# 

# **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 hay Natural Language Processing (NLP) là một nhánh của [trí tuệ nhân tạo (AI)](https://vbee.vn/blog/ai/tri-tue-nhan-tao-ai-la-gi/), hướng đến việc giúp máy tính có thể hiểu, phân tích, diễn giải và tương tác với ngôn ngữ của con người. Công nghệ xử lý ngôn ngữ này cho phép máy móc xử lý ngôn ngữ tự nhiên dưới dạng văn bản hoặc giọng nói, giúp máy tính giao tiếp với con người một cách tự nhiên và trực quan hơn.

Lịch sử của Xử lý Ngôn ngữ Tự nhiên (NLP) khá phong phú và đã trải qua nhiều giai đoạn phát triển quan trọng. Dưới đây là một cái nhìn tổng quan về các giai đoạn chính trong quá trình phát triển của NLP:

### 1. **Thời kỳ đầu (1950s - 1960s)**

* **1950**: Alan Turing giới thiệu bài kiểm tra Turing, một tiêu chuẩn để đánh giá khả năng của máy tính trong việc thể hiện hành vi thông minh giống con người.
* **1957**: Noam Chomsky xuất bản "Syntactic Structures", đặt nền móng cho lý thuyết ngữ pháp biến đổi và ảnh hưởng sâu rộng đến NLP.
* **1950s - 1960s**: Các hệ thống dịch máy ban đầu được phát triển, nhưng kết quả vẫn còn rất thô sơ.

### 2. **Thời kỳ giữa (1970s - 1980s)**

* **1970s**: Giới thiệu các mô hình ngữ nghĩa học, như mô hình mạng ngữ nghĩa và khung.
* **1980**: Đánh dấu bước tiến quan trọng với sự xuất hiện của các mô hình thống kê và tiếp cận học máy.

### 3. **Thời kỳ hiện đại (1990s - 2010s)**

* **1990s**: Bùng nổ của các mô hình ngôn ngữ dựa trên xác suất và mô hình thống kê. Sự phát triển của các mô hình n-gram và các công cụ phân tích ngữ pháp xác suất.
* **2000s**: Sự phát triển của các mô hình máy học tiên tiến như mạng nơ-ron, máy vector hỗ trợ (SVM) và mô hình ẩn Markov (HMM).
* **2010s**: Cuộc cách mạng của deep learning với các mô hình như Word2Vec, GloVe, và đặc biệt là sự ra đời của mô hình Transformer và BERT (Bidirectional Encoder Representations from Transformers) của Google vào năm 2018.

### 4. **Thời kỳ tiên tiến (2020s - nay)**

* **2020s**: Sự phát triển mạnh mẽ của các mô hình ngôn ngữ lớn (LLM) như GPT-3 của OpenAI và các phiên bản tiên tiến của Transformer. Những mô hình này có khả năng xử lý và tạo ra văn bản tự nhiên với mức độ chính xác và tinh tế cao.
* **Hiện tại và Tương lai**: Các nghiên cứu tiếp tục tập trung vào việc cải thiện hiệu suất, độ chính xác, và khả năng hiểu ngữ nghĩa của các mô hình NLP. Các ứng dụng của NLP ngày càng đa dạng, từ dịch máy, chatbot, phân tích cảm xúc đến trợ lý ảo.

NLP đã và đang tiến hóa nhanh chóng, và chúng ta có thể mong đợi nhiều cải tiến đột phá trong tương lai gần. Nếu bạn muốn biết thêm chi tiết về bất kỳ giai đoạn nào hoặc có bất kỳ câu hỏi nào khác, hãy cho tôi biết nhé!

Dưới đây là thông tin về các **ứng dụng của NLP (Natural Language Processing - Xử lý ngôn ngữ tự nhiên)** trong đời sống và công việc, được tóm tắt lại:

### **1.1.2 Ứng dụng của NLP (Xử lý ngôn ngữ tự nhiên)**

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực con của trí tuệ nhân tạo (AI) liên quan đến việc giúp máy tính hiểu, phân tích và tạo ra ngôn ngữ tự nhiên như con người. NLP có nhiều ứng dụng trong các ngành công nghiệp khác nhau. Dưới đây là một số ứng dụng phổ biến của NLP:

### 1. **Dịch máy (Machine Translation)**

* Dịch văn bản từ ngôn ngữ này sang ngôn ngữ khác. Ví dụ như **Google Translate**, **Microsoft Translator**, và **DeepL** sử dụng NLP để dịch chính xác các câu văn từ một ngôn ngữ sang ngôn ngữ khác.

### 2. **Phân tích cảm xúc (Sentiment Analysis)**

* NLP có thể phân tích cảm xúc trong văn bản để xác định thái độ, tình cảm, hoặc cảm xúc của người viết. Các ứng dụng này được sử dụng trong việc phân tích đánh giá khách hàng, phản hồi từ mạng xã hội, hoặc trong chiến dịch tiếp thị.

### 3. **Tìm kiếm thông tin (Information Retrieval)**

* NLP giúp cải thiện khả năng tìm kiếm thông tin từ các cơ sở dữ liệu, văn bản, hoặc tài liệu. Ví dụ, các công cụ tìm kiếm như **Google** sử dụng NLP để cung cấp kết quả tìm kiếm phù hợp nhất với yêu cầu của người dùng.

### 4. **Trợ lý ảo và Chatbot**

* Các hệ thống trợ lý ảo như **Siri**, **Alexa**, **Google Assistant**, hoặc các **chatbot** sử dụng NLP để hiểu và trả lời câu hỏi từ người dùng. NLP giúp chúng hiểu ngôn ngữ tự nhiên và thực hiện các tác vụ dựa trên yêu cầu của người dùng.

### 5. **Nhận dạng giọng nói (Speech Recognition)**

* NLP được ứng dụng trong nhận dạng giọng nói, chuyển đổi lời nói thành văn bản. Các công nghệ như **Google Speech-to-Text** hoặc **Apple Dictation** sử dụng NLP để cải thiện độ chính xác của việc nhận diện và chuyển đổi giọng nói thành văn bản.

### 6. **Tóm tắt văn bản (Text Summarization)**

* NLP giúp tóm tắt các đoạn văn bản dài thành những bản tóm tắt ngắn gọn, giúp tiết kiệm thời gian và cải thiện hiệu quả tìm kiếm thông tin. Các ứng dụng như báo cáo tóm tắt, hoặc tính năng tóm tắt của các dịch vụ tin tức sử dụng NLP để trích xuất những điểm chính trong bài viết.

### 7. **Khám phá thông tin (Information Extraction)**

* NLP có thể được sử dụng để trích xuất thông tin cụ thể từ một nguồn dữ liệu lớn, ví dụ như nhận diện các thực thể (như tên, địa điểm, thời gian) hoặc các mối quan hệ giữa các thực thể trong văn bản.

### 8. **Phân loại văn bản (Text Classification)**

* NLP được sử dụng để phân loại các văn bản vào các nhóm hoặc danh mục nhất định. Ví dụ, phân loại email vào các thư mục như "spam" hay "hộp thư đến", phân loại các bài báo theo chủ đề, hoặc phân loại các phản hồi của khách hàng.

### 9. **Tạo văn bản tự động (Text Generation)**

* NLP có thể được sử dụng để tạo ra văn bản tự động, như trong việc tạo ra các bài viết, thư từ, hoặc thậm chí là sáng tác nội dung dựa trên một số từ khóa hoặc chủ đề nhất định. Một ví dụ điển hình là việc sử dụng **GPT-3** hoặc **GPT-4** trong việc tạo nội dung sáng tạo.

### 10. **Phân tích ngữ nghĩa (Semantic Analysis)**

* NLP giúp máy tính hiểu và phân tích ngữ nghĩa của các từ và câu trong văn bản, giúp chúng có thể hiểu ngữ cảnh và mối quan hệ giữa các từ trong một câu.

### 11. **Nhận diện đối tượng trong văn bản (Named Entity Recognition - NER)**

* NER là một kỹ thuật trong NLP giúp nhận diện các thực thể như tên người, tên địa điểm, thời gian, hay tổ chức trong văn bản. Kỹ thuật này được sử dụng trong nhiều ứng dụng, chẳng hạn như hệ thống quản lý thông tin, báo cáo tài chính, và nhiều hơn nữa.

### 12. **Phân tích cú pháp (Syntax Parsing)**

* NLP giúp phân tích cấu trúc cú pháp của các câu trong văn bản để hiểu rõ hơn về mối quan hệ giữa các từ trong câu, từ đó hỗ trợ các ứng dụng như dịch máy, kiểm tra ngữ pháp, hay tạo ra các câu chính xác hơn.

Các ứng dụng trên cho thấy tầm quan trọng và tiềm năng của NLP trong việc cải thiện và tự động hóa các tác vụ liên quan đến ngôn ngữ. NLP ngày càng trở nên phổ biến và đang được áp dụng rộng rãi trong nhiều lĩnh vực, từ dịch vụ khách hàng, giáo dục, đến y tế và nghiên cứu khoa học.

Dưới đây là phần **1.1.3 Ưu điểm và nhược điểm của NLP** (Natural Language Processing - Xử lý ngôn ngữ tự nhiên):

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

**Xử lý ngôn ngữ tự nhiên (NLP)** có nhiều lợi ích, nhưng cũng không thiếu thách thức. Sau đây là một số **ưu điểm** và **nhược điểm** chính của NLP:

### **Ưu điểm của NLP**

1. **Tự động hóa các tác vụ liên quan đến ngôn ngữ**
   * NLP giúp tự động hóa nhiều tác vụ như dịch ngôn ngữ, phân tích văn bản, tạo ra văn bản tự động, hay hỗ trợ khách hàng, giúp tiết kiệm thời gian và công sức của con người.
2. **Tiết kiệm thời gian và chi phí**
   * NLP giảm thiểu sự can thiệp của con người trong các công việc như trả lời câu hỏi khách hàng, phân tích dữ liệu văn bản, hay xử lý thông tin. Điều này làm giảm chi phí vận hành và tăng hiệu quả công việc.
3. **Cải thiện trải nghiệm người dùng**
   * Các ứng dụng như **chatbots** và **trợ lý ảo** sử dụng NLP để tương tác với người dùng một cách tự nhiên và hiệu quả, mang lại trải nghiệm liền mạch và thân thiện hơn trong việc hỗ trợ khách hàng hoặc thực hiện các tác vụ.
4. **Khả năng xử lý dữ liệu lớn**
   * NLP giúp xử lý và phân tích một lượng lớn dữ liệu không có cấu trúc (như văn bản trên mạng xã hội, báo cáo tài chính, nghiên cứu khoa học, v.v.), giúp trích xuất thông tin có giá trị từ các nguồn dữ liệu này.
5. **Ứng dụng đa dạng trong nhiều ngành**
   * NLP được ứng dụng trong nhiều lĩnh vực như **dịch máy**, **phân tích cảm xúc**, **phân loại văn bản**, **tìm kiếm thông tin**, **y tế**, **tài chính**, và **marketing**, mang lại nhiều giá trị cho các ngành công nghiệp khác nhau.

### **Nhược điểm của NLP**

1. **Khó khăn trong việc hiểu ngữ cảnh**
   * Một trong những vấn đề lớn trong NLP là khả năng hiểu ngữ cảnh và sự đa nghĩa của từ. Cùng một từ có thể có nhiều nghĩa khác nhau tùy vào cách sử dụng trong câu, và việc máy tính hiểu đúng trong mọi tình huống là một thách thức lớn.
2. **Khả năng xử lý các cấu trúc ngữ pháp phức tạp**
   * Ngôn ngữ tự nhiên có thể có cấu trúc rất phức tạp, và việc phân tích cú pháp của các câu dài hoặc không chính xác đôi khi gây ra sự hiểu sai hoặc kết quả không chính xác.
3. **Khó khăn trong việc nhận diện cảm xúc và ý định**
   * Mặc dù NLP có thể phân tích cảm xúc (ví dụ như xác định là tích cực hay tiêu cực), nhưng việc hiểu chính xác cảm xúc sâu sắc hơn (như hài hước, mỉa mai, hoặc các ngữ nghĩa ẩn) vẫn là một thách thức.
4. **Dữ liệu không đầy đủ hoặc thiếu chất lượng**
   * Để huấn luyện các mô hình NLP hiệu quả, cần phải có một lượng lớn dữ liệu văn bản chất lượng. Tuy nhiên, dữ liệu này không phải lúc nào cũng có sẵn hoặc không đủ đa dạng để bao quát tất cả các tình huống.
5. **Vấn đề bảo mật và quyền riêng tư**
   * Trong khi xử lý văn bản, đặc biệt là trong các ứng dụng như phân tích cảm xúc hoặc trích xuất thông tin, việc thu thập và xử lý dữ liệu nhạy cảm có thể dẫn đến các mối lo ngại về bảo mật và quyền riêng tư.
6. **Độ chính xác không cao trong một số tình huống**
   * Các mô hình NLP vẫn có thể mắc lỗi trong việc phân tích các câu văn phức tạp hoặc khi ngữ cảnh không rõ ràng. Điều này có thể dẫn đến các kết quả không chính xác trong những tình huống đòi hỏi sự hiểu biết sâu sắc về ngữ nghĩa.

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

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

Trong lĩnh vực Xử lý Ngôn ngữ Tự nhiên (NLP), có một số bài toán cơ bản mà các hệ thống NLP cần phải giải quyết. Dưới đây là các bài toán quan trọng và cơ bản trong NLP:

#### 1. **Dịch máy (Machine Translation)**

* **Mô tả:** Là bài toán chuyển đổi văn bản từ ngôn ngữ này sang ngôn ngữ khác mà không làm mất đi nghĩa của câu gốc.
* **Ví dụ:** Dịch Google Translate, Microsoft Translator, DeepL.

#### 2. **Phân tích cú pháp (Syntax Parsing)**

* **Mô tả:** Xác định cấu trúc cú pháp của câu, tức là cách các từ trong câu kết hợp với nhau để tạo ra một cấu trúc hợp lý.
* **Ví dụ:** Xác định chủ ngữ, vị ngữ, tân ngữ trong câu để hiểu quan hệ giữa các từ.

#### 3. **Nhận diện thực thể có tên (Named Entity Recognition - NER)**

* **Mô tả:** Xác định và phân loại các thực thể (entities) như tên người, tổ chức, địa điểm, thời gian, v.v., trong một đoạn văn bản.
* **Ví dụ:** Trong câu "Apple đã ra mắt iPhone 14 tại Cupertino vào tháng 9 năm 2024", "Apple" là tổ chức, "iPhone 14" là sản phẩm, và "Cupertino" là địa điểm.

#### 4. **Phân loại văn bản (Text Classification)**

* **Mô tả:** Phân loại văn bản vào các nhóm hoặc thể loại nhất định. Đây là bài toán phổ biến trong các ứng dụng như phân loại email (spam hoặc không spam), phân loại tin tức (chính trị, thể thao, văn hóa, v.v.).
* **Ví dụ:** Phân loại bài viết blog vào các thể loại như công nghệ, giải trí, thể thao.

#### 5. **Tóm tắt văn bản (Text Summarization)**

* **Mô tả:** Tạo ra một bản tóm tắt ngắn gọn của một văn bản dài, giữ lại những thông tin quan trọng nhất trong nội dung.
* **Ví dụ:** Tóm tắt các báo cáo tài chính dài thành những điểm chính dễ hiểu.

#### 6. **Phân tích cảm xúc (Sentiment Analysis)**

* **Mô tả:** Phân tích văn bản để xác định cảm xúc hoặc thái độ của người viết, như là tích cực, tiêu cực, hoặc trung tính.
* **Ví dụ:** Phân tích cảm xúc từ các bài đánh giá sản phẩm, bài đăng trên mạng xã hội hoặc nhận xét từ khách hàng.

#### 7. **Tạo văn bản tự động (Text Generation)**

* **Mô tả:** Tạo ra văn bản mới tự động dựa trên một số đầu vào hoặc chủ đề nhất định. Đây là bài toán chính của các mô hình ngôn ngữ như GPT.
* **Ví dụ:** Tạo bài viết, báo cáo, câu chuyện ngắn, hoặc thậm chí sáng tác thơ.

#### 8. **Tìm kiếm thông tin (Information Retrieval)**

* **Mô tả:** Tìm kiếm các thông tin hoặc tài liệu liên quan từ một cơ sở dữ liệu hoặc nguồn tài liệu lớn dựa trên một truy vấn của người dùng.
* **Ví dụ:** Công cụ tìm kiếm Google, hệ thống tìm kiếm trong các cơ sở dữ liệu khoa học.

#### 9. **Trả lời câu hỏi (Question Answering)**

* **Mô tả:** Trả lời các câu hỏi của người dùng dựa trên một đoạn văn bản hoặc cơ sở dữ liệu có sẵn.
* **Ví dụ:** Siri, Google Assistant, hoặc các hệ thống trả lời câu hỏi từ văn bản.

#### 10. **Kiểm tra ngữ pháp (Grammar Checking)**

* **Mô tả:** Kiểm tra và sửa lỗi ngữ pháp trong văn bản, bao gồm các lỗi chính tả, cú pháp, và ngữ pháp.
* **Ví dụ:** Các công cụ như Grammarly giúp người dùng phát hiện và sửa lỗi ngữ pháp trong văn bản.

#### 11. **Chuyển đổi văn bản thành giọng nói (Text-to-Speech - TTS)**

* **Mô tả:** Chuyển đổi văn bản thành âm thanh hoặc giọng nói. Đây là một trong những ứng dụng phổ biến trong các trợ lý ảo.
* **Ví dụ:** Các công nghệ TTS như Google TTS, Apple VoiceOver giúp người khiếm thị hoặc người không thể đọc văn bản dễ dàng nghe thông tin.

### **Kết luận:**

Các bài toán cơ bản trong NLP cung cấp những nền tảng quan trọng giúp phát triển các ứng dụng như chatbot, dịch máy, phân tích cảm xúc, tạo văn bản tự động, và nhiều hơn nữa. Những bài toán này ngày càng trở nên quan trọng trong việc xây dựng hệ thống AI thông minh có khả năng tương tác với con người thông qua ngôn ngữ tự nhiên.

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:**

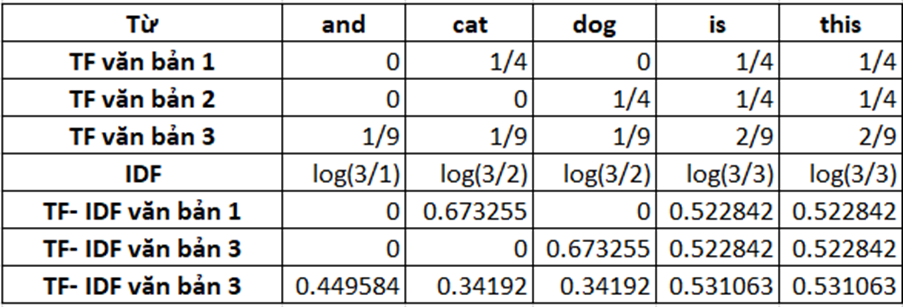
Khi đó:

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 1.1 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.

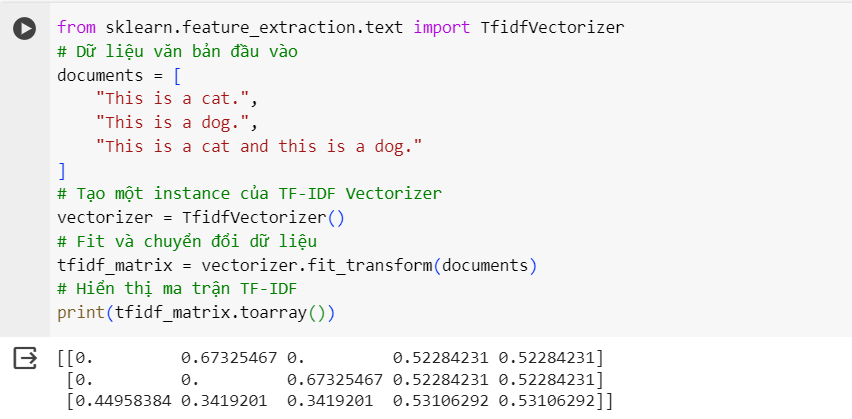
1. Tính toán IDF

* 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 1.2 Chương trình chạy demo mô hình TF – IDF

B. Kỹ thuật Consine Similarity

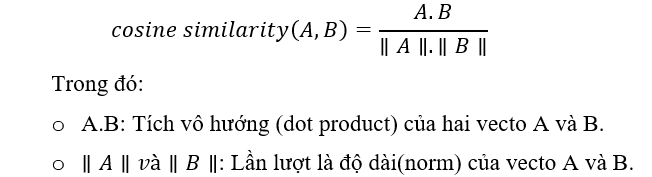
***1,*** ***Định nghĩa***

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:**

******

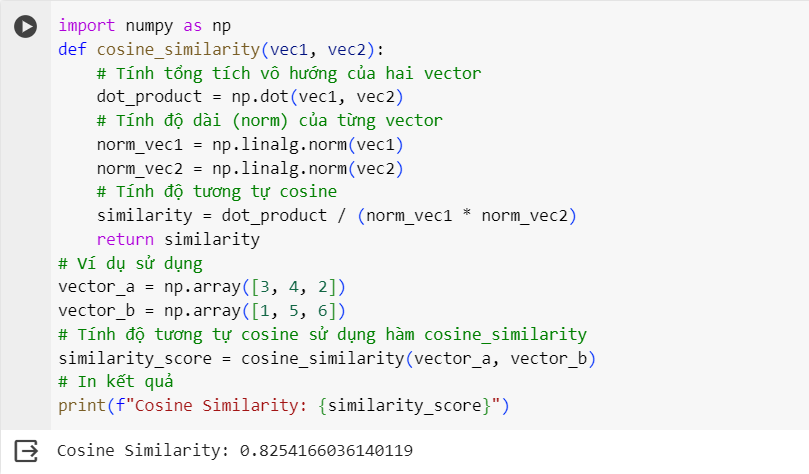
Để tính cosine similarity giữa hai vecto trong Python, chúng ta có thể sử dụng thư viện NumPy để thực hiện các phép toán đại số tuyến tính.

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 1.3 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."***

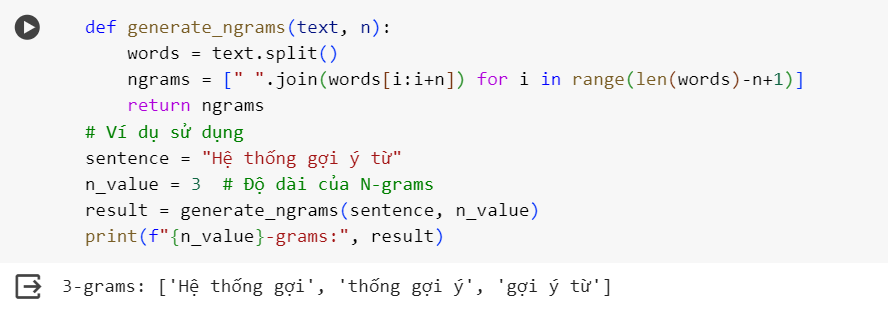
* Unigrams (1-gram): ["I", "love", "natural", "language", "processing"].
* Bigrams (2-gram): ["I love", "love natural", "natural language", "language processing"].
* Trigrams (3-gram): ["I love natural", "love natural language", "natural language processing].

***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 1.4 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.

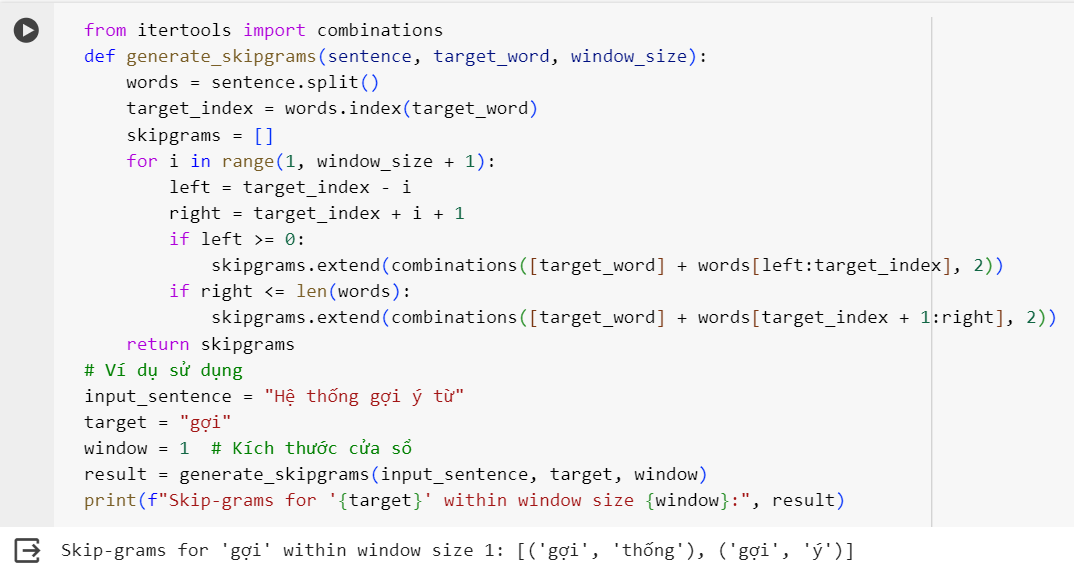
***3, Ý nghĩa và ứng dụng của Skip-grams***

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 1.5 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, được phát triển bởi **Guido van Rossum** và lần đầu tiên được phát hành 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, bao gồm lập trình hướng đối tượng (OOP), lập trình thủ tục, và lập trình hàm. Điều này giúp lập trình viên dễ dàng lựa chọn phương pháp phát triển phù hợp với yêu cầu của dự án.

Cú pháp của Python rất đơn giản và rõ ràng, làm cho việc viết mã trở nên dễ dàng và hiệu quả, ngay cả với những người mới bắt đầu học lập trình. Python còn nổi bật vì khả năng dễ dàng bảo trì mã nguồn, giúp giảm thiểu sự phức tạp trong các ứng dụng lớn và tạo ra mã dễ đọc, dễ hiểu.

Với cộng đồng người dùng rộng lớn và năng động, Python cung cấp một kho thư viện và công cụ phong phú, hỗ trợ các tác vụ từ phân tích dữ liệu, phát triển web, học máy, đến tự động hóa các quy trình. Các thư viện phổ biến như **NumPy**, **Pandas**, **TensorFlow**, và **PyTorch** giúp Python trở thành một trong những ngôn ngữ chính trong lĩnh vực **trí tuệ nhân tạo** và **học máy**, cung cấp các công cụ mạnh mẽ cho việc xử lý dữ liệu, xây dựng mô hình học máy và học sâu.

Ngoài ra, Python còn được sử dụng rộng rãi trong các lĩnh vực khác như **quản lý hệ thống**, **phát triển web**, **tự động hóa**, và **phát triển trò chơi**. Đi kèm với sự phát triển không ngừng của ngôn ngữ này, Python đã trở thành một trong những ngôn ngữ lập trình phổ biến và được ưa chuộng nhất trên toàn cầu nhờ vào tính linh hoạt, hiệu suất cao, và cộng đồng hỗ trợ mạnh mẽ.

### **1.2.2. Lịch sử phát triển**

Python đã được [Guido van Rossum](https://vi.wikipedia.org/w/index.php?title=Guido_van_Rossum&action=edit&redlink=1) tạo ra vào những năm [1980](https://vi.wikipedia.org/wiki/1980)[[39]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-venners-interview-pt-1-39) tại Trung tâm Toán học – Tin học ([Centrum Wiskunde & Informatica](https://vi.wikipedia.org/w/index.php?title=Centrum_Wiskunde_%26_Informatica&action=edit&redlink=1), CWI) ở [Hà Lan](https://vi.wikipedia.org/wiki/H%C3%A0_Lan) như là một ngôn ngữ kế tục ngôn ngữ [ABC](https://vi.wikipedia.org/w/index.php?title=ABC_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)&action=edit&redlink=1) – một ngôn ngữ được lấy cảm hứng từ [SETL](https://vi.wikipedia.org/w/index.php?title=SETL&action=edit&redlink=1),[[40]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-AutoNT-12-40) có khả năng [xử lí ngoại lệ](https://vi.wikipedia.org/w/index.php?title=X%E1%BB%AD_l%C3%AD_ngo%E1%BA%A1i_l%E1%BB%87&action=edit&redlink=1) và giao tiếp với hệ điều hành [Amoeba](https://vi.wikipedia.org/w/index.php?title=Amoeba_(h%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh)&action=edit&redlink=1).[[41]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-pythonfaq-1-41) Nó bắt đầu được triển khai vào tháng 12 năm [1989](https://vi.wikipedia.org/wiki/1989).[[42]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-timeline-of-python-42) Van Rossum đã tự mình gánh vác trách nhiệm cho dự án, với vai trò là nhà phát triển chính, cho đến ngày 12 tháng 7 năm [2018](https://vi.wikipedia.org/wiki/2018), khi ông thông báo rằng ông sẽ rời bỏ trách nhiệm của ông và cả danh hiệu "[Nhà độc tài nhân từ cho cuộc sống](https://vi.wikipedia.org/w/index.php?title=Nh%C3%A0_%C4%91%E1%BB%99c_t%C3%A0i_nh%C3%A2n_t%E1%BB%AB_cho_cu%E1%BB%99c_s%E1%BB%91ng&action=edit&redlink=1)" của Python, một danh hiệu mà cộng đồng Python đã trao tặng cho ông vì sự tận tụy lâu dài của ông với vai trò là người ra quyết định chính cho dự án.[[43]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-lj-bdfl-resignation-43) Vào tháng 1 năm [2019](https://vi.wikipedia.org/wiki/2019), các nhà phát triển phần lõi Python đã bầu ra một "Hội đồng chèo lái" gồm năm thành viên để dẫn dắt dự án.[[44]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-44)[[45]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-45)

Python 2.0 được ra mắt vào ngày 16 tháng 10 năm 2000, với nhiều tính năng mới mẻ, bao gồm một [bộ dọn rác](https://vi.wikipedia.org/wiki/B%E1%BB%99_d%E1%BB%8Dn_r%C3%A1c) phát hiện theo chu kỳ và khả năng hỗ trợ [Unicode](https://vi.wikipedia.org/wiki/Unicode).[[46]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-newin-2.0-46)

Python 3.0 được ra mắt vào ngày mùng 3 tháng 12 năm [2008](https://vi.wikipedia.org/wiki/2008). Đây là một phiên bản lớn của Python không [tương thích ngược](https://vi.wikipedia.org/w/index.php?title=T%C6%B0%C6%A1ng_th%C3%ADch_ng%C6%B0%E1%BB%A3c&action=edit&redlink=1) hoàn toàn.[[47]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-3.0-release-47) Nhiều tính năng lớn của nó đã được chuyển mã ngược (backport) về loạt phiên bản Python 2.6.x và 2.7.x.[[48]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-pep-3000-48) Các bản phát hành của Python 3 có đi kèm với công cụ 2to3, có tác dụng tự động hoá việc [dịch mã](https://vi.wikipedia.org/wiki/D%E1%BB%8Bch_m%C3%A3) Python 2 sang Python 3.[[49]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-49)

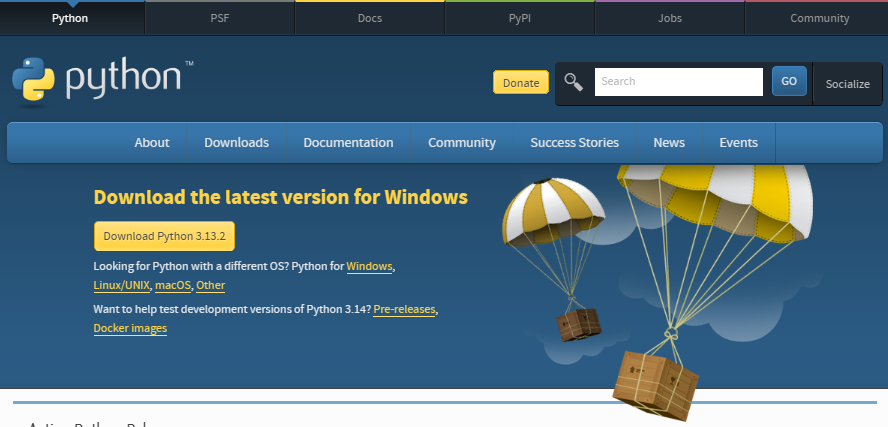
Python 3.9.2 và 3.8.8 được xúc tiến[[50]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-50) vì tất cả các phiên bản trước của Python (bao gồm cả 2.7)[[51]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-51) gặp một số vấn đề bảo mật, có thể dẫn đến [thực thị mã từ xa](https://vi.wikipedia.org/w/index.php?title=Th%E1%BB%B1c_th%E1%BB%8B_m%C3%A3_tu%E1%BB%B3_%C3%BD&action=edit&redlink=1)[[52]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-52) và ["đầu độc" bộ nhớ đệm](https://vi.wikipedia.org/w/index.php?title=%C4%90%E1%BA%A7u_%C4%91%E1%BB%99c_b%E1%BB%99_nh%E1%BB%9B_%C4%91%E1%BB%87m&action=edit&redlink=1).[[53]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-53)

Trong năm 2022, Python 3.10.4 và 3.9.12 được xúc tiến[[54]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-54) cùng với 3.8.13 và 3.7.13, nguyên nhân là do một vài vấn đề về bảo mật.[[55]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-55) Khi Python 3.9.13 được phát hành vào tháng Năm năm 2022, loạt phiên bản 3.9 (cùng với loạt 3.8 và 3.7) được thông báo rằng sẽ chỉ nhận được các bản vá bảo mật trong tương lai.[[56]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-56) Vào ngày 7 tháng Chín năm 2022, bốn bản cập nhật mới được phát hành do có khả năng xảy ra một cuộc [tấn công từ chối dịch vụ](https://vi.wikipedia.org/wiki/T%E1%BA%A5n_c%C3%B4ng_t%E1%BB%AB_ch%E1%BB%91i_d%E1%BB%8Bch_v%E1%BB%A5): 3.10.7, 3.9.14, 3.8.14 và 3.7.14.[[57]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-57)[[58]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-58)

Tính đến tháng 11 năm 2024, Python 3.13 là bản phát hành ổn định mới nhất. Một số thay đổi đáng chú ý từ bản 3.12 bao gồm các thay đổi về ngôn ngữ và thư viện chuẩn.[[59]](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)#cite_note-59)

### **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 1.6 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

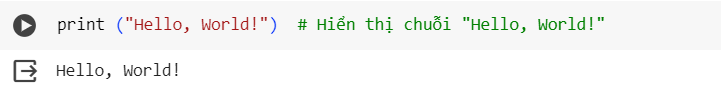
### **1.2.6. Python có lợi ích gì?**

Những lợi ích bạn nêu ra về Python rất chính xác và quan trọng. Dưới đây là một số điểm bổ sung để làm rõ hơn những lợi ích mà Python mang lại:

1. **Cú pháp rõ ràng**:
   * Python sử dụng cú pháp rất dễ hiểu, giúp lập trình viên không phải lo lắng nhiều về các chi tiết phức tạp như dấu ngoặc nhọn {} hay dấu chấm phẩy ; như trong các ngôn ngữ khác. Điều này giúp tập trung vào logic chương trình thay vì cú pháp, và làm giảm số lỗi khi viết mã.
2. **Hỗ trợ đa mô hình lập trình**:
   * **Lập trình hướng đối tượng (OOP)**: Python cho phép tạo các lớp (class) và đối tượng (object), giúp tăng tính tái sử dụng mã và cải thiện khả năng bảo trì phần mềm.
   * **Lập trình hàm (Functional programming)**: Python hỗ trợ các khái niệm của lập trình hàm như hàm ẩn danh (lambda), higher-order functions, và các tính năng như map, filter, reduce.
   * **Lập trình theo thủ tục**: Python cũng rất thích hợp cho lập trình theo thủ tục (procedural programming), giúp bạn viết các chương trình đơn giản và dễ hiểu.
3. **Cộng đồng lớn**:
   * Cộng đồng Python rất mạnh mẽ và luôn phát triển. Các diễn đàn như Stack Overflow, GitHub, và Reddit chứa rất nhiều bài viết và câu hỏi đáp để hỗ trợ lập trình viên khi gặp khó khăn.
   * Python có rất nhiều thư viện mã nguồn mở sẵn có cho hầu hết các ứng dụng, giúp lập trình viên giải quyết nhiều vấn đề phức tạp mà không phải viết lại mã từ đầu.
4. **Năng suất cao**:
   * Với cú pháp ngắn gọn, Python giúp giảm thiểu thời gian viết mã, làm cho quá trình phát triển phần mềm trở nên nhanh chóng và hiệu quả.
   * Các thư viện và framework phong phú như Django, Flask (cho phát triển web), Pandas, NumPy (cho khoa học dữ liệu), và TensorFlow, PyTorch (cho học máy) giúp lập trình viên tiết kiệm thời gian và công sức khi phải phát triển các tính năng phức tạp.
   * Python dễ dàng tích hợp với các ngôn ngữ khác như C, C++ để tối ưu hiệu suất khi cần thiết, giúp chương trình vừa nhanh chóng phát triển vừa có thể mở rộng.

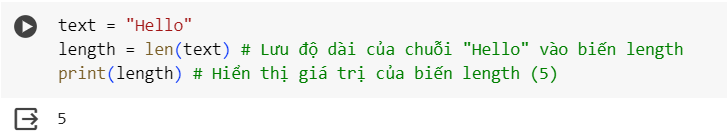
### **1.2.7. Các khái niệm và lệnh Python cơ bản**

* *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



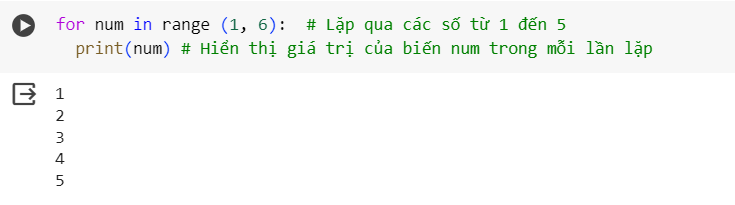
Hình 1.7 Chương trình chạy demo lệnh print()

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



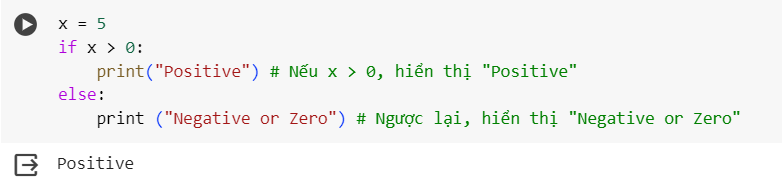
Hình 1.8 Chương trình chạy demo lệnh len()

* **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.



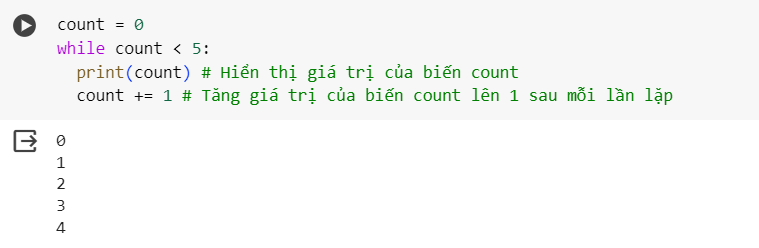
Hình 1.9 Chương trình chạy demo lệnh range()

* **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.



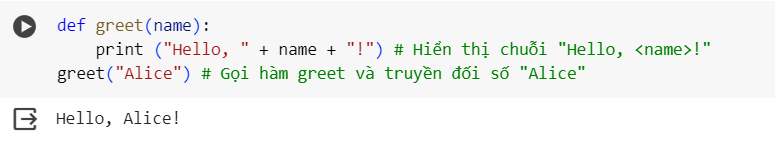
Hình 1.10 Chương trình chạy demo lệnh if - else()

* **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.



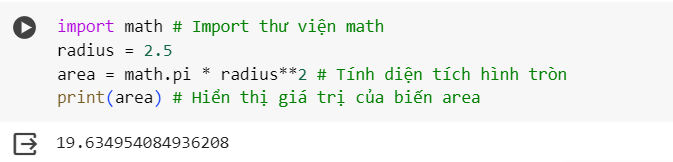
Hình 1.11 Chương trình chạy demo vòng lặp while()

* **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 1.12 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 1.13 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 1.14 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 2.1 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 đ9ánh 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"].

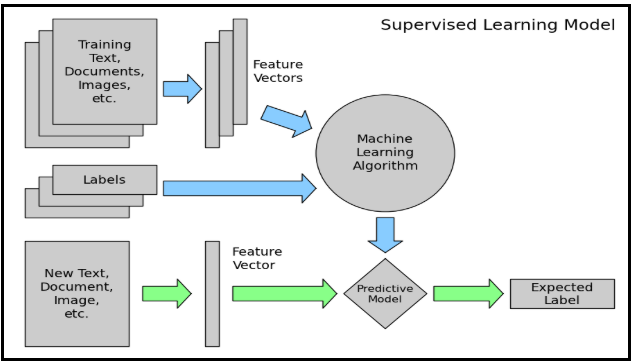
Những quá trình này đóng vai trò quan trọng trong việc hiểu và xử lý ngôn ngữ tự nhiên, cung cấp cơ sở dữ liệu cho nhiều ứng dụng như dịch thuật, tìm kiếm thông tin, phân tích ý kiến, và nhiều ứng dụng khác nằm trong lĩnh vực xử lý ngôn ngữ tự nhiên.

## 2.3 PHÂN LOẠI TỪ

**Phân loại** (Classification) là một bài toán trong học máy (machine learning), trong đó mục tiêu là gán nhãn cho dữ liệu đầu vào vào một trong các lớp (categories) hoặc nhóm (classes) đã được xác định trước. Mỗi dữ liệu đầu vào sẽ được phân loại vào một trong các lớp này dựa trên các đặc trưng của nó.

## I. Tổng quát

### 1. Mô hình khái quát cho bài toán phân lớp

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/review-text-classification.png)

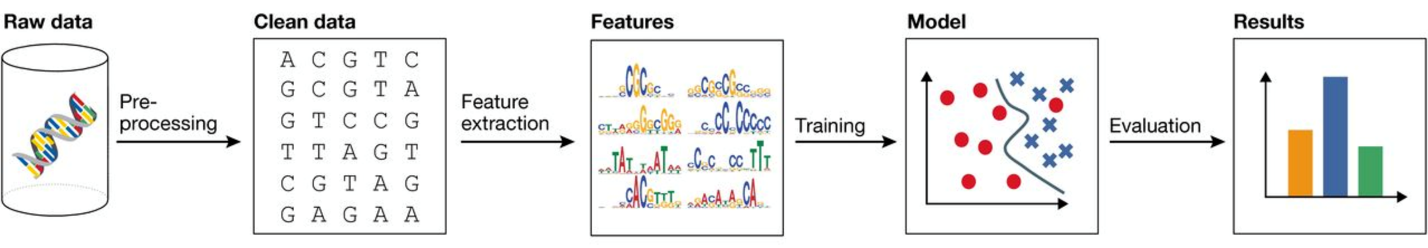
Ban đầu sẽ chia bộ dữ liệu (dataset) thành 2 phần: data train và data test. Tổng thể một bài toán phân lớp sẽ chia thành 2 công việc:

* Huấn luyện:  
  Dùng bộ **Training Text** (data train) rút trích thành các bộ **Features Vertor** đưa vào **Machine Learining Algorithm** từ đó thành một **Predictive Model** dùng để phân loại sau này, cuối cùng kết quả sẽ thu được **Expect Label** tên lớp cần phân loại.
* Hiện thực kết quả:  
  Dùng **Text** (data test) rút trích thành các bộ **Features Vertor** đưa vào **Predictive Model** sẽ nhận được **Expect Label** tên lớp cần phân loại.

Mô hình bài toán phân lớp khá đơn giản, bao gồm các thành phần:

* Training text: Đây là văn bản đầu vào thông qua đó mô hình supervised learning có thể học (learn) và dự đoán (predict) được phân lớp/ phân loại (categories/classes).
* Feature Vector: là một vertor chứa thông tin mô tả các đặc điểm của dữ liệu đầu vào.
* Labels: Đây là các danh mục/ lớp (categories/classes) được xác định trước mà mô hình sẽ dự đoán.
* ML Algo: Đây là thuật toán mà qua đó mô hình có thể xử lý phân loại văn bản (CNN, RNN, HAN,...).
* Predictive Model: Một mô hình đã được train/learn dựa trên bộ dữ liệu train và mô hình này có thể thực hiện dự đoán nhận biết được nhãn (categories/classes) nào khi nhập từ bộ dữ liệu test.

### 2. Quy trình tổng quan hiện thực bài toán phân loại văn bản

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/computational-biology.png)

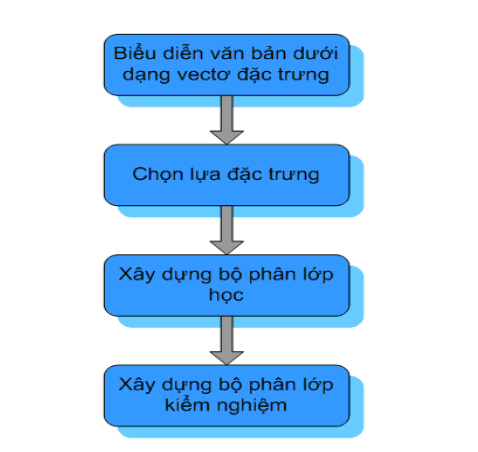
Luồng xử lý cơ bản:

**Crawler data** (cào dữ liệu) -> **text normalization** (chuẩn hóa dữ liệu) -> **data preprocessing** (tiền xử lý dữ liệu) -> **features** (trích xuất đặc trưng) -> **learn/train model** (chọn model machine learning và huấn luyện) -> **evaluation/results** (đánh giá kết quả).

Trong đó:

* **Crawler data** (cào dữ liệu): Là công đoạn chuẩn bị tập dataset (bộ dữ liệu để sử dụng) được lấy từ nhiều nguồn khác nhau như website. Ví dụ, lấy 3 triệu bài báo từ 5 trang web tin tức nổi tiếng nhất Việt Nam.
* **Text normalization** (chuẩn hóa dữ liệu): Công đoạn loại bỏ các thành phần không cần thiết từ dữ liệu mới crawler được có thể hiểu là làm sạch dữ liệu xóa đi dữ liệu rác cuối cùng nhận được đoạn văn bản chỉ có text. Ví dụ, xóa đi tag HTML, xóa link, xóa ký tự đặc biệt "\n \t &#64",...
* **Data preprocessing** (tiền xử lý dữ liệu): Chuyển dữ liệu/ văn bản nhận được ở giai đoạn trên thành dữ liệu đầu vào (data input) thích hợp cho đúng với mô hình (model machine learning) sử dụng phân loại văn bản. Ví dụ, các công việc cần thực hiện trước khi đưa vào thuật toán phân loại văn bản tiếng Việt như: tách từ, chuẩn hóa từ, loại bỏ stopwords, vertor hóa từ. Đây là công đoạn quan trọng trong bài toán phân loại văn bản.
* **Features** (trích xuất đặc trưng): Với bài toán phân loại trên thực tế, khi muốn phân loại cần phải dựa theo một đặc điểm nào đó như giới tính, hình dạng, kích thước dựa trên sự quan sát hoặc số liệu cụ thể. Trong bài toán phân loại cũng vậy, nhưng nó đòi hỏi việc phải tự động phát hiện ra các đặc điểm của đối tượng rồi mới thực hiện phân loại cho phù hợp. Ví dụ, phân loại hoa Hồng, phải phát hiện ra mỗi hoa đó có đặc điểm như thế nào xét cả về hình dạng, màu, kích thước, giống, mùi hương. Một đối tượng có rất nhiều đặc điểm, vậy dựa trên một hoặc nhiều đặc điểm nào để phân loại? Vì thế công đoạn này sẽ rúc trích hay lựa chọn bộ đặc điểm nào tối ưu nhất, dễ nhận dạng nhất, dễ phát hiện ra đối tượng đó nhất. Cuối cùng công đoạn này sẽ thu được một tập dữ liệu đã được trích xuất sau đó đưa vào thuật toán machine learning phân loại. Có 2 loại feature:
  + Feature Selection (chọn lựa đặc trưng): là chọn ra một tập đặc trưng con từ không gian đặc trưng gốc.
  + Feature Extraction (rút trích đặc trưng): là biến đổi (transform) không gian đặc trưng gốc thành một không gian đặc trưng nhỏ hơn để giảm số chiều đặc trưng. So với phương pháp chọn đặc trưng, rút trích không chỉ giảm số chiều mà còn thành công trong việc giải quyết vấn đề tính nhiều nghĩa (polysemy) và tính đồng nghĩa (synonym) của từ ở mức độ có thể chấp nhận.
* **Learn/train model** (chọn model machine learning và huấn luyện): Lựa chọn một thuật toán tối ưu nhất cho bài toán phân loại văn bản.
* **Evaluation/results** (đánh giá kết quả): Công đoạn cuối, đánh giá kết quả nhận được.

### 3. Quy trình cụ thể hiện thực bài toán phân loại văn bản

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/flow-class-part.png)

[*LuanVanDaiHoc\_2006\_CNTT\_DHKHTN-HCM\_Vu\_Nguyen\_protected.pdf*](https://github.com/duyvuleo/VNTC/blob/master/Report/LuanVanDaiHoc_2006_CNTT_DHKHTN-HCM_Vu_Nguyen_protected.pdf)

Hầu hết các phương pháp máy học áp dụng cho bài toán phân loại văn bản đều sử dụng cách biểu diễn văn bản dưới dạng véc tơ đặc trưng. Điểm khác biệt duy nhất chính là không gian đặc trưng được chọn lựa. Tuy nhiên ở đây ta thấy nảy sinh một vấn đề cơ bản: Số lượng từ xuất hiện trong văn bản sẽ rất lớn. Như vậy, mỗi véc tơ có thể có hàng ngàn đặc trưng, hay nói cách khác mỗi véc tơ sẽ có số chiều rất lớn. Do vậy các véc tơ sẽ không đồng nhất về kích thước.

Để giải quyết vấn đề thông thường chúng ta sẽ chọn lựa những đặc trưng được đánh giá là hữu ích, bỏ đi những đặc trưng không quan trọng. Đối với phân loại văn bản, quá trình này rất quan trọng bởi vì véc tơ văn bản có số chiều rất lớn (>>10000), trong đó số thành phần dư thừa cũng rất nhiều. Vì vậy các phương pháp chọn lựa đặc trưng rất hiệu quả trong việc giảm chiều của véc tơ đặc trưng văn bản, chiều của véc tơ văn bản sau khi được giảm chỉ còn lại khoảng 1000 đến 5000 mà không mất đi độ chính xác phân loại. (copy & paste)

## II. Phân tích bài toán phân loại văn bản trong tiếng Việt

Việc áp dụng phương pháp phân loại trong tiếng Việt còn khó khăn:

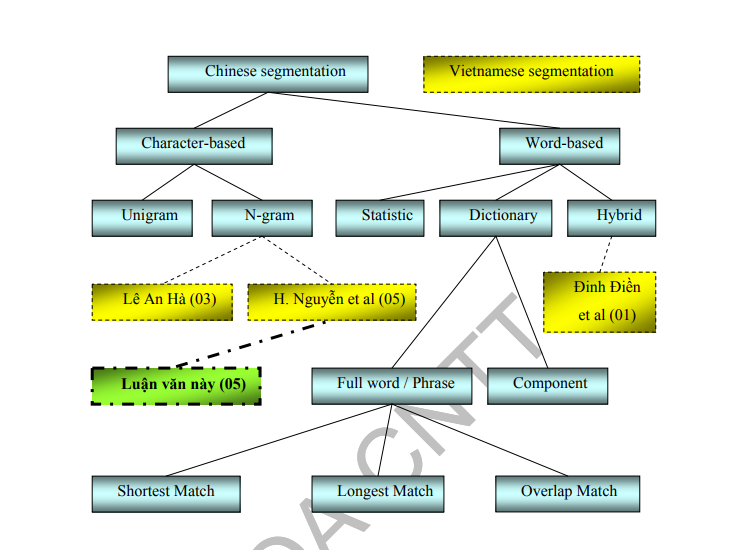
* Không có tập dữ liệu chuẩn cho việc phân loại.
* Chưa có thống nhất về font và dấu câu.
* Biểu diễn văn bản Tiếng Việt còn nhiều trở ngại do bị phụ thuộc nhiều vào phương pháp tách từ. Vì phương pháp này áp dụng đối với tiếng Việt không đạt hiệu quả cao như tiếng Anh. Vì vậy, để có thể áp dụng tốt các thuật toán phân loại hiệu quả cao trong ngôn ngữ tiếng Anh thì phải tìm ra một phương pháp tách từ tốt. Trong tiếng Anh đơn vị nhỏ nhất là "từ" trong khi tiếng Việt là "tiếng" và khoảng trắng ngăn cách không thể hiện rõ đâu là "từ" vì phải phụ thuộc vào ngữ cảnh. Vì vậy, trước khi phân loại phải phải tìm hiểu về cách tách từ trong tiếng Việt.

### Khái niệm chung

* **POS tagging** (gán nhãn từ loại): là việc phân loại các từ trong một câu (danh từ, trạng từ, tính từ hay động từ, v.v..), xem thêm: [nlp-part-of-speech-pos](https://chienuit.wordpress.com/2016/04/17/nlp-part-of-speech-pos/)
* **Word segmentation** (tách từ): tách một đoạn text (một chuỗi liên tiếp các ký tự) thành những từ (word hay token) riêng lẻ, xem thêm: [nlp-word-segmentation](https://chienuit.wordpress.com/2016/04/21/nlp-word-segmentation/), [Vietnamese Word Segmentation](http://vlsp.org.vn:8080/vitk/tok/index.xhtml)
* **Named Entity Recognition** (gán nhãn boundary (ranh giới) và type (thể loại)): phân loại text trong văn bản thành những loại xác định trước như là tên người, tổ chức, địa điểm, thời gian, số lượng, giá trị tiền tệ, phần trăm. “American Airlines[ORG]”, một đơn vị của “AMR Corp.[ORG]”, ngay lâp tức có những động thái phù hợp, phát ngôn viên “Tim Wagner[PER]” cho biết, xem thêm: [Nhận\_dạng\_thực\_thể\_có\_tên](https://vi.wikipedia.org/wiki/Nh%E1%BA%ADn_d%E1%BA%A1ng_th%E1%BB%B1c_th%E1%BB%83_c%C3%B3_t%C3%AAn), [named-entity-recognition](https://medium.com/@liennguyen_51419/17-1-named-entity-recognition-part-2-1-2287823bee3d)

### 1. Tách từ

Bối cảnh tách từ trong tiếng Việt và tiếng Hoa.

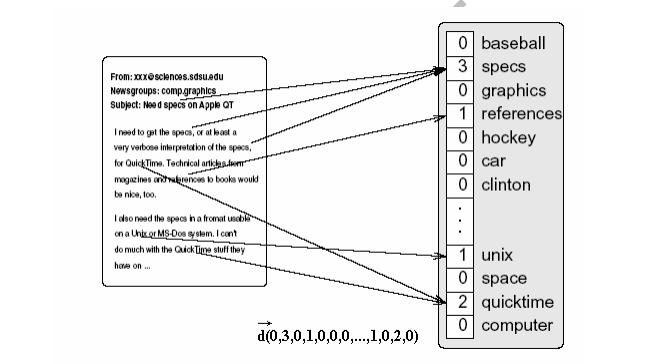
[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/character.png)

[*phan-loai-tintucbaodientu - 3.2.1*](https://www.slideshare.net/duyvong3/phan-loai-tintucbaodientu-40099422)

Các phương pháp tiếp cận tách từ:

* Dựa trên từ (Word-based approaches)
* Dựa trên ký tự (Character-based approaches)

### 2. Biểu diễn văn bản dưới dạng vertor đặc trưng

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/slipt-words.png)

<https://www.slideshare.net/duyvong3/phan-loai-tintucbaodientu-40099422> 2.2.1

Trong xử lý ngôn ngữ tự nhiên, việc biểu diễn một từ thành một vector đóng một vai trò cực kỳ quan trọng. Nó lợi ích rất nhiều trong việc thể hiện sự tương đồng, đối lập về ngữ nghĩa giữa các từ, giúp mô hình hóa vector cho 1 câu hay đoạn văn, tìm các câu có nghĩa tương đồng. Bước đầu tiên của hầu hết thuật toán phân loại văn bản là chuyển văn bản thành một dạng mô tả khác sao cho phù hợp với thuật toán cần sử dụng. Hầu hết các thuật toán phân loại đều sử dụng cách biểu diễn văn bản sử dụng vertor đặc trưng.

**Word embbding là gì?** Mở đầu như chúng ta đã biết máy tính được cấu tạo từ những con số, do đó nó chỉ có thể đọc được dữ liệu số mà thôi. Trong natural language processing thì để xử lý dữ liệu text chúng ta cũng phải chuyển dữ liệu từ text sang numeric, tức là đưa nó vào một không gian mới người ta thường gọi là embbding. Trước đây người ta mã hóa theo kiểu one hot encoding tức là tạo một vocabualary cho dữ liệu và mã hóa các word trong document thành những vectoc, nếu word đó có trong document thì mã hóa là 1 còn không có sẽ là 0. Kết quả tạo ra một sparse matrix, tức là matrix hầu hết là 0.Các mã hóa này có nhiều nhược điểm đó là thứ nhất là số chiều của nó rất lớn (NxM, N là số document còn M là số vocabulary), thứ 2 các word không có quan hệ với nhau. Điều đó dẫn đến người ta nghĩ ra một model mới có tên là Word embbding,

### 3. Cách đánh giá thuật toán/ model

Có nhiều cách để đánh giá mô hình:

#### 3.1. Accuracy

Chia thành hai phần **training** và **testing** áp dụng một mô hình để train từ tập dữ liệu **training**. Tiếp theo sử dụng mô hình đó dự đoán trên tập **testing** và cuối cùng là tìm ra tỉ lệ số dữ liệu dự đoán đúng / tổng số dữ liệu testing.

#### 3.2. Confusion matrix

Cách tính sử dụng accuracy như ở trên chỉ cho biết được bao nhiêu phần trăm lượng dữ liệu được phân loại đúng mà không chỉ ra được cụ thể mỗi loại được phân loại như thế nào, lớp nào được phân loại đúng nhiều nhất, và dữ liệu thuộc lớp nào thường bị phân loại nhầm vào lớp khác. Để có thể đánh giá được các giá trị này, chúng ta sử dụng một ma trận được gọi là confusion matrix.

Confusion matrix thể hiện có bao nhiêu điểm dữ liệu thực sự thuộc (actual) vào một class, và được dự đoán (predict) là rơi vào một class. Có tổng cộng 10 điểm dữ liệu và ma trận 3x3.

Total: 10 | Predicted | Predicted | Predicted |

| as: 0 | as: 1 | as: 2 |

-----------|-----------|-----------|-----------|---

True: 0 | 2 | 1 | 1 | 4

-----------|-----------|-----------|-----------|---

True: 1 | 1 | 2 | 0 | 3

-----------|-----------|-----------|-----------|---

True: 2 | 0 | 1 | 2 | 3

-----------|-----------|-----------|-----------|---

Ma trận thu được được gọi là confusion matrix. Nó là một ma trận vuông với **kích thước mỗi chiều bằng số lượng lớp dữ liệu**. Giá trị tại hàng thứ i, cột thứ j là số lượng điểm **lẽ ra thuộc vào class i nhưng lại được dự đoán là thuộc vào class j**. Như vậy, nhìn vào hàng thứ nhất (0), ta có thể thấy được rằng trong số bốn điểm thực sự thuộc lớp 0, chỉ có hai điểm được phân loại đúng, hai điểm còn lại bị phân loại nhầm vào lớp 1 và lớp 2.

Có thể suy ra ngay rằng tổng các phần tử trong toàn ma trận này chính là số điểm trong tập kiểm thử. Các phần tử trên đường chéo của ma trận là số điểm được phân loại đúng của mỗi lớp dữ liệu.

Suy ra **accuracy** chính bằng tổng các phần tử trên đường chéo chia cho tổng các phần tử của toàn ma trận.

Xem thêm: [evaluation-confusion-matrix](https://machinelearningcoban.com/2017/08/31/evaluation/#-confusion-matrix)

#### 3.3. Precision, Recall và F1-Score

Trong những bài toán này, người ta thường định nghĩa lớp dữ liệu quan trọng hơn cần được xác định đúng là lớp Positive (P-dương tính), lớp còn lại được gọi là Negative (N-âm tính). Ta định nghĩa True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN) dựa trên confusion matrix chưa chuẩn hoá như sau:

| Predicted | Predicted |

| as Positive | as Negative |

------------------|---------------------|---------------------|

Actual: Positive | True Positive (TP) | False Negative (FN) |

------------------|---------------------|---------------------|

Actual: Negative | False Positive (FP) | True Negative (TN) |

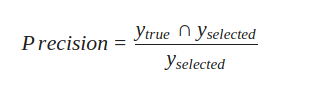
------------------|---------------------|---------------------|

Cách tính Precision và Recall.

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/pr.png)

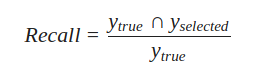
Precision cao đồng nghĩa với việc độ chính xác của các điểm tìm được là cao. Recall cao đồng nghĩa với việc True Positive Rate cao, tức tỉ lệ bỏ sót các điểm thực sự positive là thấp.

##### a. Precision (tỷ lệ chính xác) - bao nhiêu cái đúng được lấy ra

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/percision.png)

Xem xét trên tập dữ liệu kiểm tra (data-test) xem có **bao nhiêu dữ liệu được mô hình dự đoán đúng**. Tức là, số phát hiện đúng chia cho số đem đi kiểm thử. Đây chính là chỉ số accuracy - độ chính xác như bên trên. Giá trị càng cao, càng tốt.

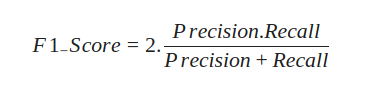
##### b. Recall (tỷ lệ tái hiện) - bao nhiêu cái được lấy ra là đúng

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/recall.png)

Recall được định nghĩa là tỉ lệ số điểm **true positive** trong số những điểm thực sự (actual) là **positive** (TP + FN).

Thể hiện tỉ lệ dự đoán chính xác của một dữ liệu.

##### c. F1-Score - trung bình điều hòa (harmonic mean)

[](https://github.com/huynhminhtan/thuattoanthongminh/blob/master/assets/f1-score.png)

Đây được gọi là một trung bình điều hòa (harmonic mean) của các tiêu chí Precision và Recall. Nó có xu hướng lấy giá trị gần với giá trị nào nhỏ hơn giữa 2 giá trị Precision và Recall và đồng thời nó có giá trị lớn nếu cả 2 giá trị Precision và Recall đều lớn. Chính vì thế F1-Score thể hiện được một cách khách quan hơn performance của một mô hình học máy.

## 2.4. 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.

## 2.5. 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.

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

#### 2.5.2.1. 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 2.2 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.

#### 2.5.2.2 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"*

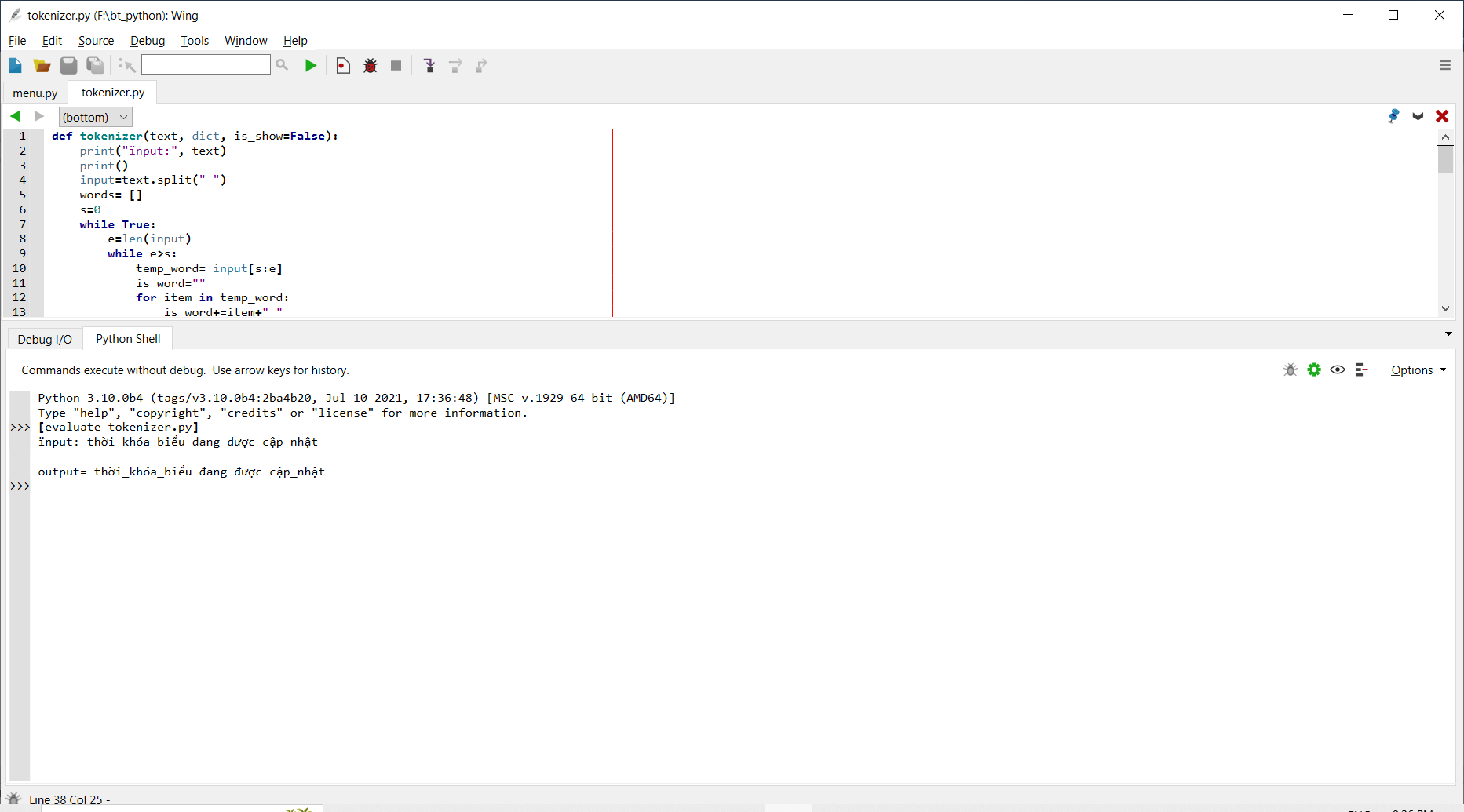
*#từ điển*

*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.

#### 2.5.2.3 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:

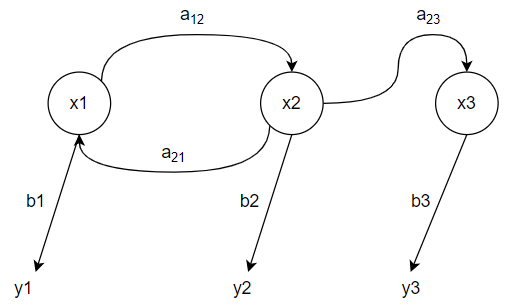
- Kiểm tra kết quả của mô hình trên dữ liệu kiểm tra để đảm bảo độ chính xác.

- 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.

2.5.2.4 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:

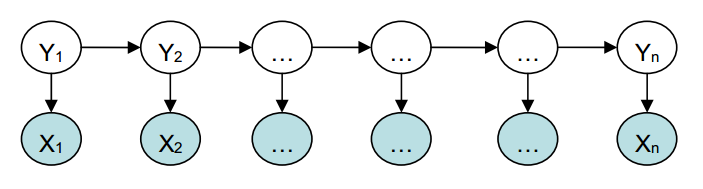
xi: các trạng thái

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:



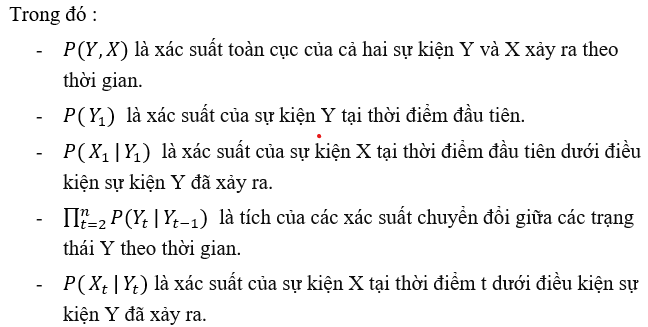
Hình 2.3 Biểu đồ chuyển tiếp trạng thái của mô hình Markov ẩn

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.

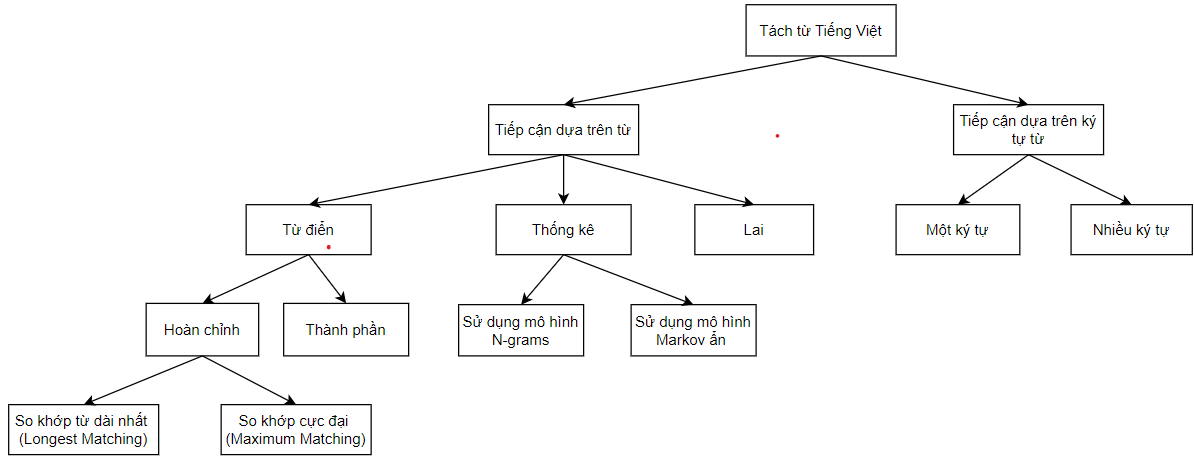
* Trạng thái hiện tại sẽ phụ thuộc vào trạng thái ngay trước đó với giả thiết là dữ liệu quan sát tại thời điểm t chỉ phụ thuộc vào trạng thái t. Ta có thể tính P(Y, X) theo công thức:



* 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 Hướng tiếp cận bài toán tách từ**



Hình 2.1 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.

### **2.5.1 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 2.3 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:

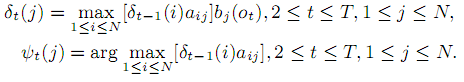


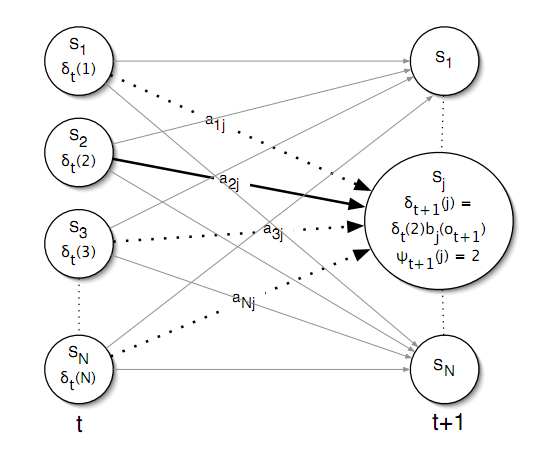
Thuật toán Viterbi được thực hiện như sau:

1. Khởi tạo:

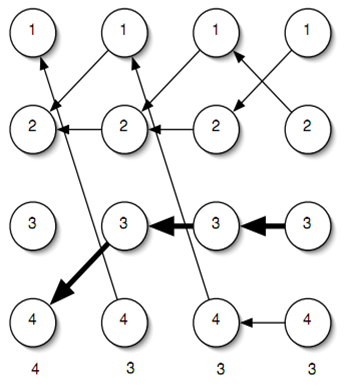


1. Đệ quy:



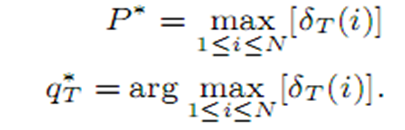


Hình 2.4 Bước đệ quy của thuật toán Viterbi

****

Hình 2.5 Bước quay lui của thuật toán Viterbi

1. Kết thúc:



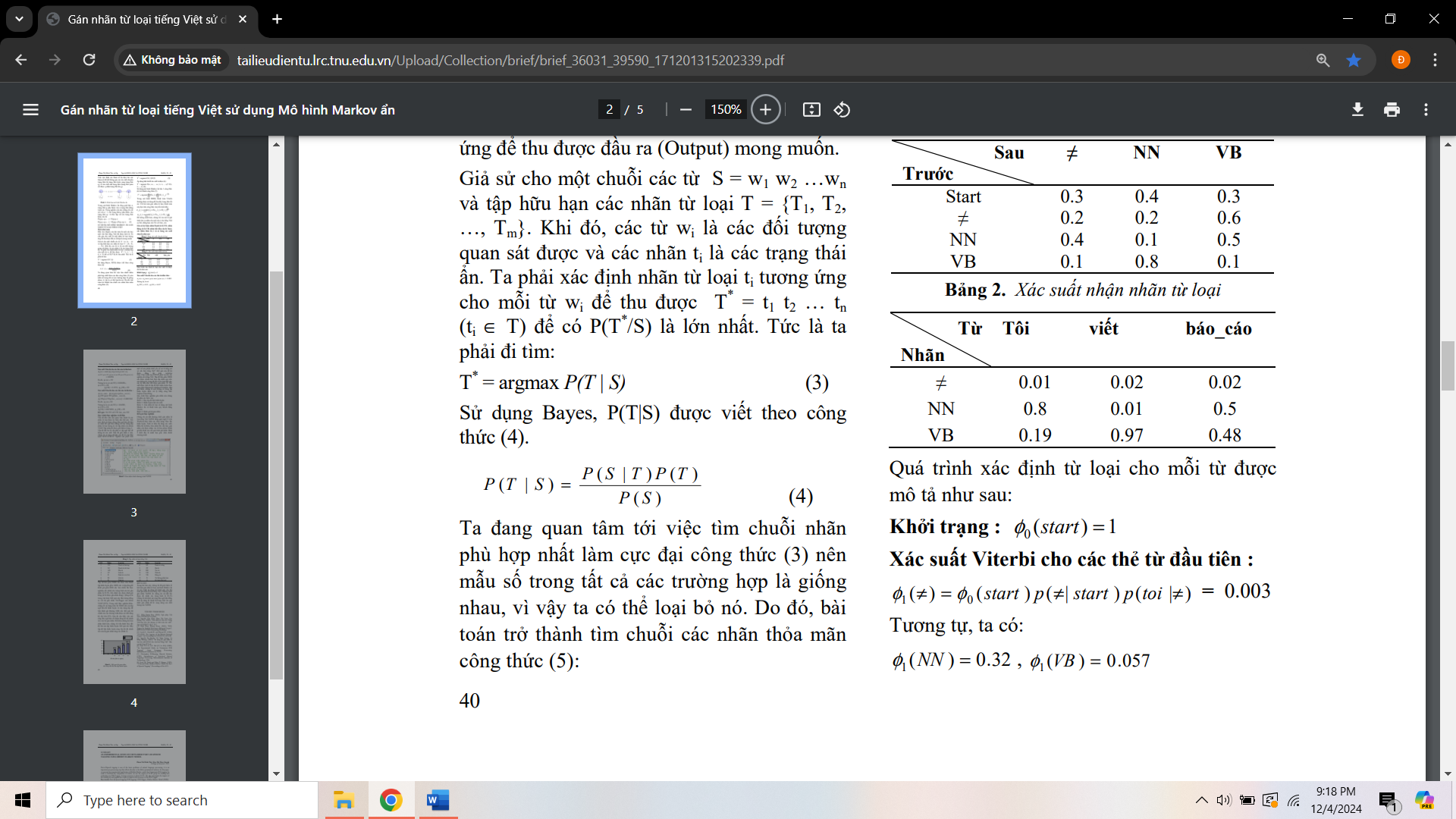
1. Quay lui trình tự trạng thái tối ưu:



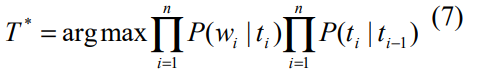
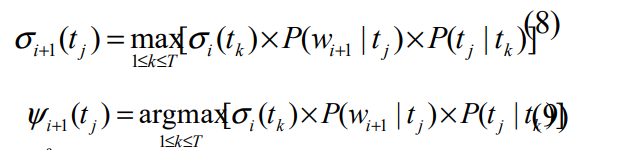
**c, Mô hình markov ẩn (hidden markov model – hmm) kết hợp thuật toán Viterbi trong gán nhãn từ loại**

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 2.4 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 2.5 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**

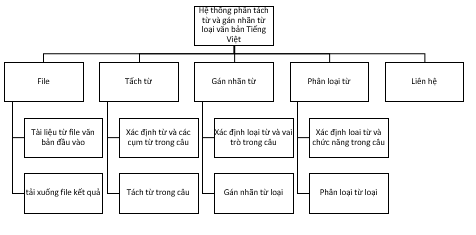
Thông qua các yêu cầu nghiệp vụ, phần mềm có các chức năng sau:

* 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.

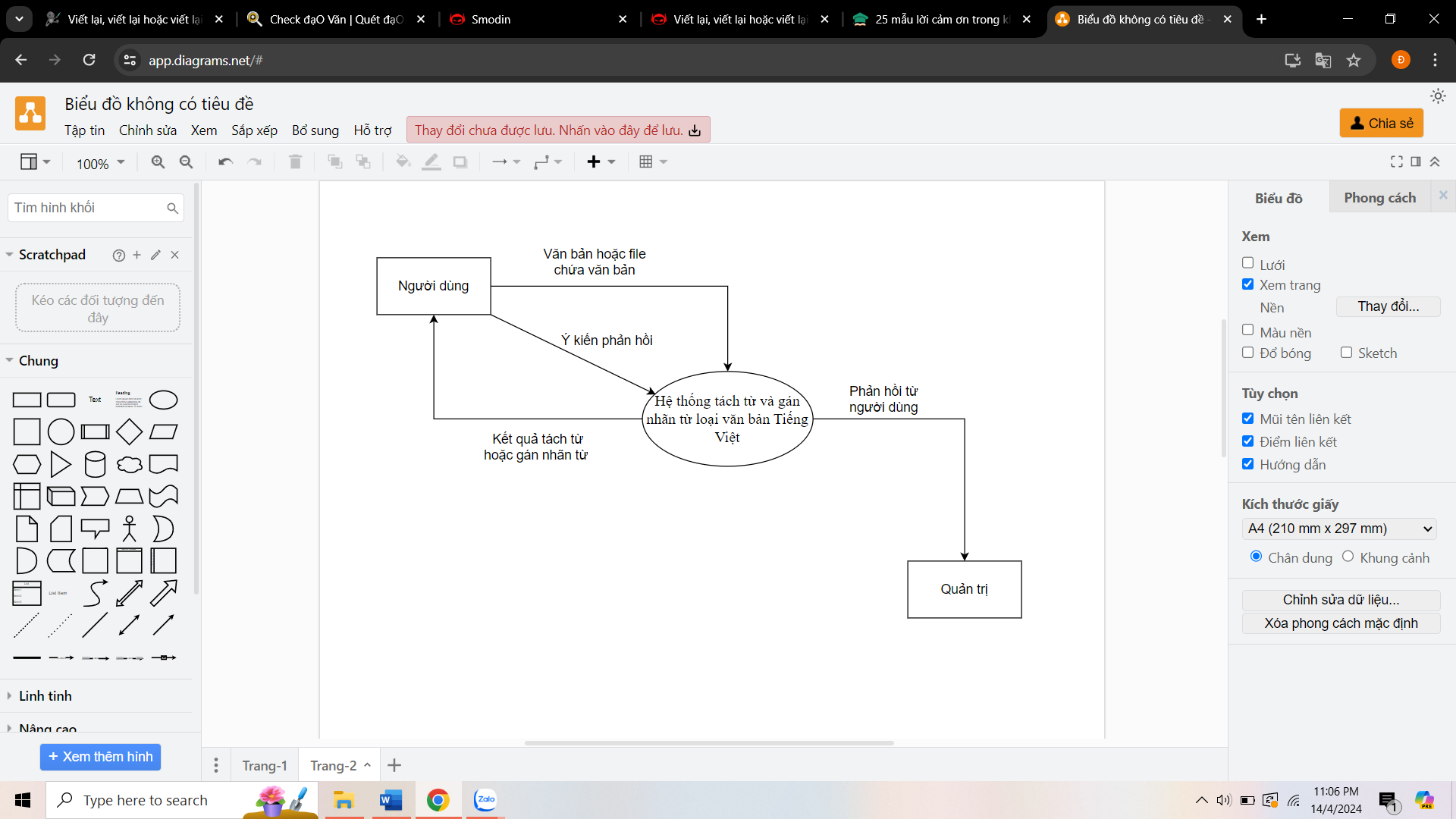
**3.1.3 Nội dung công việc nghiên cứu**

a) Biểu đồ phân cấp chức năng

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