**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**--------\*\*\*--------**

A blue button with white text

Description automatically generated

**BÁO CÁO MÔN**

**XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

**TÊN ĐỀ TÀI**

**CÀI ĐẶT VÀ HUẤN LUYỆN MÔ HÌNH SKIP-GRAM**

**TRÊN NGỮ LIỆU TIẾNG VIỆT**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | **Nguyễn Tuấn Đăng** |
| **Sinh viên thực hiện:** | **Phan Phước Vinh** |

**Thành phố Hồ Chí Minh, năm 2024**

****

**Mục Lục**

[**LỜI MỞ ĐẦU** 5](#_Toc184503232)

[**PHẦN 1: GIỚI THIỆU VỀ ĐỀ TÀI VÀ GIẢI THÍCH KHÁI NIỆM** 6](#_Toc184503233)

[**I)** **Giới thiệu về đề tài:** 6](#_Toc184503234)

[**II)** **Giải thích khái niệm:** 6](#_Toc184503235)

[**1)** **Word Embeeding (Vector biểu diễn từ):** 7](#_Toc184503236)

[**2)** **Context Window (Cửa sổ ngữ cảnh):** 8](#_Toc184503237)

[**3)** **Cosine Similarity (Độ tương đồng Cosine)** 8](#_Toc184503238)

[**PHẦN 2: TIỀN XỬ LÝ DỮ LIỆU** 9](#_Toc184503239)

[**I)** **Mục tiêu và yêu cầu:** 9](#_Toc184503240)

[**II)** **Các bước cụ thể:** 9](#_Toc184503241)

[**PHẦN 3: THIẾT KẾ MÔ HÌNH SKIP-GRAM** 13](#_Toc184503242)

[**I)** **Tổng quan về mô hình:** 13](#_Toc184503243)

[**II)** **Giới thiệu về mô hình sẽ được thiết kế:** 13](#_Toc184503244)

[**1)** **Ma trận trọng số (Weight Matrices):** 14](#_Toc184503245)

[**2)** **Softmax Function:** 14](#_Toc184503246)

[**3)** **Forward Pass:** 15](#_Toc184503247)

[**4)** **Backward Pass và Gradient Calculation:** 15](#_Toc184503248)

[**5)** **Huấn luyện mô hình (Training):** 16](#_Toc184503249)

[**6)** **Embedding và tính khoảng cách (Cosine Similarity):** 16](#_Toc184503250)

[**7)** **Lưu và tải mô hình:** 16](#_Toc184503251)

[**PHẦN 4: HUẤN LUYỆN MÔ HÌNH, KẾT QUẢ VÀ ĐÁNH GIÁ** 17](#_Toc184503252)

[**I)** **Triển khai mô hình và lý do chọn các tham số:** 17](#_Toc184503253)

[**II)** **Quy trình huấn luyện:** 17](#_Toc184503254)

[**1)** **Tạo các cặp dữ liệu Skip-gram:** 17](#_Toc184503255)

[**2)** **Huấn luyện mô hình:** 17](#_Toc184503256)

[**3)** **Loss trong quá trình huấn luyện:** 18](#_Toc184503257)

[**4)** **Tính toán cosine similarity:** 18](#_Toc184503258)

[**III)** **Kết quả thu được từ huấn luyện:** 18](#_Toc184503259)

[**1)** **Loss trong quá trình huấn luyện:** 18](#_Toc184503260)

[**2)** **Tính cosine similarity:** 19](#_Toc184503261)

[**IV)** **Phân tích và đánh giá kết quả:** 19](#_Toc184503262)

[**PHẦN 5: KHÓ KHĂN, BÀI HỌC KINH NGHIỆM VÀ KẾT LUẬN** 20](#_Toc184503263)

[**I)** **Khó khăn:** 20](#_Toc184503264)

[**1)** **Dữ liệu không đủ đa dạng** 20](#_Toc184503265)

[**2)** **Tốc độ xử lý tách từ khi chạy đơn luồng chậm** 20](#_Toc184503266)

[**3)** **Bộ nhớ không đủ để lưu dữ liệu pairs** 20](#_Toc184503267)

[**4)** **Tốc độ huấn luyện mô hình chậm** 21](#_Toc184503268)

[**II)** **Bài học kinh nghiệm:** 21](#_Toc184503269)

[**1)** **Dữ liệu là yếu tố quyết định** 21](#_Toc184503270)

[**2)** **Cần tối ưu hóa tốc độ xử lý dữ liệu** 21](#_Toc184503271)

[**3)** **Quản lý bộ nhớ hiệu quả** 21](#_Toc184503272)

[**4)** **Đa dạng hóa mô hình và tối ưu hóa tốc độ huấn luyện** 21](#_Toc184503273)

[**III)** **Kết luận:** 22](#_Toc184503274)

# **LỜI MỞ ĐẦU**

Trong kỷ nguyên chuyển đổi số, xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) đang dần khẳng định vai trò cốt lõi trong việc phát triển các ứng dụng thông minh, từ trợ lý ảo, dịch máy cho đến phân tích ý kiến trên mạng xã hội. Một trong những thách thức lớn của NLP là làm thế nào để biểu diễn từ ngữ sao cho máy tính có thể hiểu và xử lý một cách hiệu quả. Phương pháp **word embeddings**, với khả năng biểu diễn từ dưới dạng vector trong không gian số, đã trở thành nền tảng quan trọng giúp giải quyết bài toán này.

Đề tài “Cài đặt và huấn luyện mô hình Skip-gram trên ngữ liệu tiếng Việt” được thực hiện với mong muốn khám phá một phương pháp biểu diễn từ tiên tiến – Skip-gram – trong bối cảnh ngôn ngữ tiếng Việt. Tiếng Việt là một ngôn ngữ phức tạp với hệ thống dấu, hình thái từ đa dạng và cấu trúc ngữ pháp linh hoạt, tạo ra những thách thức riêng biệt khi áp dụng các phương pháp NLP phổ biến. Mục tiêu của đề tài là xây dựng một hệ thống học biểu diễn từ hiệu quả, góp phần nâng cao hiệu suất của các ứng dụng NLP tiếng Việt.

Trong suốt quá trình thực hiện, em đã nhận được sự hướng dẫn tận tình từ thầy **Nguyễn Tuấn Đăng**. Thầy không chỉ chia sẻ những kiến thức chuyên môn mà còn truyền cảm hứng và định hướng rõ ràng, giúp em vượt qua các khó khăn trong nghiên cứu. Em xin gửi lời cảm ơn chân thành đến thầy vì những hỗ trợ quý báu đó.

Báo cáo này không chỉ trình bày chi tiết về cách tiếp cận và thực hiện đề tài, mà còn chia sẻ những bài học kinh nghiệm và khó khăn trong suốt quá trình triển khai. Em hy vọng tài liệu này sẽ là một đóng góp nhỏ cho lĩnh vực xử lý ngôn ngữ tiếng Việt và đồng thời mở ra các hướng nghiên cứu mới trong tương lai.

Em rất mong nhận được ý kiến đóng góp từ thầy và các bạn để hoàn thiện hơn nữa.

# **PHẦN 1: TỔNG QUAN**

## **Giới thiệu về đề tài:**

Trong thời đại công nghệ thông tin hiện nay, việc xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) ngày càng đóng vai trò quan trọng trong nhiều lĩnh vực như tìm kiếm thông tin, dịch thuật tự động, hay phát triển trợ lý ảo. Đặc biệt, khả năng biểu diễn ngữ nghĩa của từ ngữ trong không gian số học đã trở thành nền tảng quan trọng cho nhiều mô hình học máy hiện đại.

Dự án này tập trung nghiên cứu và triển khai mô hình **Skip-gram**, một trong những phương pháp biểu diễn từ phổ biến và hiệu quả. Skip-gram không chỉ giúp biểu diễn ngữ nghĩa từ ngữ trong không gian vector mà còn xây dựng mối quan hệ giữa các từ dựa trên ngữ cảnh của chúng. Mô hình này đã chứng minh tính ưu việt trong việc biểu diễn các từ có ngữ nghĩa tương tự một cách gần nhau trong không gian vector, hỗ trợ mạnh mẽ các ứng dụng NLP phức tạp như phân tích cảm xúc, phát hiện chủ đề, và xây dựng chatbot.

Dự án nhằm giải quyết bài toán học biểu diễn từ bằng cách sử dụng bộ dữ liệu thực tế là các bài báo, từ đó tạo ra một mô hình ngôn ngữ cơ bản có thể so sánh mối quan hệ giữa các từ thông qua **cosine similarity**. Bên cạnh đó, dự án cũng phân tích những thách thức và bài học rút ra trong quá trình thực hiện, qua đó đóng góp kiến thức cho cộng đồng nghiên cứu và phát triển NLP.

Với ý nghĩa ứng dụng rộng rãi và tính thực tiễn cao, dự án không chỉ mang lại cái nhìn sâu sắc về mô hình Skip-gram mà còn mở ra hướng phát triển các mô hình học sâu hiệu quả hơn trong tương lai.

## **Giải thích khái niệm:**

Trong xử lý ngôn ngữ tự nhiên (NLP), việc biểu diễn từ ngữ dưới dạng số hóa là một bước nền tảng để máy tính có thể hiểu và xử lý ngôn ngữ con người. **Skip-gram** là một trong những phương pháp nổi bật để đạt được mục tiêu này, dựa trên việc học các biểu diễn vector (word embeddings) của từ ngữ.

Khái niệm chính của Skip-gram rất đơn giản nhưng lại vô cùng mạnh mẽ: mô hình học cách dự đoán từ ngữ xung quanh (ngữ cảnh) dựa trên một từ trung tâm. Kết quả của quá trình này là các vector biểu diễn từ trong không gian số học, sao cho các từ có ý nghĩa tương tự sẽ có vector gần nhau. Ví dụ, trong không gian vector, các từ như *"ông"* và *"cha"* sẽ có khoảng cách gần hơn so với *"ông"* và *"mẹ"*.

Kiến trúc Skip-gram bao gồm hai thành phần chính:

1. **Tập từ vựng (Vocabulary):** Đây là tập hợp các từ xuất hiện trong dữ liệu huấn luyện. Mỗi từ sẽ được gán một vector số để biểu diễn.
2. **Ma trận trọng số:** Bao gồm hai ma trận - một ma trận ánh xạ từ từ trung tâm sang vector (W1) và một ma trận ánh xạ vector sang ngữ cảnh (W2).

Trong quá trình huấn luyện, Skip-gram sử dụng một hàm mất mát dựa trên độ tương đồng giữa từ ngữ thực tế và từ ngữ dự đoán. Điều này giúp mô hình điều chỉnh các vector sao cho chúng ngày càng chính xác hơn trong việc nắm bắt ý nghĩa từ ngữ.

Skip-gram không chỉ đơn thuần là một mô hình mà còn là công cụ quan trọng để hiểu và khai thác sâu hơn các mối quan hệ tiềm ẩn giữa các từ trong văn bản. Đây là lý do tại sao Skip-gram trở thành lựa chọn hàng đầu trong nhiều bài toán NLP và tiếp tục đóng vai trò quan trọng trong các nghiên cứu hiện đại.

**Sau đây chúng ta cùng đi sâu và chi tiết hơn về các khái niệm:**

### **Word Embeeding (Vector biểu diễn từ):**

Word embedding là phương pháp ánh xạ từ ngữ sang không gian vector số nhiều chiều. Trong không gian này, các từ có ý nghĩa tương tự sẽ gần nhau, trong khi các từ khác biệt thì sẽ cách xa nhau.

**Vì sao cần word embedding?**

Máy tính không thể trực tiếp hiểu ngôn ngữ con người, vì vậy việc chuyển từ thành dạng số là cần thiết. Tuy nhiên, các phương pháp cũ như **one-hot encoding** hoặc **bag-of-words** lại không hiệu quả vì chúng không biểu diễn được mối quan hệ giữa các từ. Word embedding giải quyết vấn đề này bằng cách học các vector biểu diễn chứa thông tin ngữ nghĩa và ngữ cảnh.

**Vai trò của Skip-gram trong word embedding:**

Skip-gram là một trong những mô hình nổi bật được sử dụng để học word embedding. Phương pháp này không chỉ giúp biểu diễn mối quan hệ giữa các từ, mà còn làm nổi bật các quan hệ ngữ nghĩa phức tạp, ví dụ như:

* *"king"* - *"man"* + *"woman"* ≈ *"queen"*

### **Context Window (Cửa sổ ngữ cảnh):**

Cửa sổ ngữ cảnh là phạm vi các từ xung quanh một từ trung tâm mà mô hình Skip-gram sẽ sử dụng để huấn luyện.

**Ý nghĩa của cửa sổ ngữ cảnh:**

* Nếu cửa sổ ngữ cảnh nhỏ, mô hình sẽ học được các mối quan hệ ngắn hạn giữa từ trung tâm và các từ ngữ cảnh gần.
* Nếu cửa sổ lớn, mô hình có khả năng nắm bắt các mối quan hệ dài hạn hơn nhưng có thể bị nhiễu.

**Lựa chọn kích thước cửa sổ:**

Kích thước cửa sổ thường được lựa chọn tùy thuộc vào loại dữ liệu:

* Đối với dữ liệu ngắn như tiêu đề, cửa sổ nhỏ (2-5) sẽ hiệu quả hơn.
* Đối với văn bản dài như bài báo, cửa sổ lớn (5-10) giúp mô hình nắm bắt nhiều thông tin hơn.

### **Cosine Similarity (Độ tương đồng Cosine)**

Cosine similarity là thước đo được sử dụng để đánh giá mức độ tương đồng giữa hai vector trong không gian.

**Công thức tính:**

**Cosine Similarity**

Trong đó:

* : Hai vector cần so sánh.
* : Độ dài của vector

**Vai trò trong Skip-gram:**

* Được sử dụng để đánh giá mối quan hệ giữa các từ trong không gian vector.
* Các cặp từ có cosine similarity cao được xem là có ngữ nghĩa tương tự.

# **PHẦN 2: DỮ LIỆU**

Tiền xử lý dữ liệu là bước thiết yếu trong bất kỳ mô hình học máy nào. Dữ liệu thô thường chứa nhiều thông tin không cần thiết, cần làm sạch và chuyển đổi về dạng phù hợp để huấn luyện mô hình hiệu quả. Phần này sẽ giới thiệu quy trình tiền xử lý dữ liệu được thực hiện trên tập dữ liệu, bao gồm các công việc như làm sạch văn bản, tách từ, lọc thông tin không phù hợp và chuẩn bị từ vựng.

## **Mục tiêu và yêu cầu:**

Quy trình tiền xử lý dữ liệu trong dự án này hướng đến việc:

* **Loại bỏ thông tin thừa hoặc không cần thiết** như URL, số, ký tự đặc biệt và thông tin trong các dấu ngoặc.
* **Tách từ và gán nhãn từ loại** để làm sạch dữ liệu và chỉ giữ các từ có ý nghĩa trong ngữ cảnh học máy.
* **Chạy xử lý song song** để cải thiện tốc độ khi làm việc với tập dữ liệu lớn.
* **Tạo từ điển từ vựng** (vocab) dựa trên tần suất xuất hiện của các từ trong tập dữ liệu sau khi tiền xử lý xong.

## **Các bước cụ thể:**

**Bước 1:** Làm sạch văn bản:

Đầu tiên, ta cần làm sạch dữ liệu thô từ các thông tin được thu thập. Việc làm sạch bao gồm loại bỏ các URL, các ký tự không cần thiết và chuẩn hóa cấu trúc văn bản để làm dữ liệu sẵn sàng cho các bước tiếp theo.

Hàm clean\_text thực hiện nhiệm vụ này:



**Bước 2:** Token hóa và gán nhãn từ loại:

Sau khi làm sạch văn bản, từng văn bản sẽ được tách từ thành các token nhỏ thông qua **ViTokenizer**, sau đó gán nhãn từ loại bằng ViPosTagger. Để giảm tiếng ồn trong dữ liệu, các từ mang nhãn là danh từ riêng (Np) sẽ được loại bỏ.



**Bước 3:** Xử lý dữ liệu song song:

Dữ liệu thường rất lớn, và việc xử lý tuần tự sẽ mất nhiều thời gian. Để khắc phục điều này, ở đây sử dụng kỹ thuật **xử lý song song** với module multiprocessing.



Trong đoạn mã trên:

* Sử dụng **multiprocessing.Pool** để tạo nhiều worker xử lý dữ liệu song song.
* **tqdm** được dùng để theo dõi tiến trình của quá trình xử lý song song.

**Bước 4:** Chuẩn hóa dữ liệu và lọc thông tin rỗng:

Sau khi đã token hóa và lọc từ loại, dữ liệu sẽ được chuyển đổi thành chữ thường để đảm bảo nhất quán và loại bỏ các từ rỗng hoặc không chứa thông tin.



**Bước 5:** Tạo từ điển từ vựng:

Tiến trình tiếp theo là tạo từ điển từ vựng từ toàn bộ các từ trong dữ liệu đã làm sạch. Việc tạo từ điển giúp ánh xạ từ ngữ tự nhiên thành chỉ số trong không gian vector để mô hình có thể huấn luyện.



**Bước 6: Lưu dữ liệu đã xử lý**

Cuối cùng, dữ liệu đã làm sạch và từ điển được lưu vào các file JSON để mô hình có thể đọc và sử dụng trong quá trình huấn luyện.



# **PHẦN 3: THIẾT KẾ MÔ HÌNH SKIP-GRAM**

## **Tổng quan về mô hình:**

Skip-gram là một kỹ thuật trong mô hình Word2Vec được sử dụng để học các vector biểu diễn từ ngữ trong không gian số học. Mô hình Skip-gram hoạt động bằng cách học dựa trên việc dự đoán các từ trong ngữ cảnh lân cận từ trung tâm. Điều này giúp biểu diễn các từ có ý nghĩa gần nhau trong không gian vector có thể được tính toán thông qua khoảng cách giữa chúng.

Trong dự án này, chúng tôi đã thiết kế và triển khai mô hình Skip-gram từ đầu bằng cách sử dụng một mô hình đơn giản với hai ma trận trọng số:

* **W1 (Trọng số cho từ trung tâm):** Biểu diễn thông tin của từ trung tâm.
* **W2 (Trọng số cho ngữ cảnh):** Biểu diễn thông tin của ngữ cảnh lân cận.

## **Giới thiệu về mô hình sẽ được thiết kế:**

Mô hình Skip-gram được triển khai trong dự án sử dụng hai ma trận trọng số, W1 và W2. Quá trình huấn luyện của mô hình dựa trên việc tối ưu hóa hàm mất mát bằng **Softmax** để dự đoán các từ trong ngữ cảnh của từ trung tâm và cải thiện các trọng số thông qua phép lan truyền ngược **(backpropagation)**.

Các bước chính trong mô hình bao gồm:

1. **Forward Pass:** Dự đoán xác suất của các từ ngữ cảnh dựa vào từ trung tâm thông qua hàm Softmax.
2. **Backward Pass:** Tính toán sai số và cập nhật trọng số thông qua lan truyền ngược.
3. **Huấn luyện mô hình:** Dùng các tập dữ liệu về các cặp từ (center-context) để huấn luyện mô hình.

### **Ma trận trọng số (Weight Matrices):**

Mô hình Skip-gram sử dụng hai ma trận trọng số chính, được ký hiệu là W1 và W2. Chúng đóng vai trò quan trọng trong việc học biểu diễn từ:

1. **Ma trận W1 (Trọng số trung tâm):**
   * Đây là ma trận biểu diễn thông tin từ trung tâm trong mô hình.
   * Kích thước của ma trận là **(Vocabulary Size x Embedding Dimension)**, với Vocabulary Size là tổng số từ trong từ điển và Embedding Dimension là chiều dài của vector biểu diễn từ.
   * W1 được sử dụng để ánh xạ các chỉ số của từ trung tâm thành các vector trong không gian số học.
2. **Ma trận W2 (Trọng số ngữ cảnh):**
   * Biểu diễn thông tin ngữ cảnh lân cận tương ứng.
   * Kích thước của ma trận là **(Embedding Dimension x Vocabulary Size)**.
   * W2 dùng để tính toán xác suất các từ ngữ cảnh lân cận thông qua phép nhân với từ trung tâm.

Hai ma trận này được khởi tạo ngẫu nhiên và cập nhật thông qua quá trình huấn luyện.

### **Softmax Function:**

Softmax là một hàm kích hoạt quan trọng trong mô hình Skip-gram. Nó chuyển đổi các giá trị thô (logits) thành xác suất, mô phỏng xác suất xuất hiện của các từ ngữ cảnh dựa vào từ trung tâm.  
**Công dụng chính:**

* Biến đổi các đầu ra thô từ mô hình thành một xác suất trong khoảng [0, 1].
* Tổng của tất cả xác suất này sẽ bằng 1, đảm bảo tính xác suất hợp lý trong dự đoán của mô hình.

Softmax thường được áp dụng sau phép nhân giữa từ trung tâm và trọng số ngữ cảnh W2.

### **Forward Pass:**

Forward pass là quá trình tính toán từ đầu vào thông qua mô hình để dự đoán các xác suất ngữ cảnh lân cận của từ trung tâm:

1. Dùng trọng số W1 để lấy thông tin từ trung tâm và ánh xạ thành vector nhúng.
2. Tính toán logits thông qua phép nhân giữa vector từ trung tâm và W2.
3. Áp dụng hàm softmax để chuyển logits thành các xác suất, thể hiện xác suất dự đoán các từ trong ngữ cảnh.

Forward pass cung cấp thông tin dự đoán của mô hình cho mỗi từ trung tâm.

### **Backward Pass và Gradient Calculation:**

Backward pass là bước quan trọng để cập nhật trọng số thông qua việc tính toán gradient và tối ưu hóa sai số:

1. **Tính sai số (Error):**
   * Sai số được tính bằng cách lấy hiệu giữa xác suất dự đoán từ softmax và một vector one-hot biểu diễn thông tin ngữ cảnh thực tế.
2. **Gradient:**
   * Dựa vào sai số và các trọng số W1, W2 để tính toán gradient.
   * Gradient được tính thông qua phép nhân giữa sai số và thông tin từ mô hình.
3. **Cập nhật trọng số:**
   * Gradient được tính toán và nhân với tốc độ học để cập nhật W1 và W2.

Cập nhật trọng số này sẽ giúp mô hình học dần dần sao cho xác suất dự đoán các từ ngữ cảnh ngày càng chính xác hơn.

### **Huấn luyện mô hình (Training):**

Quá trình huấn luyện diễn ra trong nhiều epoch với các cặp từ trung tâm-ngữ cảnh được tạo từ dữ liệu đầu vào:

* **Epochs:** Là số vòng lặp qua toàn bộ tập huấn luyện.
* **Batch size:** Dùng để chia dữ liệu thành các tập nhỏ và huấn luyện theo từng batch.

Trong huấn luyện, các trọng số W1 và W2 được cập nhật thông qua gradient và tối ưu hóa sai số qua mỗi batch.

### **Embedding và tính khoảng cách (Cosine Similarity):**

1. **Vector biểu diễn (Embedding):**
   * Dùng trọng số W1 sau khi huấn luyện để lấy thông tin biểu diễn từ ngữ trong không gian số học.
   * Mỗi từ sẽ có một vector biểu diễn duy nhất.
2. **Cosine Similarity:**
   * Được tính giữa các vector biểu diễn để đánh giá khoảng cách ngữ nghĩa giữa hai từ.
   * Nếu hai từ có cosine similarity gần 1, chúng gần về mặt ngữ nghĩa; nếu gần -1, chúng xa về mặt ngữ nghĩa.

### **Lưu và tải mô hình:**

Mô hình Skip-gram cung cấp chức năng lưu các trọng số (W1 và W2) sau khi huấn luyện xong để tái sử dụng sau này. Điều này giúp tiết kiệm thời gian và tài nguyên trong các giai đoạn sau của dự án hoặc trong các tác vụ phân tích.

# **PHẦN 4: HUẤN LUYỆN MÔ HÌNH, KẾT QUẢ VÀ ĐÁNH GIÁ**

## **Triển khai mô hình và lý do chọn các tham số:**

|  |  |  |
| --- | --- | --- |
| **Tham số** | **Giá trị** | **Lý do lựa chọn** |
| Kích thước vector nhúng (embedding\_dim) | 100 | Đây là một giá trị thông dụng trong nhiều nghiên cứu và đủ để biểu diễn thông tin ngữ nghĩa của từ. |
| Batch size | 1024 | Batch size lớn giúp tăng hiệu quả tính toán và cải thiện khả năng hội tụ trong quá trình huấn luyện. |
| Epochs | 12 | Huấn luyện trong 12 vòng lặp để đảm bảo mô hình học đủ thông tin từ dữ liệu. |
| Window size | 4 | Xác định phạm vi ngữ cảnh của từ trung tâm, giúp tìm mối quan hệ từ gần và có ý nghĩa. |
| Learning rate | 0.01 | Learning rate vừa đủ để mô hình cập nhật trọng số một cách ổn định. |

## **Quy trình huấn luyện:**

### **Tạo các cặp dữ liệu Skip-gram:**

* Dùng hàm skipgram\_pairs để tạo các cặp dữ liệu trung tâm và ngữ cảnh.
* **Window size = 4**: Ngữ cảnh bao gồm từ trong khoảng 4 từ trước và 4 từ sau từ trung tâm. Điều này giúp mô hình học được thông tin ngữ nghĩa trong vùng lân cận gần nhất.

### **Huấn luyện mô hình:**

* Dùng thuật toán **Skip-gram model** để huấn luyện với cặp dữ liệu được tạo ở trên.
* Huấn luyện với batch size = 1024 và lặp qua 12 epochs.

### **Loss trong quá trình huấn luyện:**

* **Loss giảm đều qua từng epoch**: Loss trung bình giảm từ 7.3930 ở epoch đầu tiên xuống 6.5913 ở epoch cuối cùng. Điều này cho thấy mô hình đang học hiệu quả và cập nhật các trọng số phù hợp để tối ưu hóa.

### **Tính toán cosine similarity:**

* Sau khi huấn luyện xong, tính toán **cosine similarity** giữa các cặp từ cụ thể để đánh giá xem mô hình có học được các mối quan hệ ngữ nghĩa tốt không.
* Ví dụ:
  + Tính cosine similarity giữa **"ông" và "cha"**, **"ông" và "mẹ"**, **"giáo\_viên" và "buồn\_bã"**, v.v.

## **Kết quả thu được từ huấn luyện:**

### **Loss trong quá trình huấn luyện:**

Loss giảm đều qua 12 vòng lặp với các thông tin như sau:

A graph with a line

Description automatically generated

* **Epoch 1**: Avg Loss = 7.3930
* **Epoch 12**: Avg Loss = 6.5913

Việc giảm loss đều chỉ ra mô hình đang học tốt và phù hợp với dữ liệu.

### **Tính cosine similarity:**

Các kết quả tính cosine similarity cho thấy mối quan hệ ngữ nghĩa giữa một số từ quan trọng:

* **Similarity giữa "ông" và "cha"**: 0.4867  
  Giải thích: "ông" và "cha" có ý nghĩa gần nhau trong ngữ cảnh gia đình.
* **Similarity giữa "ông" và "mẹ"**: 0.2227  
  Giải thích: Mặc dù có liên quan về ngữ cảnh, nhưng không gần như "ông" và "cha".
* **Similarity giữa "giáo\_viên" và "buồn\_bã"**: 0.1936  
  Dấu hiệu mối liên kết ngữ nghĩa thấp.
* **Similarity giữa "hạnh\_phúc" và "vua"**: 0.0472  
  Không có mối liên hệ rõ ràng giữa hai từ này.
* **Similarity giữa "học" và "giáo\_dục"**: 0.3103  
  Thể hiện mối liên kết giữa học và giáo dục.
* **Similarity giữa "anh" và "chị"**: 0.7202  
  Đây là mối quan hệ gần gũi và có liên quan mạnh mẽ trong ngữ cảnh văn hóa và xã hội.

## **Phân tích và đánh giá kết quả:**

* **Loss giảm đều qua các epochs**

Loss giảm cho thấy mô hình đang học thông tin từ dữ liệu và đang tối ưu hóa các trọng số một cách hiệu quả. Loss trung bình kết thúc tại mức 6.5913 sau 12 epochs.

* **Vector nhúng có thể phản ánh ngữ nghĩa**

Thông qua tính toán cosine similarity, mô hình đã học được các mối quan hệ ngữ nghĩa giữa các từ, như "ông" gần với "cha" và "học" gần với "giáo\_dục". Điều này khẳng định mô hình đã học được thông tin ngữ nghĩa và biểu diễn chúng trong không gian vector.

* **Các từ có độ tương đồng cao**
* **"anh" và "chị"** có similarity = 0.7202, cho thấy mô hình đã học đúng về mối quan hệ giữa hai từ này trong ngữ cảnh xã hội và gia đình.
* **"học" và "giáo\_dục"** có similarity = 0.3103, mô hình hiểu được mối liên quan giữa hai từ này trong ngữ cảnh học thuật.

# **PHẦN 5: KHÓ KHĂN, BÀI HỌC KINH NGHIỆM VÀ KẾT LUẬN**

## **Khó khăn:**

### **Dữ liệu không đủ đa dạng**

Dữ liệu được thu thập và xử lý có hạn chế về ngữ cảnh và từ vựng. Điều này dẫn đến mô hình học không đủ thông tin để biểu diễn đầy đủ ý nghĩa ngữ nghĩa của từ, ảnh hưởng đến khả năng biểu diễn và khả năng tính toán similarity giữa các từ.

Giải pháp trong tương lai là mở rộng bộ dữ liệu, thêm nhiều văn bản từ nhiều nguồn ngôn ngữ và ngữ cảnh khác nhau.

### **Tốc độ xử lý tách từ khi chạy đơn luồng chậm**

Quá trình tách từ và tạo các cặp dữ liệu Skip-gram thông qua các câu lệnh xử lý đơn luồng tốn nhiều thời gian. Việc xử lý dữ liệu lớn với hàng chục nghìn câu khiến tốc độ trở nên chậm.

Điều này cho thấy cần thiết kế lại các thuật toán xử lý dữ liệu để hỗ trợ **đa luồng (multithreading)** hoặc các phương pháp tối ưu hóa.

### **Bộ nhớ không đủ để lưu dữ liệu pairs**

Với hàng chục nghìn câu và hàng chục nghìn từ vựng, việc lưu trữ tất cả các cặp từ trung tâm và ngữ cảnh (word pairs) trong bộ nhớ tốn một lượng bộ nhớ lớn.

Giải pháp tiềm năng là áp dụng các kỹ thuật lưu trữ dữ liệu hiệu quả hơn, chẳng hạn lưu trực tiếp vào ổ đĩa hoặc chia nhỏ dữ liệu thành các batch nhỏ hơn để xử lý.

### **Tốc độ huấn luyện mô hình chậm**

Mô hình Skip-gram huấn luyện tốn rất nhiều thời gian với các thông số như batch size lớn, window size và vocab size. Thêm vào đó, việc lặp qua 12 epoch với dữ liệu lớn làm tăng đáng kể thời gian huấn luyện.

Giải pháp trong tương lai có thể bao gồm tối ưu hóa thuật toán huấn luyện, áp dụng đa luồng hoặc sử dụng GPU để tăng tốc độ tính toán.

## **Bài học kinh nghiệm:**

### **Dữ liệu là yếu tố quyết định**

Dữ liệu không đủ đa dạng làm giảm chất lượng mô hình. Vì vậy, khi xây dựng mô hình học từ ngữ nghĩa, cần đầu tư thêm vào thu thập và làm phong phú bộ dữ liệu.

### **Cần tối ưu hóa tốc độ xử lý dữ liệu**

Thay vì chạy các tác vụ xử lý dữ liệu tuần tự, nên áp dụng các kỹ thuật **đa luồng (multithreading)** hoặc tối ưu hóa thuật toán để giảm thời gian chờ và đẩy nhanh tốc độ xử lý dữ liệu.

### **Quản lý bộ nhớ hiệu quả**

Lưu trữ và xử lý dữ liệu lớn cần thiết kế bộ nhớ hiệu quả hơn. Thay vì giữ toàn bộ dữ liệu trong bộ nhớ, nên lưu trữ từng batch nhỏ hoặc tối ưu hóa kích thước dữ liệu.

### **Đa dạng hóa mô hình và tối ưu hóa tốc độ huấn luyện**

Thời gian huấn luyện lâu dài do mô hình và dữ liệu lớn. Do đó, trong các dự án tiếp theo, cần thử nghiệm các kỹ thuật như **GPU acceleration** hoặc các chiến lược huấn luyện tối ưu để cải thiện tốc độ huấn luyện.

## **Kết luận:**

Qua quá trình huấn luyện mô hình Skip-gram và tính toán cosine similarity, em đã thành công trong việc biểu diễn từ và học thông tin ngữ nghĩa từ dữ liệu. Kết quả cho thấy mô hình đã học được thông tin ngữ nghĩa trong không gian vector và có thể tính toán các độ tương đồng giữa các từ.

Tuy nhiên, các thách thức như dữ liệu chưa đủ đa dạng, tốc độ huấn luyện và bộ nhớ cần cải thiện là những vấn đề cần tập trung giải quyết trong các nghiên cứu và dự án tiếp theo. Những bài học kinh nghiệm từ quá trình triển khai mô hình sẽ là cơ sở để cải tiến và phát triển các mô hình học từ trong tương lai.

Việc lưu trữ mô hình và các trọng số đã hoàn thành thành công với kết quả đáng khích lệ, mở ra cơ hội tiếp tục nghiên cứu và ứng dụng cho các bài toán liên quan đến ngôn ngữ tự nhiên và mô hình học máy.

# **TÀI LIỆU THAM KHẢO**

* 1. "Speech and Language Processing" by Daniel Jurafsky and James H. Martin.
  2. "Efficient Estimation of Word Representations in Vector Space" by Mikolov et al. (2013).
  3. "An overview of gradient descent optimization algorithms" by Ruder (2016).
  4. "Python Data Science Handbook" by Jake VanderPlas.
  5. NumPy Documentation.
  6. Matplotlib Documentation.
  7. "UIT-VSFC: Vietnamese Students’ Feedback Corpus for Sentiment Analysis" by Nguyen et al. (2020).
  8. "Building Word2Vec from Scratch in Python" by Chris McCormick.

# **PHỤ LỤC**

## **Mô tả thuật toán Skip-gram:**

**Quy trình Skip-gram**

* Input: Một từ trung tâm ***Wt*​** và một cửa sổ ngữ cảnh có kích thước ***c***.
* Output: Xác suất của các từ trong ngữ cảnh xuất hiện gần từ trung tâm.

**Các bước thực hiện:**

* Bước 1: Biểu diễn từ trung tâm ***W1***​ dưới dạng one-hot vector.
* Bước 2: Nhân one-hot vector với ma trận nhúng ***W1​*** để tạo vector nhúng.
* Bước 3: Từ vector nhúng, tính đầu ra bằng cách nhân với ma trận ***W2***​ và áp dụng hàm softmax.
* Bước 4: Sử dụng hàm mất mát cross-entropy để đo lường độ chênh lệch giữa dự đoán và từ ngữ cảnh thực tế.
* Bước 5: Cập nhật trọng số ​ ***W1***​ và ***W2***​​ bằng thuật toán lan truyền ngược và gradient descent.

## **Mã giả:**



## **Ví dụ minh họa:**

**Dataset (tập dữ liệu):** “Học lập trình rất thú vị và bổ ích.”

**Vocab (từ vựng):** {'học', 'lập', 'trình', 'rất', 'thú', 'vị', 'bổ', 'ích'}

**Dữ liệu huấn luyện (Context Window = 2):**

|  |  |
| --- | --- |
| **Từ trung tâm** | **Từ ngữ cảnh** |
| học | lập |
| học | trình |
| lập | học |
| lập | trình |
| lập | rất |
| rất | lập |
| … | … |

## **Đánh giá:**

Công thức tính cosine similarity giữa hai vector :

**Cosine Similarity**

Trong đó:

* : Hai vector cần so sánh.
* : Độ dài của vector

## **Toàn bộ mã của dự án:**

**Github:** [WNormanTPN/SkipGramModel: Assignment of NLP class](https://github.com/WNormanTPN/SkipGramModel)

<https://github.com/WNormanTPN/SkipGramModel>

**Dataset:** [Vietnamese Online News Dataset](https://www.kaggle.com/datasets/haitranquangofficial/vietnamese-online-news-dataset)

<https://www.kaggle.com/datasets/haitranquangofficial/vietnamese-online-news-dataset>