# MỤC LỤC

[MỤC LỤC 1](#_Toc95296387)

[DANH MỤC HÌNH 3](#_Toc95296388)

[DANH MỤC BẢNG 5](#_Toc95296389)

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 6](#_Toc95296390)

[1.1. Tóm tắt văn bản là gì 6](#_Toc95296391)

[1.2. Ý nghĩa của việc tóm tắt văn bản 6](#_Toc95296392)

[1.3. Các hướng tiếp cận 7](#_Toc95296393)

[1.4. Các phương pháp đánh giá 8](#_Toc95296394)

[CHƯƠNG 2. MỘT SỐ NGHIÊN CỨU VỀ TÓM TẮT VĂN BẢN 9](#_Toc95296395)

[2.1. Tóm tắt văn bản theo hướng trích chọn 9](#_Toc95296396)

[2.1.1. Phương pháp chủ đề đại diện dựa trên tần suất 10](#_Toc95296397)

[2.1.2. Phương pháp đặc trưng đại diện 11](#_Toc95296398)

[2.2. Tóm tắt văn bản theo hướng tóm lược 12](#_Toc95296399)

[CHƯƠNG 3. MẠNG NƠ RON NHÂN TẠO 14](#_Toc95296400)

[3.1. Mạng nơ ron nhân tạo ANN 14](#_Toc95296401)

[3.1.1. Cấu trúc mạng nơ ron nhân tạo 14](#_Toc95296402)

[3.1.2. Hoạt động của mạng ANN 15](#_Toc95296403)

[3.2. Mạng nơ ron hồi quy RNN 17](#_Toc95296404)

[3.3. Mạng nơ ron có nhớ LSTM 19](#_Toc95296405)

[CHƯƠNG 4. XÂY DỰNG HỆ THỐNG TÓM TẮT VĂN BẢN THEO HƯỚNG TÓM LƯỢC 26](#_Toc95296406)

[4.1. Quy trình tóm tắt văn bản theo hướng tóm lược sử dụng mạng LSTM 26](#_Toc95296407)

[4.2. Xây dựng bộ dữ liệu cho tóm tắt văn bản tiếng Việt 27](#_Toc95296408)

[4.3. Word Embedding 30](#_Toc95296409)

[4.3.1. Embedding dựa trên tần suất xuất hiện của từ 31](#_Toc95296410)

[4.3.2. Word2Vec 35](#_Toc95296411)

[4.4. Xây dựng mô hình 40](#_Toc95296412)

[4.5. Thuật toán phân cụm K-means 45](#_Toc95296413)

[4.6. Thuật toán LSA ( Latent Semantic Analysis) 45](#_Toc95296414)

[4.7. Thuật toán TextRank 47](#_Toc95296415)

[CHƯƠNG 5. KẾT QUẢ CÀI ĐẶT 48](#_Toc95296416)

[5.1. Cài đặt thuật toán 48](#_Toc95296417)

[5.2. Cài đặt giao diện web 49](#_Toc95296418)

[CHƯƠNG 6. ĐÁNH GIÁ, HƯỚNG PHÁT TRIỂN 51](#_Toc95296419)

[6.1. Đánh giá 51](#_Toc95296420)

[6.2. Khó khăn 51](#_Toc95296421)

[6.3. Định hướng phát triển 51](#_Toc95296422)

[TÀI LIỆU THAM KHẢO 52](#_Toc95296423)

# DANH MỤC HÌNH

[Hình 2.1. Mô hình sequence-to-sequence với cơ chế attention 9](#_Toc95296303)

[Hình 2.2. Ví dụ văn bản tóm tắt được sinh bởi mô hình pointer-generator networks 9](#_Toc95296304)

[Hình 3.1. Cấu trúc mạng nơ ron nhân tạo 10](#_Toc95296305)

[Hình 3.2. Nguyên lý hoạt động của mạng ANN 12](#_Toc95296306)

[Hình 3.3. Đồ thị của các hàm kích hỏa phổ biến và đạo hàm của chúng 13](#_Toc95296307)

[Hình 3.4. Ví dụ bài toán dự đoán từ 14](#_Toc95296308)

[Hình 3.5. Cấu trúc mạng RNN tiêu biểu 14](#_Toc95296309)

[Hình 3.6. Kiến trúc LSTM 16](#_Toc95296310)

[Hình 3.7. Kiến trúc mạng LSTM 17](#_Toc95296311)

[Hình 3.8. Ống nhớ trong khối LSTM 18](#_Toc95296312)

[Hình 3.9. Cổng bỏ nhớ của LSTM 18](#_Toc95296313)

[Hình 3.10. LSTM tính toán giá trị lưu tại cell state 19](#_Toc95296314)

[Hình 3.11. Cập nhật giá trị Cell State 20](#_Toc95296315)

[Hình 3.12. Đầu ra của khối LSTM 21](#_Toc95296316)

[Hình 4.1. Mô hình bài toán tóm tắt văn bản 22](#_Toc95296317)

[Hình 4.2. Quy trình thực hiện tóm tắt văn bản với LSTM 23](#_Toc95296318)

[Hình 4.3. Thu thập dữ liệu cho tóm tắt văn bản tiếng Việt 24](#_Toc95296319)

[Hình 4.4. Chi tiết top 10 thể loại có nhiều bài viết nhất 26](#_Toc95296320)

[Hình 4.5. Ma trận M được xây dựng theo phương pháp Count vector 29](#_Toc95296321)

[Hình 4.6. Cách hoạt động của CBOW 32](#_Toc95296322)

[Hình 4.7. Mô hình Skip-gram 33](#_Toc95296323)

[Hình 4.8. Kiến trúc mạng mô hình Skip-gram 34](#_Toc95296324)

[Hình 4.9. Ma trận trọng số lớp ẩn word2vec 36](#_Toc95296325)

[Hình 4.10. Lớp ẩn hoạt động như một bảng tra cứu 36](#_Toc95296326)

[Hình 4.11. Tương quan giữa 2 từ thực hiện với word2vec 36](#_Toc95296327)

[Hình 4.12. Mô hình chuỗi sang chuỗi 37](#_Toc95296328)

[Hình 4.13. Mô hình bộ mã hóa – giải mã 38](#_Toc95296329)

[Hình 4.14. Kiến trúc mô hình tóm tắt văn bản tiếng việt sử dụng LSTM 39](#_Toc95296330)

[Hình 4.15. Mục tiêu thuật toán phân cụm 41](#_Toc95296331)

[Hình 4.16. Chi tiết thuật toán LSA 42](#_Toc95296332)

[Hình 4.17. Thuật toán google page rank 43](#_Toc95296333)

[Hình 5.1. Tổng quan giao diện 45](#_Toc95296334)

[Hình 5.2. Giao diện tóm tắt văn bản 46](#_Toc95296335)

# DANH MỤC BẢNG

[**Bảng 4.1. Count Matrix M có kích thước 2x6** 28](#_Toc95296336)

[**Bảng 4.2. Minh họa phương pháp TF-IDF** 30](#_Toc95296337)

[**Bảng 5.1. Kết quả cài đặt đánh giá trên bộ 1000 bài viết** 45](#_Toc95296338)

# GIỚI THIỆU ĐỀ TÀI

Cùng với sự trăng trưởng mạnh mẽ của mạng Internet, con người ngày càng bị quá tải bởi khối lượng lớn các thông tin và tài liệu trực tiuyến. Điều này đã thúc đẩy rất nhiều nghiên cứu về tóm tắt văn bản tự động. Theo Radev và cộng sự, một tóm tắt được định nghĩa như là một văn bản được tạo từ một hoặc nhiều văn bản, truyền đạt các thông tin quan trọng từ các văn bản gốc và thông thường bản tóm tắt có độ dài khá ngắn, ngắn hơn nhiều so với 50% độ dài của văn bản gốc

## Tóm tắt văn bản là gì

Tóm tắt văn bản tự động là tác vụ để tạo ra một tóm tắt chính xác và hợp ngữ pháp trong khi vẫn giữ được các thông tin chính và ý nghĩa của văn bản gốc. Trong các năm gần đây, có rất nhiều hướng tiếp cận đã được nghiên cứu cho tóm tắt văn bản tự động và đã được áp dụng rộng rãi trong nhiều lĩnh vực. Ví dụ, máy tìm kiếm sinh ra các trích đoạn như là các bản xem trước của tài liệu [2], các website tin tức sinh ra các đoạn mô tả ngắn gọn cho bài viết (thường là tiêu đề của bài viết) [20]. Mục tiêu của tóm tắt văn bản là tạo ra bản tóm tắt giống như cách con người tóm tắt, đây là bài toán đầy thách thức, bởi vì khi con người thực hiện tóm tắt một văn bản, chúng ta thường đọc toàn bộ nội dung rồi dựa trên sự hiểu biết và cảm thụ của mình để viết lại một đoạn tóm tắt nhằm làm nổi bật các ý chính của văn bản gốc. Nhưng vì máy tính khó có thể có được tri thức và khả năng ngôn ngữ như của con người, nên việc thực hiện tóm tắt văn bản tự động là một công việc phức tạp.

## Ý nghĩa của việc tóm tắt văn bản

- Giúp phân loại văn bản với văn bản có kích thước chênh lệch quá nhiều (ví dụ độ dài của văn bản dao động từ 5-100 câu → rút gọn các văn bản quá dài về 5-10 câu) hay văn bản quá dài → chỉ lấy ý chính phân loại.

- Tóm tắt tin tức → giới thiệu cho người đọc những gì quan trọng và “tinh túy” nhất từ bài viết.

- Tạo headline tự động: sử dụng trong các hệ thống tổng hợp tin tức tự động, tự tạo ra tiêu đề cho bài viết đầu vào.

- Ứng dụng cho các bài toán tìm kiếm: khi người dùng tìm kiếm 1 cụm từ thì chỉ tìm kiếm trong dữ liệu đã được tóm tắt, vì người dùng thường tìm kiếm những thứ có ý nghĩa và quan trọng.

## Các hướng tiếp cận

Nhìn chung, có hai hướng tiếpcận cho tóm tắt văn bản tự động là trích chọn (extraction) và tóm lược (avtraction). Theo [32], tóm tắt văn bản có thể được phân loại dựa trên đầu vào (đơn hay đa văn bản), mục đích (tổng quát, theo lĩnh vực cụ thể, hay dựa trên truy vấn) và loại đầu ra (trích chọn hay tóm lược).

Phương pháp tóm tắt trích chọn thực hiện đánh giác các phần quan trong của văn bản và đưa chúng một cách nguyên bản vào tóm tắt, do đó, phương pháp này chỉ phụ thuộc vào việc trích chọn các câu từ văn bản gốc dựa trên việc xếp hạng mức độ liên quan của các cụm từ để chỉ chọn những cụm từ liên quan nhất tới nội dung của tài liệu gốc. Trong khi đó, phương pháp tóm tắt tóm lược nhằm tạo ra văn bản tóm tắt mới có thể không gồm các từ hay các cụm từ trong văn bản gốc. Nó cố gắng hiểu và đánh giá văn bản sử dụng các kỹ thuật xử lý ngôn ngữ tự nhiên tiên tiến để tạo ra một văn bản ngắn hơn, truyền đạt được những thông tin quan trọng nhất từ văn bản gốc. Mặc dù các tóm tắt được con người thực hiện thường không giống như trích chọn, song hầu hết các nghiên cứu về tóm tắt văn bản hiện tại vẫn tập trung vào tóm tắt bằng phương pháp trích chọn vì về cơ bản các tóm tắt sinh bởi phương pháp trích chọn cho kết quả tốt hơn so với tóm tắt bằng phương pháp tóm lược. Điều này bởi vì phương pháp tóm tắt bằng tóm lược phải đối mặt với các vấn đề như thể hiện ngữ nghĩa, suy luận và sinh ngôn ngữ tự nhiên, các vấn đề này phức tạp hơn nhiều lần so với việc trích chọn câu, Hướng tiếp cận tóm tắt bằng tóm lược khó hơn so với tóm tắt bằng bằng trích chọn, song phương pháp này được kỳ vọng có thể tạo ra được các văn bản tóm tắt giống như cách con người thực hiện

## Các phương pháp đánh giá

* Đánh giá về cú pháp, ngữ nghĩa của văn bản tóm tắt: đối với hướng tiếp cận theo mô hình sinh, cần đánh giá xem văn bản tóm tắt có nghĩa không, đúng cú pháp, ngữ pháp hay không…
* Đánh giá về ngữ nghĩa: cần đánh giá xem văn bản tóm tắt có giữ được ý chính, những chủ đề quan trọng nhất của văn bản đầu vào. Về phương diện ngữ nghĩa, ta chia phương pháp đánh giá làm 2 loại:

+ Dữ liệu có nhãn- mỗi văn bản đã có 1 hoặc nhiều bài tóm tắt: một số phương pháp được sử dụng để đánh giá độ tương đồng ngữ nghĩa giữa đầu ra của thuật toán và văn bản được tóm tắt bằng tay như: N-gram Co-occurrence Statistics – ROUGE, Longest Common Subsequence hay Unit Overlap.

+ Dữ liệu không có nhãn: cách đánh giá trên dữ liệu này chủ yếu dựa trên độ tương đồng ngữ nghĩa giữa văn bản đầu vào và văn bản đầu ra của mô hình. Phương pháp content-based dựa trên độ tương đồng ngữ nghĩa trực tiếp giữa văn bản tóm tắt và văn bản được tóm tắt. Main Topic Similarity dựa trên chính mô hình chủ đề LSA, đánh giá xem liệu văn bản tóm tắt có giữ được những chủ đề chính của văn bản đầu vào.

# MỘT SỐ NGHIÊN CỨU VỀ TÓM TẮT VĂN BẢN

## Tóm tắt văn bản theo hướng trích chọn

Như đã đề cập trong chương 1, các kỹ thuật tóm tắt bằng trích chọn sinh ra các đoạn tóm tắt bằng cách chọn một tập các câu trong văn bản gốc. Các đoạn tóm tắt này chứa các câu quan trọng nhất của đầu vào. Đầu vào có thể là đơn văn bản hoặc đa văn bản. Trong khuôn khổ của luận văn này, đầu vào của bài toán tóm tắt văn bản là đơn văn bản.

Các hệ thống tóm tắt văn bản theo hướng trích chọn thường gồm các tác vụ: xây dựng một đại diện trung gian (intermediate representation) của văn bản đầu vào thể hiện các đặc điểm chính của văn bản; tính điểm (xếp hạng) các câu dựa trên đại diện trung gian đã xây dựng; chọn các câu đơn vào tóm tắt [23].

Mỗi hệ thống tóm tắt văn bản tạo ra một số đại diện trung gian của văn bản mà nó sẽ thực hiện tóm tắt và tìm các nội dung nổi bật dựa trên đại diện trung gian này. Có hai hướng tiếp cận dựa trên dại diện trung gian là chủ đề đại diện (topic representation) và các đặc trưng đại diện (indicator representation). Các phương pháp dựa trên chủ đề đại diện biến đổi văn bản đầu vào thành một đại diện trung gian và tìm kiếm các chủ đề được thảo luận trong văn bản. Kỹ thuật tóm tắt dựa trên chủ đề đại diện tiêu biểu là phương pháp tiếp cận dựa trên tần xuất (frequency). Phương pháp dựa trên các đặc trưng đại diện thực hiện mô tả các câu trong văn bản như một danh sách các đặc trưng quan trọng chẳng hạn như độ dài câu, vị trí của câu trong tài liệu hay câu có chứa những cụm từ nhất định. Khi các đại diện trung gian đã được tạo ra, một điểm số thể hiện mức độ quan trọng sẽ được gán cho mỗi câu. Đối với phương pháp dựa trên chủ đề đại diện, điểm số của một câu thể hiện mức độ giải thích của câu đối với một vài chủ đề quan trọng nhất của văn bản. Trong hầu hết các phương pháp dựa trên đặc trưng đại diện, điểm số được tính bằng tổng hợp các dấu hiệu từ các đặc trưng khác nhau. Các kỹ thuật học máy thường được sử dụng để tìm trọng số cho các đặc trưng.

Cuối cùng hệ thống tóm tắt sẽ lựa chọn các câu quan trọng nhất để tạo ra bản tóm tắt. Có thể áp dụng các thuật toán tham lam để chọn các câu quan trọng nhất từ văn bản gốc, hoặc biến việc lựa chọn câu thành một bài toán tối ưu trong đó xem xét ràng buộc tối đa hóa tầm quan trọng tổng thể và sự gắn kết ngữ nghĩa trong khi tối thiểu hóa sự dư thừa. Có nhiều yếu tố khác cần được cân nhắc khi lựa chọn các câu quan trọng, ví dụ ngữ cảnh của bản tóm tắt hay loại tài liệu cần tóm tắt (bài báo tin tức, email, báo cáo khoa học). Các tiêu chí này có thể trở thành các trọng số bổ sung cho việc lựa chọn các câu quan trọng đưa vào bản tóm tắt.

### Phương pháp chủ đề đại diện dựa trên tần suất

#### Word probability

Xác suất của từ (word probability) là dạng đơn giản nhất sử dụng tần xuất trên văn bản đầu vào như là một chỉ số quan trọng. Phương pháp này khá phụ thuộc vào độ dài của văn bản đầu vào, ví dụ, một từ xuất hiện ba lần trong một văn bản 10 từ có thể là từ quan trọng song có thể nó là một từ bình thường trong văn bản 1000 từ.

Xác suất của một từ w: p(w) được tính dựa trên số lần xuất hiện của từ w, n(w), trong toàn bộ các từ thuộc văn bản đầu vào N.

(2.1)

Hệ thống SumBasic [18] được phát triển dựa trên ý tưởng sử dụng xác suất của từ để tính toán câu quan trọng. Với mỗi câu Sj trong văn bản đầu vào, nó gán một trọng số bằng xác suất trung bình của các từ chứa nội dung trong câu (một danh sách các từ không mang thông tin – stop words – sẽ bị loại khỏi quá trình đánh trọng số):

(2.2)

Tiếp theo nó sẽ chọn các câu có điểm số tốt nhất gồm những từ có xác suất cao nhất. Bước này đảm bảo rằng các từ có xác suất cao nhất đại diện cho chủ đề của văn bản đầu vào sẽ được đưa vào bản tóm tắt. Sau khi chọn một câu đưa vào tóm tắt, xác suất của mỗi từ trong câu được hiệu chỉnh:

(2.3)

Việc hiệu chỉnh này thể hiện rằng xác suất một từ xuất hiện hai lần trong bản tóm tắt là thấp hơn so với xác suất từ xuất hiện chỉ một lần. Quá trình lặp lại cho đến khi đạt được độ dài cần thiết của văn bản tóm tắt.

#### Phương pháp TF-IDF

Phương pháp dựa trên xác suất của từ phụ thuộcvào danh sách stop word để loại bỏ các từ không quan trọng khỏi bản tóm tắt. Việc quyết định từ nào sẽ đưa bào danh sách stop word sẽ ảnh hưởng tới hiệu năng của phương pháp word probability. Phương pháp TF-IDF (Term Frequency – Inverse Document Frequency) đã được nghiên cứu phát triển để giải quyết hạn chế của phương pháp xác suất từ. Phương pháp này sẽ đánh giá đọ quan trọng của một từ bằng cách đánh trọng số cho từ. Các từ quan trọng trong văn bản sẽ được đánh trọng số cao, còn các từ phổ biến trong rất nhiều tài liệu (common words) sẽ được đánh trọng số thấp để loại bỏ khỏi danh sách đánh giá lựa chọn đưa vào văn bản tóm tắt. Trọng số của mỗi từ trong tài liêu d được tính như sau

(2.4)

Trong đó, là term frequency của từ w trong tài liệu d, là số tài liệu chứa từ w và D là tổng số tài liệu. Như vậy, các từ xuất hiện trong hầu hết các tài liệu sẽ có giá trị IDF gần bằng 0. Trọng số TF\*IDF của từ là một chỉ số tốt để đánh giá mức độ quan trọng.

### Phương pháp đặc trưng đại diện

Phương pháp đặc trưng đại diện nhằm mô hình các đại diện của văn bản dựa trên một tập các đặc trưng và sử dụng chúng để xếp hạng các câu của văn bản đầu vào. Các phương pháp dựa trên đồ thị và kỹ thuật học máy thường được sử dụng để quyết định mức độ quan trọng của các câu sẽ đưa vào văn bản tóm tắt.

#### Phương pháp đồ thị cho tóm tắt văn bản

Phương pháo dựa trên đồ thị thể hiện văn bản như là một đồ thị liên thông. Các câu tạo thành các đỉnh của đồ tị và các cạnh giữa các câu thể hiện sự liên quan giữa 2 câu với nhau. Một kỹ thuật thường được sử dụng để nói hai đỉnh đó là đo lường sự tương đồng giữa hai câu và nếu nó lớn hơn một ngưỡng nhất định thì chúng liên thông nhau. Đồ thị này thể hiện kết quả ở hai phần: thứ nhất, một phần đồ thị con được tạo bởi các thủ đề rởi rạc trong văn bản; thứ hai, các câu được kết nối tới nhiều cây khác trong đồ thị là các câu quan trọng có thể lựa chọn đưa vào văn bản tóm tắt. Một phương pháp dựa trên đồ thị tiêu biểu đó là TextRank [24].

Phương pháp dựa trên đồ thị không cần các kỹ thuật xử lý ngôn ngữ tự nhiên đặc thù cho từng ngôn ngữ ngoài việc tách câu và từ, nên nó có thể áp dụng cho nhiều ngôn ngữ khác nhau.

#### Kỹ thuật học máy cho tóm tắt văn bản

Phương pháp áp dụng học máy cho tóm tắt văn bản thực hiện giải bài toán phân loại ,nhị phân. Tư tưởng của chúng là phân loại các câu trong văn bản đầu vào gồm hai tập là tập các câu tóm tắt và tập các câu không là tóm tắt dựa vào các đặc trưng mà chúng có. Tập dữ liệu huấn luyện gồm các văn bản và các tóm tắt trích chọn tương ứng.

Xác suất một câu được chọn vào văn bản tóm tắt là điểm số của câu. Việc lựa chọn các hàm phân loại đóng vai trò quan trọng trong việc tính điểm cho các câu. Một số đặc trưng phân loại thường được sử dụng trong tóm tắt văn bản gồm có vị trí của câu trong văn bản, độ dài của câu, tồn tại của các từ viết hoa, độ tương đồng của câu với tiêu đề của văn bản… Có nhiều kỹ thuật học máy được áp dụng trong tóm tắt văn bản, tiêu biểu là áp dụng của mô hình Markov ẩn (Hidden Markov Model) [14].

## Tóm tắt văn bản theo hướng tóm lược

Những năm gần đây với sự phát triển của phần cứng máy tính, cùng với nhiều kỹ thuật tiên tiến dựa trên mạng nơ ron nhân tạo và kiến trúc mạng học sâu, một số nghiên cứu về tóm tắt văn bản bằng tóm lược đã được thực hiện với mục tiêu tạo được văn bản tóm tắt giống như cách con người thực hiện.

Nallapati và cộng sự [22] áp dụng mô hình chuỗi sang chuỗi (sequence-tosequence) với cơ chế attention kết hợp với các đặc trưng ngôn ngữ (part-of-speech, name-entity và TF-IDF) để thực hiện tóm tắt văn bản theo hướng tóm lược (hình 2.1). Kết quả cho thấy mô hình có khả năng sinh ra các từ không có trong văn bản đầu vào, nhiều ví dụ cho thấy mô hình có thể sinh ra được đoạn tóm tắt gần giống với con người viết.

Diagram

Description automatically generated

Hình 2.1. Mô hình sequence-to-sequence với cơ chế attention

Tác giả See và cộng sự trong [28] đề xuất cải tiến mạng pointer-generator trên mô hình chuỗi sang chuỗi cho phép thực hiện sao chép một (các từ) từ văn bản gốc vào văn bản tóm tắt trong trường hợp mô hình sinh ra một từ không có trong tập từ vựng (unknown word). Mô hình được thử nghiệm trên bộ dữ liệu tiếng anh các bàibáo của CNN/DailyMail cho kết quả khá khả quan.

Text

Description automatically generated

Hình 2.2. Ví dụ văn bản tóm tắt được sinh bởi mô hình pointer-generator networks

# MẠNG NƠ RON NHÂN TẠO

## Mạng nơ ron nhân tạo ANN

Mạng nơ ron nhân tạo (ANN – Artificial Neural Network) là một mô phỏng xử lý thông tin, được nghiên cứu ra từ hệ thống thần kinh của con người, giống như bộ não để xử lý thông tin. Mạng ANN bao gồm số lượng lớn các mối gắn kết cấp cao để xử lý các thông tin trong mối liên hệ rõ ràng. Nó có khả năng học bởi kinh nghiệm từ huấn luyện, lưu những kinh nghiệm thành tri thức và áp dụng trong những dữ liệu mới trong tương lai.

### Cấu trúc mạng nơ ron nhân tạo

Mỗi nơ ron (gọi là nút mạng) là yếu tố cơ bản nhất cấu tạo nên mạng nơ ron, tham gia vào xử lý thông tin trong mạng. Các nơ ron trong mạng liên kết với nhau, xử lý và chuyển tiếp thông tin dựa trên các trọng số liên kết và hàm kích hoạt.

Diagram

Description automatically generated

Hình 3.1. Cấu trúc mạng nơ ron nhân tạo

Cấu trúc mạng nơ ron nhân tạo về cơ bản gồm ba lớp: lớp đầu vào (input layer), lớp ẩn (hidden layer) và lớp đầu ra (output layer) được minh họa như hình 3.1. Khi một mạng ANN có nhiều hơn hai lớp ẩn thì được gọi là một mạng nơ ron sâu (deep neural network hay DNN) [8].

### Hoạt động của mạng ANN

Đầu vào: dữ liệu vào của mạng ANN tùy thuộc vào ứng dụng mà mô hình cần xử lý. Ví dụ với bài toán kinh điển nhận dạng ký tự viết tay, đầu vào là các ảnh chụp các số viết tay từ 0 đến 9. Đầu ra của mạng ANN là lời giải cho bài toán cần giải quyết, ví dụ với bài toán nhận dạng ký tự chữ viết tay thì đầu ra sẽ là dự đoán tương ứng cho ảnh đầu vào, ví dụ ảnh đầu vào là số 7 viết tay, thì đầu ra là kết quả đúng nếu dự đoán là số 7, và sai nếu trả kết quả là một số khác số 7 (ví dụ số 1 hay số 4).

Hoạt động của mạng ANN được minh họa trong hình 3.2 [15]. Thông tin tới một nơ ron được nhân với một trọng số (mỗi đầu vào có thể được nhân với một trọng số khác nhau), sau đó nơ ron sẽ tính tổng các đầu vào đã tính trọng số và tham số hiệu chỉnh (bias) và xử lý tổng này thông qua một hàm kích hoạt (activation function) hay còn gọi là chuyển đổi (transfer function).

Diagram

Description automatically generated

Hình 3.2. Nguyên lý hoạt động của mạng ANN

Quá trình tính toán được thực hiện bằng công thức:

Trong đó: là giá trị đầu vào tại từng thời điểm k, là giá trị trọng số của đầu vào i, b là tham số hiệu chỉnh (bias), F là một hàm kích hoạt và y(k) là giá trị đầu ra tương ứng

Một số hàm kích hoạt thường được sử dụng là hàm bước nhảy (step function), hàm logit (hay hàm sigmoid), hàm tanh và hàm Rectified Linear Unit (ReLU) [8].

Đồ thị của các hàm kích hoạt này và đạo hàm của nó được thể hiện trong hình 3.3. [8].

Chart

Description automatically generated

Hình 3.3. Đồ thị của các hàm kích hỏa phổ biến và đạo hàm của chúng

## Mạng nơ ron hồi quy RNN

Trong một mạng nơ ron truyền thống, chúng ta giả sử rằng tất cả các dữ liệu đầu vào và dữ liệu đầu ra là độc lập với nhau, nhưng trong nhiều bài toán thực tế thì giả sử này hoàn toàn sai. Ví dụ trong bài toán dự đoán từ tiếp theo trong một câu như minh họa hình 3.4 [5], việc suy diễn sẽ có căn cứ và có xác suất đúng ngữ cảnh là cao hơn nếu biết các từ trước nó. Việc sử dụng thông tin có tính chuỗi tuần tự chính là tư tưởng cho việc nghiên cứu và phát triển mạng nơ ron hồi quy RNN (Recurrent Neural Network) [6].

Graphical user interface

Description automatically generated with medium confidence

Hình 3.4. Ví dụ bài toán dự đoán từ

Các mạng RNN được gọi là hồi quy (hay hồi tiếp) bởi vì chúng thực thi cùng một tác vụ cho mỗi thành phần của chuỗi với đầu ra phụ thuộc vào các kết quả tính toán trước đó. Có thể hiểu theo một cách khác đó là RNN có bộ nhớ mà đã lưu trữ các thông tin đã xử lý trước đó. Về lý thuyết thì mạng RNN có thể xử lý thông tin cho một chuỗi dài tùy ý, song trên thực tế thì khả năng này khá giới hạn trong chỉ vài bước [6]. Một mạng RNN tiêu biểu có cấu trúc như hình 3.5:

A picture containing text, clock, gauge

Description automatically generated

Hình 3.5. Cấu trúc mạng RNN tiêu biểu

Hình 3.5 minh họa một mạng RNN trải ra thành một mạng đầy đủ [6]. Điều này có nghĩa là ta có thể trải một mạng RNN để xử lý cho một chuỗi đầy đủ. Ví dụ, nếu một chuỗi là một câu gồm năm từ, thì mạng có thể trải ra thành năm lớp mạng nơron, mỗi lớp xử lý một từ. Các công thức tính toán trong mạng RNN cụ thể như sau:

- là đầu vào tại thời điểm t, ví dụ, có thể là một véc tơ one-hot tương ứng với từ thứ hai của một câu.

- là trạng thái ẩn tại thời điểm t. Nó giống như là bộ nhớ của mạng, st được tính dựa vào trạng thái ẩn trước đó và đầu vào của bước hiện tại: = f(U). Hàm f thường là một hàm phi tuyến như là hàm tanh hoặc hàm ReLU, thường được khởi tạo là 0 khi tính toán trạng thái ẩn thứ nhất.

- là đầu ra (output) tại bước t. Ví dụ với bài toán dự đoán từ tiếp theo trong câu thì có thể là một véc tơ xác suất các từ trong từ điển: = softmax(V).

Không giống với mạng nơ ron thông thường với các tham số khác nhau tại mỗi lớp mạng (layer), mạng RNN sử dụng cùng một bộ tham số (U, V, W) trong tất cả các bước. Điều này ám chỉ rằng nó sẽ thực hiện cùng một tác vụ tại mỗi bước, nhưng với các đầu vào khác nhau. Chính đặc trưng này làm giảm đi đáng kể số lượng các tham số cần học trong mạng. Mạng RNN có thể có đầu ra tại mỗi bước, nhưng tùy theo bài toán cần xử lý mà các kết quả này có cần thiết hay không; tương tự với đầu vào, mạng RNN không nhất thiết cần có đầu vào tại mỗi thời điểm. Đặc trưng quan trọng nhất của RNN là trạng thái ẩn của nó, với khả năng nắm giữ thông tin về một chuỗi liên tiếp [6].

## Mạng nơ ron có nhớ LSTM

Khi quan sát một mạng nơ ron, chức năng của nó giống như một chiếc hộp đen. Dữ liệu được đưa vào một chiều và kết quả được đưa ra ở một chiều khác, quá trình quyết định đưa ra kết quả chỉ phụ thuộc vào các đầu vào hiện tại. Nhìn chung, mạng nơ ron không hoàn toàn là không có khả năng nhớ, vì về cơ bản chúng học các tham số logic trong quá trình huấn luyện [29]. Tuy nhiên khả năng nhớ này là rất hạn chế và không phù hợp đối với các trường hợp khi cần sử dụng trạng thái nhớ trung gian để sử dụng sau này, ví dụ như tóm tắt nội dung chính của bài báo.

Cách cơ bản nhất để một mạng nơ ron chấp nhận dữ liệu theo thời gian (time seriesdata) đó là kết nối vài mạng nơ ron lại với nhau, mỗi mạng nơ ron xử lý một bước theo thứ tự thời gian. Tức là thay vì đưa dữ liệu đầu vào rời rạc, dữ liệu được đưa theo một cửa sổ thời gian, hay một ngữ cảnh, vào mạng nơ ron.

Về lý thuyết thì mạng nơ ron hồi tiếp (recurrent neural network) có thể hoạt động, nhưng thực tế trong nhiều nghiên cứu chỉ ra hạn chế của mạng RNN là sự hội tụ và phân kỳ gradient (vanishing gradient và exploding gradient) [29]. Hạn chế này khiến RNN không hiệu quả đối với các bài toán cần xử lý dữ liệu theo thời gian đòi hỏi trạng thái nhớ trung gian.

LSTM (Long short term memory) [12] ra đời để giải quyết hạn chế của RNN bằng

việc đưa vào mạng một đơn vị nhớ được gọi là memory unit hay Cell.

Diagram

Description automatically generated

Hình 3.6. Kiến trúc LSTM

Kiến trúc một khối LSTM được thể hiện trên Hình 3.6 [4]. Đầu vào gồm ba thành phần. Xt là đầu vào tại bước hiện tại. là đầu ra từ một khối LSTM trước và là “nhớ” của khối trước, và đây cũng chính là điểm quan trọng nhất của LSTM

Đầu ra của nó gồm ht là kết quả của khối LSTM hiện tại và Ct là nhớ của nó. Như vậy, một khối đơn LSTM đưa ra quyết định dựa vào việc xem xét đầu vào hiện tại, kết quả và nhớ của khối trước và nó sinh ra một đầu ra mới cũng như là nhớ của nó.

Một mô hình mạng LSTM [4] được minh họa trong hình 3.7.

Diagram

Description automatically generated

Hình 3.7. Kiến trúc mạng LSTM

Điểm quan trọng nhất của LSTM chính là trạng thái nhớ (cell state), thể hiện ở đường kẻ ngang trên cùng của Hình 3.8.

Véc tơ nhớ được đưa vào một ống nhớ (memory pipe) qua một cổng gọi là

cổng bỏ nhớ (forget gate), cổng bỏ nhớ thực chất là một toán hạng nhân ma trận

(element-wise multiplication operation). sẽ được nhân với một véc tơ, và nếu

kết quả là gần 0, thì kết quả nhớ Ct-1 sẽ bị loại bỏ, ngược lại nếu kết quả là 1 thì

sẽ được đi tiếp. Hình 3.8 minh họa hoạt động của ống nhớ trong khối LSTM [4].

A picture containing text, clock

Description automatically generated

Hình 3.8. Ống nhớ trong khối LSTM

Cụ thể cách hoạt động của LSTM [4] như sau:

Đầu tiên khối LSTM là quyết định thông tin nào sẽ loại bỏ khỏi cell state. Quá trình quyết định này do một lớp sigmoid gọi là “forget gate layer” thực hiện. Cổng bỏ nhớ lấy đầu vào là h−1 và và cho đầu ra là một giá trị nằm trong khoảng [0, 1] cho cell state −1. Nếu kết quả đầu ra là 1 thể hiện cho việc “giữ lại thông tin”, và 0 thể hiện rằng “thông tin bị loại bỏ”.

A picture containing text, clock

Description automatically generated

Hình 3.9. Cổng bỏ nhớ của LSTM

Tiếp theo LSTM quyết định thông tin mới sẽ được lưu lại tại cell state như thế nào.

Việc này được gồm hai phần, một là lớp sigmoid gọi là “input gate layer” (lớp đầu vào) quyết định giá trị sẽ được cập nhật, và một lớp tanh tạo ra một véc tơ các giá trị mới, , mà có thể được thêm vào cell state.

A picture containing graphical user interface

Description automatically generated

Hình 3.10. LSTM tính toán giá trị lưu tại cell state

Kế tiếp, trạng thái cell state cũ được cập nhật tại trạng thái cell state mới theo công thức:

Trạng thái nhớ cũ được nhân với giá trị kết quả của cổng bỏ nhớ ft, thực hiện việc loại bỏ những gì đã được quyết định loại bỏ ở bước trước. Giá trị thể hiện giá trị ứng viên mới cho cell state được quyết định bởi hệ số giãn nở it cụ thể cho việc cập nhật giá trị cho mỗi cell state. Hình 3.11 minh họa việc cập nhật giá trị cho cell state tại bước này

A picture containing text, clock

Description automatically generated

Hình 3.11. Cập nhật giá trị Cell State

Bước cuối cùng, khối LSTM quyết định đầu ra của nó dựa trên cell state được minh họa trong hình 3.12. Lớp sigmoid được dùng để tính toán thành phần của cell state sẽ được xuất ra. Sau đó, giá trị cell state được đưa vào hàm tanh (kết quả sẽ thuộc khoảng [-1,1]) và nhân với kết quả đầu ra của cổng sigmoid, để quyết định cái gì sẽ được khối LSTM xuất ra. Công thức tính toán cho các thành phần của bước này như sau:

3.8

A picture containing text, clock

Description automatically generated

Hình 3.12. Đầu ra của khối LSTM

Mạng LSTM là kết hợp của các khối LSTM kết nối kế tiếp nhau qua theo chuỗi thời gian. Hoạt động của mỗi khối LSTM tại một thời điểm được đảm trách bởi các cổng: cổng bỏ nhớ ft, cổng đầu vào it và cổng đầu ra ot, trong đó cổng bỏ nhớ chính là điểm đáng chú ý nhất của LSTM, đem lại khả năng sử dụng thông tin tính toán từ các thời điểm trước đó.

# XÂY DỰNG HỆ THỐNG TÓM TẮT VĂN BẢN THEO HƯỚNG TÓM LƯỢC

Bài toán tóm tắt văn bản theo hướng tóm lược có thể được phát biểu như sau: đầu vào của bài toán là một văn bản x gồm M từ: , ,, …, . Chúng ta sẽ ánh xạ chuỗi M từ này thành một chuỗi đầu ra y gồm N từ: , ,, …, ; trong đó N< M dựa trên một tập từ vựng ccó kích thước cố định V. Các từ thuộc N không nhấtt định phải thuộc M. Mục tiêu là tìm một chuỗi đầu ra y làm cực đại hóa xác suất có điều kiện của y theo chuỗi đầu vào x:

(4.1)

Hình 4.1 minh họa mô hình bài toán tóm tắt văn bản tự động.

A picture containing text, clock, screenshot

Description automatically generated

Hình 4.1. Mô hình bài toán tóm tắt văn bản

## Quy trình tóm tắt văn bản theo hướng tóm lược sử dụng mạng LSTM

Để thực hiện tóm tắt văn bản theo hướng tóm lược sử dụng mạng LSTM, chúng tôi thực hiện các bước như sau:

- Thu thập dữ liệu phù hợp: dữ liệu phù hợp cho bài toán tóm tắt văn bản tiếng việt áp dụng LSTM là bộ dữ liệu gồm một cặp tương ứng: văn bản đầy đủ và văn bản tóm tắt mẫu (do con người thực hiện tóm tắt).

- Xử lý dữ liệu: làm sạch dữ liệu, loại bỏ các ký tự không cần thiết, các lỗi phân tách câu.

- Word embedding: véc tơ hóa dữ liệu về dạng số để đưa vào mô hình LSTM30

- Xây dựng mô hình LSTM: xây dựng mô hình xử lý chuỗi văn bản đầu vào, mục tiêu là tạo ra chuỗi văn bản tóm tắt bằng cách áp dụng các khối LSTM.

- Huấn luyện và đánh giá mô hình sử dụng bộ dữ liệu đã được xử lý phía trên.

Các bước được tiến hành như thể hiện trong hình 4.2, chi tiết các bước được thể hiện trong các mục tiếp theo của luận văn.

Diagram

Description automatically generated

Hình 4.2. Quy trình thực hiện tóm tắt văn bản với LSTM

## Xây dựng bộ dữ liệu cho tóm tắt văn bản tiếng Việt

Bài toán tóm tắt văn được đã được rất nhiều tác giả nghiên cứu, đặc biệt là đối với tóm tắt văn bản tiếng Anh. Với tóm tắt văn bản tiếng Anh, bộ dữ liệu kinh điển được sử dụng là bộ dữ liệu Gigaword với khoảng bốn triệu bài báo (Graff và các cộng sự, 2003 [9]), chi phí mua giấy phép sử dụng bộ dữ liệu này là 6,000 USD nên chỉ có những tổ chức lớn mới có khả năng tiếp cận kho dữ liệu này. Một kho dữ liệu khác thường được sử dụng cho tóm tắt văn bản tiếng Anh đó là bộ dữ liệu các bài báo của CNN/Daily Mail với hơn 90,000 bài báo CNN và hơn 200,000 bài báo Daily Mail [11].

Tuy nhiên, đối với tóm tắt văn bản tiếng Việt, hiện tại chưa có kho dữ liệu chính thức nào được công bố, đây là thách thức lớn đối với chúng tôi. Vì vậy, để chuẩn bị dữ liệu thực hiện bài toán tóm tắt văn bản tiếng Việt, chúng tôi tiến hành thu thập dữ liệu là các bài báo trên một số website tin tức của Việt Nam. Dữ liệu mà chúng tôi quan tâm đó là phần tóm tắt dưới tiêu đề của bài báo, và nội dung văn bản của bài báo.

Hình 4.3 minh họa một website mà chúng tôi thực hiện thu thập dữ liệu.

Graphical user interface, website

Description automatically generated

Hình 4.3. Thu thập dữ liệu cho tóm tắt văn bản tiếng Việt

Hầu hết các trang báo hiện tại đều có một phần “sapo” cho bài báo tương ứng. Đoạn Sapo này có thể được viết bởi 1 hoặc nhiều câu văn hoàn chỉnh khác nhau. Những câu văn này có thể ngắn, có thể dài nhưng nó phải mang tính khái quát để người đọc hiểu được nội dung phần thông tin mà bạn cung cấp phía dưới ([Sapo Là Gì?](https://vietbaixuyenviet.com/sapo-la-gi/) ). Tuy phần sapo không phải cũng như không tóm tắt đầy đủ nội dung của bài viết nhưng đa số sapo thể hiện được nội dung của bài viết (trực tiếp hoặc gián tiếp). Do vậy ta có thể sử dụng dữ liệu nội dung bài viết – sapo của bài viết tương ứng để làm dữ liệu đầu vào cho bài toán tóm tắt văn bản. Với hy vọng mô hình có thể tự động tạo sapo cho bài viết dựa vào nội dung bài viết.

Đề tài tập trung crawl dữ liệu từ trang [soha.vn](https://soha.vn/) do số lượng bài viết lớn, uy tín và các bài viết đều có phần sapo tương ứng.

Về công cụ, Scrapy (python) là một công cụ rất mạnh với khả năng song song hóa cao, khả năng xử lý lỗi hiệu quả. Dựa trên cấu trúc cây XML, mỗi phần của bài viết (sapo, title, nội dung…) sẽ có một địa chỉ XPath tương ứng từ đó Scrapy sẽ giúp ta lấy được các thành phần này.

Đầu vào của Scrapy là danh sách các đường dẫn của bài viết nên việc thu thập đường dẫn cũng là một phần quan trọng. Để thuận tiện, ta sử dụng đường dẫn của các bài báo chứa ngay trong các bài đang crawl (sử dụng cấu trúc tập hợp để loại bỏ các đường dẫn đã crawl), để extract ra các đường dẫn này ta sử dụng biểu thức chính quy (regex). Phương pháp này giúp ta “duyệt theo chiều rộng” các bài viết trong cơ sở dữ liệu của trang web này với mỗi node là một bài viết tương ứng.

Với dữ liệu thu được từ các website tin tức trực tuyến của Việt Nam, chúng tôi tiến hành tiền xử lý để làm sạch dữ liệu và loại bỏ các ký tự nhiễu trong văn bản như sau:

* Loại bỏ các dấu gạch đầu dòng, các dấu gạch ngang trong văn bản.
* Loại bỏ các dấu hai chấm “:” trước mỗi danh sách liệt kê.
* Loại bỏ các dấu ba chấm, các dấu ngoặc đơn và phần chú thích thêm trong ngoặc đơn, các dấu nháy đơn, các dấu nháy kép.
* Thay thế các dấu chấm phẩy “;” phân tách ý thành dấu chấm ngắt câu “.”
* Thêm dấu chấm kết thúc câu cho những chú thích dưới ảnh không có dấu kết thúc câu.
* Tách các câu trong phần tóm tắt của bài báo bằng phân tách các câu dựa trên kết thúc câu bởi dấu chấm, dấu chấm hỏi và dấu chấm than.
* Tách văn bản thành các token.
* Chuyển đổi các phần tóm tắt và bài báo từ dạng văn bản thông thường thành dạng nhị phân và ghi vào file.

Sau khoảng 12 tiếng crawl, với 1 trang web khởi tạo đã duyệt qua khoảng 300000 link, 196497 link thành công (các link không có nội dung hoặc sapo được coi là không thành công).

Chart, icon

Description automatically generated

Hình 4.4. Chi tiết top 10 thể loại có nhiều bài viết nhất

Chi tiết dữ liệu đã crawl: [link google drive](https://drive.google.com/file/d/1SuYzCb8kpNQRkM24nLdStGp1c9iwPBnQ/view)

## Word Embedding

Word embedding là quá trình chuyển đổi văn bản thành các con số và có thể có nhiều đại diện dạng số khác nhau thể hiện cùng một văn bản. Word embedding là kỹ thuật để thể hiện các từ thành các véc tơ có kích thước cố định, sao cho các từ có nghĩa tương tự hoặc gần nghĩa được thể hiện bằng các véc tơ gần nhau (tính theo khoảng cách euclid) [13].

Nhiều thuật toán học máy và hầu hết tất cả các kiến trúc học sâu (deep learning) không thể xử lý trực tiếp các xâu hay các văn bản thông thường. Chúng yêu cầu đầu vào là các con số để thực thi các tác vụ của mình như phân loại văn bản, dịch Word embedding về cơ bản sẽ thực hiện ánh xạ một từ trong một từ điển thành một véc tơ [27]. Chính vì vậy có thể hiểu word embedding là quá trình véc tơ hóa một từ, hay tổng quát là véc tơ hóa văn bản. Một véc tơ đại diện của một từ có thể là một véc tơ one-hot, véc tơ one-hot chỉ có một giá trị là 1, còn lại tất cả đều là 0, giá trị 1 thể hiện vị trí của từ trong từ điển.

Ví dụ, giả sử ta có hai câu: “ Tôi thích chơi piano” và “Tôi thích chơi guitar”.

Đầu tiên chúng ta tách chọn các từ phân biệt trong hai câu, tập các từ phân biệt thu được là tập V = { Tôi, thích, chơi, piano, guitar} gồm 5 từ. Các từ trong tập V có thể được véc tơ hóa dạng one-hot như sau:

Tôi = [1,0,0,0,0], thích = [0,1,0,0,0], chơi = [0,0,1,0,0], piano = [0,0,0,1,0] , guitar = [0,0,0,0,1].

Phần tiếp theo luận văn sẽ giới thiệu một số kỹ thuật word embedding.

### Embedding dựa trên tần suất xuất hiện của từ

#### Count vector

Xem xét một corpus C của D tài liệu (d1, d2, …, dN) và N token phân biệt được trích chọn từ tập từ vựng C [27] . N token sẽ thể hiện từ điển và kích thước của Count vector ma trận M sẽ được xác định bằng D x N. Mỗi dòng trong ma trận M gồm tần xuất xuất hiện của token trong một tài liệu Di. Giả sử

D1: He is a lazy boy. She is also lazy.

D2: Tom is a lazy person.

Từ điển được tạo có thể là một danh sách các token phân biệt trong corpus= [‘He’, ‘She’, ‘lazy’, ‘boy’, ‘Tom’, ‘person’]

Ta có D = 2 và N = 6. Count matrix M có kích thước 2x6 được thể hiện trong bảng 4.1:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | He | She | lazy | boy | Tom | person |
| D1 | 1 | 1 | 2 | 1 | 0 | 0 |
| D2 | 0 | 0 | 1 | 0 | 1 | 1 |
|  |  |  |  |  |  |  |

**Bảng 4.1. Count Matrix M có kích thước 2x6**

Một cột có thể được hiểu như là một véc tơ từ cho một từ tương ứng trong ma trận M. Ví dụ, véc tơ cho từ ‘lazy’ trong ma trận trên là [2,1]. Và một dòng thể hiện tương ứng cho một tài liệu trong corpus và các cột tương ứng là các token trong từ điển.

Việc xây dựng ma trận M như trên phụ thuộc vào hai yếu tố: cách từ điển được xây dựng và cách đếm của các từ [27]. Thứ nhất, một corpus trong thực tế có thể gồm hàng triệu tài liệu, và với số lượng tài liệu lớn như vậy thì hàng trăm triệu từ phân biệt có thể được trích chọn ra. Do đó, ma trận M xây dựng như trên sẽ rất thưa và không hiệu quả cho việc tính toán. Vì lý do này, một từ điển thường được xây dựng từ khoảng 10000 từ dựa trên tần xuất xuất hiện của nó trong corpus. Thứ hai, cách đếm một từ có thể được tính bằng số lần từ đó xuất hiện trong tài liệu hoặc có mặt của từ đó trong tài liệu. Cách thức đếm tần xuất xuất hiện của từ trong tài liệu thường được dùng hơn, vì nó cũng tương đồng với cách xây dựng từ điển. Hình 4.4 dưới dây thể hiện hình ảnh của ma trận M:

Diagram

Description automatically generated

Hình 4.5. Ma trận M được xây dựng theo phương pháp Count vector

#### Phương pháp vector hóa TF-IDF

TF-IDF là viết tắt của term frequency–inverse document frequency đây là phương pháp khác dựa trên tần xuất xuất hiện của từ nhưng có cách véc tơ hóa khác so với count vector, đó là nó không chỉ xem xét sự xuất hiện của một từ trong một tài liệu mà trong toàn bộ corpus [27].

Các từ phổ biến (common words) như ‘is’, ‘the, ‘a’,… thường có tần xuất xuất hiện cao hơn so với các từ quan trọng trong một tài liệu. Ví dụ, một tài liệu X về Ambert Einstein thì từ ‘Einstein’ có tần xuất xuất hiện cao hơn trong các tài liệu khác, nhưng các từ phổ biến như ‘the’ luôn có tần xuất cao hơn trong hầu hết các tài liệu.

Ý tưởng của phương pháp này là chúng ta sẽ đặt trọng số thấp cho các từ phổ biến

xuất hiện trong hầu hết các tài liệu và đặt trọng số cao cho các từ mà chỉ xuất hiện

trong một số tài liệu của tập tài liệu đang xét [27]. Xét hai tài liệu D1 và D2 với tần

xuất của các token được cho ở Bảng 4.2 dưới đây:

|  |  |
| --- | --- |
| Tài liệu D1 | |
| Token | Count |
| This | 1 |
| Is | 1 |
| About | 2 |
| Einstein | 4 |

|  |  |
| --- | --- |
| Tài liệu D2 | |
| Token | Count |
| This | 1 |
| Is | 2 |
| About | 1 |
| Me | 1 |

**Bảng 4.2. Minh họa phương pháp TF-IDF**

TF thể hiện sự đóng góp của từ trong một tài liệu, tức là các từ liên quan tới tài liệu thì sẽ xuất hiện nhiều lần trong tài liệu. Công thức tính TF được tính như sau:

TF = (số lần xuất hiện của token T trong một tài liệu) / (tổng số token trong tài liệu đó).

Áp dụng công thức trên cho token “This” ta có TF(This.D1) = 1/8 và TF(This.D2) = 1/5.

IDF = log(N/n) trong đó N là tổng số tài liệu xem xét và n là tổng số tài liệu chứa token T. Ta có IDF(This) = log(2/2) = 0.

IDF(Einstein) = log(2/1) = 0.301

TF-IDF(This.D1) = (1/8)\*(0) = 0

TF-IDF(This.D2) = (1/5)\*(0) = 0

TF-IDF(Einstein.D1) = (4/8)\*0.301 = 0.15

Phương pháp TF-IDF đánh giá một từ nếu xuất hiện trong tất cả các tài liệu thi khả năng từ đó không liên quan tới một tài liệu cụ thể, nhưng nếu một từ chỉ xuất hiện trong một vài tài liệu thì từ đó có khả năng là một từ quan trọng trong tài liệu chứa nó.

### Word2Vec

Trong rất nhiều bài toán xử lý ngôn ngữ tự nhiên, các từ thường được đại diện bằng điểm TF-IDF. Mặc dù các điểm này mang lại ý tưởng về độ quan trọng tương ứng của các từ trong một văn bản, chúng không thể hiện được ngữ nghĩa của các từ.

Word2vec [1] là một kỹ thuật trong đó áp dụng một lớp mạng nơ ron cùng với một tập dữ liệu huấn luyện không đánh nhãn, để tạo ra một véc tơ cho mỗi từ trong tập dữ liệu chứa cả những thông tin về ngữ nghĩa. Các véc tơ này hữu ích vì hai yếu tố quan trọng của chúng:

- Chúng ta có thể đo lường độ tương đồng ngữ nghĩa giữa hai từ bằng cách đo độ tương đồng cosine giữa hai véc tơ tương ứng.

- Chúng ta có thể sử dụng các véc tơ như là các đặc trưng cho các bài toán xử lý ngôn ngữ tự nhiên có giám sát như phân loại văn bản hay phân tích quan điểm.

Ví dụ, các từ đồng nghĩa thường có các véc tơ khá tương đồng dựa trên độ tương đồng cosine và các từ trái nghĩa thường là các véc tơ hoàn toàn không tương đồng.

Hơn nữa, các véc tơ từ thường có xu hướng tuân theo các luật suy diễn, ví dụ: “Woman is to queen as man is to king” có thể suy ra:

Vqueen – Vwoman + Vman ~ Vking

Trong đó Vquen, Vwoman, Vman, và Vking là lần lượt là các véc tơ từ của queen, woman,

man và king. Qua ví dụ này có thể thấy rằng các véc tơ từ có thể hàm chứa các

thông tin ngữ nghĩa quan trọng của các từ mà chúng đại diện [1].

Word2Vec là phương pháp véc tơ hóa từ do Mikolov và cộng sự nghiên cứu và phát triển [21]. Đây là phương pháp dựa trên dự đoán từ, trong đó cơ sở của việc dự đoán dựa vào xác suất của các từ, độ tương tự và liên quan giữa các từ. Word2Vec kết hợp hai kỹ thuật là CBOW (Continuous bag of words) và mô hình Skip-gram (Skip-gram model). Ý tưởng của word2vec là việc đại diện các từ sử dụng các từ xung quanh từ đó. Điều này tương tự với việc con người biết nghĩa của một từ dựa trên các từ gần nó. Ví dụ xét câu ‘Tôi thích chơi X’, với X là một từ chưa biết. Tuy nhiên, dù chưa biết nghĩa của từ X, nhưng ta có thể biết ‘X’ là một thứ gì đó mà ta có thể ‘chơi’ được và nó cũng tạo cảm giác ‘thích’ [13].

#### CBOW( Continous Bag of Word)

Cách hoạt động của CBOW đó là dự đoán xác suất của một từ được cho trong một ngữ cảnh (context) dựa trên các từ gần nó. Một ngữ cảnh có thể là một từ đơn hoặc một tập các từ.

Diagram

Description automatically generated

Hình 4.6. Cách hoạt động của CBOW

CBOW là một mạng nơ ron nông (Shallow Neural Network) với chỉ 1 lớp ẩn hoạt động như một lớp chiếu (projection layer) của lớp đầu vào. Mục tiêu là để dự đoán được từ đích dựa trên các từ xung quanh nó. Đầu vào của CBOW là N từ, với N là kích thước của cửa sổ của ngữ cảnh được định nghĩa trước và đầu ra là từ dự đoán sử dụng lớp Softmax [13].

#### Mô hình Skip-gram

Diagram

Description automatically generated

Hình 4.7. Mô hình Skip-gram

Skip-gram cũng là một mạng nơ ron chỉ gồm một lớp ẩn. Mục tiêu của mô hình này là dự đoán các từ gần với một từ đích. Đầu vào của mô hình là một véc tơ one-hot của từ đích, và đầu ra của nó là N từ với N là kích thước cửa sổ của ngữ cảnh được định nghĩa trước [13].

Trong các bài toán thực tế, mô hình skip-gram thường được áp dụng do nó đem lại độ chính xác cao hơn [21]. Chi tiết cách thực hiện word2vec với mô hình skip-gram [19] như sau.

Đầu tiên chúng ta cần xây dựng tập từ vựng từ các văn bản huấn luyện, ví dụ tập từ vựng gồm 10000 từ phân biệt. Để có thể đưa các từ vào mạng nơ ron huấn luyện, các từ cần được véc tơ hóa, mỗi từ sẽ được thể hiện bằng một véc tơ one-hot.

Vector này sẽ có 10000 phần tử với mỗi phần tử thể hiện vị trí tương ứng của từ trong tập từ vựng. Ví dụ véc tơ one-hot cho từ “ants” sẽ có phần tử có giá trị bằng 1 tương ứng với vị trí của từ “ants” trong tập từ vựng, các vị trí khác có giá trị bằng 0. Kiến trúc mạng nơ ron được thể hiện trong hình 4.7.

Diagram

Description automatically generated

Hình 4.8. Kiến trúc mạng mô hình Skip-gram

Lớp ẩn trong ví dụ hình 4.7 gồm 300 nơ ron không sử dụng một hàm kích hoạt nào nhưng đầu ra thì sử dụng một hàm softmax. Lớp ẩn được thể hiện bằng một ma trận trọng số gồm 10000 hàng (tương ứng với mỗi từ trong tập từ vựng) và 300 cột (tương ứng với mỗi nơ ron ẩn). Số nơ ron ẩn được gọi là số đặc trưng hay số chiều của word2vec là một siêu tham số có thể được tùy chỉnh tùy theo từng bài toán.

Các hàng của ma trận trọng số của lớp ẩn, thực chất chính là các véc tơ từ, đây chính là mục tiêu của word2vec. Với word2vec, chúng ta tiến hành huấn luyện một mạng nơ ron đơn giản với chỉ một lớp ẩn để tiến hành véc tơ hóa các từ trong tập từ vựng. Tuy nhiên, chúng ta không thực sự sử dụng kết quả đầu ra của mạng nơ ron sau khi huấn luyện, mà sẽ sử dụng trọng số của lớp ẩn. Ma trận trọng số của lớp ẩn giống như một bảng tìm kiếm các từ được thể hiện bằng các véc tơ từ tương ứng được minh họa như hình 4.8.

Với đầu vào là một từ được thể hiện bằng một véc tơ one-hot, việc đưa véc tơ này qua lớp ẩn về bản chất chính là việc tìm kiếm trên ma trận trọng số của lớp ẩn một véc tơ có số đặc trưng bằng số cột của ma trận trọng số.

Chart, bar chart

Description automatically generated

Hình 4.9. Ma trận trọng số lớp ẩn word2vec

Khi nhân một véc tơ one-hot 1x10000 với ma trận 10000x300 thì kết quả của phép nhân ma trận chỉ ảnh hưởng bởi hàng có phần tử 1 của véc tơ one-hot. Hình 4.9 minh họa kết quả nhân véc tơ one-hot với ma trận trọng số của lớp ẩn.

A picture containing text, clock

Description automatically generated

Hình 4.10. Lớp ẩn hoạt động như một bảng tra cứu

Đầu ra của word2vec là một bộ phân loại sử dụng hàm softmax. Hàm softmax cho kết quả là một giá trị thuộc khoảng 0 tới 1, chính là xác suất của mỗi đầu ra, tổng các giá trị này bằng 1. Hình 4.10 minh họa hoạt động của mô hình thể hiện xác suất từ ‘car’ là từ lân cận từ ‘ants’

A picture containing diagram

Description automatically generated

Hình 4.11. Tương quan giữa 2 từ thực hiện với word2vec

## Xây dựng mô hình

Tư tưởng của bài toán tóm tắt văn bản theo hướng tóm lược là đưa ra văn bản tóm tắt là một chuỗi các từ (hay token) dựa trên chuỗi các từ của văn bản đầu vào, đây chính là mô hình chuỗi sang chuỗi (sequence-to-sequence). Mô hình chuỗi sang chuỗi được thể hiện như trong hình 4.11, trong đó các nút mạng RNN có thể lấy đầu vào là một chuỗi và sinh ra một chuỗi đầu ra [8].

Diagram, schematic

Description automatically generated

Hình 4.12. Mô hình chuỗi sang chuỗi

Mô hình chuỗi sang chuỗi có thể được xây dựng bằng kết hợp của hai mạng RNN, một là một mạng chuỗi sang véc tơ (sequence-to-vector) thường được gọi là bộ mã hóa (encoder), theo sau là một mạng véc tơ sang chuỗi (vector-to-sequence) thường được gọi là bộ giải mã (decoder). Hình 4.12. minh họa cho mô hình chuỗi sang chuỗi thực hiện bằng bộ mã hóa-giải mã [8]. Mô hình bộ mã hóa-giải mã được áp dụng thành công trong nhiều bài toán xử lý ngôn ngữ tự nhiên, trong đó đáng chú ý là các nghiên cứu về dịch máy [3], [30].

A picture containing text, clock

Description automatically generated

Hình 4.13. Mô hình bộ mã hóa – giải mã

Nallapati và cộng sự [22] áp dụng mô hình bộ mã hóa cho bài toán tóm tắt văn bản tiếng Anh cho thấy kết quả khả thi của mô hình với bài toán tóm tắt văn bản tự động. Vì vậy, cách tiếp cận của chúng tôi để xây dựng mô hình cho bài toán tóm tắt văn bản tiếng Việt tự động là sử dụng mô hình chuỗi sang chuỗi thực hiện bằng bộ mã hóa-giải mã với các khối LSTM cho cả bộ mã hóa và bộ giải mã.

Bộ mã hóa được xây dựng từ 2 lớp mạng LSTM nạp chồng, mỗi nút mạng là một khối LSTM hai chiều (Bidirectional LSTM) kích thước 256. Bộ giải mã là một mạng LSTM với mỗi nút mạng là một khối LSTM một chiều (unidirectional LSTM). Kiến trúc mô hình chúng tôi xây dựng dựa trên kết quả nghiên cứu của tác giả See và cộng sự [28] và được thể hiện như hình 4.13.

Diagram

Description automatically generated

Hình 4.14. Kiến trúc mô hình tóm tắt văn bản tiếng việt sử dụng LSTM

Các token của văn bản đầu vào được lần lượt đưa vào bộ mã hóa, sinh ra một chuỗi các trạng thái ẩn của bộ mã hóa. Word embedding được khởi tạo ngẫu nhiên theo phân phối chuẩn và được học để điều chỉnh các hệ số trong quá trình huấn luyện. Bộ giải mã nhận các word embedding của các từ ở thời điểm trước: trong quá trình huấn luyện chính là các từ của văn bản tóm tắt tham chiếu và trong quá trình chạy thì các từ ở thời điểm trước chính là các từ được sinh bởi bộ giải mã. Để bộ giải mã có thể học cách tự sinh các từ cho văn bản tóm tắt, chúng tôi sử dụng cơ chế chú ý (attention) giống như tác giả Bahdanau và cộng sự thực hiện [3]. Cơ chế attention dựa trên phân phối xác suất của các từ trong văn bản gốc, giúp bộ giải mã xác định được vị trí của từ sẽ được lựa chọn cho văn bản tóm tắt. Một véc tơ ngữ cảnh (context vector) được tính tại mỗi trạng thái bộ giải mã dựa trên các trạng thái ẩn của bộ mã hóa và trạng thái ẩn trước đó của bộ giải mã.

(4.2)

Trong đó là véc tơ ngữ cảnh, là chuỗi trạng thái ẩn của bộ mã hóa, là phân

phối attention.

(4.3)

Với

, và bias là các tham số được điều chỉnh trong quá trình huấn luyện.

Véc tơ ngữ cảnh là một vector có kích thước cố định thể hiện những gì đã được đọc từ văn bản gốc, kết hợp với trạng thái ẩn của bộ giải mã để tính phân bố xác suất của một token trong tập từ vựng Pvocab.

Do thực tế việc sinh ra từ tiếp theo của văn bản tóm tăt có khả năng đối mặt với một từ không tìm thấy trong tập từ vựng (Out Of Vocabulary – OOV). Để xử lý vấn đề này, See và cộng sự [28] đề xuất cơ chế mạng con trỏ (pointer network) hoạt động giống như một bộ chuyển đổi cho phép bộ giải mã quyết định sinh một từ có trong tập từ vựng đưa vào văn bản tóm tắt hay là sao chép một từ từ văn bản đầu vào. Xác suất một từ được sinh trong văn bản tóm tắt được tính như sau:

(4.4)

Trong đó:

(4.5)

Với cho mỗi thời điểm t được tính từ véc tơ ngữ cảnh , trạng thái

của bộ giải mã và đầu vào của bộ giải mã ; ( , , , *b*) là các tham số

được học trong quá trình huấn luyện.

Trong công thức tính , nếu một từ là từ không có trong tập từ vựng thì

= 0, từ được lấy từ văn bản gốc đưa vào văn bản tóm tắt; và nếu từ đó

không xuất hiện trong văn bản gốc thì = 0, từ được lấy từ tập từ vựng

đưa vào văn bản tóm tắt.

## Thuật toán phân cụm K-means



Hình 4.15. Mục tiêu thuật toán phân cụm

- Kmeans là một thuật toán học máy không giám sát – Unsupervised Learning. Đầu vào của thuật toán phân cụm nói chung là các điểm dữ liệu và đầu ra là các điểm với cụm tương ứng của nó sao cho các điểm thuộc cùng một cụm có nhiều mối quan hệ, liên kết với nhau hơn nhiều so với các điểm khác cụm.

- Với thuật toán Kmeans ta cần truyền thêm tham số là số cụm. Kmeans sử dụng Euclid để xác định sự giống nhau giữa hai điểm dữ liệu, hai điểm càng gần nhau thì độ tương tự giữa hai điểm này càng cao.

- Dựa trên ý tưởng như vậy, Kmeans sử dụng hàm lỗi là tổng khoảng cách từ 1 điểm tới tâm cụm tương ứng của nó và tối ưu dựa trên thuật toán lặp. Với ý tưởng đơn giản, dễ triển khai nhưng lại đem lại kết quả tốt trên nhiều bài toán. Hiện tại Kmeans vẫn luôn nằm trong top những thuật toán Machine Learning phổ biến nhất.

## Thuật toán LSA ( Latent Semantic Analysis)

- LSA là một thuật toán thuộc mô hình chủ đề, phân loại topic. Ý tưởng của thuật toán dựa trên phép phân tích suy biến ma trận biểu diễn văn bản đầu vào (A) để xác định không gian tuyến tính con của A mà vẫn giữ được hầu hết những biến động của ma trận A. Nói cách khác, thuật toán này đi giảm chiều ma trận dữ liệu đầu vào sử dụng phép phân tích ma trận.

- Gỉa sử ma trận biểu diễn mối quan hệ giữa document-word là Amn∈Rm×n. Về cơ bản phép phân tích suy biến sẽ biến đổi ma trận gốc A thành tích của các ma trận suy biến trái U, ma trận đường chéo S và ma trận suy biến phải V theo công thức:

A picture containing diagram

Description automatically generated

Hình 4.16. Chi tiết thuật toán LSA

- Trong đó t là tham số số lượng topics được lựa chọn. Khi đó:

- Ma trận U∈Rm×t là ma trận document-topic.

- Ma trận V∈Rn×t là ma trận term-topic.

- Mỗi dòng của U sẽ biểu diễn phân phối của văn bản và mỗi dòng của V là phân phối của mỗi từ theo t topics ẩn.

- Các cột của U và V đại diện cho các topics.

• Để tìm ra 2 văn bản có giống nhau hoặc 2 từ giống nhau chúng ta sẽ tính tương quan của các véc tơ dòng tương ứng với các văn bản của ma trận U hoặc các véc tơ dòng tương ứng với từ của ma trận V.

Phép phân tích này giảm được khá nhiều chiều đối với trường hợp bộ văn bản lớn. Tuy nhiên nó vẫn tồn tại những nhược điểm đó là:

• Các tính toán hoàn toàn dựa trên phân tích ma trận.

• Các topic được xác định dựa trên các phép chiếu tuyến tính nên mối quan hệ giữa topic-document và topic-term là tuyến tính.

• Chi phí tinh toán của mô hình rất tốn kém nếu kích thước dữ liệu lớn

## Thuật toán TextRank

• TextRank được bắt nguồn từ thuật toán PageRank, thuật toán nhằm xếp hạng các trang web của Google.

•

• Thuật toán này chỉ định giá trị nhất định cho mỗi thành phần của một tập hợp các văn bản liên kết với nhau, ví dụ như World Wide Web.

• Mục đích "đo" tầm quan trọng tương đối của các liên kết trong tập hợp đó.

• Áp dụng cho bất kỳ tập hợp văn bản nào có trích dẫn đối ứng và liên kết cụ thể.

• Giá trị (weight) mà nó gán cho bất kỳ thành phần E được gọi là PageRank của E và ký hiệu là PR(E).

Diagram

Description automatically generated

Hình 4.17. Thuật toán google page rank

Giá trị **Pagerank**hình thành từ thuật toán toán học dựa trên webgraph: các trang world wide web được coi như các đỉnh và các đường link là các cạnh. Khi hình thành webgraph người ta có tính đến những trang của các cơ quan có thẩm quyền như cnn.com hay usa.gov. Giá trị xếp hạng cho thấy tầm quan trọng của từng trang cụ thể. Mỗi đường link tới trang web sẽ được tính như 1 sự hỗ trợ làm tăng thêm giá trị Pagerank.  
Giá trị Pagerank của trang được định nghĩa đệ quy và phụ thuộc vào số lượng và giá trị của các trang mà có link dẫn đến trang đó (incoming links).Một trang web có chứa nhiều link liên kết từ các trang web có giá trị PageRank cao thì giá trị *PageRank*của trang đó cũng sẽ cao.Có rất nhiều bài viết đã được xuất bản ra công chúng dựa trên nghiên cứu gốc của Page và Brin.

Dựa trên ý tưởng này, thuật toán TextRank ra đời nhằm xếp hạng các câu trong một văn bản dựa trên độ tương tự giữa các câu. Mỗi đỉnh sẽ là một câu, cạnh nối là độ tương tự giữa hai câu. Từ đó xác định được các đỉnh với rank cao nhất-các câu quan trọng nhất.

# KẾT QUẢ CÀI ĐẶT

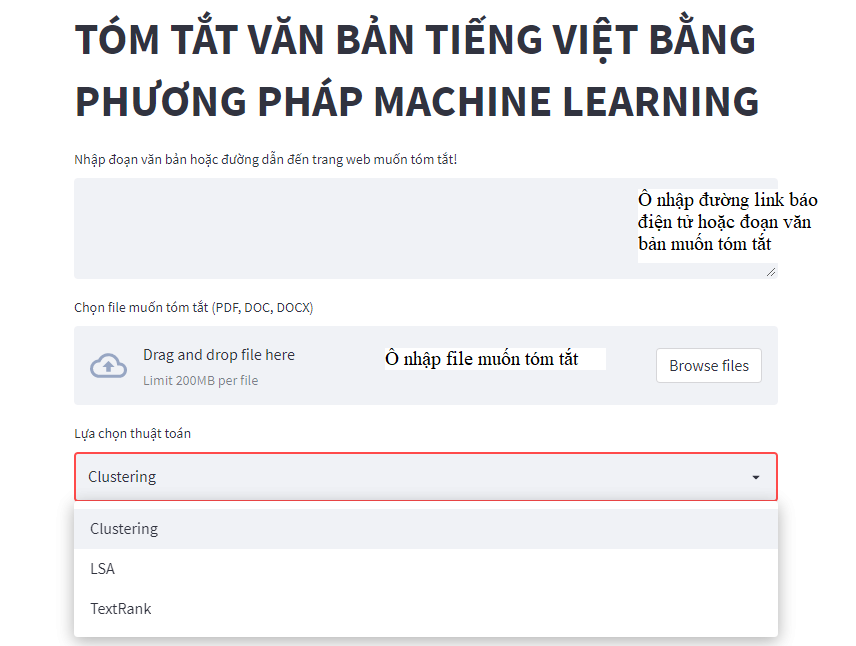
## Cài đặt thuật toán

Với hướng tiếp cận extraction-based, với đơn vị extract là câu ta sử dụng 3 phương pháp mã hóa cho câu là CountVectorize, Tf-idf Vectorize và Word2Vec; Mỗi câu sẽ xem như một điểm dữ liệu đầu vào sử dụng cho mỗi thuật toán. Dữ liệu đầu vào là không có nhãn nên sẽ sử dụng phương pháp đánh giá đơn giản dựa trên nội dung-Content Based.

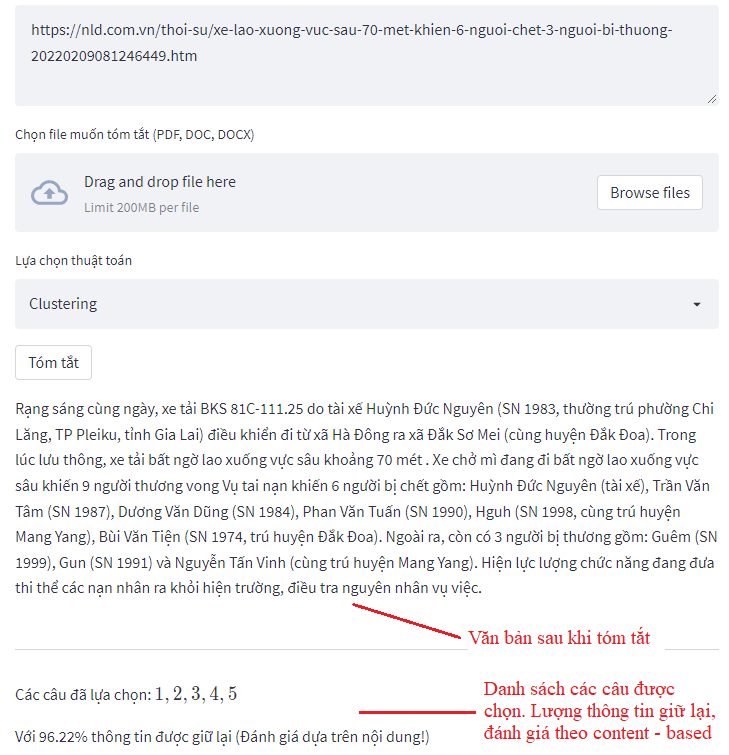
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **vectorize method** | **summarize method** | **min** | **max** | **mean** | **std** |
| **0** | count | clustering | 0.549474 | 0.971858 | 0.828509 | 0.072546 |
| **1** | count | lsa | **0.635774** | **0.978254** | **0.863468** | 0.05693 |
| **2** | count | textrank | 0.606043 | 0.959311 | 0.843669 | 0.05943 |
| **3** | tfidf | clustering | 0 | 0.965379 | 0.809225 | 0.100389 |
| **4** | tfidf | lsa | 0.347293 | 0.961283 | 0.760018 | 0.092178 |
| **5** | tfidf | textrank | 0.59064 | 0.959311 | 0.842451 | 0.062583 |
| **6** | word2vec | clustering | 0.594531 | 0.96886 | 0.831716 | 0.067225 |
| **7** | word2vec | lsa | 0.635774 | 0.978254 | 0.861422 | 0.058301 |
| **8** | word2vec | textrank | 0.633165 | 0.976642 | 0.848594 | 0.06024 |

**Bảng 5.1. Kết quả cài đặt đánh giá trên bộ 1000 bài viết**

## Cài đặt giao diện web



Hình 5.1. Tổng quan giao diện



Hình 5.2. Giao diện tóm tắt văn bản

# ĐÁNH GIÁ, HƯỚNG PHÁT TRIỂN

## Đánh giá

Nhìn chung, trong ba phương pháp tóm tắt văn bản sử dụng thuật toán machine learning ở trên ta đều cho được kết quả trong thời gian rất ngắn (4 mico second cho 1 văn bản khoàng 30 câu). Với thuật toán hiện tại phương pháp mã hóa tf-idf tỏ ra không phù hợp, dễ dàng nhận thấy một từ khi xuất hiện trong nhiều câu có thể nó chính là chủ đề chính, từ quan trọng chứ không phải stops word, từ không quan trọng.

Trong ba phương pháp trên LSA tỏ ra hiệu quả hơn cả, chất lượng văn bản tóm tắt cũng ổn định hơn. Nhưng phương pháp này tồn tại một vấn đề, là vấn đề chung của Extraction-based đó là “chọn bao nhiêu câu là đủ?”. Có thể cải tiến theo cách chọn sao cho trị riêng của ma trận phân tích không nhỏ hơn một ngưỡng nào đó so với giá trị lớn nhất của trị riêng.

## Khó khăn

Các khó khăn gặp phải với bài toán tóm tắt văn bản bằng cách trích dẫn:

- Số lượng câu trích dẫn là bao nhiêu

- Đâu là cách đánh giá tốt nhất cho văn bản đầu ra

- Cần cải tiến các thuật toán Machine learning sao cho phù hợp với bài toán

Các khó khăn nếu áp dụng phương pháp abtraction-based:

- Làm sao thu thập lượng lớn dữ liệu có nhãn

- Thuật toán nào là phù hợp

- Tài nguyên cho việc training model Deep Learning

- Làm sao để đánh giá ngữ pháp cũng như ngữ nghĩa văn bản đầu ra

## Định hướng phát triển

Các hướng phát triển của đề tài:

- Tối ưu model Machine Learning: sử dụng thuật toán phân cụm khác (ví dụ DBSCAN), chọn số lượng câu trích dẫn phù hợp với từng văn bản đầu vào.

- Sử dụng hàm tính độ tương tự khác cho thuật toán Text Rank

- Cài đặt các thuật toán Deep Learning, dựa trên các mô hình sinh, Auto Encoder. Cần thu thập, xử lý dữ liệu trước và tìm hiểu các phương pháp đánh giá mô hình.

# TÀI LIỆU THAM KHẢO

**Tiếng Anh**

[1]. Alex M. (2015), Word2Vec Tutorial Part I: The Skip-gram Model. *Retrieved from* [*http://mccormickml.com/2016/04/27/word2vec-resources/#alex-minnaarstutorials.*](http://mccormickml.com/2016/04/27/word2vec-resources/#alex-minnaarstutorials.)

[2]. Andrew T., Yohannes T., David H., and Hugh E.W. (2007), “Fast generation of

result snippets in web search”, *In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 127-134.

[3]. Bahdanau D., Cho K., Bengio Y. (2015), "Neural machine translation by jointly learning to align and translate". *In International Conference on Learning Representations* (ICLR).

[4]. Christopher O. (2015), Understanding LSTM Networks. *Retrieved from* [*http://colah.github.io/posts/2015-08-Understanding-LSTMs/*](http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

[5]. Corochann (2017), Recurrent Neural Network (RNN) introduction. *Retrieved from* [*http://corochann.com/recurrent-neural-network-rnn-introduction-1286.html*](http://corochann.com/recurrent-neural-network-rnn-introduction-1286.html)

[6]. Denny B. (2015), Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs. *Retrieved from* [*http://www.wildml.com/2015/09/recurrent-neural-networkstutorial-part-1-introduction-to-rnns/*](http://www.wildml.com/2015/09/recurrent-neural-networkstutorial-part-1-introduction-to-rnns/)

[7]. Diederik P. K., Jimmy L.B. (2015), "Adam: A Method for Stochastic

Optimization". *International Conference on Learning Representations*.

[8]. Géron A. (2017), *Hands-on Machine Learning with Scikit-Learn and Tensorflow – Concepts, Tools, and Techniques to Build Intelligent Systems*. Published by O‟Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[9]. Graff D., Kong J., Chen K., and Maeda K. (2003). English gigaword. Linguistic Data Consortium, Philadelphia.62

[10]. Heinzerling B., Johannsen A. (2016), A Python wrapper for the ROUGE summarization evaluation package. *Retrieved from* [*https://pypi.org/project/pyrouge/*](https://pypi.org/project/pyrouge/)

[11]. Hermann K.M., Kocisky T., Grefenstette E., Espeholt L., Kay W., Suleyman M., Blunsom P. (2015). “Teaching machines to read and comprehend”. *In Neural Information Processng Systems*.

[12]. Hochreiter S., Schmidhuber J. (1997), “LONG SHORT-TERM MEMORY”.

*Neural Computation 9(8),* pp. 1735-1780.

[13]. Ibrahim A.H. (2017), Understanding Word2vec for Word Embedding I. *Retrieved from* [*https://ahmedhanibrahim.wordpress.com/2017/04/25/thesistutorials-i-understanding-word2vec-for-word-embedding-i/*](https://ahmedhanibrahim.wordpress.com/2017/04/25/thesistutorials-i-understanding-word2vec-for-word-embedding-i/)

[14] John M.C., Dianne P.O. (2001), “Text summarization via hidden markov

models”. *In Proceedings of the 24th annual international ACM SIGIR conference*

*on Research and development in information retrieval. ACM*, pp. 406-407.

[15]. Krenker A., Bester J., Kos A. (2011), "Introduction to the Artificial Neural

Networks". *Artificial Neural Networks - Methodological Advances and Biomedical*

*Applications*, ISBN: 978-953-307-243-2, InTech.

[16]. Lin C.Y. (2004). "Rouge: A package for automatic evaluation of summaries".

*In Proceedings of Workshop on Text Summarization Branches Out, PostConference Workshop of ACL*.

[17]. Lucas O.Y. (2016). “Newspaper3K Article scraping library”. *Retrieved from* [*https://github.com/codelucas/newspaper*](https://github.com/codelucas/newspaper).

[18]. Lucy V., Hisami S., Chris B., and Ani N. (2007), “Beyond SumBasic: Taskfocused summarization with sentence simplification and lexical expansion”,

*Information Processing & Management*, 43 (6), pp. 1606-1618.

[19]. McCormick C. (2016), Word2Vec Tutorial - The Skip-Gram Model.

*Retrieved from* [*http://www.mccormickml.com*](http://www.mccormickml.com).

[20]. Mehdi A., Seyedamin P., Mehdi A., Saeid S.,Elizabeth D. T., Juan B. G., Krys K. (2017), “Text Summarization Techniques: A Brief Survey”, *arXiv*.

[21]. Mikolov T., Chen K, Corrado G., Dean J. (2013), Efficient Estimation of Word Representations in Vector Space. *International Conference on Learning Representations.*

[22]. Nallapati R., Zhou B., Santos C.D., (2016), "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond", *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, pp. 280-290.

[23]. Nenkova A., McKeown K. (2012), “A survey of text summarization techniques”, *In Mining Text Data. Springer, pp*. 43-76.

[24]. Rada M., Paul T. (2004), "TextRank: Bringing order into texts", *Association*

*for Computational Linguistics*.

[25]. Radev D.R., Hovy E., and McKeown K. (2002), “Introduction to the special

issue on summarization”, *Computational linguistics*, 28(4), pp. 399-408.

[26]. Rush A.M., Chopra S., Weston J. (2015), "A Neural Attention Model for

Sentence Summarization". *In Empirical Methods in Natural Language Processing*.

[27]. Sarwan N.S. (2017), An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec. Retrieved from <https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-countword2veec/>

[28]. See A., Peter J. L., Christopher D.M. (2017), "Get To The Point: Summarization with Pointer-Generator Networks", *arXiv*.

[29]. Shi Yan (2016), Understanding LSTM and its diagrams. *Retrieved from*

[*https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714*](https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714)

[30]. Sutskever I., Vinyals O., Quoc V.L. (2014), “Sequence to Sequence Learning

with Neural Networks”, *arXiv*.

[31]. Trung V.T. (2017). “Python Vietnamese Toolkit”. Retrieved from <https://pypi.python.org/pypi/pyvi>

[32]. Yogan J. K., Ong S. G., Halizah B., Ngo H. C. and Puspalata C. S. (2016), "A Review on Automatic Text Summarization Approaches", *Journal of Computer Science, 12 (4)*, pp. 178-190.