọng nhất của thuật toán cosine similarity.
* Tính độ dài (norm) của mỗi vector sử dụng hàm np.linalg.norm.
* Tính độ tương tự cosine bằng cách chia tổng tích vô hướng cho tích của độ dài hai vector.

***4, Cài đặt mô hình***



Hình 1. Chương trình chạy demo mô hình Consine Similarity

C. Kỹ thuật N-grams

***1, Định nghĩa***

Kỹ thuật N-gram: là một phương pháp trong xử lý ngôn ngữ tự nhiên (NLP) để phân tích và dự đoán chuỗi các từ hoặc ký tự trong văn bản dựa trên một ngữ cảnh có kích thước n xác định trước.

N-gram: Là một chuỗi gồm n phần tử kế tiếp trong một văn bản. N-gram có thể được hình thành từ các từ (word n-grams) hoặc các ký tự (character n-grams).

N: Đại diện cho số lượng phần tử trong mỗi chuỗi. Ví dụ, trong bigram (2-gram), n = 2, tức là chuỗi gồm 2 từ hoặc 2 ký tự liên tiếp nhau.

***2, Ý nghĩa***

Mô hình ngôn ngữ: N-grams thường dùng tạo ra mô hình ngôn ngữ để dự đoán từ tiếp theo dựa trên ngữ cảnh. Ví dụ, dự đoán từ tiếp theo trong một câu dựa trên các từ trước đó (trigram hoặc bigram model).

Phân tích ngôn ngữ: N-grams cũng được sử dụng để phân tích cấu trúc và tần suất xuất hiện của các từ hoặc ký tự trong văn bản. Việc này rất có ích trong các ứng dụng tìm kiếm thông tin, phân loại văn bản, và xử lý ngôn ngữ tự nhiên.

***3, Một số mô hình N-grams phổ biến***

* **Unigram:** với n=1, là tính tần suất xuất hiện của một kí tự (từ), như: “k”, “a”, “I” ,…
* **Bigrams:** với n=2, là mô hình dùng phổ biến trong việc phân tích các hình thái ngôn ngữ.
* **Trigrams:** với n-3, n càng lớn thì độ chính xác càng cao nhưng kéo theo đó thì độ phức tạp của bài toán cũng lớn hơn.

Ví dụ:

Với câu: ***"I love natural language processing."***

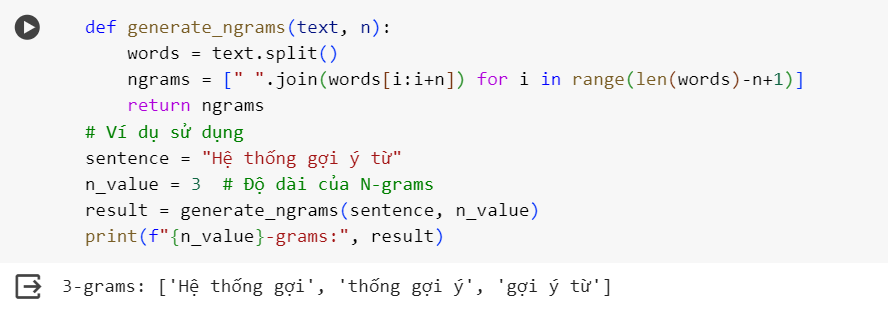
* Unigrams (1-gram): ["I", "love", "natural", "language", "processing"].
* Bigrams (2-gram): ["I love", "love natural", "natural language", "language processing"].
* Trigrams (3-gram): ["I love natural", "love natural language", "natural language processing].

***4, Ứng dụng trong NLP***

**- Mô hình ngôn ngữ:** phương pháp n-grams dùng đưa ra dự đoán từ tiếp theo trong một chuỗi từ.

**- Phân tích văn bản:** Xác định cấu trúc ngữ pháp, tần suất xuất hiện của từng từ hoặc nhóm từ trong văn bản để hỗ trợ việc tìm kiếm thông tin, gợi ý tìm kiếm, và phân loại văn bản.

***5, Cài đặt mô hình***



Hình 1. Chương trình chạy demo mô hình N-grams

D. Kỹ thuật Skip-grams

***1, Định nghĩa***

Kỹ thuật Skip-grams là một kỹ thuật trong xử lý ngôn ngữ tự nhiên (NLP) để tạo ra các ngữ cảnh xung quanh một từ cụ thể. Trong NLP, nhiều phương pháp sử dụng mô hình ngôn ngữ để ánh xạ các từ thành các vectơ. Skip-grams là một trong những phương pháp này.

Skip-grams: tập trung vào việc dự đoán từ tiếp theo trong một chuỗi từ dựa trên từ hiện tại.

Thay vì chỉ xem xét các từ lân cận trực tiếp, skip-grams cho phép bỏ qua một số từ trong ngữ cảnh xung quanh từ hiện tại.

***2, Cách hoạt động của Skip-grams***

Xác định từ trung tâm (Center word): Skip-grams chọn một từ làm trung tâm và cố gắng dự đoán các từ xung quanh nó.

Tạo các cặp từ (Word pairs): Các cặp từ được tạo ra từ việc kết hợp từ trung tâm với các từ trong ngữ cảnh xung quanh, bỏ qua một số từ theo khoảng cách đã chọn.

Huấn luyện mô hình:

* Một mô hình máy học (thường là neural network) được huấn luyện để dự đoán từ xung quanh từ trung tâm.
* Mô hình cố gắng học cách biểu diễn từ sao cho khi cho vào từ trung tâm, nó có thể dự đoán được các từ trong ngữ cảnh xung quanh.

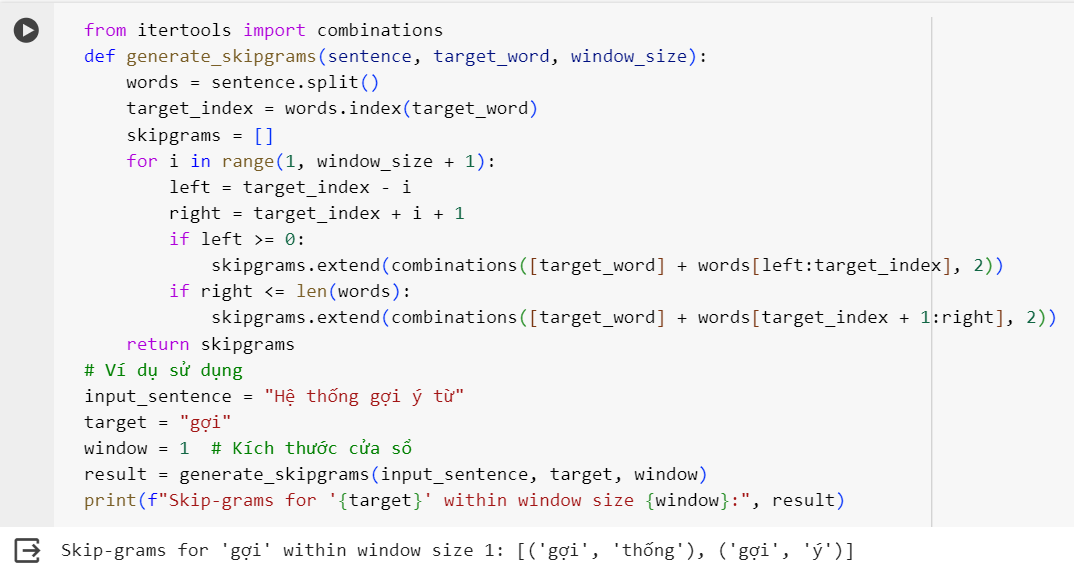
***3, Ý nghĩa và ứng dụng của Skip-grams***

Biểu diễn từ vựng: Skip-grams giúp biểu diễn từ vựng dưới dạng các vectơ có ý nghĩa trong không gian vector.

Mô hình ngôn ngữ: Cung cấp các biểu diễn từ vựng có ý nghĩa, hữu ích trong việc dự đoán từ tiếp theo trong một chuỗi từ.

Ứng dụng trong các nhiệm vụ NLP: Skip-grams có thể được sử dụng trong các nhiệm vụ như hệ thống gợi ý, phân loại văn bản, và phân tích ngữ cảnh trong các vấn đề NLP.

***4, Cài đặt mô hình***



Hình 1. Chương trình chạy demo mô hình Skip-grams

## 1.2 NGÔN NGỮ VÀ THƯ VIỆN SỬ DỤNG

### **1.2.1. Giới thiệu về ngôn ngữ lập trình Python**

Python là một ngôn ngữ lập trình thông dịch, mở rộng và dễ đọc. Nó được tạo ra bởi Guido van Rossum và lần đầu tiên xuất hiện vào năm 1991. Python được thiết kế với mục tiêu tối ưu hóa sự đọc hiểu và dễ sử dụng, đồng thời hỗ trợ nhiều mô hình lập trình như lập trình hàm, lập trình hướng đối tượng và lập trình hàm lồng ghép.

Python có cú pháp đơn giản và rõ ràng, giúp cho việc viết mã trở nên dễ dàng và hiệu quả. Ngôn ngữ này được thiết kế để đẩy mạnh sự tăng trưởng và sự linh hoạt. Python có cộng đồng lớn và năng động, cung cấp nhiều thư viện và công cụ phong phú để giải quyết các vấn đề khác nhau.

Python dùng phổ biến vào các ngành khoa học như máy học, trí tuệ nhân tạo. Cụ thể như trong lĩnh vực trí tuệ nhân tạo và học máy, các thư viện như NumPy, Pandas, TensorFlow và PyTorch cung cấp các công cụ mạnh mẽ cho việc xử lý dữ liệu và xây dựng mô hình học máy. Python cũng được sử dụng trong tự động hóa, quản lý hệ thống, phân tích dữ liệu, trò chơi và nhiều lĩnh vực khác. Đi kèm với tốc độ đi lên không ngừng là sự hỗ trợ mạnh mẽ từ cộng đồng những nhà phát triển trong lĩnh vực công nghệ thông tin, ngày nay ngôn ngữ lập trình Python trở thành một trong những ngôn ngữ lập trình được yêu thích và lựa chọn sử dụng nhiều nhất trên toàn cầu.

### **1.2.2. Lịch sử phát triển**

Python có một lịch sử phát triển dài. Phiên bản đầu tiên, Python 0.9.0, được phát hành vào tháng 2 năm 1991 bởi Guido van Rossum tại Trung tâm Nghiên cứu Mathematisch Centrum ở Hà Lan. Guido van Rossum tạo ra Python với mục tiêu tạo ra một ngôn ngữ lập trình mới, dễ đọc, dễ hiểu và mạnh mẽ.

Ban đầu, Python được phát triển như một dự án cá nhân, nhưng nhanh chóng thu hút sự quan tâm và đóng góp từ cộng đồng lập trình viên. Python đã trải qua nhiều phiên bản và bản cập nhật kể từ đó, với các phiên bản chính như Python 1.0, Python 2 và Python 3.

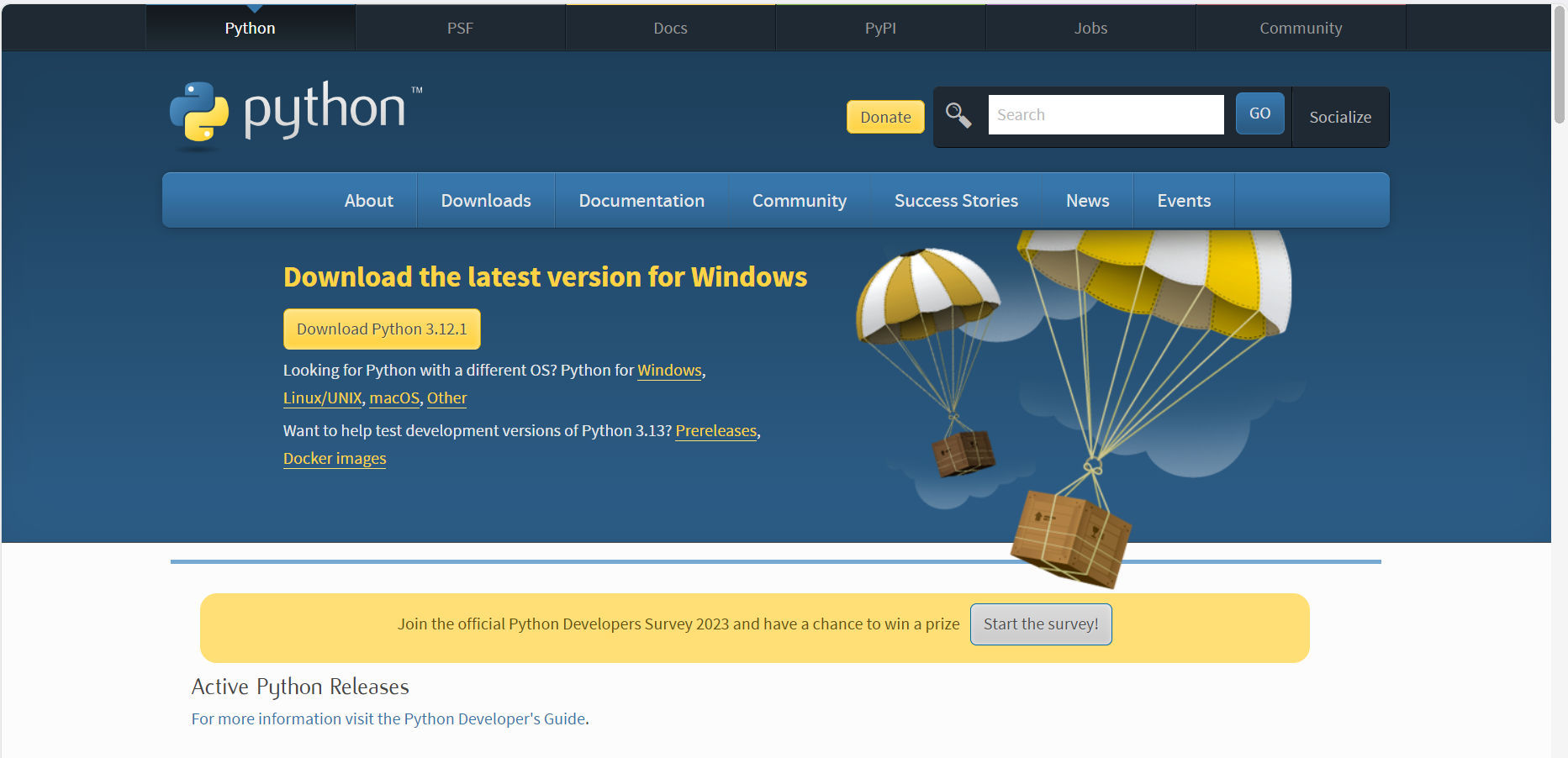
Python 2, được phát hành vào năm 2000, trở thành phiên bản phổ biến trong một thời gian dài. Tuy nhiên, sự phát triển của Python 2 đã dẫn đến một số vấn đề về sự tương thích và thiếu những cải tiến quan trọng. Do đó, Python 3 được giới thiệu vào năm 2008 nhằm giải quyết các vấn đề này và đưa ra một ngôn ngữ Python mới và cải tiến hơn.

Python 3 đưa ra nhiều cải tiến và điều chỉnh so với Python 2, nhưng có một số sự không tương thích khiến việc chuyển đổi khá phức tạp đối với các dự án lớn. Tuy nhiên, từ năm 2020, Python 2 đã chính thức ngừng được hỗ trợ, và sự tập trung của cộng đồng hiện tại là tạo ra các phiên bản mới và phát triển Python 3.

Python tiếp tục phát triển và cải thiện qua các phiên bản mới. Hiện tại, phiên bản mới nhất là Python 3.12.1, được phát hành vào ngày 8 tháng 12 năm 2023. Mỗi phiên bản đều đem lại những cải tiến và tính năng mới cho ngôn ngữ.

### **1.2.3. Cách cài đặt Python**

* *Trên Windows:*
* Tải xuống và cài đặt thông qua website chính thức của Python tại đường link sau: <https://www.python.org/downloads>.



Hình Giao diện trang chủ download Python

* Chọn cài đặt bản Python phù hợp với hệ điều hành Windows của bạn (thường là phiên bản mới nhất).
* Chạy tệp tin cài đặt đã tải xuống và làm theo hướng dẫn trên màn hình. Đảm bảo chọn "Add Python to PATH" khi được yêu cầu để có thể sử dụng Python từ dòng lệnh.
* Sau khi cài đặt hoàn tất, bạn có thể kiểm tra phiên bản Python bằng cách mở Command Prompt và chạy lệnh sau: **python –version**
* *Trên Linux:*
* Nếu bạn đang sử dụng Fedora, hãy chạy lệnh sau để cài đặt Python: **$ sudo dnf install python**
* Nếu bạn đang sử dụng hệ điều hành dựa trên Debian, hãy chạy lệnh sau để cài đặt Python: **$ sudo apt-get install python**
* *Trên Mac:*
* Python đã được cài đặt sẵn trên hệ điều hành macOS. Bạn có thể kiểm tra phiên bản Python hiện có bằng cách mở Terminal và chạy lệnh sau: **$ python –version**
* Nếu bạn muốn cài đặt phiên bản Python mới hoặc quản lý các phiên bản Python khác nhau, bạn có thể sử dụng công cụ quản lý gói như Homebrew hoặc Pyenv.

### **1.2.4. Đặc điểm**

* Python là một ngôn ngữ thông dịch

Python hỗ trợ thông dịch mã nguồn, do đó nó sẽ dịch và chạy từng dòng lệnh trong mã nguồn. Nếu gặp lỗi, nó sẽ ngừng chạy giúp lập trình viên dễ dàng phát hiện và sửa lỗi.

* Python là một ngôn ngữ dễ sử dụng

Python đặc biệt với việc sử dụng từ ngữ phổ biến trong tiếng Anh thay vì dấu ngoặc ôm, tạo điều kiện cho việc sắp xếp mã thông qua thụt đầu dòng.

* Python là một ngôn ngữ linh hoạt

Sự linh hoạt của Python là điểm đáng chú ý, khi lập trình viên không cần phải khai báo loại biến trước, mà để cho Python tự động xác định chúng trong quá trình chạy mã. Điều này giúp tăng tốc độ viết mã.

* Python là một ngôn ngữ cấp cao

Với đặc điểm gần gũi với ngôn ngữ con người, Python giảm bớt gánh nặng cho lập trình viên về các khía cạnh kỹ thuật như kiến trúc và quản lý bộ nhớ.

* Python là một ngôn ngữ lập trình hướng đối tượng

Python không chỉ là một ngôn ngữ lập trình hướng đối tượng mà còn hỗ trợ nhiều phong cách lập trình khác như lập trình hàm và lập trình cấu trúc, mở ra nhiều cơ hội cho các phong cách lập trình khác nhau.

### **1.2.5. Thư viện**

Thư viện trong Python là một tổ hợp các mã thường được sử dụng, giúp nhà phát triển tránh việc phải viết lại từ đầu mỗi khi tạo chương trình. Python đi kèm với Thư viện chuẩn, chứa nhiều hàm có thể tái sử dụng mặc định. Ngoài ra, còn có hơn 137.000 thư viện Python khác, phục vụ cho các ứng dụng đa dạng như phát triển web, khoa học dữ liệu và máy học (ML).

Thư viện Python sử dụng cho xử lý ngôn ngữ tự nhiên.

* **TextBlob**

TextBlob là một thư viện mã nguồn mở được sử dụng để xử lý dữ liệu văn bản. TextBlob cho phép bạn chọn thuật toán bạn muốn sử dụng thông qua giao diện lập trình ứng dụng (API) đơn giản của nó. API của TextBlob hỗ trợ các chức năng như gán nhãn từ loại, trích xuất cụm danh từ, phân loại, dịch thuật, phân tích quan điểm và nhiều hơn nữa.

Để phân tích quan điểm, thư viện TextBlob cung cấp hai cách triển khai:

* PatternAnalyzer: (Mặc định) Dựa trên thư viện mẫu.
* NaiveBayesAnalyzer: Công cụ NLTK giúp phân loại văn bản thành các loại khác nhau dựa trên nội dung.

Ưu và nhược điểm của việc sử dụng TextBlob cho NLP:

Ưu điểm:

* Dễ dùng cho người mới bắt đầu
* Cung cấp nền tảng cho NLTK
* Giao diện dễ sử dụng

Nhược điểm:

* Hiệu suất thấp kế thừa từ NLTK
* Không tốt cho việc sử dụng sản xuất quy mô lớn
* **Natural Language Toolkit (NLTK)**

NLTK, viết tắt của Bộ công cụ ngôn ngữ tự nhiên (Natural Language Toolkit), được coi là một trong những thư viện Python hàng đầu cho Xử lý Ngôn ngữ Tự nhiên (NLP). Nó cung cấp các công cụ cần thiết để thực hiện các tác vụ như phân loại, gắn thẻ, bắt nguồn, phân tích cú pháp và lý luận ngữ nghĩa. Đặc biệt, NLTK thường được lựa chọn bởi những người mới bắt đầu muốn khám phá lĩnh vực NLP và học máy.

Với tính linh hoạt cao, NLTK giúp tạo ra các hàm NLP phức tạp. Nó cung cấp một loạt các thuật toán để lựa chọn phù hợp với từng vấn đề cụ thể. NLTK cũng hỗ trợ nhiều ngôn ngữ khác nhau và cung cấp các thực thể được đặt tên cho nhiều ngôn ngữ khác nhau.

Do NLTK tập trung vào xử lý chuỗi, nó nhận đầu vào là chuỗi và trả về chuỗi hoặc danh sách chuỗi là kết quả.

Ưu và nhược điểm của việc sử dụng NLTK cho NLP:

Ưu điểm:

* Thư viện NLP nổi tiếng nhất
* Tiện ích mở rộng của bên thứ ba
* NLP giúp người dùng đặt câu hỏi về bất kỳ chủ đề nào và nhận được phản hồi trực tiếp trong vòng vài giây.
* NLP đưa ra câu trả lời chính xác cho câu hỏi có nghĩa là nó không cung cấp thông tin không cần thiết và không mong muốn.
* NLP giúp máy tính giao tiếp với con người bằng ngôn ngữ của họ.
* Hiệu quả về thời gian.
* Hầu hết các công ty sử dụng NLP để nâng cao hiệu quả của các quy trình tài liệu, độ chính xác của tài liệu và xác định thông tin từ cơ sở dữ liệu lớn.

Nhược điểm:

* NLP có thể không hiển thị ngữ cảnh.
* NLP là không thể đoán trước
* NLP có thể yêu cầu nhiều lần gõ phím hơn.
* NLP không thể thích ứng với miền mới và nó có một chức năng hạn chế, đó là lý do tại sao NLP chỉ được xây dựng cho một nhiệm vụ cụ thể và duy nhất.
* NLP đôi khi chậm
* NLP không có mô hình mạng Nơron
* NLP chỉ tách văn bản theo câu
  + - * + **SpaCy**

SpaCy là một thư viện mã nguồn mở trong lĩnh vực Xử lý Ngôn ngữ Tự nhiên (NLP) được thiết kế một cách rõ ràng để sử dụng trong môi trường sản xuất. Nó cho phép các nhà phát triển tạo ra các ứng dụng có khả năng xử lý và hiểu văn bản có khối lượng lớn. Thư viện này thường được sử dụng trong việc xây dựng các hệ thống hiểu ngôn ngữ tự nhiên và hệ thống khai thác thông tin.

Một trong những ưu điểm chính của SpaCy là khả năng hỗ trợ mã hóa cho hơn 49 ngôn ngữ thông qua việc đi kèm với các mô hình thống kê được đào tạo trước và vectơ từ. Các trường hợp sử dụng phổ biến của SpaCy bao gồm tự động hoàn tất tìm kiếm, tự động sửa lỗi, phân tích các đánh giá trực tuyến, trích xuất các chủ đề chính và nhiều ứng dụng khác.

Ưu và nhược điểm của việc sử dụng spaCy cho NLP:

Ưu điểm:

* Nhanh chóng
* Dễ sử dụng
* Tuyệt vời cho các nhà phát triển mới bắt đầu
* Dựa vào mạng lưới thần kinh cho các mô hình đào tạo

Nhược điểm:

* Không linh hoạt như các thư viện khác như NLTK
  + - * + **Gensim:**

Gensim, một bộ công cụ hàng dầu của Python trong lĩnh vực Xử lý Ngôn ngữ Tự nhiên (NLP), ban đầu được phát triển để tạo mô hình chủ đề. Tuy nhiên, thư viện này hiện đã mở rộng sử dụng của mình cho nhiều tác vụ NLP khác, như lập chỉ mục tài liệu. Gensim được xây dựng dựa trên các thuật toán có khả năng xử lý đầu vào lớn hơn bộ nhớ RAM.

Với các giao diện trực quan dễ dùng, Gensim hiệu quả trong triển khai đa luồng của các thuật toán như Phân tích Ngữ nghĩa Tiềm ẩn (LSA) và Phân bổ Dirichlet Tiềm ẩn (LDA). Một số ứng dụng phổ biến của thư viện bao gồm tìm kiếm sự tương đồng giữa các văn bản và biến đổi từ và tài liệu thành vectơ.

Ưu và nhược điểm của việc sử dụng Gensim cho NLP:

Ưu điểm:

* Giao diện trực quan
* Khả năng mở rộng
* Thực thi tốt một số thuật toán thông dụng như LDA và LSA

Nhược điểm:

* Được thiết kế cho mô hình văn bản không giám sát
* Thường cần sử dụng với các thư viện khác như NLTK

### **1.2.6. Python có lợi ích gì?**

**Cú pháp rõ ràng:** Python có cú pháp đơn giản, dễ đọc và dễ hiểu, giúp người lập trình viết mã một cách dễ dàng và nhanh chóng.

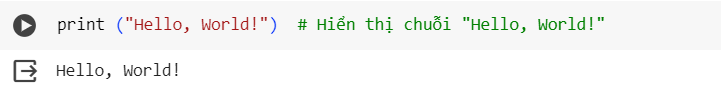
**Hỗ trợ đa mô hình lập trình:** Python hỗ trợ hướng đối tượng, lập trình hàm, và lập trình theo thủ tục. Điều này cho phép lập trình viên sử dụng phong cách lập trình phù hợp với yêu cầu của dự án.

**Cộng đồng lớn:** Python có một cộng đồng lập trình viên rộng lớn, với nhiều nguồn tài nguyên và hỗ trợ trực tuyến. Cộng đồng này cung cấp các thư viện, framework và giải pháp cho các vấn đề lập trình phổ biến.

**Năng suất cao:** Python giúp lập trình viên tăng năng suất nhờ vào cú pháp đơn giản, thư viện phong phú và khả năng tương tác cao. Nó cho phép bạn viết mã ngắn gọn và hiệu quả, giảm thời gian và công sức trong quá trình phát triển phần mềm.

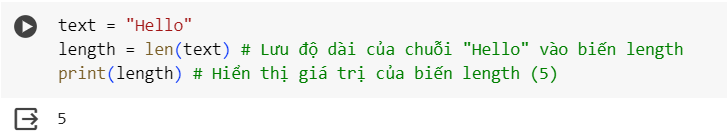
### **1.2.7. Các khái niệm và lệnh Python cơ bản**

* *Biến và kiểu dữ liệu:*
* Biến trong Python được sử dụng để lưu trữ giá trị. Bạn có thể gán một giá trị cho biến bằng cách sử dụng dấu "=".
* Kiểu dữ liệu trong Python bao gồm: số nguyên (int), số thực (float), chuỗi ký tự (str), danh sách (list), bộ (tuple), từ điển (dictionary), và nhiều kiểu dữ liệu khác.
* *Câu lệnh điều kiện và vòng lặp:*
* **Câu lệnh điều kiện if:** Kiểm tra một điều kiện và thực hiện một khối mã nếu điều kiện đúng.
* **Câu lệnh else:** Được sử dụng sau câu lệnh if và thực hiện một khối mã khác nếu điều kiện trong câu lệnh if không đúng.
* **Câu lệnh elif:** Được sử dụng sau câu lệnh if và kiểm tra một điều kiện khác nếu điều kiện trong câu lệnh if không đúng.
* **Vòng lặp for:** Thực hiện một khối mã lặp lại cho mỗi phần tử trong một dãy giá trị.
* **Vòng lặp while:** Thực hiện một khối mã lặp lại miễn là một điều kiện là đúng.
* *Hàm và phạm vi biến:*
* Hàm trong Python là một khối mã được đặt tên và có thể được gọi để thực hiện một tác vụ cụ thể. Hàm có thể nhận đối số và trả về giá trị.
* Phạm vi biến (scope) trong Python xác định phạm vi sử dụng của một biến. Có phạm vi toàn cục (global scope) và phạm vi cục bộ (local scope) cho biến.
* *Thư viện và module:*
* Thư viện trong Python là một bộ sưu tập các module và chức năng đã được viết sẵn để sử dụng.
* Module trong Python là một tập hợp các định nghĩa và câu lệnh Python được lưu trong một tệp tin có đuôi .py. Module cho phép tái sử dụng mã và tách chức năng thành các phần độc lập.
* *Lệnh nhập và xuất dữ liệu:*
* **Lệnh nhập dữ liệu từ người dùng:** Sử dụng hàm input () để nhận dữ liệu từ người dùng qua bàn phím.
* **Lệnh xuất dữ liệu:** Sử dụng hàm print () để hiển thị dữ liệu ra màn hình.
* *Các lệnh cơ bản:*
* **Lệnh print():** Dùng để hiển thị dữ liệu ra màn hình



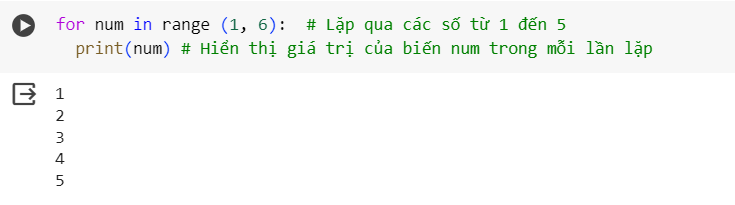
Hình 1. Chương trình chạy demo lệnh print()

* **Lệnh len():** Dùng để trả về độ dài của một chuỗi, danh sách hoặc bất kỳ đối tượng có thể đếm được nào khác.



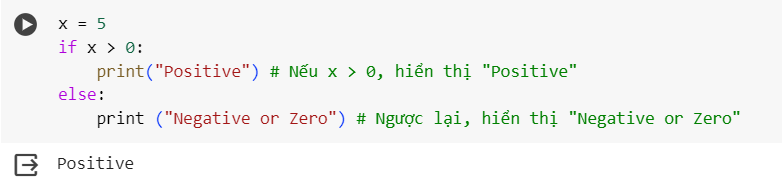
Hình 1. Chương trình chạy demo lệnh len()

* **Lệnh range():** Tạo ra một dãy số nguyên liên tiếp từ một giá trị bắt đầu đến một giá trị kết thúc với một bước nhảy cụ thể.



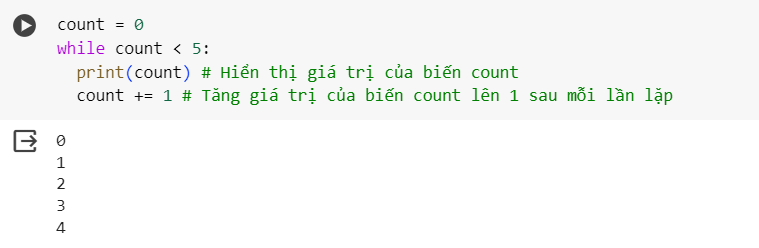
Hình 1. Chương trình chạy demo lệnh range()

* **Câu lệnh if - else:** Dùng để thực hiện một hành động hoặc một nhóm hành động dựa trên một điều kiện hoặc một loạt các điều kiện.



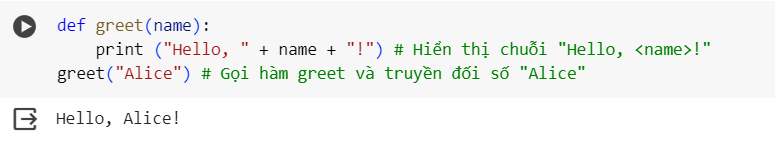
Hình 1. Chương trình chạy demo lệnh if - else()

* **Vòng lặp while:** Dùng để lặp lại một khối mã cho đến khi một điều kiện không còn đúng nữa. Mã trong vòng lặp sẽ được thực thi cho đến khi điều kiện không đúng.



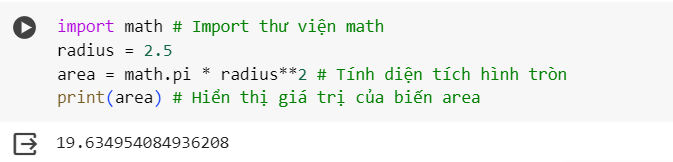
Hình 1. Chương trình chạy demo vòng lặp while()

* **Hàm:**Dùng để đóng gói một khối mã thành một đơn vị độc lập có thể được gọi và sử dụng nhiều lần. Hàm có thể nhận đối số (tham số) và thực hiện một số hành động cụ thể.



Hình 1. Chương trình chạy demo sử dụng hàm

* **Sử dụng thư viện và module:** Thư viện và module cung cấp các tập hợp các hàm, lớp và biến đã được viết sẵn để thực hiện các tác vụ cụ thể. Bằng cách import một thư viện hoặc module, bạn có thể sử dụng các chức năng và tài nguyên có sẵn trong đó.



Hình 1. Chương trình chạy demo sử dụng thư viện và module

* **Lệnh nhập và xuất dữ liệu:** Dùng để tương tác với người dùng hoặc đọc/ghi dữ liệu từ/đến các nguồn khác nhau. Lệnh input () được sử dụng để nhận dữ liệu từ người dùng thông qua bàn phím, trong khi lệnh print() được sử dụng để hiển thị dữ liệu ra màn hình hoặc ghi vào tệp tin.



Hình 1. Chương trình chạy demo lệnh nhập và xuất dữ liệu

# **CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG TÁCH TỪ VÀ GÁN NHÃN TỪ LOẠI**

## 2.1. KHÁI QUÁT VỀ TỪ VỰNG TIẾNG VIỆT

Tiếng Việt là ngôn ngữ đơn lập, không biến đổi hình thái, bao gồm các từ để diễn đạt ý nghĩa khác nhau trong cuộc sống hàng ngày, trong học tập, trong công việc và nhiều lĩnh vực khác.

• Từ điển tiếng Việt (Vietlex) có số lượng trên 40,181 từ.

Trong đó:

• 81.55% âm tiết là từ: từ đơn

• 15.69% các từ trong từ điển là từ đơn

• 70.72% từ ghép có 2 âm tiết

• 5.62% từ ghép có 3 âm tiết

• 6.93% từ ghép có 4 âm tiết

• 1.04% từ ghép có 5 âm tiết

|  |  |  |
| --- | --- | --- |
| Độ dài | Số lượng từ | % |
| 1 | 6,303 | 15.69 |
| 2 | 228,416 | 70.72 |
| 3 | 32,259 | 5.62 |
| 4 | 42,784 | 6.93 |
| 5 | 5419 | 1.04 |
| Tổng | 40,181 | 100 |

Bảng 2. Độ dài của từ tính theo âm tiết

• Từ đơn: dùng một âm tiết làm một từ.

Ví dụ: người, cây, hoa, ...

• Từ ghép: được cấu tạo bởi các âm tiết với nhau, giữa các âm tiết được ghép có quan hệ về ngữ nghĩa. Từ ghép được chia làm hai loại:

- Từ ghép đẳng lập là từ ghép cấu tạo bởi các âm tiết có quan hệ giống nhau về mặt ngữ nghĩa.

Ví dụ: ăn uống, bếp núc, ...

- Từ ghép chính phụ là từ ghép có cấu tạo được chia thành âm tiết chính và âm tiết phụ. Thành tố phụ có vai trò phân loại, bổ sung nghĩa cho thành tố chính.

Ví dụ: tàu hoả, hoa hồng, xấu bụng, bạn thân, ...

• Từ láy: là từ được cấu tạo bởi các thành phần ngữ âm được lặp lại về âm, về vần hoặc cả âm và vần. Từ láy được chia làm hai loại:

- Từ láy toàn bộ là từ có âm tiết được lặp lại hai lần.

Ví dụ: ào ào, xanh xanh, …

- Từ láy bộ phận là từ có hai âm tiết giống nhau về âm hoặc vần.

Ví dụ: mênh mông, lấp lánh, …

• Biến thể của từ: là dạng lâm thời biến động hoặc dạng "lời nói" của từ, gồm nhiều dạng biến thể: dạng số nhiều, dạng từ loại khác, dạng biến đổi của động từ, dạng biến đổi của tính từ, dạng đồng nghĩa và trái nghĩa, …

Ví dụ: dạng biến đổi của tính từ: đẹp → đẹp hơn → đẹp nhất

• Rút gọn một từ dài thành từ ngắn hơn

Ví dụ: thông tin liên lạc → liên lạc

• Lâm thời phá vỡ cấu trúc của từ, phân bố lại yếu tố tạo từ với những yếu tố khác ngoài từ chen vào.

Ví dụ:

phương pháp giáo dục hiện đại → hiện đại phương pháp giáo dục

ngặt nghẽo → cười ngặt cười nghẽo

danh lợi + ham chuộng → ham danh chuộng lợi

▪ Các diễn tả gồm nhiều từ (vd, “bởi vì”) cũng được coi là 1 từ

▪ Tên riêng: tên người và vị trí được coi là 1 đơn vị từ vựng

▪ Các mẫu thường xuyên: số, thời gian, …

## 2.2. BÀI TOÁN TÁCH TỪ VÀ GÁN NHÃN TỪ LOẠI

* **Tách từ (Tokenization):**

Tách từ là quá trình chia một đoạn văn bản thành các đơn vị nhỏ hơn, gọi là "tokens." Các tokens có thể là từ, chấm câu, hoặc cụm từ, tùy thuộc vào mức độ tách từ được áp dụng. Quá trình này giúp biểu diễn văn bản dưới dạng các đơn vị xử lý, là cơ sở cho nhiều công việc trong xử lý ngôn ngữ tự nhiên.

Ví dụ: Đoạn văn bản "con mèo nằm ở hiên nhà" sau khi được tách từ sẽ trở thành các tokens như: ["con", "mèo", "nằm", "ở", "hiên", "nhà"]

Nhưng đây chưa phải là phương pháp tối ưu nhất vì tiếng Việt hầu như là các cụm từ ghép. Ví dụ: thay vì tách ra là ["con", "mèo"] thì ta phải tách thành cụm [“con mèo”]

* Gán nhãn từ loại (Part-of-Speech Tagging - POS):

Gán nhãn từ loại là quá trình đánh dấu từng từ trong một đoạn văn bản với loại từ tương ứng, như danh từ, động từ, tính từ, giúp hiểu cấu trúc ngữ pháp và ngữ nghĩa của câu.

Input : Một đoạn văn bản đã tách từ + tập nhãn  
 Output : Cách gán nhãn cuối cùng và chính xác nhất

Ví dụ: Cho các tokens như: ["con", "mèo", "nằm", "ở", "hiên", "nhà"]. Sau quá trình gán nhãn từ loại, chúng ta có thể có các dạng nhãn như: ["Danh từ", " Danh từ ", "Động từ", " Danh từ ", "Danh từ"] tương ứng với ["con", "mèo", "nằm", "ở", "hiên", "nhà"].

Những quá trình này đóng vai trò quan trọng trong việc hiểu và xử lý ngôn ngữ tự nhiên, cung cấp cơ sở dữ liệu cho nhiều ứng dụng như dịch thuật, tìm kiếm thông tin, phân tích ý kiến, và nhiều ứng dụng khác nằm trong lĩnh vực xử lý ngôn ngữ tự nhiên.

## 2.3. TỔNG QUAN VỀ CÁCH TIẾP CẬN GIẢI BÀI TOÁN

Quá trình này gồm 3 bước xử lý:

* Bước 1: Đây là giai đoạn tiền xử lý, phân tách câu văn, xâu ký tự thành chuỗi các từ hoặc các cụm từ. Quá trình này đôi khi rất đơn giản nhưng cũng có thể trở nên phức tạp tuỳ theo loại ngôn ngữ và quan niệm về đơn vị từ vựng. Ví dụ như ngôn ngữ tiếng Anh, việc phân tách từ phần lớn dựa vào các ký hiệu trắng, tuy nhiên vẫn còn gây tranh cãi do xuất hiện những từ ghép và cụm từ. Tiếng Việt cũng không ngoại lệ, thậm chí dấu trắng còn không được coi là dấu hiệu để xác định ranh giới giữa các đơn vị từ vựng do tần suất xuất hiện từ ghép cao.
* Bước 2: Khởi tạo nhãn dán, tức là tìm cho mỗi từ hoặc cụm từ đã được tách ra tất cả các nhãn từ loại nó có thể có. Tập nhãn từ loại có thể thu từ cơ sở dữ liệu từ điển, thư viện. Đối với một số từ mới chưa xuất hiện trong cơ sở dữ liệu thì có thể dùng một nhãn từ loại ngầm định hoặc tất cả. Tiếng Việt là ngôn ngữ không biến đổi hình thái, nên chúng ta không cần dựa vào hình thái từ để đoán nhận lớp từ loại tương ứng của từ đang xét.
* Bước 3: Kết quả gán cuối. Đây là giai đoạn lựa chọn cho mỗi từ một nhãn phù hợp nhất với ngữ cảnh trong tập nhãn khởi tạo, loại bỏ đi sự nhập nhằng ở giai đoạn khởi tạo nhãn dán.

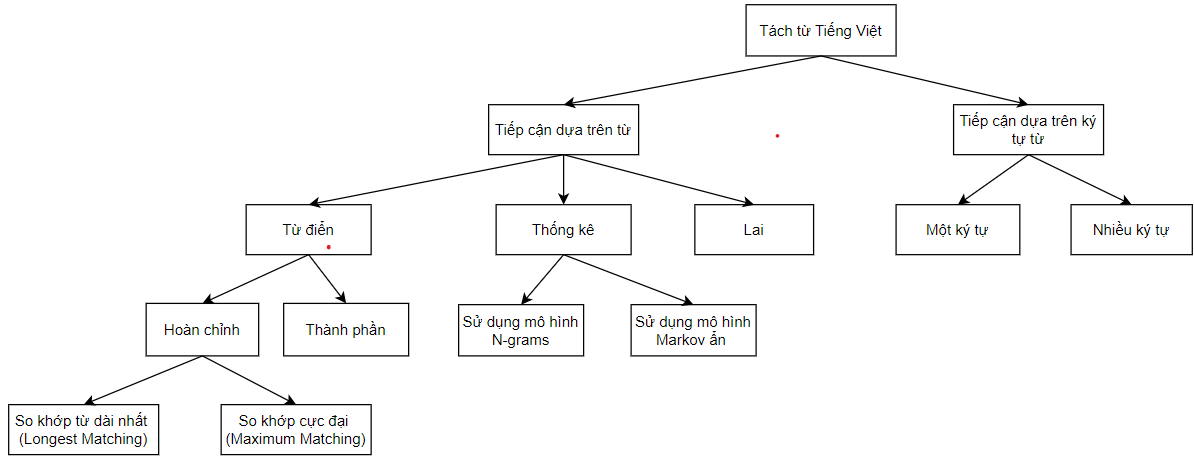
## 2.4. HỆ THỐNG TÁCH TỪ

Tiếng Việt gồm có từ đơn (gồm một âm tiết) và từ ghép (đa âm tiết), điều này làm cho việc sử dụng khoảng trắng để phân biệt ranh giới giữa các từ trở nên không khả thi. Các âm tiết thường được kết hợp lại với nhau để tạo thành các từ khác nhau, phụ thuộc vào ngữ cảnh của văn bản.

Để nhận dạng chính xác ranh giới của các từ (tách từ), phục vụ cho các nhiệm vụ phân tích dữ liệu văn bản như gom nhóm, phân loại văn bản, các nhà nghiên cứu đã đề xuất nhiều phương pháp tách từ. Các phương pháp này bao gồm phương pháp so khớp cực đại, mô hình MarKov ẩn, chuyển dịch trạng thái hữu hạn có trọng số, so khớp từ dài nhất, và nhiều phương pháp khác.

Trong bài luận này, chúng em sẽ sử dụng một mô hình tách từ trong ngôn ngữ tự nhiên dựa trên phương pháp so khớp cực đại.

### **2.4.1 Hướng tiếp cận bài toán tách từ**



Hình 2. Sơ đồ hướng tiếp cận bài toán tách từ

Có 2 hướng tiếp cận : tiếp cận dựa trên từ và tiếp cận dựa trên ký tự từ. Hướng tiếp cận dựa trên từ với mục tiêu là chia văn bản thành các đơn vị nhỏ hơn, thường là các từ hoặc các đơn vị ngữ cảnh khác, để dễ dàng xử lý và hiểu văn bản một cách hiệu quả. Hướng tiếp cận này được chia thành 3 nhóm: dựa trên từ điển (dictionary-based), dựa trên thống kê (statistics-based) và kết hợp nhiều phương pháp (hydrid-based).

* Hướng tiếp cận dựa trên từ điển: Ý tưởng của hướng tiếp cận này là phải so sánh các cụm từ được tách ra từ văn bản với các từ trong từ điển. Từ điển có thể được chia thành hai loại: từ điển hoàn chỉnh và từ điển thành phần. Trong từ điển hoàn chỉnh, có hai phương pháp chính là so khớp từ dài nhất (Longest Matching) và so khớp cực đại (Maximum Matching). Mặc dù đơn giản và dễ hiểu, nhưng hướng tiếp cận này chưa đạt hiệu quả cao vì chưa xử lý được nhiều trường hợp nhập nhằng và khả năng phát hiện từ mới trong văn bản chưa cao. Hiện nay, so khớp cực đại được coi là phương pháp quan trọng và hiệu quả nhất trong hướng tiếp cận từ điển.
* Hướng tiếp cận dựa vào thống kê: Phương pháp này dựa vào thông tin thống kê như tần số của từ hoặc ký tự, hoặc xác suất xuất hiện cùng nhau trong một tập dữ liệu. Do đó, hiệu quả của các giải pháp thống kê phụ thuộc chủ yếu vào dữ liệu huấn luyện cụ thể được sử dụng.
* Hướng tiếp cận kết hợp nhiều phương pháp: Hướng tiếp cận này kết hợp các phương pháp tiếp cận khác nhau để tận dụng các ưu điểm của từng kỹ thuật và hướng tiếp cận khác nhau nhằm nâng cao hiệu quả. Tuy nhiên, việc kết hợp này có thể mất nhiều thời gian xử lý, tốn không gian đĩa và có chi phí cao.
* Hướng tiếp cận dựa trên ký tự: Phương pháp này rút trích một số lượng nhất định các tiếng trong văn bản, ví dụ như rút trích một ký tự hoặc nhiều ký tự. Mặc dù đơn giản, nhưng có thể mang lại kết quả quan trọng.

### **2.4.2 Các phương pháp sử dụng trong bài toán tách từ**

#### 2.4.2.1. Phương pháp so khớp từ dài nhất (Longest Matching)

* Phương pháp so khớp từ dài nhất hoạt động dựa trên tư tưởng tham lam. Với mỗi câu, thuật toán duyệt qua từng âm tiết từ trái sang phải, kiểm tra xem có nhóm các âm tiết nào tồn tại trong từ điển hay không. Chuỗi âm tiết dài nhất mà tìm thấy trong từ điển sẽ được xác định là một từ. Quá trình này được lặp lại cho tới khi câu được xử lý hoàn toàn. Tuy nhiên, phương pháp này chỉ hoạt động đúng khi không có sự nhập nhằng, tức là không có trường hợp một phần của từ sau có thể kết hợp với từ trước để tạo thành một từ khác trong từ điển.
* Độ phức tạp tính toán: O (n.V)

Trong đó : n – số âm tiết trong chuỗi

V – số từ trong từ điển

* Giải thuật:

***Input:*** *Chuỗi ký tự;*

***Output:*** *Chuỗi từ, cụm từ (từ có chiều dài dài nhất);*

*V là danh sách các tiếng chưa xét;*

*T là bộ từ điển.*

***While*** *V ≠ ∅ do*

**Begin**

*Wmax= từ đầu tiên trong danh sách V;*

*Foreach (v thuộc từ gồm các tiếng bắt đầu trong V)*

*If (length(v) > length(Wmax) and (v thuộc T))*

*Then Wmax=v;*

*Loại bỏ đi các từ Wmax ở đầu danh sách V;*

**End**.

* Ví dụ: “Tôi là sinh viên trường Đại học Kinh tế Kỹ thuật Công nghiệp”

|  |  |  |
| --- | --- | --- |
| Bước | Từ dài nhất có thể | Các từ còn lại |
| 1 | Tôi | là sinh viên trường Đại học Kinh tế Kỹ thuật Công nghiệp |
| 2 | là | sinh viên trường Đại học Kinh tế Kỹ thuật Công nghiệp |
| 3 | sinh viên | trường Đại học Kinh tế Kỹ thuật Công nghiệp |
| 4 | trường | Đại học Kinh tế Kỹ thuật Công nghiệp |
| 5 | Đại học | Kinh tế Kỹ thuật Công nghiệp |
| 6 | Kinh tế | Kỹ thuật Công nghiệp |
| 7 | Kỹ thuật | Công nghiệp |
| 8 | Công nghiệp |  |

Bảng 2. Kết quả ví dụ giải thuật phương pháp so khớp từ dài nhất

* Đánh giá phương pháp:
* Ưu điểm: Cài đặt đơn giản; độ phức tạp tính toán hợp lý; không yêu cầu dữ liệu huấn luyện.
* Nhược điểm: Phụ thuộc vào từ điển; chưa giải quyết được vấn đề nhập nhằng.

#### 2.4.2.2 Phương pháp so khớp cực đại (Maximum Matching)

* Ý tưởng chính của phương pháp này là duyệt một câu từ trái qua phải và chọn từ có nhiều tiếng nhất có mặt trong từ điển tiếng Việt, rồi cứ thế tiếp tục cho từ kế tiếp cho đến hết câu.
* Phương pháp so khớp cực đại dạng đơn giản : giải quyết nhập nhằng từ đơn.
* Giả sử ta có chuỗi ký tự S = { C1, C2, C3,..., Cn } với C1, C2, C3,..., Cn là các tiếng được tách bởi khoảng trắng trong câu.
* Duyệt từ đầu chuỗi, xác định đâu là từ. Kiểm tra xem C1 có phải là từ có trong từ điển hay không, sau đó kiểm tra C1C2 có trong từ điển hay không.
* Tiếp tục như vậy C1C2C3, C1C2C3C4,..., C1C2C3...Cn, với n là số tiếng lớn nhất của một từ có thể có nghĩa (nghĩa là có trong từ điển tiếng Việt).
* Sau đó, chúng ta chọn từ có nhiều tiếng nhất có mặt trong từ điển và đánh dấu từ đó. Tiếp tục quá trình trên với tất cả các từ còn lại trong câu và trong toàn bộ văn bản.
* Phương pháp so khớp cực đại dạng phức tạp : là phân đoạn từ. Phương pháp này về cơ bản cũng giống như so khơp cực đại dạng đơn giản. Tuy nhiên, dạng này có thể tránh được một số nhập nhằng gặp phải trong dạng đơn giản. Độ chính xác cao lên đến 99.69% và 93.21 nhập nhằng được giải quyết.
* Ví dụ chuỗi ký tự C1, C2, C3, C4,..., Cn. với C1, C2, C3, C4,..., Cn là các tiếng được tách bởi khoảng trắng trong câu.
* Giả sử C1 là từ, C1C2 cũng là một từ. Khi đó ta kiểm tra các kí tự trong chuỗi C1, C2, C3, C4,..., Cn để tìm tất cả các đoạn 3 từ bắt đầu với C1 hoặc C1C2.
* Giả sử ta được:

C1C2 C3C4

C1C2 C3C4 C5

C1C2 C3C4 C5C6

Chuỗi dài nhất là chuỗi thứ ba. Do đó từ đầu tiên của chuỗi thứ ba (C1C2) sẽ được chọn. Sau đó tiếp tục thực hiện lại các bước cho đến khi thu được chuỗi từ hoàn chỉnh.

* Cài đặt giải thuật bằng ngôn ngữ Python:

*def tokenizer(text, dict, is\_show=False):*

*print("ïnput:", text)*

*print()*

*input=text.split(" ")*

*words= []*

*s=0*

*while True:*

*e=len(input)*

*while e>s:*

*temp\_word= input[s:e]*

*is\_word=""*

*for item in temp\_word:*

*is\_word+=item+" "*

*is\_word=is\_word[:-1]*

*e-=1*

*if is\_word.lower() in dict:*

*words.append(is\_word)*

*break*

*if e==s:*

*words.append(is\_word)*

*break*

*if e>= len(input):*

*break*

*#hien thi qua trinh tach tu*

*if is\_show:*

*print("s= ",s)*

*print("e= ",e)*

*print(words[len(words-1)])*

*pritn("-"\*100)*

*s=e+1*

*output=""*

*for item in words:*

*output+= item.replace(" ","\_")*

*output+=" "*

*output=output[:-1]*

*return output*

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

*ex1="thời khóa biểu đang được cập nhật"*

*ex2="môn học xử lý ngôn ngữ tự nhiên"*

*ex3="con ngựa đá con ngựa đá"*

*ex4="học sinh học sinh học"*

*ex5="tôi xin chào bạn"*

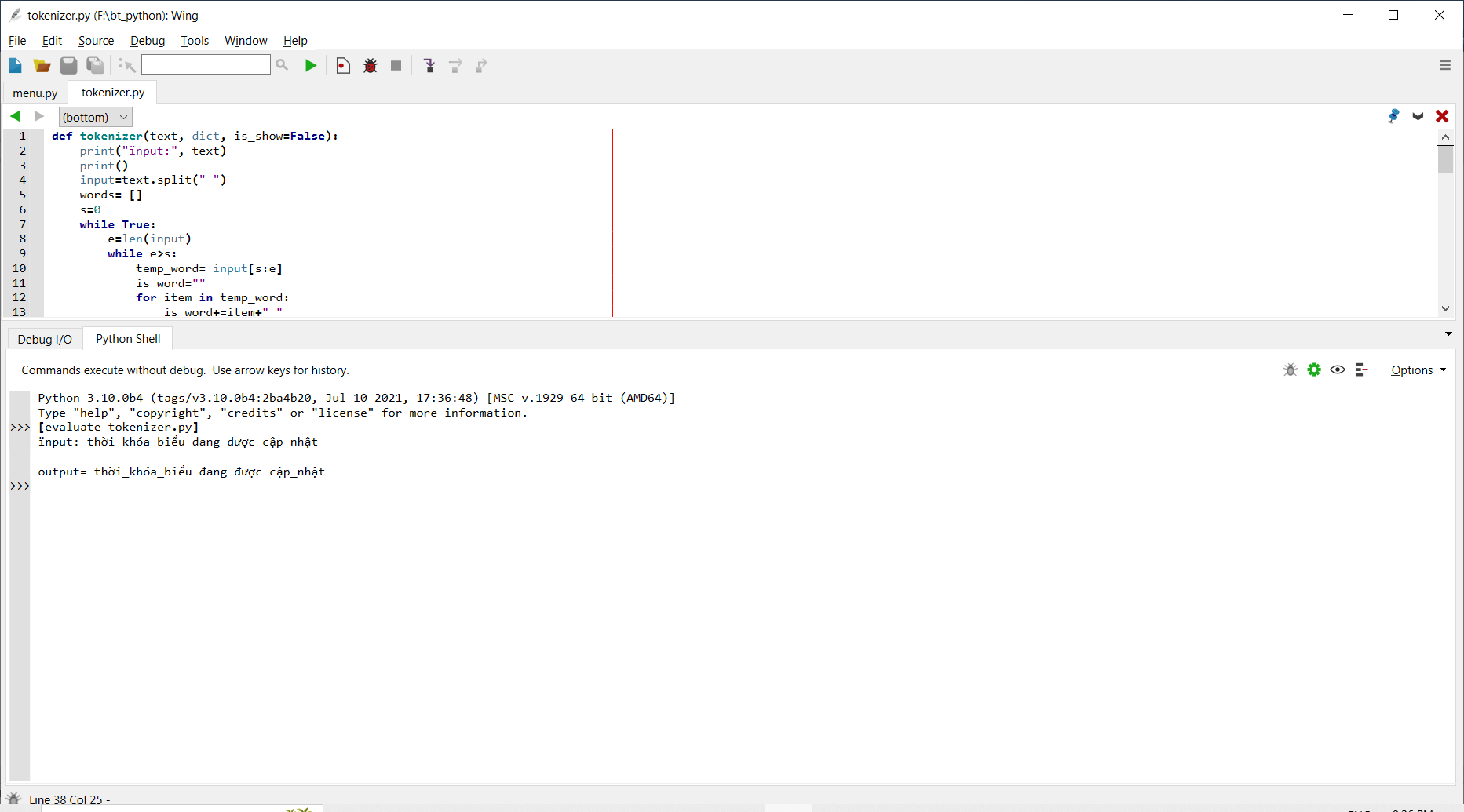
*#từ điển*

*dict= {"thời khóa biểu":0, "đang":1,"được":2,"cập nhật":3, "môn học":4, "môn":5, "học":6, "xử lý":7, "ngôn ngữ":8, "tự nhiên":9, "con":10, "con ngựa":11, "ngựa":12, "đá":13,"học":14, "học sinh":15,"sinh học":16, "dân tộc":17, "viện trưởng":18,"giáo viên":19, "đạo diễn":20, "xứ sở":21, "nguồn lực":22, "thủ đô":23,"số lượng":24, "thuần nhất":25,"môi giới":26, "đơn giản":27,"tiến bộ":28, "chính sách":29, "thường xuyên":30, "tình yêu":31,"tôi":32,"xin chào":33, "bạn":34}*

*test1=tokenizer(ex1,dict)*

*print("output=",test1)*

* Kết quả chạy chương trình:



* Đánh giá phương pháp:
* Ưu điểm: giải thuật đơn giản, nhanh chóng, chỉ cần dựa vào từ điển để thực hiện.
* Nhược điểm: Độ chính xác của giải thuật so khớp từ dài nhất này phụ thuộc hoàn toàn vào tính đầy đủ và tính chính xác của bộ từ điển, yêu cầu bộ từ điển lớn, khó xử lý các từ viết tắt, tên riêng, số, ngày giờ hoặc từ rút gọn.

#### 2.4.2.3 Phương pháp tách từ sử dụng mô hình N-gram

* Mô hình ngôn ngữ N-gram tách từ có hiệu quả khi thể hiện mối quan hệ ngữ cảnh của từ. Trong mô hình đó, mỗi từ sẽ phụ thuộc xác suất vào n-1 từ trước nó.
* Mô hình N-gram dùng xác định xác suất xuất hiện của các từ hoặc các cụm từ liên tiếp trong một văn bản, từ đó giúp tách văn bản thành các từ riêng lẻ hoặc các cụm từ, trong đó với mỗi câu thì phân đoạn tốt nhất theo mô hình này là phân đoạn có xác suất P(W) được tính theo công thức dưới đây là lớn nhất:

Xác suất về sự phụ thuộc của một từ vào n từ trước đó được thống kê dựa trên một corpus đủ lớn, tùy vào giả thiết về tính phụ thuộc mà ta có các mô hình bigram hoặc trigram tương ứng.

Phương pháp này là một trong những phương pháp thống kê chủ yếu để giải quyết bài toán phân tách từ khi không có thông tin từ điển và dữ liệu gán nhãn.

* Cách thực hiện:
* Chuẩn bị dữ liệu đầu vào:

- Dữ liệu đầu vào thường là văn bản đã được tiền xử lý, bao gồm việc loại bỏ các ký tự không mong muốn và các dấu câu nếu cần.

* Xây dựng mô hình N-grams:

- Chia văn bản thành các chuỗi từ có độ dài N (ví dụ: bigrams, trigrams).

Đếm tần suất xuất hiện của mỗi chuỗi N-gram trong dữ liệu.

* Xác định vị trí của khoảng trống:

- Dựa trên tần suất xuất hiện của các N-grams, tính xác suất cho việc một khoảng trống xuất hiện ở một vị trí cụ thể giữa các từ.

- Xác định vị trí của các khoảng trống dựa trên xác suất cao nhất hoặc một ngưỡng xác suất.

* Tách từ dựa trên vị trí khoảng trống:

- Sử dụng vị trí của khoảng trống để tách từ văn bản thành các từ riêng lẻ.

* Kiểm tra và điều chỉnh:

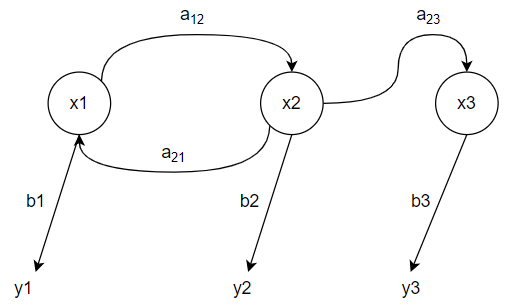
- Kiểm tra kết quả của mô hình trên dữ liệu kiểm tra để đảm bảo độ chính xác.

- Điều chỉnh các tham số của mô hình N-grams nếu cần thiết để cải thiện hiệu suất trên dữ liệu thực tế.

* Đánh giá phương pháp:
* Ưu điểm: Dễ triển khai và hiệu quả; hiệu suất tốt với văn bản có cấu trúc đơn giản; độc lập ngữ cảnh; chi phí tính toán thấp.
* Nhược điểm: Khó xử lý ngôn ngữ phức tạp; phụ thuộc vào kích thước N; khó khăn trong việc xử lý các từ hiếm gặp.

2.4.2.4 Phương pháp tách từ sử dụng mô hình Markov ẩn

* Mô hình Markov ẩn (Hidden Markov Model - HMM) thuộc loại mô hình thống kê được sử dụng để mô hình hóa hệ thống, trong đó quá trình được biểu diễn theo một quá trình Markov với các tham số không biết trước. Nhiệm vụ chính là dự đoán ranh giới của các từ trong văn bản, dựa trên mô hình Markov ẩn và các thông tin ngữ cảnh.
* Trong một mô hình Markov điển hình, các trạng thái được quan sát trực tiếp bởi quan sát viên, do đó các xác suất chuyển tiếp giữa các trạng thái là các tham số duy nhất trong mô hình này.



Hình 2.2 Mô hình Markov ẩn

Giải thích các tham số:

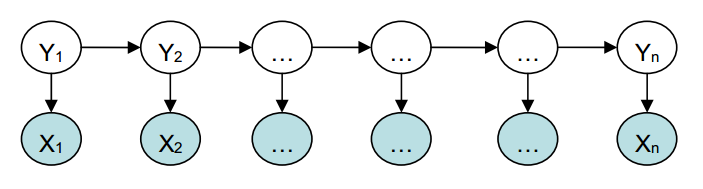
xi: các trạng thái

aij: Xác xuất chuyển tiếp

bij: xác xuất đầu ra

yi: dữ liệu quan sát

* Mô hình Markov ẩn sẽ bổ sung vào các đầu ra, mỗi trạng thái có xác suất phân bố trên các thể hiện đầu ra. Ta có thể tìm được chuỗi các trạng thái mô tả tốt nhất cho mỗi dữ liệu quan sát được bằng cách tính xác xuất điều kiện P(Y|X) theo công thức sau:



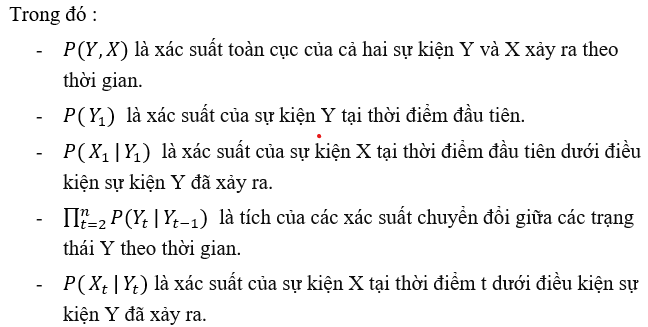
Hình 2.3 Biểu đồ chuyển tiếp trạng thái của mô hình Markov ẩn

Trong đó:

- Yn: biểu diễn trạng thái thời điểm thứ t=n trong chuỗi trạng thái Y.

- Xn: biểu diễn dữ liệu quan sát được tại thời điểm t= n trong chuỗi X.

* Trạng thái hiện tại sẽ phụ thuộc vào trạng thái ngay trước đó với giả thiết là dữ liệu quan sát tại thời điểm t chỉ phụ thuộc vào trạng thái t. Ta có thể tính P(Y, X) theo công thức:



* Cách thực hiện:
* **Chuẩn bị dữ liệu:**
* Chuẩn bị dữ liệu văn bản đã tách từ (hoặc không tách từ) để làm dữ liệu huấn luyện
* **Xây dựng mô hình HMM:**
* Xác định số lượng trạng thái ẩn trong mô hình. Mỗi trạng thái có thể đại diện cho một loại từ (ví dụ: từ đơn, từ ghép).
* Ước lượng các ma trận chuyển đổi giữa các trạng thái, xác suất quan sát từng trạng thái.
* **Chuẩn bị dữ liệu huấn luyện:**
* Chia dữ liệu văn bản thành các đoạn ngắn, mỗi đoạn chứa một loạt từ.
* **Huấn luyện mô hình HMM:**
* Sử dụng thuật toán Expectation-Maximization (EM) để ước lượng tham số của mô hình HMM dựa trên dữ liệu huấn luyện.
* **Tách từ với mô hình huấn luyện:**
* Áp dụng mô hình HMM đã huấn luyện để tách từ cho văn bản mới.
* **Điều chỉnh và kiểm tra:**
* Kiểm tra độ chính xác của mô hình trên dữ liệu kiểm tra và điều chỉnh tham số nếu cần thiết.
* Đánh giá phương pháp:
* Ưu điểm : Có khả năng xử lý được các ngôn ngữ tự nhiên phức tạp; có thể tích hợp được các thông tin từ ngữ cảnh trước đó; thích hợp cho việc xử lý dữ liệu số và chuỗi thời gian.
* Nhược điểm: Khả năng mô hình hóa bị hạn chế, đòi hỏi dữ liệu lớn, **khó xử lý các từ hiếm gặp và từ không xuất hiện trong dữ liệu huấn luyện, khó xử lý các ngôn ngữ có cấu trúc phức tạp**.

## 2.5 HỆ THỐNG GÁN NHÃN TỪ LOẠI

### **2.5.1 Các nhãn từ loại**

|  |  |  |  |
| --- | --- | --- | --- |
| **Stt** | **Mã từ loại** | **Miêu tả** | **Ví dụ** |
| 1 | N | Danh từ (noun) | Mèo, chó, bút,... |
| 2 | Np | Danh từ riêng (proper noun) | Hạ Long, Thái Bình,.. |
| 3 | V | Động từ (Verb) | Muốn, đi, chơi, ăn, uống,…. |
| 4 | E | Giới từ (Preposition) | Của, để, từ, đến... |
| 5 | P | Đại từ (Pronoun) | tôi, tao, mày, nó, ấy, bao nhiêu, ai, kia, gì, nào, vậy, thế, sao,... |
| 6 | C | Liên từ (Conjunction) | và, hoặc, nhưng, nếu, thì, vì, nên ... |
| 7 | A | Tính từ (Adjective) | Nhanh chóng, xanh, đỏ, buồn,… |
| 8 | CH | Dấu câu (Punctuation) | . , : - @ # $ % & \* ! < > ( ) { } [ ] ... |

Bảng 2. Danh sách nhãn từ loại

### **2.5.2 Mô hình markov ẩn (hidden markov model – hmm) kết hợp thuật toán Viterbi**

**a, Mô hình markov ẩn**

Các mô hình thống kê được gọi là Mô hình Markov ẩn (HMM) dùng để mô tả các hệ thống có trạng thái thay đổi không thể quan sát được theo thời gian. Nó dựa trên ý tưởng rằng có một quy trình cơ bản với các trạng thái ẩn, mỗi trạng thái đều có một kết quả đã biết. Xác suất chuyển đổi giữa các trạng thái ẩn và phát ra các ký hiệu có thể quan sát được xác định bởi mô hình.

Mô hình Markov ẩn (HMM) là mô hình thống kê mô tả mối quan hệ xác suất giữa một chuỗi các trạng thái quan sát được và một chuỗi các trạng thái ẩn. Nó thường được sử dụng trong các tình huống mà hệ thống hoặc quy trình cơ bản tạo ra các quan sát không xác định hoặc ẩn, do đó có tên là "Mô hình Markov ẩn".

Mô hình Markov ẩn ứng dụng trong rất nhiều lĩnh vực như nhận dạng tín hiệu, nhận dạng giọng nói, … Hơn nữa nó còn ứng dụngrất thành công trong lĩnh vực xử lý ngôn ngữ tự nhiên (Natural Languages Processing) như: gán nhãn từ loại (part-of-speech tagging), phân lớp (phrase chunking), trích rút thông tin (extracting target information from document).

* Định nghĩa của mô hình Markov ẩn là:



* S là tập hợp tất cả các trạng thái:



* V là tập hợp tất cả các quan sát được:



* Q là chuỗi các trạng thái có thể xảy ra, có chiều dài T



* Tương ứng với nó là chuỗi các quan sát có thể quan sát được



* A là bảng chuyển đổi, chứa những giá trị xác suất chuyển đổi từ trạng thái i sang trạng thái j, và những xác suất chuyển đổi này độc lập với thời gian:



* B là bảng xác suất quan sát, chứa những giá trị xác suất của quan sát k từ trạng thái i, độc lập với thời gian:



* Π là bảng xác suất đầu tiên:



❖ HMM bao gồm hai loại biến: trạng thái ẩn và quan sát.

- Trạng thái ẩn là các biến cơ bản tạo ra dữ liệu quan sát nhưng không thể quan sát trực tiếp được.

- Trạng thái quan sát là các biến đo lường và quan sát.

- Lập mô hình mối quan hệ giữa các trạng thái ẩn và các quan sát bằng cách sử dụng phân bố xác suất. Mô hình Markov ẩn (HMM) là mối quan hệ giữa các trạng thái ẩn và các quan sát sử dụng hai bộ xác suất: xác suất chuyển tiếp và xác suất phát xạ.

- Xác suất chuyển tiếp mô tả xác suất chuyển từ trạng thái ẩn này sang trạng thái ẩn khác.

- Xác suất phát xạ mô tả xác suất quan sát đầu ra ở trạng thái ẩn.

❖ Thuật toán mô hình Markov ẩn có thể được thực hiện qua các bước sau:

- Bước 1: Xác định không gian trạng thái và không gian quan sát

Không gian trạng thái là tập hợp tất cả các trạng thái ẩn có thể có và không gian quan sát là tập hợp tất cả các quan sát có thể có.

- Bước 2: Xác định phân bố trạng thái ban đầu

Đây là phân bố xác suất của trạng thái ban đầu.

- Bước 3: Xác định xác suất chuyển trạng thái

Chính là xác suất chuyển từ trạng thái này sang trạng thái khác. Nó sẽ tạo thành ma trận chuyển tiếp, mô tả xác suất chuyển từ trạng thái này sang trạng thái khác.

- Bước 4: Xác định khả năng quan sát:

Đây là xác suất tạo ra mỗi quan sát từ mỗi trạng thái. Điều này tạo thành ma trận phát xạ, mô tả xác suất tạo ra mỗi quan sát từ mỗi trạng thái.

- Bước 5: Huấn luyện mô hình

Xác suất chuyển trạng thái và các tham số có thể quan sát được ước tính bằng thuật toán Baum-Welch hoặc thuật toán tiến-lùi. Điều này được thực hiện bằng cách cập nhật lặp đi lặp lại nhiều lần các tham số cho đến khi hội tụ.

- Bước 6: Giải mã chuỗi trạng thái ẩn có khả năng xảy ra nhất

Thuật toán Viterbi được sử dụng để tính toán chuỗi trạng thái ẩn có khả năng xảy ra nhất dựa trên dữ liệu được quan sát. Điều này có thể được sử dụng để dự đoán các quan sát trong tương lai, phân loại trình tự hoặc phát hiện các mẫu trong dữ liệu trình tự.

- Bước 7: Đánh giá mô hình

Hiệu suất của mô hình HMM có thể được tính toán bằng cách sử dụng nhiều các số liệu khác nhau ví dụ như độ chính xác, độ chính xác, khả năng thu hồi hoặc điểm F1.

**b, Thuật toán Viterbi**

Thuật toán Viterbi là một thể hiện khác của thuật toán biểu đồ mắt cáo, nó giống như thuật toán Forward, ngoài việc lựa chọn các giá trị xác suất chuyển đổi lớn nhất tại mỗi bước thay thế cho việc tính tổng của chúng.

Đầu tiên, chúng ta phải xác định:

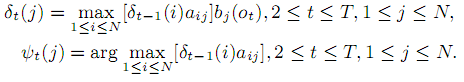


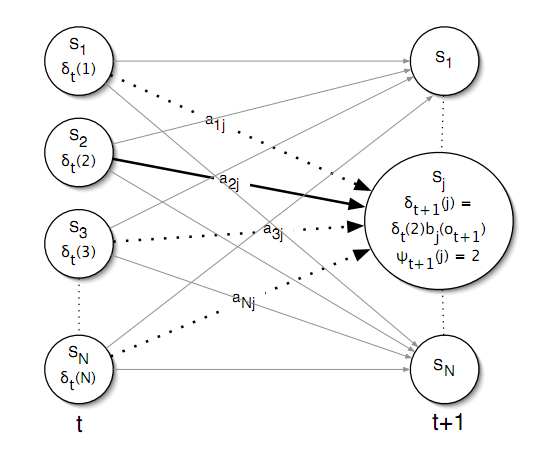
Thuật toán Viterbi được thực hiện như sau:

1. Khởi tạo:

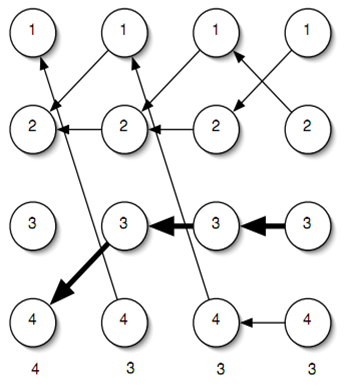


1. Đệ quy:



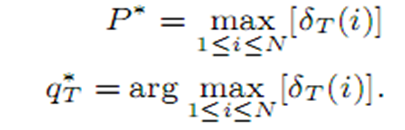


Hình 2.4 Bước đệ quy của thuật toán Viterbi

****

Hình 2.5 Bước quay lui của thuật toán Viterbi

1. Kết thúc:



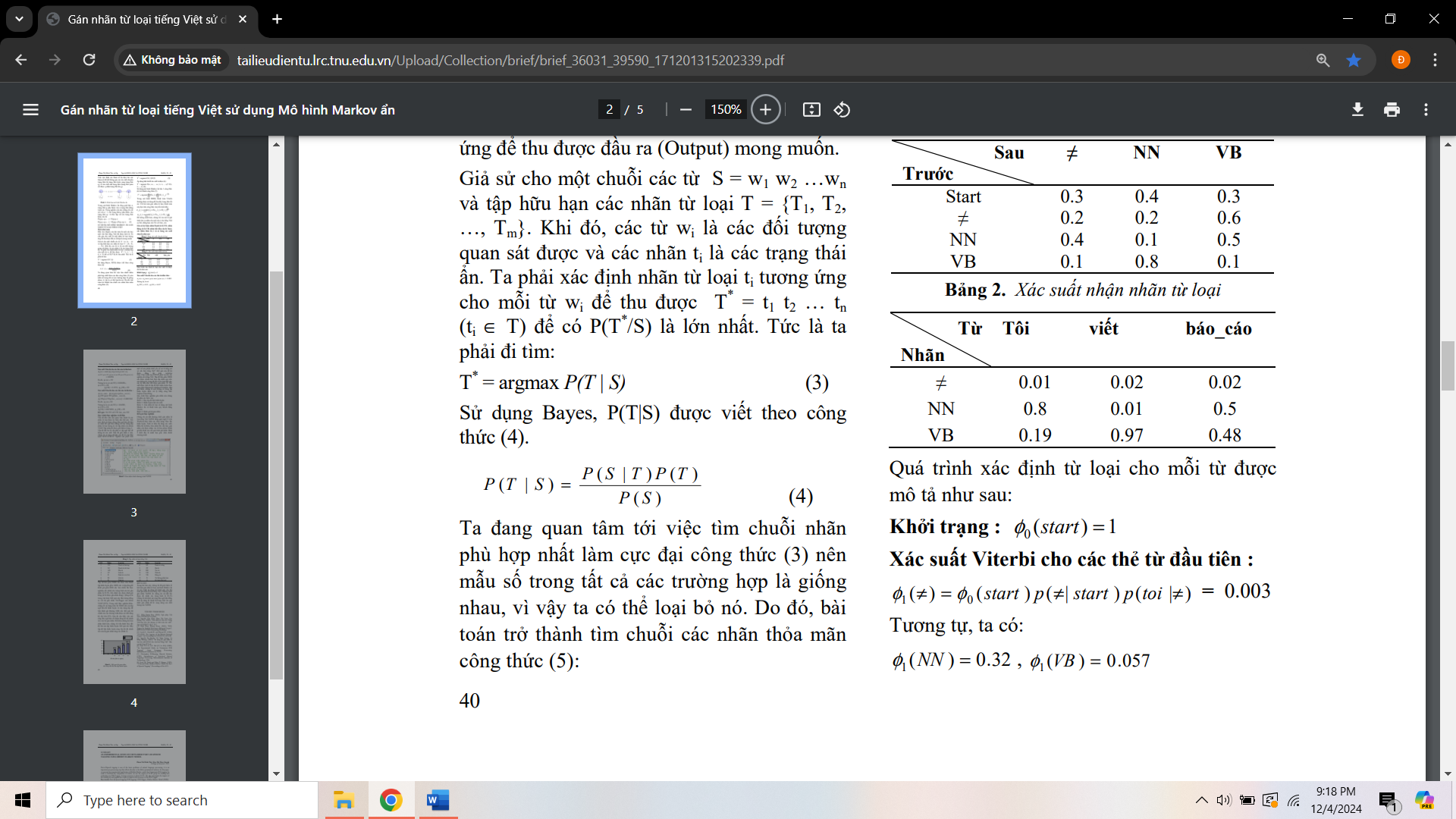
1. Quay lui trình tự trạng thái tối ưu:



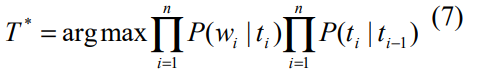
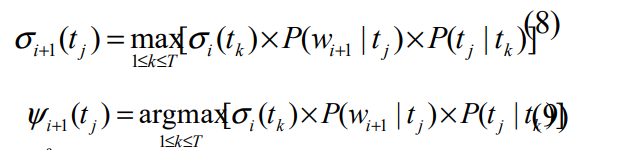
**c, Mô hình markov ẩn (hidden markov model – hmm) kết hợp thuật toán Viterbi trong gán nhãn từ loại**

Đầu vào (Input) của thuật toán là một câu hay một đoạn văn tiếng Việt đã được tách từ. Ta phải thực hiện việc gán cho mỗi từ một nhãn từ loại tương ứng để thu được đầu ra (Output) mong muốn.  
 Ta giả thiết rằng cho một chuỗi các từ S = w1 w2 …wn và tập hữu hạn các nhãn từ loại T = {T1, T2, …, Tm}. Trong đó, các từ wi là các đối tượng quan sát được và các nhãn ti là các trạng thái ẩn. Tiếp theo cần phải tìm nhãn từ loại ti tương ứng cho mỗi từ wi để có kết quả T\* = t1 t2 … tn (ti ∈ T) sao cho thảo mãn P(T\*/S) là có giá trị lớn nhất.

Tức là ta phải đi tìm: T\* = argmax *P(T | S)* (3)  
 Sử dụng Bayes, P(T|S) được viết theo công thức (4).

  
 Chú ý tới việc tìm chuỗi nhãn phù hợp nhất làm cực đại công thức (3) nên mẫu số trong tất cả các trường hợp là giống nhau, vì vậy ta có thể loại bỏ nó. Do đó, bài toán trở thành tìm chuỗi các nhãn thỏa mãn công thức (5)

(5) T\* = argmax *P(S | T)P(T)* Áp dụng luật chuỗi xác suất ta được (6) :

(6) T\* = argmax *P(w1, w2,…, wn | t1, t2,…, tn)\* P(t1,t2,…, tn)*   
 Sử dụng mô hình Markov ẩn bậc 1 công thức (6) trở thành công thức (7)Trong mô hình HMM, thuật toán Viterbi thường dùng để các định dãy trạng thái tối ưu. Với thuật toán gán nhãn từ loại sẽ dựa trên công thức truy hồi sau:

Để tường minh hơn, chúng tôi xin mô tả quá trình tìm ra nhãn cho một câu ví dụ tiếng Việt cụ thể, chẳng hạn câu *Tôi viết luận\_văn.*Ký hiệu nhãn danh từ là NN, nhãn động từ là V, nhãn bắt đầu câu là Start,  
các nhãn khác là ≠ và có bảng xác suất chuyển như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Sau  Trước | ≠ | N | V |
| Start | 0.3 | 0.4 | 0.3 |
| ≠ | 0.2 | 0.2 | 0.6 |
| N | 0.4 | 0.1 | 0.5 |
| V | 0.1 | 0.8 | 0.1 |

Bảng 2. Xác suất chuyển từ loại

|  |  |  |  |
| --- | --- | --- | --- |
| Sau  Trước | Tôi | viết | Luận\_văn |
| ≠ | 0.01 | 0.02 | 0.02 |
| N | 0.8 | 0.01 | 0.5 |
| V | 0.19 | 0.97 | 0.48 |

Bảng 2. Xác suất nhận nhãn loại

Mô tả quá trình xác định từ loại cho các từ như sau:  
Khởi tạo : φ0(*start*) = 1  
**Xác suất Viterbi cho các nhãn từ loại từ đầu tiên :**φ1( ≠) = φ0 (*start) p(* ≠| *start) p*( *toi|* ≠) **=** 0.003  
Tương tự, ta có:  
φ1(*NN* ) = 32.0 , φ1(*VB* ) = 057.0

**Xác suất Viterbi cho các nhãn từ loại của từ thứ hai:**

φ 2 (≠) = max{ φ1 (≠) p( ≠ ≠) p (viet |≠ ),

φ1( N) p (≠ |N ) p( viet |≠), φ1 (V ) p( ≠ |V ) p (viet| ≠ ) = 0.00256

Do đó, ψ 2 (≠) = N

Tương tự, ta có: φ2 (N) = 0.000456 , ψ 2 (N) = V

φ2 (V) = 0.1552 , ψ 2 (V) = N

**Xác suất Viterbi cho các nhãn từ loại của từ thứ ba** :

φ 3 (≠) = max{ φ2 (≠) p( ≠ ≠) p (luan\_van |≠ ),

φ2( N) p (≠ |N ) p( luan\_van |≠), φ2 (V ) p( ≠ |V ) p (luan\_van| ≠ ) = 0.0003104

Do đó, ψ 3 (≠) = V

Tương tự, ta có: φ3 (N) = 0.06208, ψ 3 (N) = V

Φ3 (V) = 0.0074496 , ψ 3 (V) = V

**Kết quả:** Tôi viết luận\_văn

N V N

# **CHƯƠNG 3: CHƯƠNG TRÌNH THỰC NGHIỆM**

3.1. XÂY DỰNG CHỨC NĂNG, NHIỆM VỤ CỦA HỆ THỐNG

### **3.1.1. Xây dựng chức năng hệ thống**

* Người dùng cung cấp dữ liệu đầu vào cho hệ thống thông qua nhập văn bản hoặc đẩy file lên, có thể là các đoạn văn bản tiếng Việt từ nhiều nguồn khác nhau như sách, bài báo, hoặc trang web.
* Người dùng chọn tách từ hoặc gán nhãn từ để hệ thống có thể tự động tách và gán nhãn từng thành phần của văn bản.
* Sau khi nhận kết quả, người dùng có thể sử dụng hệ thống để xử lý các đoạn văn bản mới. Họ có thể lưu kết quả đã được tách và gán nhãn thành file để tải về máy.
* Người dùng có thể kiểm tra kết quả được hệ thống trả về và điều chỉnh nếu cần thiết. Điều này có thể bao gồm việc sửa đổi các nhãn đã gán hoặc cải thiện độ chính xác của hệ thống qua phần liên hệ để cung cấp thông tin phản hồi.

### **3.1.2. Mô tả chức năng hệ thống**

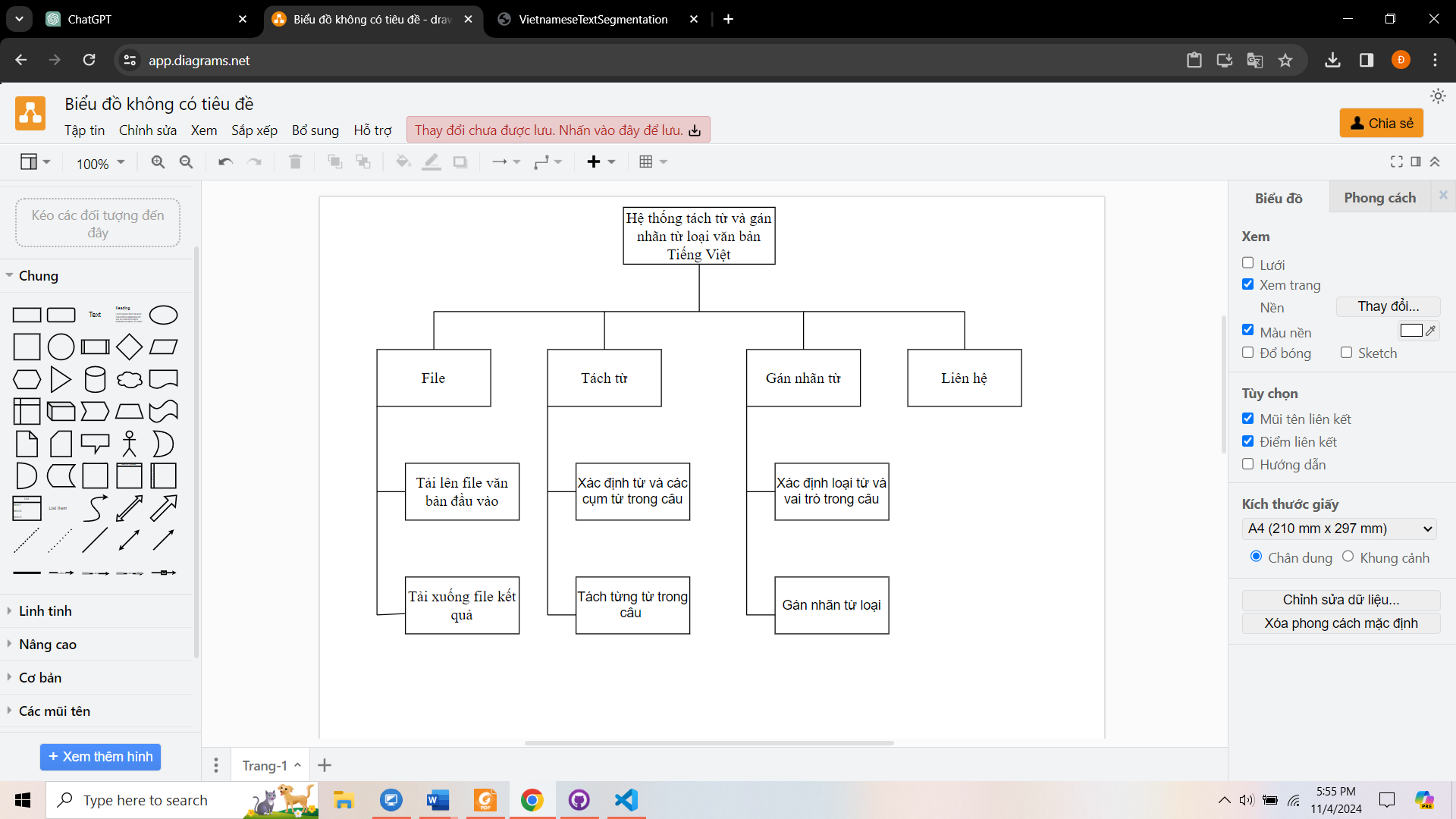
Thông qua các yêu cầu nghiệp vụ, phần mềm có các chức năng sau:

* Giới thiệu: Các thông tin về trang web
* Nhãn dán Tiếng Việt: Chức năng này cho phép người dùng xem kết quả nhãn dán từ loại.
* Liên hệ: Chức năng này cho phép người dùng phản hồi ý kiến của mình về hệ thống.
* Tải file lên: Chức năng này cho phép người dùng tải file có chứa văn bản Tiếng Việt.
* Tách từ: Chức năng này cho phép người dùng tách từ văn bản Tiếng Việt vừa nhập hoặc tải lên.
* Gán nhãn từ: Chức năng này cho phép người dùng gán nhãn từ loại văn bản Tiếng Việt vừa nhập hoặc tải lên.

Yêu cầu phi chức năng

* Giao diện thân thiện với người dùng.
* Giao diện ngôn ngữ tiếng Việt.
* Tính bảo mật, độ tin cậy, hiệu năng.

**3.1.3 Nội dung công việc nghiên cứu**

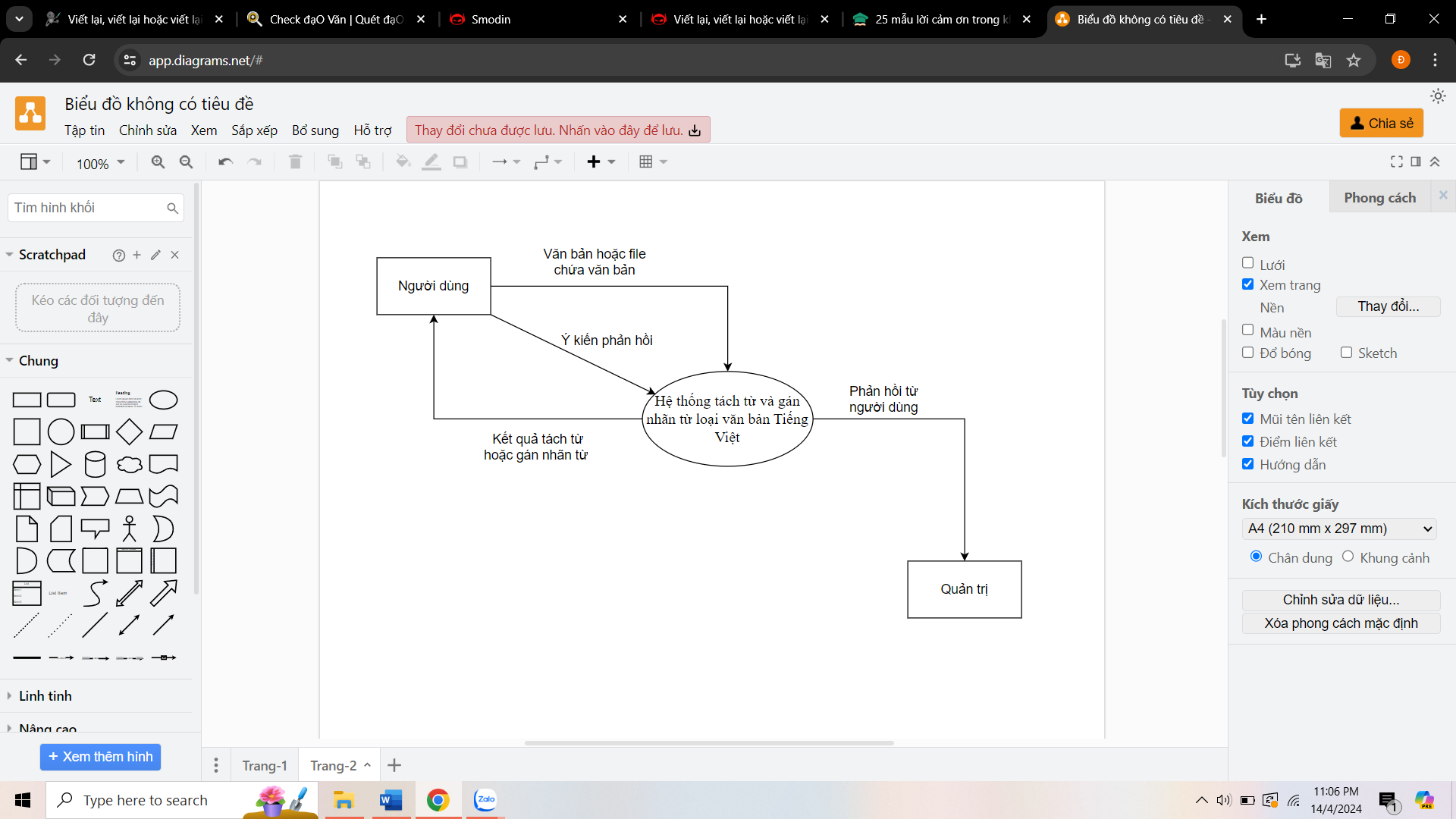
a) Biểu đồ phân cấp chức năng

Hình 3.1. Biểu đồ phân cấp chức năng hệ thống

b) Biểu đồ luồng dữ liệu mức khung cảnh

Dựa trên biểu đồ phân cấp chức năng, ta không thể thấy rõ được mối quan hệ giữa các dữ liệu với nhau. Để có thể nhìn rõ các chi tiết liên hệ với dữ liệu cần để thực hiện các công việc của bài toán và nhóm sử dụng biểu đồ luồng dữ liệu, ta cần quan sát biểu đồ mức khung cảnh để thấy rõ các tác nhân của hệ thống, các luồng dữ liệu tổng quan.

Người dùng gửi yêu cầu tới hệ thống, hệ thống nhận diện văn bản và tiến hành thực hiện tách từ hoặc gán nhãn từ loại. Sau quá trình xử lý sẽ gửi trả kết quả cho người dùng.



Hình 3.2. Biểu đồ luồng dữ liệu mức khung cảnh

## 3.2. MÔI TRƯỜNG CÀI ĐẶT

Hệ điều hành : Visual Studio Code version 1.88  
Ram : 8GB  
Bộ xử lý : AMD Ryzen™ 5-4500U (2.30GHz upto 4.00GHz, 8MB)

## 3.3. CÁC THƯ VIỆN PYTHON SỬ DỤNG TRONG CHƯƠNG TRÌNH

### **3.3.1 Thư viện Flask**

Python Flask là loại framework web siêu nhẹ dựa trên ngôn ngữ lập trình Python, nó được sử dụng để thiết kế nhằm phát triển các ứng dụng web. Flask mang phong cách thiết kế trực quan, đơn giản, linh hoạt và dễ triển khai với mục đích cung cấp giải pháp phát triển ứng dụng web hiệu quả.

Flask có các tính năng cơ bản như định tuyến URL, xây dựng giao diện người dùng, đính tuyến yêu cầu và phản hồi HTTP. Ngoài ra, thư viện cũng hỗ trợ các tính năng mở rộng bằng các thư viện mở rộng Flask (Flask extensions) để thêm các chức năng như xác minh người dùng, kết nối cơ sở dữ liệu, quản lý phiên và nhiều tính năng khác.

Các hàm được sử dụng trong chương trình:

- Render\_template: là một hàm trong Flask được sử dụng để tạo và trả về các trang HTML dựa trên các templates. Render\_template() cho phép sử dụng [công cụ mẫu Jinja](http://jinja.pocoo.org/), điều này sẽ giúp quản lý HTML dễ dàng hơn nhiều bằng cách viết mã HTML của bạn trong các file .html cũng như sử dụng logic trong mã HTML của bạn.. Bạn có thể truyền các danh sách, các đối tượng hoặc bất kỳ dữ liệu nào từ hàm xử lý của Flask đến template để hiển thị thông tin cho người dùng.

- Request: Trong Flask, Request là một đối tượng đại diện cho yêu cầu HTTP được gửi đến ứng dụng. Bằng cách sử dụng Request, bạn có thể truy cập thông tin như dữ liệu form, tham số trên URL, header của request, và nhiều thông tin khác liên quan đến yêu cầu từ client.

### **3.3.2 Thư viện Pyvi**

Pyvi là một thư viện Python được lập trình để xử lý văn bản tiếng Việt trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). Thư viện này cung cấp các công cụ mạnh mẽ cho việc tách từ, gán nhãn từ, chuẩn hóa văn bản, tìm kiếm từ đồng nghĩa và đồng tiền tố, phân loại văn bản, và nhiều tính năng khác, giúp đơn giản hóa quá trình xử lý văn bản tiếng Việt trong các dự án Python.

Ưu điểm:

* Dễ sử dụng: Pyvi được thiết kế để dễ dàng tích hợp vào các dự án Python khác nhau và sử dụng các chức năng một cách thuận tiện.
* Hiệu suất cao: Sử dụng kỹ thuật "maximal matching" và mô hình máy học, Pyvi có hiệu suất cao và thời gian thực thi nhanh trong việc xử lý văn bản tiếng Việt.
* Tính linh hoạt: Thư viện cho phép người dùng tùy chỉnh và cải thiện bộ từ điển và các tính năng khác theo nhu cầu cụ thể của dự án.
* Tích hợp tốt với các công cụ NLP khác: Pyvi có thể kết hợp với các thư viện và công cụ NLP khác như spaCy, NLTK và scikit-learn để xây dựng các ứng dụng phức tạp trong lĩnh vực xử lý ngôn ngữ tự nhiên.

Nhược điểm:

* Hạn chế trong việc xử lý: Mặc dù hiệu suất của Pyvi là cao, nhưng vẫn có thể gặp khó khăn trong xử lý các trường hợp đặc biệt hoặc từ vựng phức tạp.
* Phụ thuộc vào bộ từ điển: Hiệu suất của Pyvi phụ thuộc nhiều vào chất lượng của bộ từ điển, và sự không đồng nhất trong bộ từ điển có thể ảnh hưởng đến độ chính xác của kết quả phân tích.

### **3.3.3 Thư viện Underthesea**

Underthesea là một thư viện Python được phát triển để xử lý ngôn ngữ tự nhiên (NLP) cho tiếng Việt. Thư viện này cung cấp các công cụ và mô hình cho việc tách từ, gán nhãn từ (part-of-speech tagging), phân tích cú pháp và các tác vụ NLP khác, giúp đơn giản hóa quá trình xử lý văn bản tiếng Việt trong các dự án Python.

Ưu điểm:

* Chính xác và hiệu quả: Underthesea sử dụng mô hình HMM và các công nghệ NLP tiên tiến, cung cấp kết quả chính xác và hiệu quả trong việc xử lý văn bản tiếng Việt.
* Dễ sử dụng: Thư viện được thiết kế để dễ dàng tích hợp vào các dự án Python và sử dụng một cách thuận tiện.
* Hỗ trợ cộng đồng: Underthesea là một dự án mã nguồn mở, cho phép đóng góp và phát triển từ cộng đồng người dùng, giúp nâng cao chất lượng và tính đa dạng của thư viện.

Nhược điểm:

* Hạn chế trong việc xử lý: Mặc dù hiệu suất của Underthesea là cao, nhưng vẫn có thể gặp khó khăn trong xử lý các trường hợp đặc biệt hoặc từ vựng phức tạp.
* Yêu cầu tài nguyên máy tính: Các tác vụ phức tạp có thể đòi hỏi nhiều tài nguyên máy tính, đặc biệt là khi xử lý văn bản lớn.

### **3.3.4 Phương thức sử dụng các thư viện trong Python:**

|  |
| --- |
| from flask import Flask, render\_template, request  from pyvi import ViTokenizer  from underthesea import pos\_tag  import re |

## 3.4 GIẢI THUẬT DÙNG CHO TÁCH TỪ VÀ GÁN NHÃN TỪ

### **3.4.1 Hàm ViTokenizer**

ViTokenizer là một hàm từ thư viện pyvi, được sử dụng để tách một đoạn văn bản tiếng Việt thành các từ hoặc cụm từ.

Giải thuật của ViTokenizer sử dụng một số kỹ thuật như "maximal matching" để phân tách từ văn bản.

Cách hoạt động của ViTokenizer:

* ViTokenizer sử dụng một bộ từ điển có sẵn chứa các từ phổ biến trong tiếng Việt.
* Khi nhận vào một đoạn văn bản, ViTokenizer bắt đầu từ đầu đoạn văn bản và thử từng cách chia từ theo chiều dài từ lớn nhất đến nhỏ nhất.
* Với mỗi cách chia từ, ViTokenizer kiểm tra xem từ tạo ra có nằm trong từ điển hay không. Nếu có, cách chia từ này được chấp nhận và tiếp tục tìm cách chia từ tiếp theo. Nếu không, ViTokenizer thử các cách chia từ khác.
* Quá trình này tiếp tục cho đến khi cả đoạn văn bản được duyệt hết.
* Kết quả cuối cùng là một danh sách các từ đã được phân tách từ đoạn văn bản.

Ví dụ: Đoạn văn bản " Chào mừng đến với sản phẩm tách từ và dán nhãn của chúng tôi " sau khi được tách từ sẽ trở thành: [" Chào mừng ", " đến ", " với ", " sản phẩm ", " tách ", " từ ", " và ", " dán ", " nhãn ", " của ", " chúng tôi "].

Ưu điểm của ViTokenizer:

* Dễ sử dụng và và có thể dễ dàng tích hợp vào các dự án Python.
* Hiệu suất cao
* Người dùng có thể tùy chỉnh và cải thiện bộ từ điển của ViTokenizer để tăng tính linh hoạt và độ chính xác.

Nhược điểm của ViTokenizer:

* Hạn chế trong việc xử lý các trường hợp đặc biệt hoặc từ vựng phức tạp.
* Khó khăn trong phân biệt được các từ có cùng một tiền tố hoặc hậu tố nhưng khác nghĩa.
* Phụ thuộc vào bộ từ điển:

### **3.4.2 Hàm pos\_tag**

pos\_tag là một hàm từ thư viện underthesea, được sử dụng để gán nhãn cho từng từ trong câu, xác định loại từ và vai trò của từ đó trong câu.

Giải thuật của pos\_tag sử dụng mô hình máy học như HMM (Hidden Markov Models) được đào tạo trước để dự đoán loại từ và vai trò của từ trong câu.

Cách hoạt động của pos\_tag:

* Khi nhận một câu văn đầu vào, pos\_tag sẽ sử dụng mô hình máy học đã được đào tạo trước để dự đoán loại từ và vai trò của từ trong câu.
* Đầu vào của mô hình thường là các đặc trưng của từ (như từ đầu tiên trong câu, từ trước và sau của từ cần dự đoán, các đặc trưng ngữ cảnh khác) được trích xuất từ câu văn.
* Dựa vào các đặc trưng này, mô hình sẽ dự đoán loại từ và vai trò của từ trong câu.
* Kết quả cuối cùng là một danh sách các từ đã được gán nhãn với loại từ và vai trò tương ứng.

Ví dụ: Đoạn văn bản " Chào mừng đến với sản phẩm tách từ và dán nhãn của chúng tôi " sau khi được gán nhãn từ sẽ trở thành: [("Chào mừng ", "N"), ("đến ", "V"), ("với ", "A"), ("sản phẩm ", "N"), ("tách ", "V"), ("từ ", "N"), ("và ", "C"), ("dán ", "V"), ("nhãn ", "N"), ("của ", "A"), ("chúng tôi ", "N")].

Trong đó, "N" đại diện cho danh từ, "V" đại diện cho động từ, "A" đại diện cho giới từ, "C" đại diện cho liên từ.

Ưu điểm của pos\_tag:

* Cung cấp thông tin chi tiết về loại từ và vai trò của từ trong câu.
* Có khả năng học và dự đoán tốt trên các dữ liệu mới.
* Có thể sử dụng các mô hình máy học tiên tiến để cải thiện độ chính xác.

Nhược điểm của pos\_tag:

* Yêu cầu một mô hình máy học đã được đào tạo trước, có thể tốn kém và phức tạp trong quá trình triển khai.
* Đôi khi độ chính xác phụ thuộc nhiều vào chất lượng và số lượng dữ liệu huấn luyện.

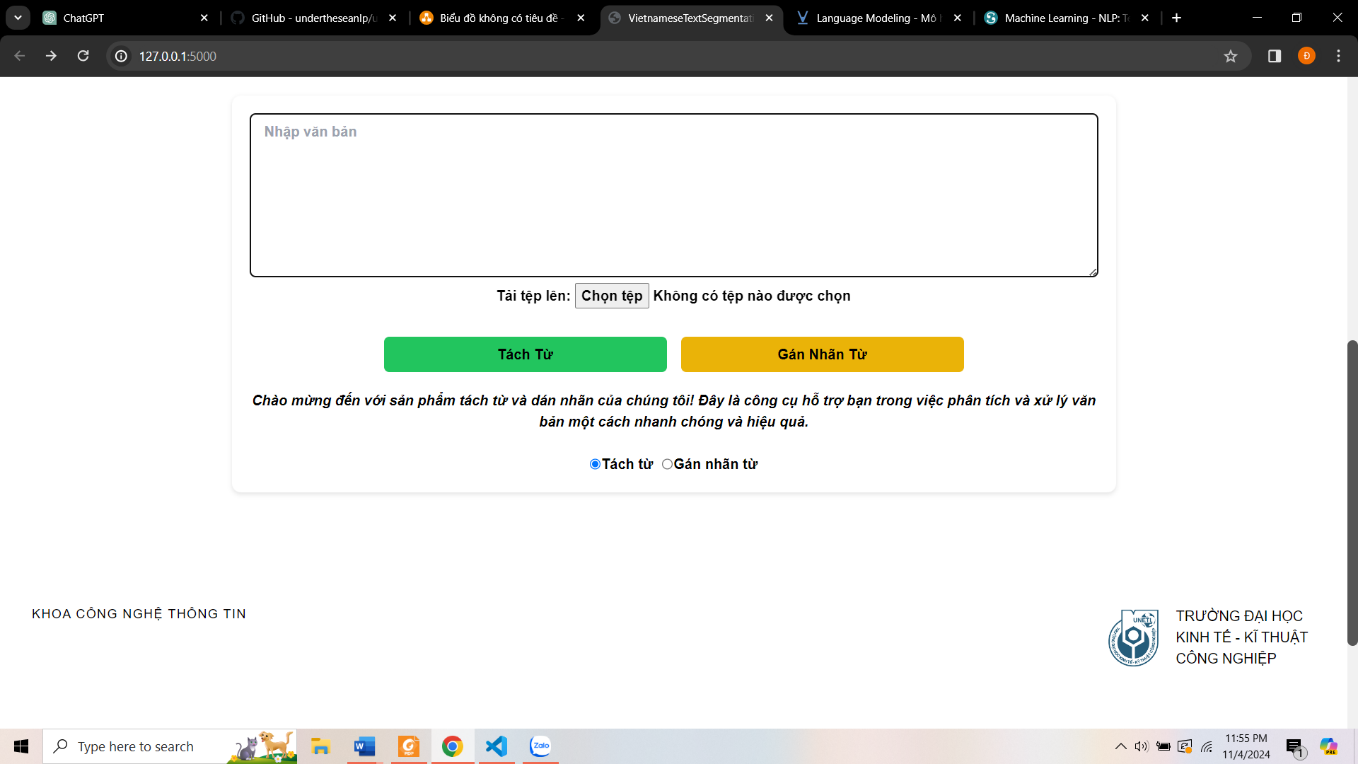
## 3.5 CHƯƠNG TRÌNH CÀI ĐẶT

|  |
| --- |
| app = Flask(\_\_name\_\_)  def remove\_underscores(segmented\_text):      return [re.sub('\_', ' ', word) for word in segmented\_text]  def get\_pos\_mapping():      return {          'P': 'Đại từ (pronoun)',          'V': 'Động từ (verb)',          'M': 'Danh từ (noun)',          'N': 'Danh từ (noun)',          'E': 'Giới từ (preposition)',          'Np': 'Danh từ riêng (proper noun)',          'C': 'Liên từ (conjunction)',          'A': 'Tính từ (adjective)',          'CH': 'Dấu câu (punctuation)'      }  def process\_input\_text(input\_text):      segmented\_text = ViTokenizer.tokenize(input\_text).split()      segmented\_text\_without\_underscore = remove\_underscores(segmented\_text)      tagged\_text = pos\_tag(input\_text)      return segmented\_text\_without\_underscore, tagged\_text  def print\_results(segmented\_text, tagged\_text):      pos\_mapping = get\_pos\_mapping()      results = []      for word, pos\_tag in tagged\_text:          results.append((word, pos\_mapping.get(pos\_tag, 'Không xác định')))      return results  def normalize\_input\_text(input\_text):      input\_text = re.sub(r"[^\w\s]", "", input\_text)      input\_text = re.sub(r"\s+", " ", input\_text)      input\_text = input\_text.lower()      return input\_text  @app.route('/', methods=['GET', 'POST'])  def index():      if request.method == 'POST':          input\_text = request.form['input\_text']          action = request.form.get('action', '')          try:              if 'file' in request.files:                  file = request.files['file']                  if file.filename != '':                      document = Document(file)                      input\_text = ""                      for paragraph in document.paragraphs:                          input\_text += paragraph.text + "\n"              if action == 'segment':                  segmented\_text = ViTokenizer.tokenize(input\_text)                  segmented\_text\_with\_slash = segmented\_text.replace(' ', ' / ')                  return render\_template('index.html', input\_text=input\_text, segmented\_text=segmented\_text\_with\_slash)              elif action == 'segment\_and\_tag':                  segmented\_text = ViTokenizer.tokenize(input\_text).split()                  segmented\_text\_without\_underscore = remove\_underscores(segmented\_text)                  tagged\_text = pos\_tag(input\_text)                  results = print\_results(segmented\_text\_without\_underscore, tagged\_text)                  return render\_template('index.html', input\_text=input\_text, results=results)              if action == 'export':                  export\_type = request.form.get('export\_type')                  result\_doc = Document()                  result\_doc.add\_heading('Kết quả', level=1)                  if export\_type == 'tag':                      segmented\_text = ViTokenizer.tokenize(input\_text).split()                      segmented\_text\_without\_underscore = remove\_underscores(segmented\_text)                      tagged\_text = pos\_tag(input\_text)                      result\_text = "\n".join([f"{word} : {pos}" for word, pos in tagged\_text])                      result\_doc.add\_paragraph(result\_text)                  else:                      segmented\_text = ViTokenizer.tokenize(input\_text)                      segmented\_text\_with\_slash = segmented\_text.replace(' ', ' / ')                      result\_doc.add\_paragraph(segmented\_text\_with\_slash)                  # Lưu tài liệu dưới dạng file Word                  result\_doc.save("result.docx")                  return send\_file("result.docx", as\_attachment=True)          except Exception as e:              error\_message = f"Lỗi: {e}"              return render\_template('index.html', input\_text=input\_text, error\_message=error\_message)      return render\_template('index.html')  @app.route('/gioi-thieu')  def gioi\_thieu():      return render\_template('gioi-thieu.html')  @app.route('/nhan-dan-tieng-viet')  def nhan\_dan\_tieng\_viet():      return render\_template('nhan-dan-tieng-viet.html')  @app.route('/lien-he')  def lien\_he():      return render\_template('lien-he.html')  if \_\_name\_\_ == "\_\_main\_\_":      app.run(debug=True) |

## 3.6 GIAO DIỆN CHƯƠNG TRÌNH

### **3.6.1 Giao diện trang chủ**

Hình 3.3. Giao diện màn hình trang chủ (1)



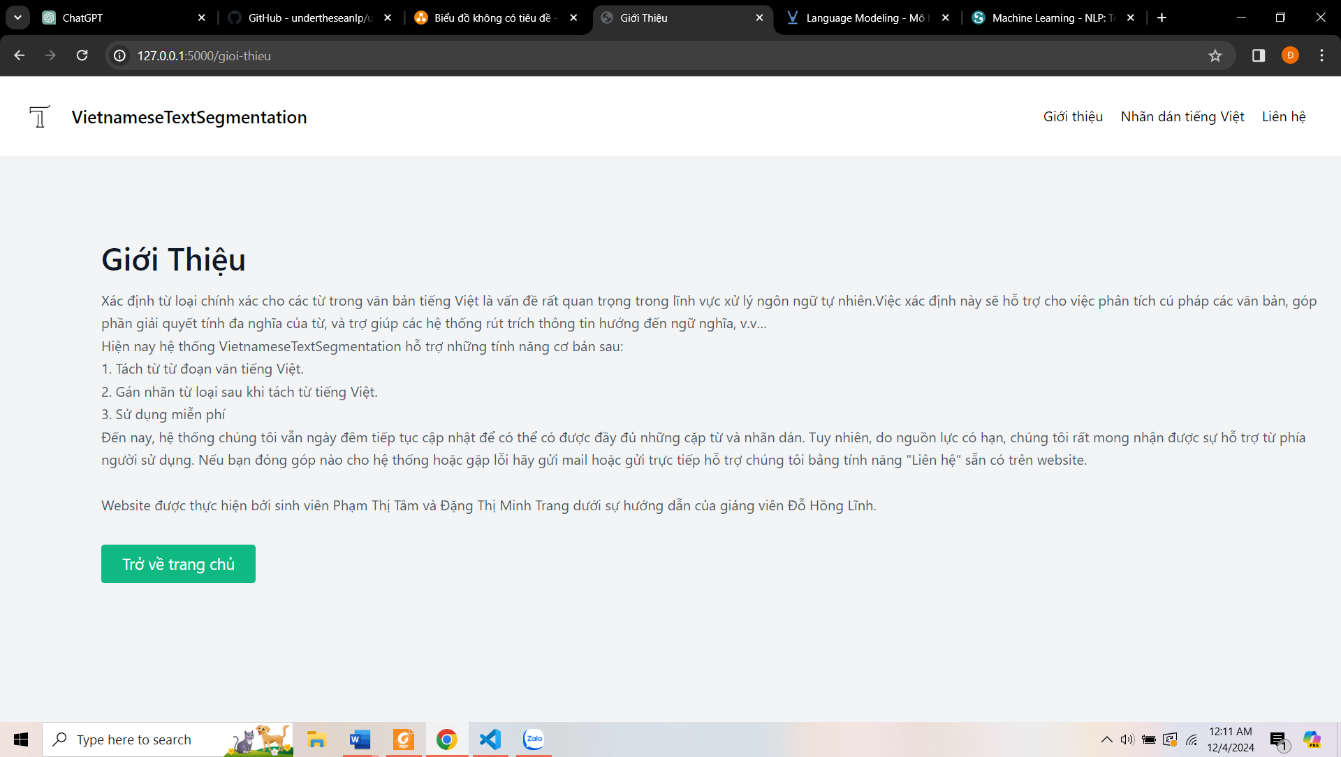
*Hình 3.4. Giao diện màn hình trang chủ (2*)

* Mô tả các đối tượng trên trang chủ:

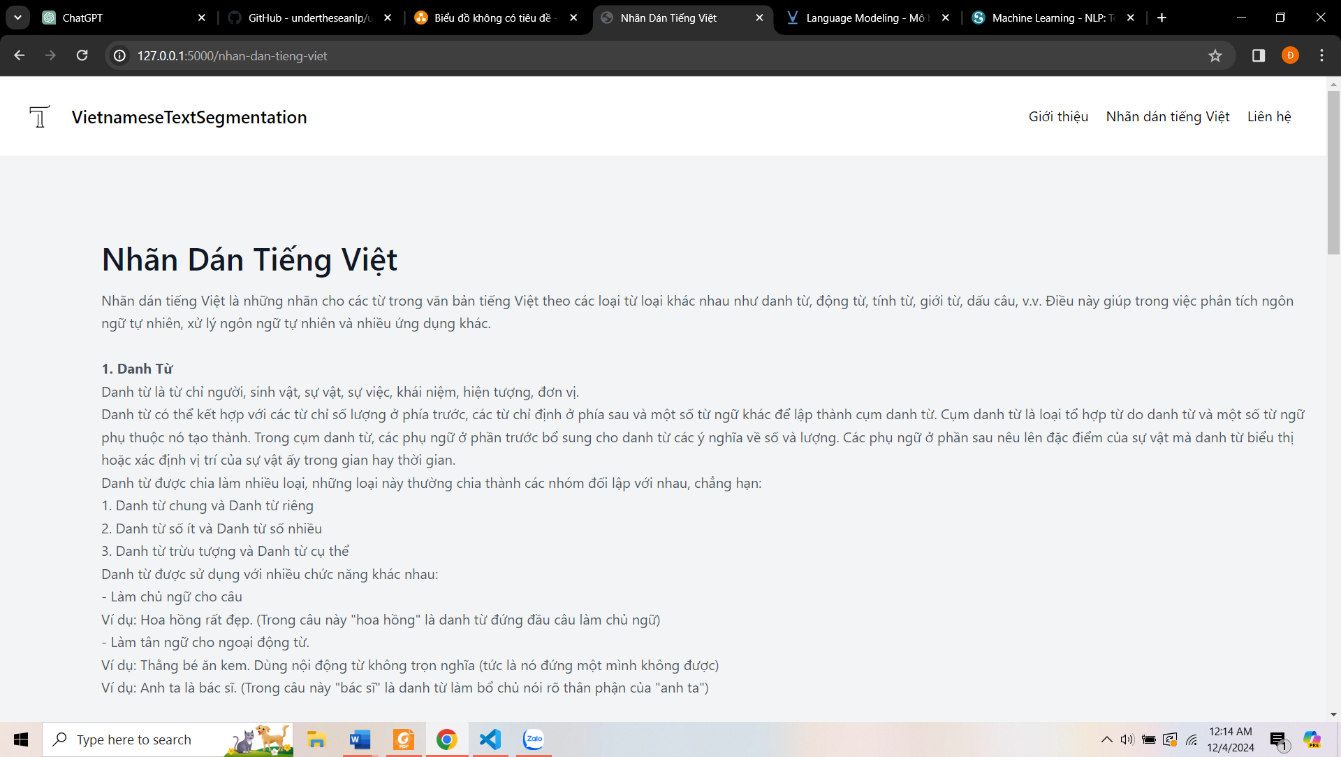
|  |  |  |
| --- | --- | --- |
| STT | Tên đối tượng | Ý nghĩa |
| 1 | Logo | Logo đại diện cho trang web |
| 2 | Giới thiệu | Giới thiệu sơ qua về trang web |
| 3 | Nhãn dán Tiếng Việt | Giới thiệu sơ qua về các nhãn dán Tiếng Việt |
| 4 | Liên hệ | Cho phép người dùng phản hồi lại về trang web |
| 5 | Ô text nhập văn bản | Cho phép người dùng nhập văn bản |
| 6 | Chọn tệp | Cho phép người dùng tải file có chứa văn bản lên hệ thống |
| 7 | Nút “Tách từ” | Cho phép người dùng tách văn bản đã nhập |
| 8 | Nút “Gán nhãn từ” | Cho phép người dùng tách và gán nhãn văn bản đã nhập |
| 9 | Chọn “Tách từ” hoặc “Gán nhãn từ” | Cho phép người dùng tải file có chứa kết quả về máy |

Bảng 3.1 Các đối tượng trên trang chủ

### **3.6.2 Menu**

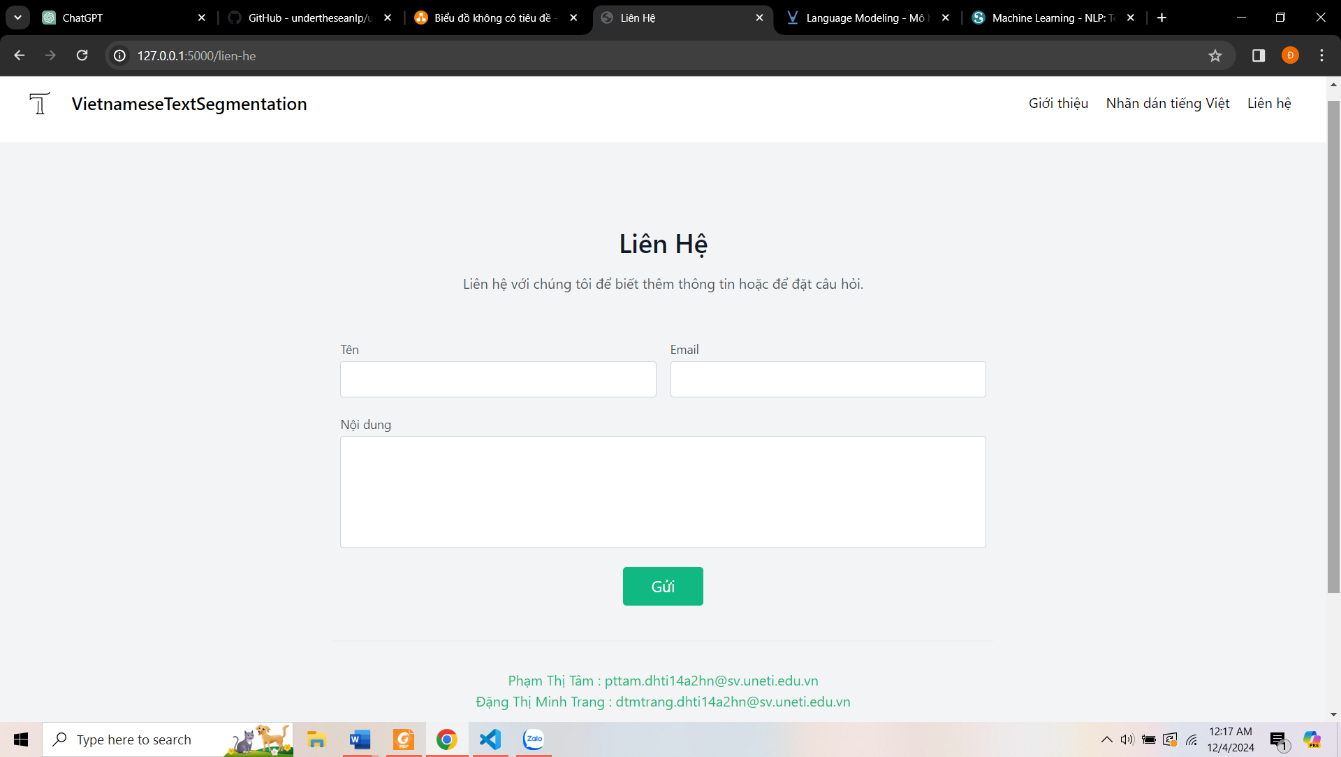
- Giới thiệu: Giới thiệu sơ qua về trang web

*Hình 3.5. Giao diện giới thiệu*

- Nhãn dán Tiếng Việt: Giới thiệu sơ qua về nhãn dán Tiếng Việt

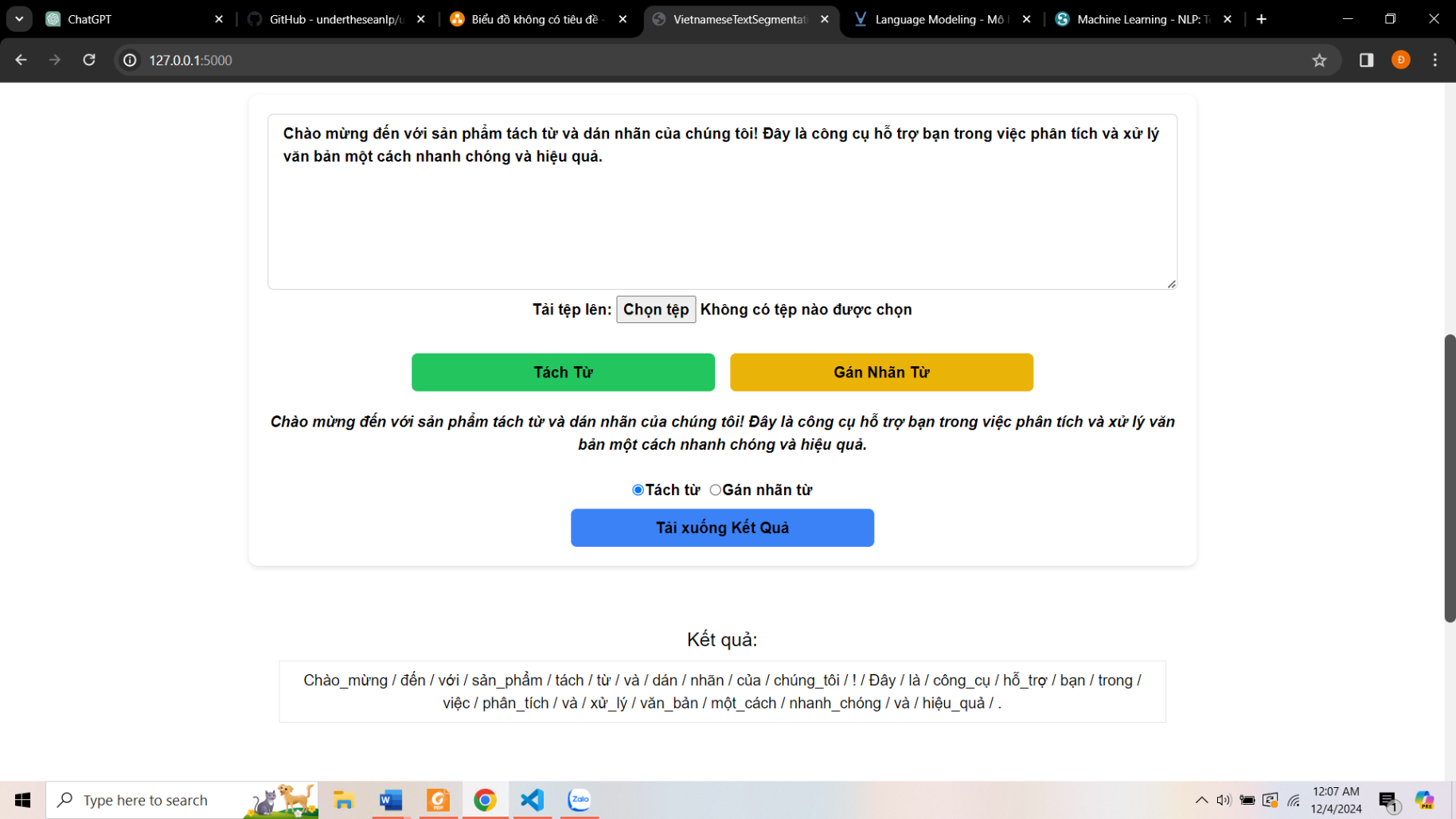
*Hình 3.6. Giao diện* nhãn dán Tiếng Việt

- Liên hệ: Cho phép người dùng phản hồi lại về trang web

Hình 3.7. Giao diện phần liên hệ

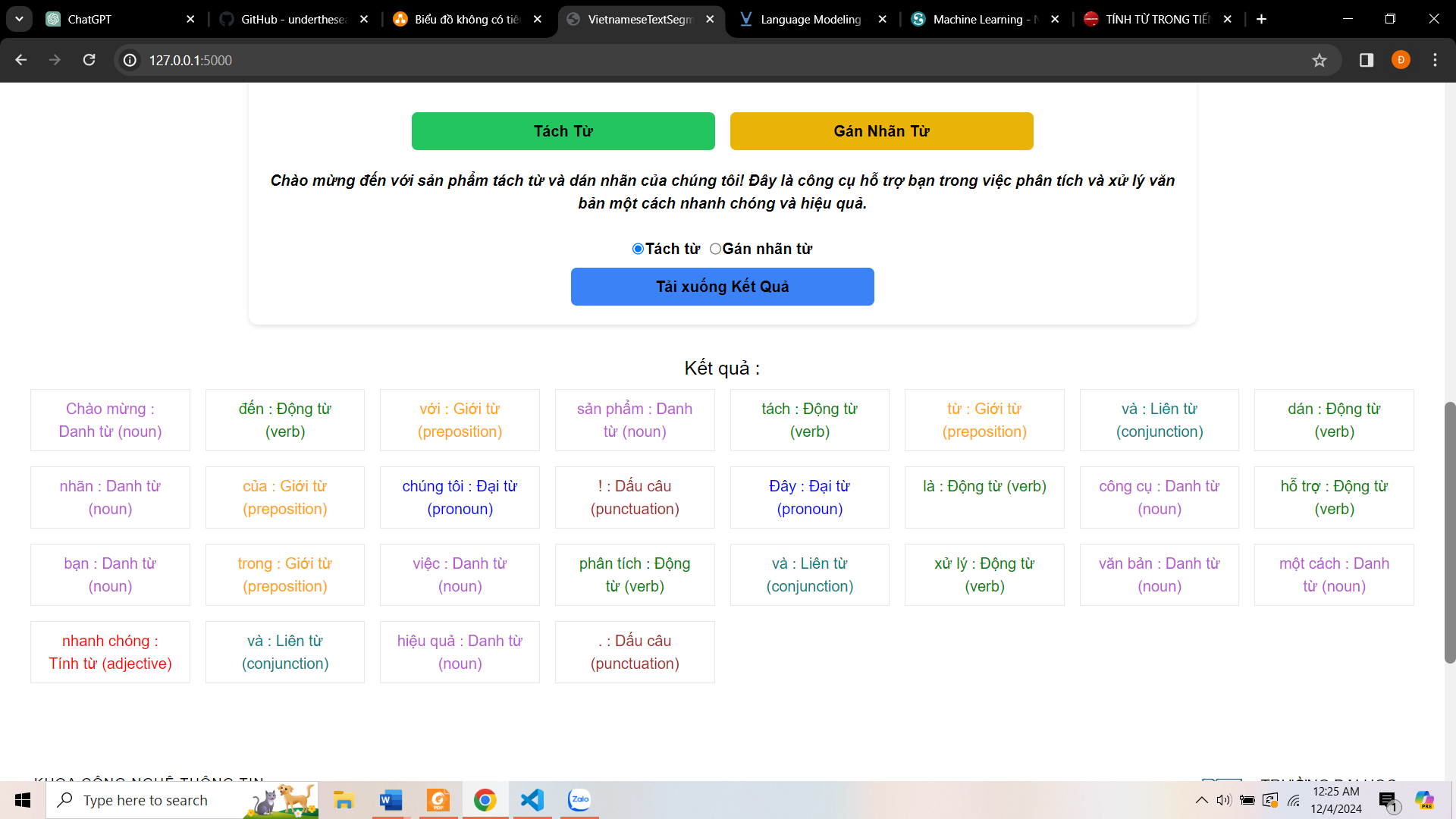
### **3.6.3 Chức năng tách từ**

Hiển thị kết quả tách từ của 1 đoạn văn bản mà người dùng nhập vào

*Hình 3.8. Kết quả tách từ tiếng Việt*

### **3.6.4 Chức năng gán nhãn từ loại**

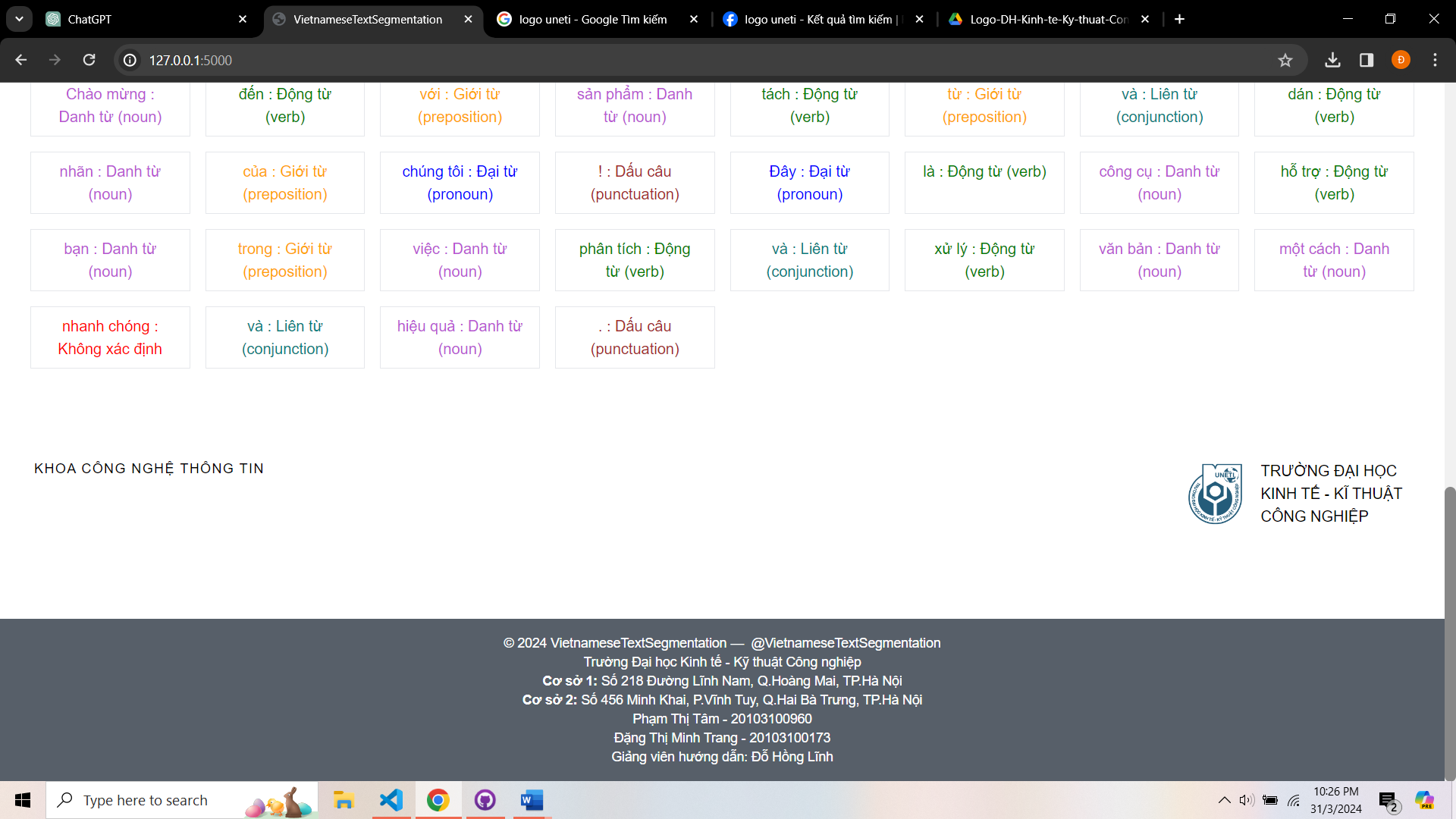
Hiển thị kết quả gán nhãn của 1 đoạn văn bản mà người dùng nhập vào



Hình 3.9. Kết quả gán nhãn từ tiếng Việt

### **3.6.5 Giao diện phần footer**

Hiển thị địa chỉ, thông tin chủ trang web



Hình 3.10. Giao diện phần footer

# **CHƯƠNG 4: TỔNG KẾT**

## 4.1. KẾT LUẬN

Sau thời gian lên ý tưởng, nghiên cứu và xây dựng luận văn tốt nghiệp với đề tài *“Xây dựng ứng dụng tách từ và gán nhãn từ loại văn bản Tiếng Việt”*, chúng em đã mở mang và học tập thêm được nhiều kiến thức, kỹ năng mới cho chính mình. Được sự hướng dẫn tận tình của giảng viên hướng dẫn Cô Đỗ Hồng Lĩnh, bên cạnh đó là sự quan tâm, giúp đỡ của các thầy, các cô trong khoa Công nghệ thông tin, cùng với nỗ lực, cố gắng của bản thân, chúng em đã hoàn thành luận văn tốt nghiệp, thực hiện các nội dung theo đúng kế hoạch đã đề ra. Cụ thể, luận văn đã trình bày và giải quyết được những vấn đề sau:  
 - Hiểu được tổng quan về xử lý ngôn ngữ tự nhiên.  
 - Tìm hiểu và nghiên cứu về từ tiếng Việt và các nhãn dán.  
 - Tìm hiểu và nghiên cứu về mô hình markov ẩn và các thư viện tiếng Việt như ViTokenizer, underthesea,….  
 - Nghiên cứu và tìm hiểu về ngôn ngữ Python.  
 - Xây dựng chương trình thực nghiệm.  
 Vì thời gian nghiên cứu của đề tài không nhiều cũng như kiến thức, kinh nghiệm và năng lựccá nhân của bản thân em về lĩnh vực này còn hạn chế, cho nên không tránh khỏi luận văn vẫn còn những thiếu sót như:  
 - Chương trình ứng dụng còn hạn chế, đóng góp khoa học không cao.  
 - Luận văn chưa đi sâu vào giải thích tường tận về một số lĩnh vực.

## 4.2. HƯỚNG PHÁT TRIỂN

Trong thời gian tới em sẽ tiếp tục trau dồi kiến thức, nghiên cứu để phát triển sản phẩm hoàn thiện và chính xác cao hơn nữa, cùng với đó là nghiên cứu ứng dụng chương trình vào các hệ thống ngôn ngữ… với độ tin cậy cao dành cho ngôn ngữ tiếng Việt.

# **TÀI LIỆU THAM KHẢO**

[1] Phạm Thị Minh Thu, Đào Thị Thuý Quỳnh, *tạp chí khoa học và công nghệ, gán nhãn từ loại tiếng việt sử dụng mô hình markov ẩn*, Trường Đại học Khoa học – ĐH Thái Nguyên.

[2] Lê Thanh Hương, *Giáo trình tách từ tiếng Việt*, Trường Đại học Bách khoa Hà Nội.

[3] Trần Thị Oanh*, Mô hình tách từ, gán nhãn từ loại và hướng tiếp cận tích hợp cho tiếng Việt*, trường Đại học Công nghệ Đại học Quốc gia Hà Nội.

[4] Thái Đông Hồ, *nghiên cứu một số kỹ thuật tách từ trong xử lý ngôn ngữ tự nhiên*, Trường Đại học Kiên Giang.

[5] Le An Ha (2003), *A method for word segmentation in Vietnamese,* In Proceedings of Curpus Linguistics. Lancaster, UK.

[6] Chih-Hao Tsai (2000), *MMSEG: A word identification system for Mandarin Chinese text Based on two variants of the Maximum Matching Algorithm*.