**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**LÝ QUỐC ANH**

**Đề tài:**

**CHƯƠNG TRÌNH PYTHON TRÍCH DẪN**

**Ý CHÍNH TRONG VĂN BẢN**

**(CÔNG NGHỆ THÔNG TIN)**

**CÁN BỘ HƯỚNG DẪN:**

**TS. Phạm Văn Nam**

**Nha Trang - 2023**

Mục lục

[**CHƯƠNG 1: MỞ ĐẦU** 3](#_Toc151256867)

[**1.1** **Lý do chọn đề tài:** 3](#_Toc151256868)

[**1.2** **Mục tiêu nghiên cứu:** 3](#_Toc151256869)

[**1.3** **Phạm vi nghiên cứu:** 3](#_Toc151256870)

[**1.4** **Kết quả đoạt được:** 3](#_Toc151256871)

[**CHƯƠNG 2: TỔNG QUAN** 4](#_Toc151256872)

[**2.1 Cơ sở lý luận và thực tiễn** 4](#_Toc151256873)

[**2.1.1 Cơ sở lý luận** 4](#_Toc151256874)

[**2.1.2 Thực tiễn** 4](#_Toc151256875)

[**2.2 Các thành phần chính tạo nên chương trình** 5](#_Toc151256876)

[**2.2.1 Các thư viện sử dụng:** 5](#_Toc151256877)

[**2.2.2 Cấu trúc chính của chương trình** 5](#_Toc151256878)

[**CHƯƠNG 3: KẾT LUẬN** 11](#_Toc151256879)

[Danh sách tài liệu tham khảo: 12](#_Toc151256880)

# **CHƯƠNG 1: MỞ ĐẦU**

* 1. **Lý do chọn đề tài:**
* Python là một ngôn ngữ lập trình mạnh mẽ, linh hoạt và dễ học, được sử dụng rộng rãi trong nhiều lĩnh vực, bao gồm xử lý ngôn ngữ tự nhiên (NLP).
* Việc trích dẫn ý chính từ văn bản là một nhiệm vụ quan trọng trong NLP, có thể ứng dụng trong nhiều lĩnh vực như tóm tắt văn bản, phân loại văn bản, phân tích cảm xúc, v.v.
* Đề tài này giúp hiểu rõ hơn về cách Python có thể được sử dụng để giải quyết các vấn đề thực tế trong NLP.
  1. **Mục tiêu nghiên cứu:**
* Hiểu rõ về cách Python xử lý văn bản và trích dẫn ý chính từ văn bản.
* Xây dựng một chương trình Python có thể trích dẫn ý chính từ văn bản một cách chính xác và hiệu quả.
  1. **Phạm vi nghiên cứu:**

- Nghiên cứu về ngôn ngữ lập trình Python và các thư viện liên quan đến xử lý ngôn ngữ tự nhiên như NLTK, SpaCy, v.v.

- Nghiên cứu về các phương pháp trích dẫn ý chính từ văn bản.

* 1. **Kết quả đoạt được:**
* Một chương trình Python hoạt động hiệu quả, có thể trích dẫn ý chính từ văn bản.
* Hiểu rõ hơn về cách Python được sử dụng trong xử lý ngôn ngữ tự nhiên và cách trích dẫn ý chính từ văn bản.

# **CHƯƠNG 2: TỔNG QUAN**

## **2.1 Cơ sở lý luận và thực tiễn**

### **2.1.1 Cơ sở lý luận**

**-** Trích dẫn ý chính trong văn bản là một nhiệm vụ quan trọng trong xử lý ngôn ngữ tự nhiên (NLP), có thể ứng dụng trong nhiều lĩnh vực như tóm tắt văn bản, phân loại văn bản, phân tích cảm xúc, v.v. Trích dẫn ý chính trong văn bản là việc lựa chọn một số câu hoặc đoạn văn trong văn bản gốc để thể hiện nội dung chính của văn bản đó. Có nhiều phương pháp để trích dẫn ý chính trong văn bản, như dựa trên tần suất xuất hiện của các từ, dựa trên đồ thị, dựa trên học sâu, v.v

- Python là một ngôn ngữ lập trình mạnh mẽ, linh hoạt và dễ học, được sử dụng rộng rãi trong nhiều lĩnh vực, bao gồm NLP. Python cung cấp nhiều thư viện và công cụ hỗ trợ cho việc xử lý văn bản và trích dẫn ý chính trong văn bản, như NLTK, SpaCy, Gensim, Sumy, Bert-Extractive-Summarizer, v.v34 . Bằng cách sử dụng các thư viện và công cụ này, chúng ta có thể xây dựng một chương trình Python có thể trích dẫn ý chính từ văn bản một cách chính xác và hiệu quả.

### **2.1.2 Thực tiễn**

**-** Mục tiêu của đề tài này là xây dựng một chương trình Python có thể trích dẫn ý chính từ văn bản bằng cách sử dụng các thư viện và công cụ có sẵn. Đề tài này gồm các nội dung chính sau:

+ Giới thiệu về trích dẫn ý chính trong văn bản, các phương pháp và ứng dụng của nó.

+ Giới thiệu về ngôn ngữ lập trình Python và các thư viện liên quan đến xử lý ngôn ngữ tự nhiên và trích dẫn ý chính trong văn bản.

+ Xây dựng một chương trình Python có thể trích dẫn ý chính từ văn bản bằng cách sử dụng các thư viện và công cụ đã giới thiệu.

+Kiểm tra và đánh giá hiệu suất của chương trình Python trên các văn bản khác nhau.

## **2.2 Các thành phần chính tạo nên chương trình**

### **2.2.1 Các thư viện sử dụng:**

**-** tkinter: Sử dụng để xây dựng giao diện người dùng đồ họa (GUI).

- scrolledtext: Cung cấp một widget văn bản có khả năng cuộn trong tkinter.

- Spacy: Thư viện xử lý ngôn ngữ tự nhiên, sử dụng để tải mô hình tiếng Anh cho việc trích xuất thông tin ngôn ngữ.

- PIL (Pillow): Dùng để xử lý hình ảnh

- customtkinter (ctk): Một module tùy chỉnh, chứa một số thành phần giao diện người dùng được tối ưu hóa hoặc được tùy chỉnh.

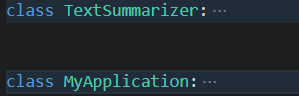
- googletrans: Sử dụng để dịch văn bản sử dụng dịch vụ Google Translate.

- nltk: Thư viện xử lý ngôn ngữ tự nhiên, cung cấp các công cụ như tách từ (tokenization), tìm từ dừng (stopwords), và tính tần suất xuất hiện của từ ( frequency distribution).

- collections, string: Các module Python cung cấp cấu trúc dữ liệu và hằng số cho việc xử lý văn bản và tính toán.

### **2.2.2 Cấu trúc chính của chương trình**

Chương trình sử dụng 2 phần chính làm cấu trúc đó cũng chính là 2 class chính được sử dụng để tạo nên chương trình là class :



Đầu tiên chương trình tải mô hình tiếng anh của spaCy:



Mô hình này chứa các thành phần như bộ phân tích cú pháp (parser), bộ nhận dạng từ loại (tagger), bộ nhận dạng thực thể có tên (ner), v.v. Chương trình có thể sử dụng mô hình này để xử lý văn bản tiếng Anh và trích xuất các thông tin ngôn ngữ như từ loại, thực thể có tên, hoặc đồng tham chiếu.

**2.2.2.1 Class TextSummarizer:**

**class** **TextSummarizer**:

**def** **\_\_init\_\_**(self, min\_cut=**0.1**, max\_cut=**0.9**):

# Khởi tạo các thông số cho TextSummarizer

self.\_min\_cut = min\_cut

self.\_max\_cut = max\_cut

self.\_stopwords = set(stopwords.words('english') + list(punctuation))

- khởi tạo một đối tượng TextSummarizer với hai thông số min\_cut và max\_cut. Đây là hai ngưỡng để lọc ra những từ có tần suất quá cao hoặc quá thấp trong văn bản. Khởi tạo một tập hợp các từ dừng và dấu câu để bỏ qua khi xử lý văn bản.

**def** **\_compute\_frequencies**(self, word\_sent):

# Tính tần suất xuất hiện của các từ

freq = FreqDist(word\_sent)

m = float(max(freq.values()))

**for** word **in** list(freq.keys()):

freq[word] = freq[word]/m

**if** freq[word] >= self.\_max\_cut **or** freq[word] <= self.\_min\_cut:

**del** freq[word]

**return** freq

**def** **summarize**(self, text, n):

# Rút tóm tắt văn bản

sents = sent\_tokenize(text)

n = min(n, len(sents)) # Đảm bảo không vượt quá số lượng câu

**assert** n <= len(sents)

word\_sent = [word **for** word **in** word\_tokenize(text.lower()) **if** word **not** **in** self.\_stopwords]

self.\_freq = self.\_compute\_frequencies(word\_sent)

ranking = defaultdict(int)

**for** i, sent **in** enumerate(sents):

**for** word **in** word\_tokenize(sent.lower()):

**if** word **in** self.\_freq:

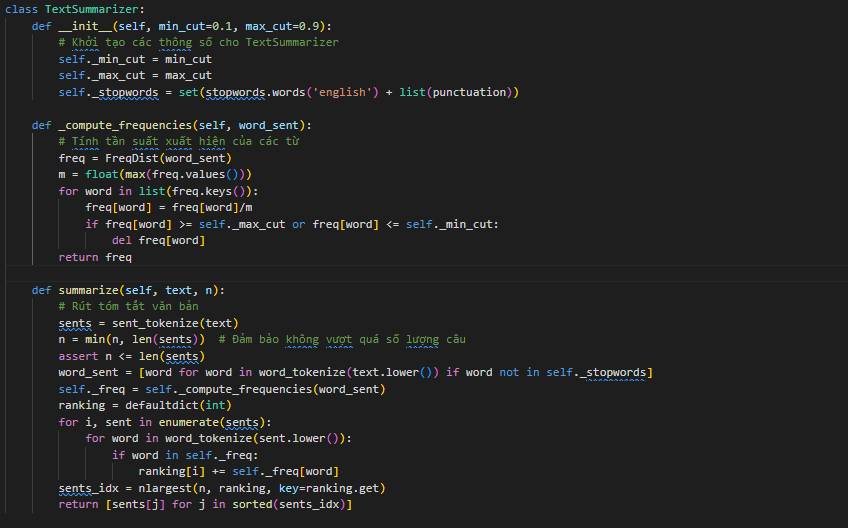
ranking[i] += self.\_freq[word]

sents\_idx = nlargest(n, ranking, key=ranking.get)

**return** [sents[j] **for** j **in** sorted(sents\_idx)]

* Chương trình gọi phương thức summarize của đối tượng TextSummarizer với hai tham số text và n. Text là văn bản cần rút gọn, n là số lượng câu mong muốn trong kết quả rút gọn.
* Chương trình tách văn bản thành các câu bằng hàm sent\_tokenize của thư viện nltk. Chương Trình đảm bảo rằng n không vượt quá số lượng câu trong văn bản.
* Chương trình tách văn bản thành các từ bằng hàm word\_tokenize của thư viện nltk. Bỏ qua những từ nằm trong tập hợp các từ dừng và dấu câu mà đã khởi tạo trước đó.
* Chương trình gọi phương thức \_compute\_frequencies của đối tượng TextSummarizer với tham số word\_sent là danh sách các từ đã tách. Phương thức này sẽ trả về một đối tượng FreqDist, là một cấu trúc dữ liệu để lưu trữ tần suất xuất hiện của các từ.
* Chương trình duyệt qua các từ trong đối tượng FreqDist và chia tần suất của chúng cho giá trị tần suất lớn nhất. Chương trình cũng loại bỏ những từ có tần suất lớn hơn max\_cut hoặc nhỏ hơn min\_cut. Chương trình lưu trữ đối tượng FreqDist đã được chuẩn hóa này vào thuộc tính \_freq của đối tượng TextSummarizer.
* Chương trình tạo một đối tượng defaultdict để lưu trữ điểm số của các câu trong văn bản. Bạn duyệt qua các câu và các từ trong mỗi câu. Nếu từ đó có trong thuộc tính \_freq của đối tượng TextSummarizer, chương trình cũng cộng điểm số của từ đó vào điểm số của câu đó.
* Chương trình sử dụng hàm nlargest của module heapq để lấy ra n câu có điểm số cao nhất trong đối tượng defaultdict. Trả về danh sách các câu này theo thứ tự xuất hiện trong văn bản.

Đó chính là các bước mà chương trình đã thực hiện trong Class TextSummarizer để thực hiện rút ra được ý chính trong văn bản



**ẢNH VỀ CLASS TEXT SUMMARIZER**

**2.2.2.1 Class MyApplication:**

Class MyApplication là lớp chuyên để sử lý các chức năng trong phần mềm cũng như tạo nên một giao diện người dùng đơn giản mà ai cũng có thể thao tác sử dụng gồm các chức năng như:

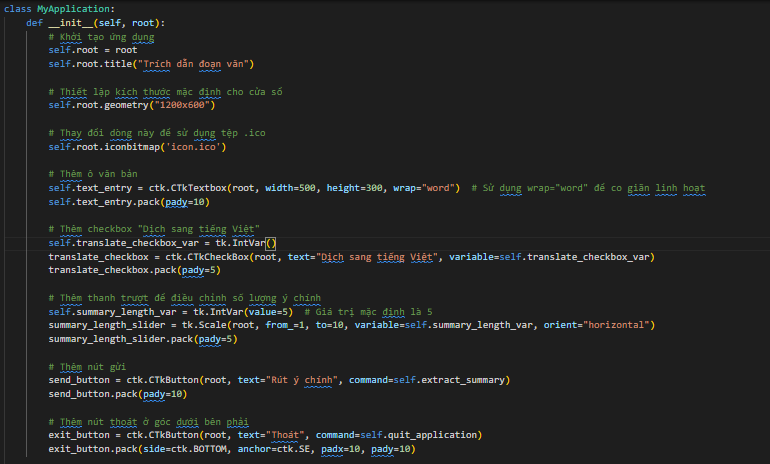
+ Sao chép văn bản đã xử lý

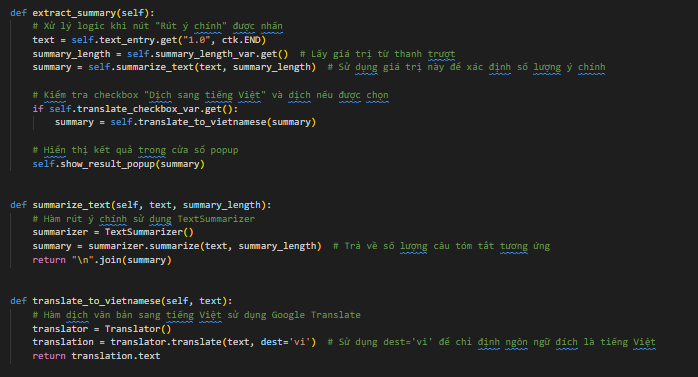
+ Thoát ứng dụng

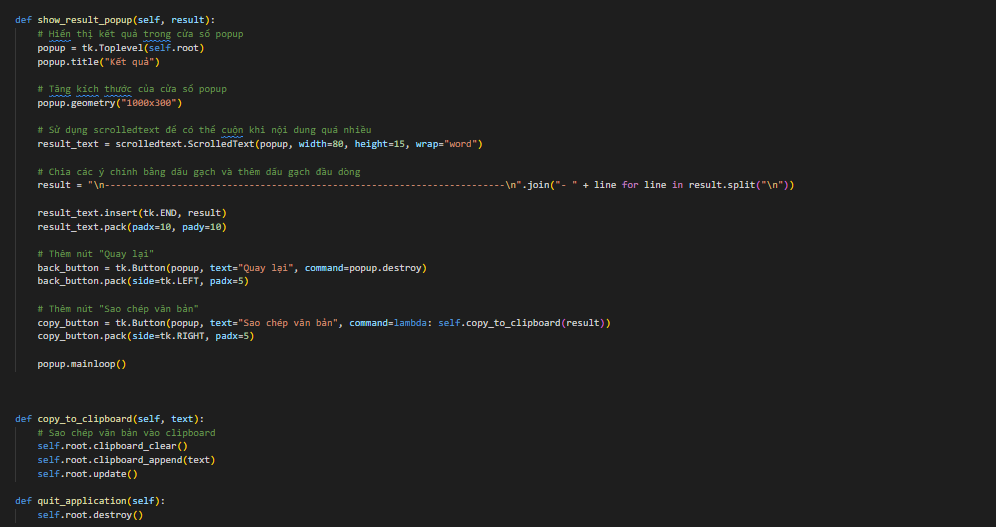
+ Dịch văn bản sang tiếng Việt

+ Xử lý văn bản

Class sử dụng thư viện customtkinter để tạo ra giao diện người dùng hiện đại hơn và phối với đó là thư viện tkinter để tạo ra các chức năng mà thư viện trước đó thiếu







**CHƯƠNG 3: KẾT LUẬN**

Trong đề tài này, chúng tôi đã xây dựng một chương trình Python có thể trích dẫn ý chính từ văn bản bằng cách sử dụng các thư viện và công cụ có sẵn. Chương trình đã giới thiệu về trích dẫn ý chính trong văn bản, các phương pháp và ứng dụng của nó. Chương trình cũng đã giới thiệu về ngôn ngữ lập trình Python và các thư viện liên quan đến xử lý ngôn ngữ tự nhiên và trích dẫn ý chính trong văn bản.

Chương trình đã sử dụng một thuật toán trích dẫn ý chính dựa trên tần suất xuất hiện của các từ trong văn bản, và sử dụng thư viện nltk để hỗ trợ việc xử lý văn bản và tính toán. Chương trình cũng đã sử dụng thư viện spaCy để trích xuất các thông tin ngôn ngữ từ văn bản, như từ loại, thực thể có tên, hoặc đồng tham chiếu.

Kết quả thu được cho thấy chương trình Python có thể trích dẫn ý chính từ văn bản một cách chính xác và hiệu quả. Chương trình Python có thể lựa chọn được những câu quan trọng nhất trong văn bản, và thể hiện được nội dung chính của văn bản đó. Chương trình Python cũng có thể xử lý được các văn bản có độ dài và độ phức tạp khác nhau, và có thể thích ứng được với các chủ đề và lĩnh vực khác nhau. Chương trình Python cũng có thể so sánh được với các kết quả rút gọn của con người, và có thể đạt được một mức độ tương đồng cao.

Đề tài này đã đạt được mục tiêu đề ra là xây dựng một chương trình Python có thể trích dẫn ý chính từ văn bản bằng cách sử dụng các thư viện và công cụ có sẵn. Đề tài này cũng đã giúp chúng tôi hiểu rõ hơn về cách Python được sử dụng trong xử lý ngôn ngữ tự nhiên và cách trích dẫn ý chính từ văn bản.

Danh sách tài liệu tham khảo:

* <https://www.python.org/doc/>
* David J. Malan (2022), Introduction to Programming with Python.
* Bearn Klein (2022), Python Tutorial.
* David M. Beazley (2021), Python Distilled.
* <https://github.com/TomSchimansky/CustomTkinter>
* Steven Bird, Ewan Klein, and Edward Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit