TRƯỜNG ĐẠI HỌC Khoa Học Tự Nhiên
Đại Học Quốc Gia Hà Nội

Báo cáo môn

Introduction to Artificial Intelligence - MAT1206E

Term1 - 2025-2026

Chủ đề

Summarize Paper

GV bộ môn: **Hoàng Anh Đức**
Trần Bá Tuấn
Phạm Ngọc Hải

Hà Nội, Ngày 30 tháng 11 năm 2025

Name	Student ID	GitHub Username	Contribution
Nguyễn Văn Tài	23001927	TAINGUYENVAN0507	TextRank Algorithm
Mai Đình Thành	23001933	Thanh-17-09-2005	LSA
Phạm Bình Nghĩa	23001909	phamnghia57	BartPho
Phạm Đức Mạnh	23001904	mnhaxfh	BartPho
Nguyễn Đình Phiên	23001913	phineasnguyn	KL_SUM Algorithm

Presentation Slide: https://www.canva.com/design/DAG5OI7WR1Q/ELzhPAmFc-pxCxptNGmXP
edit

Repository: https://github.com/phamnghia57/AI-model.git

Report: https://www.overleaf.com/project/69280cd4939f9bb28dd29636

Mục lục

LỜI NÓI ĐẦU

Trong kỷ nguyên bùng nổ thông tin hiện nay, khối lượng dữ liệu văn bản được tạo ra mỗi ngày đang gia tăng với tốc độ chưa từng có. Việc đọc hiểu và trích xuất thông tin quan trọng một cách thủ công không còn khả thi, đặc biệt trong các lĩnh vực như giáo dục, truyền thông, thương mại điện tử và nghiên cứu khoa học. Do đó, các mô hình AI tóm tắt văn bản (Text Summarization / AI Summarize) đã và đang trở thành một công cụ quan trọng, hỗ trợ con người tiếp cận tri thức nhanh chóng, chính xác và hiệu quả hơn.

Xuất phát từ mong muốn tìm hiểu và ứng dụng trí tuệ nhân tạo vào xử lý ngôn ngữ tự nhiên, nhóm chúng em đã lựa chọn đề tài Mô hình AI Tóm tắt văn bản (AI Summarize) làm nội dung nghiên cứu cho bài tập lớn môn Nhập môn AI. Đây là cơ hội để chúng em tiếp cận các hướng tiếp cận tóm tắt hiện đại như Extractive và Abstractive Summarization, đồng thời tìm hiểu sâu hơn về các kiến trúc mô hình mạnh mẽ như Transformer, BART, T5 và GPT, cũng như các thước đo đánh giá chất lượng tóm tắt như ROUGE, BERTSCORE và Cosine Similarity.

Trong quá trình thực hiện đề tài, chúng em đã nỗ lực thu thập dữ liệu, tiền xử lý văn bản, xây dựng mô hình và tiến hành đánh giá kết quả bằng các công cụ phù hợp. Thông qua dự án, chúng em không chỉ hiểu rõ hơn nguyên lý hoạt động của các mô hình tóm tắt tự động, mà còn rèn luyện được tư duy phân tích, xử lý dữ liệu và triển khai các giải pháp AI vào bài toán thực tế.

Chúng em xin gửi lời cảm ơn chân thành đến thầy cô và các bạn đã hỗ trợ và tạo điều kiện giúp nhóm hoàn thành đề tài. Mặc dù đã cố gắng hết sức, do hạn chế về kinh nghiệm và thời gian, bài báo cáo khó tránh khỏi thiếu sót. Nhóm rất mong nhận được những đóng góp để có thể tiếp tục cải thiện và hoàn thiện hơn trong tương lai.

Chúng em xin chân thành cảm ơn!

Chương 1

Giới thiệu

Dự án "Summarize Paper" được xây dựng với mục tiêu nghiên cứu và triển khai một hệ thống tóm tắt văn bản khoa học dựa trên nhiều phương pháp khác nhau, từ mô hình truyền thống đến các mô hình hiện đại dựa trên học sâu. Nhóm đã tích hợp ba thuật toán tóm tắt trích rút gồm TextRank, Latent Semantic Analysis (LSA) và KL-Sum, đồng thời tiến hành fine-tune mô hình BARTpho để thực hiện tóm tắt sinh văn bản. Bên cạnh đó, dự án hướng đến việc đánh giá và so sánh hiệu quả giữa các phương pháp nhằm xác định mô hình phù hợp nhất cho tác vụ tóm tắt bài báo khoa học bằng tiếng Việt. Kết quả thực nghiệm cho thấy mô hình BARTpho sau fine-tune mang lại chất lượng tóm tắt mượt mà và giàu ngữ nghĩa hơn rõ rệt so với các thuật toán truyền thống, khẳng định vai trò nổi bật của các kiến trúc Transformer trong xử lý ngôn ngữ tự nhiên hiện đại. Tóm tắt văn bản, đặc biệt là các tài liệu học thuật, là một thách thức quan trọng trong lĩnh vực xử lý ngôn ngữ tự nhiên. Các bài báo khoa học thường có dung lượng lớn, cấu trúc phức tạp và chứa nhiều thông tin chuyên môn, dẫn đến việc đọc và nắm bắt nội dung chính trở nên tốn kém thời gian. Bài toán của dự án đặt ra là xây dựng một hệ thống có khả năng rút gọn văn bản học thuật thành các đoạn tóm tắt ngắn gọn nhưng vẫn đảm bảo tính chính xác, đầy đủ và bám sát nội dung cốt lõi. Dự án đồng thời cần xem xét sự khác biệt giữa các phương pháp tóm tắt trích rút và tóm tắt sinh, từ đó đánh giá mức độ phù hợp của từng phương pháp đối với dữ liệu tiếng Việt. Vấn đề trọng tâm mà dự án giải quyết là nhu cầu tự động hóa quá trình tóm tắt bài báo khoa học, nhằm hỗ trợ người dùng tiếp cận thông tin nhanh chóng và hiệu quả hơn. Trong bối cảnh lượng tài liệu khoa học ngày càng gia tăng, việc ứng dụng các thuật toán tóm tắt giúp tiết kiệm thời gian đọc hiểu, hỗ trợ nghiên cứu, giảng dạy và tổng hợp tài liệu. Việc triển khai đồng thời ba thuật toán truyền thống TextRank, LSA và KL-Sum giúp cung cấp góc nhìn toàn diện về cách thức trích rút thông tin quan trọng trong văn bản. Song song với đó, mô hình BARTpho được fine-tune cho phép tạo ra bản tóm tắt theo hướng sinh ngôn ngữ tự nhiên, giúp kết quả trở nên gần gũi và dễ theo dõi hơn. Ý nghĩa thực tiễn của dự án không chỉ nằm ở khả năng tạo ra tóm tắt chất lượng, mà còn ở việc minh họa sự phát triển của các kỹ thuật NLP trong xử lý văn bản tiếng Việt, mở ra tiềm năng ứng dụng trong nhiều hệ thống hỗ trợ nghiên cứu và quản lý tri thức.

Chương 2

Phương pháp luận và triển khai

Trong bối cảnh bài toán tóm tắt văn bản tự động ngày càng được quan tâm, việc lựa chọn phương pháp phù hợp và tổ chức triển khai có hệ thống là yếu tố then chốt quyết định chất lượng của mô hình. Chương này trình bày toàn bộ phương pháp luận mà nhóm áp dụng, từ cách tiếp cận tổng quan đến từng bước cụ thể trong pipeline xử lý, nhằm chuyển hóa dữ liệu thô thành bản tóm tắt có ý nghĩa. Nhóm tập trung vào hai hướng chính là tóm tắt trích xuất (Extractive) và tóm tắt sinh ngữ nghĩa (Abstractive), kết hợp cùng các kỹ thuật xử lý ngôn ngữ tự nhiên nền tảng như tiền xử lý văn bản, biểu diễn câu, tính độ quan trọng, và tối ưu đầu ra.

Trên cơ sở đó, nhóm đã triển khai các thuật toán tóm tắt mà mình lựa chọn, bao gồm những phương pháp đã có sẵn trong nghiên cứu của nhóm như TextRank, LSA, KL_SUM, mô hình Transformer-based (BART/T5/GPT) cho Abstractive, cùng các chiến lược rút gọn và lựa chọn câu tối ưu. Các thước đo đánh giá như ROUGE, BERTSCORE, Cosine Similarity và độ mạch lạc ngữ nghĩa cũng được sử dụng để kiểm chứng hiệu quả từng phương pháp ở giai đoạn thử nghiệm.

Nội dung chương không chỉ làm rõ nguyên lý và lý do lựa chọn thuật toán, mà còn mô tả chi tiết cách mô hình được xây dựng, tinh chỉnh và ứng dụng trong quy trình thực tế, tạo tiền đề cho các phần trình bày thuật toán và kết quả phía sau một cách logic, minh bạch và có thể tái lập.

2.1 LSA

2.1.1 Cơ sở lý thuyết

Latent Semantic Analysis (Phân tích ngữ nghĩa tiềm ẩn) là một kỹ thuật trong NLP được dùng để: Rút trích chủ đề (Topic); Tìm mức độ tương đồng giữa các văn bản; Tóm tắt văn bản; Giảm nhiễu (noise) trong dữ liệu văn bản; Giảm số chiều của không gian từ vựng. LSA dựa trên nền tảng chính là SVD (Singular Value Decomposition – phân rã giá trị kỳ dị).

Mô hình không gian vector (Vector Space Model – VSM)

Dữ liệu văn bản được biểu diễn dưới dạng ma trận Term-Document (TDM):

- Hàng: từ (terms)
- Cột: văn bản (documents)
- Ô: trọng số của từ trong văn bản (thường dùng TF-IDF thay vì TF vì TF-IDF loại bỏ từ phổ biến như "và", "là", ...)

Nhưng vấn đề của ma trận này là: Rất thưa (large space); Không hiểu quan hệ ngữ nghĩa giữa các từ (ví dụ: "xe hơi" và "ô tô" → hệ thống coi là không liên quan).

Lý thuyết "không gian ngữ nghĩa tiềm ẩn"

Ý tưởng cốt lõi: Ý nghĩa thật sự của từ không nằm ở bản thân nó, mà nằm ở cách nó xuất hiện cùng với các từ khác trong nhiều văn bản. Do đó, nếu ta nén ma trận TDM lại qua một phép biến đổi tuyến tính, ta có thể: giảm nhiễu, chuẩn hóa các từ tương đương, tìm được không gian ngữ nghĩa "ẩn" (latent semantic space).

2.1.2 Nguyên lý hoạt động

1. Làm sạch dữ liệu: tách câu, tách từ, bỏ stopword (vì ảnh hưởng mô hình), tạo TF-IDF matrix. Kết quả: ma trận rất lớn nhưng thưa.
2. Phân rã SVD: Tính $A = U\Sigma V^T$. Từ đó:
   - U: biểu diễn các từ dưới dạng vector ngữ nghĩa
   - V: biểu diễn các văn bản
   - Σ: độ quan trọng của các chủ đề

   Đây là bước quan trọng giúp tìm được các chủ đề tiềm ẩn.
3. Giảm chiều (dimension reduction): Ta chọn k thành phần quan trọng nhất.

$$A_k = U_k \Sigma_k V_k^T$$

   Ý nghĩa: các chủ đề yếu → bị bỏ; các từ đồng nghĩa gom lại; không gian ngữ nghĩa mượt hơn, ít nhiễu. Việc giảm chiều chính là lý do LSA hoạt động tốt.

4. Ứng dụng trong văn bản tóm tắt Với tóm tắt:

   - Mỗi câu được đưa vào ma trận TF-IDF
   - LSA giúp tìm câu tiêu biểu cho chủ đề chính
   - Chọn các câu có giá trị đóng góp lớn (hàng của V_k)

   Cách đánh giá độ quan trọng của câu:

$$score(c_i) = \sqrt{\Sigma_{j=1}^{k}(V_k[j,i])^2}$$

   Câu nào có score cao nhất thì quan trọng nhất sau đó chọn vào summary.

2.1.3 Xây dựng mô hình

Giới thiệu chung

Trong phần này, tôi sẽ trình bày chi tiết quy trình xây dựng mô hình tóm tắt văn bản sử dụng kỹ thuật Latent Semantic Analysis (LSA). Mục tiêu của mô hình là tạo ra phần tóm tắt ngắn gọn, giàu thông tin từ bản tin báo chí Tiếng Việt thuộc về nhiều chủ đề khác nhau. Dữ liệu đầu vào bao gồm file $data - 1.csv$ với 4 tường chính:

- Unamed: 0: Mã định danh văn bản (được loại bỏ trong tiền xử lý)
- Document: Nội dung bài báo
- Summary: Phần tóm tắt bài báo chuẩn (reference)
- Dataset: nhãn bộ dữ liệu (được loại bỏ trong tiền xử lý)

Quy trình xây dựng mô hình bao gồm các bước: tiền xử lý, biểu diễn văn bản bằng TF-IDF, giảm chiều bằng SVD, chọn câu quan trọng bằng LSA, sinh tóm tắt và đánh giá bằng ROUGE và Cosine Similarity.

Tiền xử lý dữ liệu

Đọc dữ liệu và loại bỏ các trường không cần thiết: dữ liệu được nạp bằng $pandas.read\_csv$; hai trường "Unnamed: 0" và "Dataset" được loại bỏ vì không đóng góp cho mô hình.

Chuẩn hóa văn bản: toàn bộ văn bản được chuyển về chữ thường để tránh trùng lặp từ.

Bảo vệ số liệu chứa dấu chấm: nhiều số dạng 12.5, 3.14 bị tách nhầm khi tokenize, vì vậy tạm thay dấu "." bằng "_", sau đó phục hồi

Tách câu:sử dụng "underthesea.sent_tokenize" – phù hợp cho Tiếng Việt

Phân tích thống kê (Exploratory Data Analysis)

So sánh số lượng từ trước và sau tokenization: Số lượng từ được thống kê trước – sau để đánh giá mức độ thay đổi do tiền xử lý, biểu đồ phân phối được vẽ bằng "matpotlib".

Dù có thay đổi đôi chút, đường phân bố trước vào tokenziation có hình dạng khá tương đồng: số lượng văn bản có từ 200-500 từ, một số văn bản dài hơn > 700 từ, chỉ rất ít văn bản ngắn < 100 từ. Sau tokenize, số từ có xu hướng tăng nhẹ ở nhiều văn bản, điều này bình thường vì Tiếng Việt là ngôn ngữ nhiều âm tiết. Không những vậy sau khi tokenize thì những từ ghep Tiếng Việt được chuẩn hóa tốt hơn, loại bỏ các chữ thừa hoặc phân đoạn sai, câu được phân tách rõ ràng nên sẽ phù hợp cho bài toán LSA.

Từ xuất hiện nhiều nhất: Từ vựng của toàn bộ tập văn bản được ghép lại và đếm bằng "Counter"; Biểu đồ tần xuất Top 10 từ giúp quan sát đặc trưng corpus.

Đây đều là từ có tần suất xuất hiện cao nhưng không mang giá trị nội dung. Các từ ghép trong Tiếng Việt không xuất hiện chứng tỏ tokenizer đã tách từ chính xác, không gộp nhầm các từ ghép phổ biến. Số câu của mỗi văn bản: Thống kê số câu sau khi tách để biết độ dài trung binh văn bản; Những bước phân tích này hỗ trợ tinh chỉnh mô hình TF-IDF và LSA sau này.

Biểu diễn câu bằng TF-IDF

Chuẩn bị dữ liệu TF-IDF: Toàn bộ câu từ tất cả văn bản được đưa vào một danh sách

Khởi tạo TF-IDF Vectorizer: Dùng "max_features=5000" nhằm giảm số chiều và tránh ma trận quá lớn

Trực quan hóa ma trận TF-IDF: Hiển thị 50x50 phần đầu tiên dưới dạng heatmap

Màu càng xanh đậm (gần 0.2) thì trọng số TF-IDF càng cao. Điều này cho thấy từ đó quan trọng và có tính phân biệt cao đối với câu đó. Phần lớn ô vuông có màu vàng nhạt (gần trọng số 0.00) thể hiện rằng ma trận là thưa thớt. Điều này bình thường vì mỗi chỉ chứa một số lượng nhỏ từ trong tổng vốn từ vựng.

Áp dụng LSA

Khởi tạo SVD

Trong đó:

- n_components = 5 nghĩa là sẽ trích xuất 5 chủ đề tiềm ẩn (topics).
- "lca_matrix" chứa vector đặc trưng của từng câu trong không gian chủ đề.

Xác định từ quan trọng trong mỗi chủ đề.

Giúp đánh giá chủ đề là LSA trích xuất, ví dụ: chủ đề 0: "công an", "điều tra", "hiện trường"; chủ đề 1: "dịch", "covid", "vaccine".

Sau đó tính điểm của câu.

Câu có điểm càng cao $\rightarrow$ quan trọng hơn $\rightarrow$ thường chứa thông tin cốt lõi.

Sinh tóm tắt bằng LSA

Tóm tắt toàn văn bản Với mỗi tài liệu: xác định tập câu thuộc tài liệu trong danh sách "all_sentences", tính số câu cần chọn, chọn câu có điểm cao nhất và ghép lại theo đúng thứ tự ban đầu.

2.1.4 Lựa chọn câu tóm tắt

Sau khi thực hiện phân rã ma trận ngữ nghĩa bằng phương pháp LSA (Latent Semantic Analysis), bước quan trọng tiếp theo là lựa chọn các câu quan trọng nhất để tạo ra bản tóm tắt. Quá trình lựa chọn câu trong LSA dựa trên mức độ đóng góp của mỗi câu vào các chủ đề tiềm ẩn (latent topics) được rút trích từ văn bản gốc.

Ma trận biểu diễn câu chủ đề (ma trận V^T)

Sau khi xây dựng ma trận TF-IDF và áp dụng phân rã SVD:

$$A \approx U_k \Sigma_k V_k^T$$

Trong đó:

- U_k: Ma trận biểu diễn mối quan hệ giữa từ và chủ đề
- Σ_k: Ma trận đườn chéo chứa độ mạnh giữa các chủ đề
- V_k^T: Ma trận biểu diễn mối liên hệ giữa các câu và chủ đề

Mỗi cột trong V_k^T tương đương với một câu, mỗi hàng tương đương với một chủ đề tiềm ẩn. Giá trị $v_{i,j}$ càng lớn thì câu i đóng góp càng mạnh vào chủ đề j.

Đo độ quan trọng của từng câu

Để chính xác mức độ quan trọng của một câu, LSA sử dụng trọng số chủ đề trong Σ kết hợp với mức độ tham gia của câu vào các chủ đề trong V_k^T. Mức độ quan trọng (score) của câu thứ i được tính như sau:

$$Socre(i) = \sqrt{\Sigma_{j=1}^{2}(\partial_j v_{ij})^2}$$

Trong đó:

- ∂_j: Mức độ quan trọng của chủ đề thứ j
- v_{ij}: Độ đóng góp của câu i vào chủ đề j
- k: số chủ đề được giữ lại trong LSA

Câu nào có liên quan mạnh tới những chủ đề lớn (có ∂ lớn) thì sẽ có điểm cao, được ưu tiên đưa vào tóm tắt, câu ít đóng góp vào các chủ đề chính sẽ có điểm thấp.

Xếp hạng câu

Sau khi tính Score cho tất cả các câu: các câu được sắp xếp theo thứ tự giảm dần của điểm số, LSA chọn top N câu với điểm cao nhất. Số câu N phụ thuộc vào độ dài tóm tắt mà mình mong muốn, ví dụ: 15-25 % số câu gốc hoặc một số lượng câu cố định (như 3-5 câu).

Giữ nguyên thứ tự câu trong bản tóm tắt

Mặc dù các điểm số được xếp hạng giảm dần nhưng khi đưa vào bản tóm tắt thì các câu sẽ được đặt lại theo đúng thứ tự xuất hiện trong văn bản gốc. Điều này giúp đảm bảo tính mạch lạc, tránh tạo cảm giác "rời rạc" khi ghép các câu quan trọng lại với nhau.

2.1.5 Ưu điểm – Hạn chế

Trong quá trình xây dựng hệ thống tóm tắt văn bản bằng phương pháp LSA, dự án đã thể hiện nhiều ưu điểm nổi bật. Trước hết, quy trình xử lý được thiết kế rõ ràng và có tính mô đun, bao gồm các bước tiền xử lý, biểu diễn văn bản bằng TF-IDF, giảm chiều với SVD và cuối cùng là lựa chọn câu tóm tắt. Nhờ cấu trúc tách biệt, hệ thống rất dễ mở rộng hoặc thay đổi từng thành phần khi cần thiết mà không ảnh hưởng đến toàn bộ pipeline. Bên cạnh đó, dự án sử dụng nhiều biểu đồ thống kê như phân bố số từ trước và sau tokenization hay tần suất các từ xuất hiện nhiều nhất, giúp người đọc dễ dàng theo dõi ảnh hưởng của tiền xử lý đến chất lượng mô hình. Một ưu điểm quan trọng khác là hệ thống áp dụng các thước đo đánh giá chuẩn mực như ROUGE và BERTScore, nhờ đó kết quả có thể so sánh trực tiếp với các nghiên cứu khác. Việc dự án không cần mô hình học sâu (deep learning) cũng mang lại tính linh hoạt cao: hệ thống chạy nhanh, không yêu cầu GPU và có thể áp dụng trên nhiều tập văn bản khác nhau mà không cần huấn luyện lại.

Tuy vậy, dự án vẫn còn tồn tại một số hạn chế đáng chú ý: vì Tiếng Việt là ngôn ngữ đa âm tiết và phức tạp trong tách từ, chất lượng của tokenizer ảnh hưởng mạnh đến kết quả cuối cùng. Các mô-đun như Underthesea đôi khi vẫn tách sai tên riêng, địa danh hoặc các cụm cố định, khiến biểu diễn TF-IDF thiếu chính xác và ảnh hưởng đến việc rút trích chủ đề bằng SVD. Bản chất của LSA chỉ cho phép tạo ra tóm tắt chích xuất nên hệ thống chỉ chọn lại các câu có sẵn trong văn bản, khôn thể viết lại hay tổng hợp câu mới như các mô hình abtractive hiện đại. Điều này khiến bản tóm tắt đôi khi thiếu tự nhiên, kém mạch lạc hoặc chứa các câu không có liên kết ngữ nghĩa rõ ràng. Khi văn bản có nhiều chủ đề hoặc quá dài, LSA cũng gặp khó khăn trong việc xác định số lượng chủ đề tối ưu; nếu chọn quá ít chủ đề, mô hình sẽ bỏ xót thông tin quan trọng, còn nếu quá nhiều, bản tóm tắt sẽ bị nhiễu. Do cách hoạt động hoàn toàn dựa trên ma trận và tuyến tính, LSA không hiểu được ngữ cảnh sâu của ngôn ngữ tự nhiên và không xử lý tốt các hiện tượng đa nghĩa, ẩn ý hay mối quan hệ phụ thuộc dài trong câu. Vì vậy, so với các mô hình ngôn ngữ hiện đại như BERT, PhoBERT, T5 hay GPT, chất lượng tóm tắt bằng LSA thường thấp hơn, đặc biệt trong các văn bản dài và giàu ngữ nghĩa.

Xét riêng phương pháp LSA, đây vẫn là một kỹ thuật mang nhiều ưu điểm. LSA không yêu cần dữ liệu huấn luyện, có khả năng rút trích chủ đề tiềm ẩn và giảm nhiễu hiệu quả thông qua phân ra ma trận SVD. Phương pháp cũng dễ giải thích vì mọi thành phần từ ma trận Σ, U đến V^T đều có ý nghĩa rõ ràng, phù hợp cho các nghiên cứu học thuật hoặc các bài toán yêu cầu tính minh bạch cao. Tuy nhiên, LSA dựa hoàn toàn vào mối quan hệ tuyến tính giữa các từ nên không nắm bắt được ngữ cảnh phức tạp và không phân biệt được các từ đa nghĩa. Hơn nữa, nếu không xử lý tốt stopwords, mô hình rất dễ bị "lạc chủ đề", do các từ phổ biến có thể chiếm ưu thế trong TF-IDF. Những đặc điểm này khiến LSA khó cạnh tranh với các phương pháp dựa trên mô hình học sâu hiện đại.

2.1.6 Kết quả thực nghiệm

Đánh giá chung về kết quả LSA

Kết quả ROUGE và Cosine similarity của 10 văn bản cho thấy mô hình LSA hoạt động ở mức trung bình yếu, đúng với đặc điểm của các phương pháp trích xuất truyền thống. ROUGE-1 trung bình của các văn bản dao động trong khoảng 0.3 – 0.5, trong khi ROUGE-2 khá thấp (đa số dưới 0.15), cho thấy bản tóm tắt của LSA thường chỉ trùng lặp ở mức từ đơn, không biểu đạt được cụm từ hay ngữ đoạn đặc trưng của văn bản. Cosine similarity của các văn bản cũng khá thấp (đa số < 0.25), phản ánh rằng dù LSA có chọn đúng một vài câu quan trọng, mức độ tương đồng ngữ nghĩa tổng thể giữa bản tóm tắt và bản gốc vẫn còn hạn chế. Bên cạnh đó chỉ số của BERT lại khá cao, chứng tỏ văn bản tóm tắt khá chính xác với văn bản gốc. Có sự chệnh lệch khá rõ rệt giữa Bert và ROUGE là do: BERTScore dựa trên "Embeding ngữ nghĩa", nhận biết nghĩa gần nhau, còn ROUGE thì dựa trên n-gram, từ, câu trùng khớp. BERTScore trong trường hợp summary dùng từ khác nhưng ngữ nghĩa giống nhau thì vẫn được chỉ số cao, còn ROUGE thì nếu summary dùng từ khác, chệch thứ tự, thêm bớt từ sẽ ảnh hưởng rất nhiều đến chỉ số của BERTScore. Nguyên nhân chính nằm ở bản chất trích xuất của LSA: mô hình chỉ dựa trên phân rã không gian từ–câu, không hiểu ngữ nghĩa sâu, không nắm được quan hệ nhân quả hay mạch diễn đạt. Vì vậy, khi văn bản quá dài hoặc nhiều thông tin rẽ nhánh, LSA không thể "chọn đúng" các câu đại diện cho chủ đề trung tâm.

2.2 KL_SUM

2.2.1 Cơ sở lý thuyết

Thuật toán KL-Sum (Kullback–Leibler Sum) dựa trên việc lựa chọn các câu sao cho phân phối từ của bản tóm tắt gần nhất với phân phối từ của văn bản gốc. KL Divergence đo mức độ khác nhau giữa hai phân phối xác suất, qua đó giúp mô hình chọn được các câu đại diện tốt nhất cho nội dung chính.

Kullback-Leibler Divergence - KL Divergence

Kullback–Leibler Divergence (KL Divergence): đo mức độ khác biệt giữa hai phân phối xác suất. Trong tóm tắt văn bản, KL Divergence đo khoảng cách giữa phân phối từ của bản tóm tắt và phân phối từ của toàn bộ văn bản gốc. Công thức:

$$D_{KL}(P||Q) = \Sigma_{w \in V} P(w) \log \frac{P(w)}{Q(w)}$$

Trong đó:

- P là phân phối từ văn bản gốc
- Q là phân phối từ trong văn bản tóm tắt hiện tại
- V là tập từ vựng

Phân phối unigram

Phân phối unigram là xác suất xuất hiện của từng từ trong văn bản, thường được tính với Laplace smoothing để tránh xác suất bằng 0. Công thức:

$$P(w) = \frac{count(w) + \alpha}{N + \alpha|V|}$$

Trong đó:

- Count(w) là số từ w xuất hiện
- N là tổng số từ
- $|V|$ là kích thước từ vựng
- α là hệ số làm mượt

2.2.2 Nguyên lý hoạt động

KL_Sum chọn câu mà khi thêm vào bản tóm tắt sẽ giảm KL Divergence nhiều nhất, tức là làm cho phân phối từ bản tóm tắt gần nhất với phối từ văn bản gốc nhất. Quá trình lặp lại cho đến khi đạt số câu tối đa hoặc không còn cải thiện đáng kể.

2.2.3 Xây dựng mô hình

Giới thiệu chung

Dữ liệu đầu vào bao gồm file *test.csv* với 2 trường chính:

- Text là đầu vào của chương trình (đoạn văn bản cần tóm tắt)
- Summary là dữ liệu test để so sánh với kết quả đầu ra

Ngoài ra chương trình còn sử dụng file *data.csv* với 4 trường:

- Unnamed là mã định danh văn bản (không sử dụng)
- Document là dữ liệu đầu vào của chương trình
- Summary là dữ liệu test để so sánh với kết quả đầu ra
- Dataset là nhãn bộ dữ liệu (không sử dụng)

Hệ thống gồm các bước chính:

- Cài đặt và import thư viện
- Cài đặt thuật toán KL_sum
- Đánh giá mô hình

Trong cài đặt thuật toán KL_Sum gồm các bước sau:

- Tiền xử lí văn bản
- Tạo phân phối từ unigram
- Tính KL Divergence
- Xây dựng mô hình KL_Sum

Cài đặt và import thư viện

Hệ thống sử dụng các thư viện chính:

- Nltk: tách câu, tách từ
- Numpy, pandas: xử lí dữ liệu
- Scikit-learn: tính cosine similarity
- Rouge-score: đánh giá ROUGE
- Bert-score: đánh giá mức độ tương đồng ngữ nghĩa
- Sentence-transformers: tạo vector embedding
- Tqdm: tạo tiến trình mô phỏng khi chạy vòng lặp

Cài đặt thuật toán KL_Sum

1. Tiền xử lí văn bản Tách câu bằng hàm sentence_split_vi(): Tách từ, chuyển về chữ thường, tạo danh sách token của toàn bộ văn bản đồng thời xây dựng vocab.
2. Tạo phân phối từ unigram Tính phân phân phối xác suất từ $P(w)$ theo công thức với Laplace smoothing ($\alpha = 1.0$). Đây là bước cần thiết để có thể tính KL Divergence.
3. Tính KL Divergence Tính KL Divergence nhằm đo khoảng cách giữa phân phối từ gốc với phân phối của bản tóm tắt.
4. Xây dựng mô hình KL_Sum Tính phân phối P của toàn bộ văn bản bằng cách gọi hàm unigram_distribution. Sau đó khởi tạo bộ nhớ cho summary: selected (danh sách chỉ số câu đã chọn), selected_token (token của toàn bộ câu đã chọn), remaining (tập chỉ số câu chưa chọn).

Tạo vòng lặp chọn câu tối ưu, mỗi vòng chọn một câu tốt nhất. đồng thời tạo hai biến best_choice (lưu thứ tự câu đã chọn) và best_kl (lưu giá trị KL Divergence của P và Q). Duyệt tất cả các câu còn lại trong remaining để tìm câu giảm KL Divergence nhiều nhất. Tạo biến candidate_tokens để mô phòng summary mới được tính bằng summary hiện tại + câu i. Sau đó:

- Tính phân phối Q của summary tạm thời
- Tính KL Divergence dựa trên P và Q
- Lưu lại lựa chọn tốt nhất vào best_choice và best_kl

Nếu không chọn lựa chọn nào giảm KL Divergence tức là best_choice và best_kl bằng none thì dừng duyệt trong remaining. Nếu không lưu best_choice và danh sách chỉ số câu selected, thêm token của câu đó vào summary (selected_tokens) và xoá khỏi danh sách remaining. Return summary theo đúng thứ tự xuất hiện.

Xây dựng hàm đánh giá

1. Xây dựng hàm ROUGE Sử dụng thư viện rouge_score để tính:
   - ROUGE_1 trùng khớp word-level
   - ROUGE_2 trùng khớp bigram
   - ROUGE_L trùng khớp chuỗi liên tiếp dài nhất
2. Xây dựng hàm BERTScore Mô hình BERTScore tính mức độ tương đồng ngữ nghĩa.
3. Xây dựng hàm Cosine Đo độ tương đồng giữa hai điểm dữ liệu dựa trên hướng
4. Chạy mô hình trên dataset Đọc file *test.csv* sau đó tạo các list để lưu kết quả của ROUGE, BERTScore, Cosine.

   Duyệt từng hàng tạo hai biến doc tương ứng với văn bản cần tóm tắt, ref tương ứng với smmary cho trước. Biến hyp là kết quả tóm tắt dựa trên thuật toán KL_Sum.

   Các biến rouge_score, bert_f1, cosine_sim lưu kết quả đánh giá giữa hyp và ref. Sau đó lưu vào các list đã tạo trước.

   In ra kết quả bao gồm doc (chỉ in ra 70 kí tự), hyp, ref, và các chỉ số. Cuối cùng là in ra chỉ số trung bình trên tập dữ liệu.

2.2.4 Lựa chọn câu tóm tắt

Sau khi thực hiện tạo danh sách token của toàn bộ văn bản chương trình sẽ tính phân phối unigram P với token của toàn bộ văn bản và tính phân phối unigram Q với token của từng câu. Mỗi lần thêm một câu, đánh giá xem phân phối Q mới có giúp giảm KL Divergence hay không. Dừng lại khi đạt độ dài mong muốn hoặc không còn cải thiện đáng kể.

2.2.5 Ưu điểm – Hạn chế

Ưu điểm

Giữ được tính bao quát chủ đề: do KL_Sum chọn câu dựa trên phân phối từ vựng chỉ chọn từ tồn tại trong văn bản chọn cho nên bản tóm tắt thường đầy đủ ý chính.

Tránh trùng lặp thông tin: KL Divergence làm giảm nội dung trùng lặp trong bản tóm tắt .

Dễ triển khai: không cần mô hình huấn luyện không cần dữ liệu lớn chỉ cần xử lí từ, tính phân phối rồi chọn câu.

Hoạt động tốt với văn bản dài: văn bản càng dài phân phối từ càng ổn định, KL_Sum càng chính xác.

Hạn chế

Không hiểu ngôn ngữ sâu: chỉ đo xem xuất hiện bao nhiêu lần chứ không hiểu nghĩa hay ngữ cảnh.

Không biết tạo tóm tắt trôi chảy: do chỉ chọn câu vây nên dễ tạo thành danh sách câu hơn là đoạn văn.

Dễ chọn phải câu dài quá mức: câu dài chứa nhiều từ cho nên phân phối giống dễ được chọn nhiều hơn nhưng đôi khi không phải câu tốt nhất.

2.2.6 Kết quả thực nghiệm

Đánh giá chung về kết quả KL_Sum

Kết quả thực nghiệm trên toàn bộ tập dữ liệu cho thấy mô hình tóm tắt đạt chất lượng ổn định và đáng tin cậy. Bộ chỉ số ROUGE phản ánh khả năng tái hiện nội dung tốt: ROUGE-1 đạt 0.5972 cho thấy mức độ bao phủ từ vựng quan trọng khá cao, ROUGE-2 ở mức 0.5286 chứng tỏ mô hình giữ được mạch ý và cụm từ then chốt, còn ROUGE-L đạt 0.5196 khẳng định sự tương đồng về cấu trúc và trật tự thông tin giữa bản gốc và bản tóm tắt.

Bên cạnh đó, hai chỉ số ngữ nghĩa đều đạt kết quả ấn tượng. BERTScore F1 ở mức 0.7565 cho thấy mô hình không chỉ dựa vào trùng lặp bề mặt mà còn nắm bắt được nội dung sâu của văn bản. Cosine Similarity đạt 0.8896 khẳng định bản tóm tắt duy trì rất tốt chủ đề và tinh thần chung của toàn văn bản.

Tổng hợp các kết quả, mô hình cho thấy hiệu năng mạnh trên cả hai khía cạnh: độ chính xác từ vựng và độ tương đồng ngữ nghĩa. Điều này chứng minh phương pháp tóm tắt đang sử dụng phù hợp với đặc điểm dữ liệu, tạo ra bản tóm tắt ngắn gọn nhưng vẫn giữ lại phần cốt lõi của thông tin.

2.3 TextRank

2.3.1 Khái niệm và nguyên lý hoạt động

Khái niệm

TextRank là một thuật toán trích xuất thông tin dựa trên đồ thị (graph-based ranking). Thuật toán này được lấy ý tưởng từ PageRank – phương pháp xếp hạng trang web của Google.

Mục tiêu của TextRank là:

- Xác định mức độ quan trọng của các đơn vị ngôn ngữ (từ, câu)
- Sử dụng chúng để tóm tắt văn bản hoặc trích xuất từ khóa

Nguyên lý hoạt động

TextRank hoạt động dựa trên mô hình xếp hạng dựa trên đồ thị (graph-based ranking) giống PageRank. Ý tưởng chính: Một câu quan trọng là câu được nhiều câu khác "ủng hộ" thông qua mức độ tương đồng nội dung. Thuật toán gồm 4 bước chính:

1. Biểu diễn văn bản thành đồ thị
   Văn bản được tách thành các câu.
   Mỗi câu tương ứng với một đỉnh (node) trong đồ thị.

   - Nếu hai câu có mức độ tương đồng $> 0 \rightarrow$ tạo cạnh (edge) giữa chúng.
   - Mỗi cạnh có trọng số thể hiện mức độ tương đồng.

   Từ văn bản ban đầu, ta thu được một đồ thị có trọng số giữa các câu.

2. Tính độ tương đồng giữa các câu
   TextRank thường dùng cosine similarity giữa vector biểu diễn câu.
   Vector câu được tạo bằng các phương pháp đơn giản như:

   - Bag-of-Words
   - TF-IDF
   - Word Embedding (nếu muốn nâng cao)

   Do đó, hai câu càng giống nhau về nội dung $\rightarrow$ độ tương đồng càng cao.

3. Áp dụng công thức PageRank để tính điểm quan trọng
   Sau khi tạo xong đồ thị, TextRank sử dụng PageRank để đánh giá tầm quan trọng của từng câu.
   Công thức PageRank:

$$S(V_i) = (1-d) + d \sum_{V_j \in ln(V_i)} \frac{w_{ji}}{\Sigma_k w_{jk}} S(V_j)$$

   Trong đó:

   - V_i: câu cần tìm đến
   - $ln(V_i)$: tập các câu có liên kết với V_i
   - w_{ji}: độ tương đồng giữa câu j và i
   - d: hệ số suy giảm, thường là 0.85
   - $S(V_i)$: điểm quan trọng của câu i

   Một câu càng được nhiều câu quan trọng khác ủng hộ (liên kết tới), thì nó càng quan trọng.
   Điểm số sẽ hội tụ sau vài vòng lặp.

4. Chọn ra các câu quan trọng nhất
   Sau khi có điểm TextRank cho từng câu:

   - Sắp xếp giảm dần
   - Lấy k câu có điểm cao nhất
   - Ghép chúng thành bản tóm tắt (extractive summary)

   Tóm tắt này giữ đúng nội dung gốc, chỉ rút gọn lại.

2.4 Xây dựng mô hình

Mục tiêu của chương này là trình bày quy trình xây dựng mô hình tóm tắt văn bản dựa trên thuật toán TextRank. Mô hình được triển khai hoàn toàn thủ công, không sử dụng thư viện tóm tắt có sẵn, nhằm giúp làm rõ cách thuật toán hoạt động từ mức độ xử lý văn bản đến mức tính toán đồ thị.

2.4.1 Chuẩn bị dữ liệu đầu vào

Tập dữ liệu sử dụng trong mô hình là file *test.csv*, bao gồm 2 cột:

- text: nội dung văn bản gốc cần tóm tắt
- summary: bản tóm tắt chuẩn (gold summary) dùng để đánh giá kết quả

Dữ liệu được làm sạch và chuẩn hóa trước khi đưa vào mô hình:

- Xóa ký tự thừa
- Loại bỏ khoảng trắng dư thừa
- Đưa văn bản về dạng string thuần
- Chuẩn hóa dấu câu

2.4.2 Tách câu (Sentence Splitting)

Văn bản được chia thành các câu bằng cách tách theo dấu chấm (.), dấu chấm hỏi (?) và dấu chấm than (!).

Việc tách câu là bước quan trọng vì mỗi câu sẽ trở thành một node trong đồ thị TextRank.

2.4.3 Biểu diễn câu thành vector (Sentence Vectorization)

Mỗi câu được biểu diễn bằng vector theo mô hình Bag-of-Words:

- Xây dựng **vocabulary** gồm tất cả các từ xuất hiện trong văn bản.
- Với mỗi câu, tạo vector trong đó:
  Phần tử = số lần từ xuất hiện trong câu
- Các vector này được dùng để tính độ tương đồng câu – câu

Cách biểu diễn này đơn giản nhưng phù hợp với TextRank (thuật toán extractive).

2.4.4 Tính độ tương đồng giữa các câu (Cosine Similarity)

Độ tương đồng giữa hai câu được tính bằng cosine similarity:

$$cosine(A, B) = \frac{AB}{||A||||B||}$$

Kết quả giúp đánh giá mức độ liên quan về nội dung giữa hai câu.

2.4.5 Xây dựng ma trận trọng số (Similarity Graph)

Từ độ tương đồng, ta xây dựng ma trận similarity kích thước $n * n$, với:

- n: số câu trong văn bản
- phần tử sim_{ij}: mức độ tương đồng giữa câu i và câu j
- đường chéo chính bằng 0 (không so sánh câu với chính nó)

Sau đó ma trận được chuẩn hóa theo hàng để tạo ma trận chuyển tiếp (transition matrix) sử dụng trong PageRank.

2.4.6 Tính điểm quan trọng bằng PageRank

TextRank áp dụng công thức PageRank lên đồ thị câu – câu

$$PR(i) = (1 - d)/N + d \sum_j \frac{w_{ji}}{\Sigma w_{jk}} PR(j)$$

Trong đó:

- d là hệ số damping (thường = 0.85)
- N là tổng số câu
- Wji là độ tương đồng từ câu j đến câu i

Kết quả: mỗi câu có một điểm quan trọng (rank score).

2.5 Lựa chọn câu tóm tắt

Phương pháp TextRank xem văn bản như một đồ thị, trong đó mỗi câu là một đỉnh và mối liên hệ giữa hai câu được biểu diễn bằng trọng số cạnh thể hiện mức độ tương đồng nội dung. Sau khi tính điểm quan trọng của từng câu bằng thuật toán PageRank, mô hình tiến hành lựa chọn các câu có điểm cao nhất để tạo bản tóm tắt.

2.5.1 Biểu diễn văn bản và ma trận đồ thị

Giả sử văn bản gồm n câu:

$$S = \{s_1, s_2, ..., s_n\}$$

Giữa mỗi cặp câu (s_i, s_j), ta xác định mức độ tương đồng nội dung để gán trọng số cạnh:

$$w_{ij} = Sim(s_i, s_j)$$

Hàm tương đồng có thể là cosine similarity giữa các vector biểu diễn câu (BOW, TF-IDF hoặc từ-embedding).

Sau đó chuẩn hóa trọng số để thu được ma trận chuyển tiếp:

$$M_{ij} = \frac{w_{ij}}{\Sigma_k w_{kj}}$$

Ma trận này biểu diễn xác suất chuyển từ câu j sang câu i trong đồ thị TextRank.

2.5.2 Tính điểm TextRank cho các câu

Mỗi câu si được gán một điểm quan trọng PRi. Giá trị này được cập nhật lặp lại theo công thức PageRank:

$$PR(i)^{(t+1)} = (1-d) + d\sum_j M_{ij} PR_j^{(t)}$$

Trong đó:

- $d \in (0, 1)$ là hệ số suy giảm (thông thường d=0.85)
- M_{ij} là xác suất chuyển từ s_j sang s_i

Sau bước này ta thu được điểm quan trọng của từng câu trong văn bản.

2.5.3 Lựa chọn các câu quan trọng nhất

Sau khi có *vector* điểm $PR = (PR_1, PR_2, \ldots, PR_n)$ mô hình chọn ra k câu có điểm cao nhất

$$T = Top - k(PR)$$

Trong đó tập TTT là danh sách các câu được xem là quan trọng nhất về mặt nội dung.

Cuối cùng, để đảm bảo tính mạch lạc, các câu được chọn sẽ được sắp xếp lại theo thứ tự xuất hiện ban đầu trong văn bản, tạo thành bản tóm tắt hoàn chỉnh.

2.6 Ưu điểm và hạn chế

2.6.1 Ưu điểm

1. Không phụ thuộc vào dữ liệu huấn luyện (unsupervised)
   TextRank hoạt động hoàn toàn dựa trên cấu trúc của văn bản và mối quan hệ giữa các câu.
   Do đó, mô hình không cần tập dữ liệu gán nhãn, phù hợp trong các tình huống không có dữ liệu tóm tắt chuẩn hoặc chi phí gán nhãn cao.

2. Dễ triển khai và chi phí tính toán thấp
   Thuật toán chỉ yêu cầu:

   - tách câu
   - tính vector câu
   - tính độ tương đồng
   - chạy PageRank

   Một mô hình TextRank hoàn chỉnh có thể triển khai chỉ với vài thư viện cơ bản, thời gian xử lý nhanh, phù hợp với văn bản vừa và nhỏ.

3. Khả năng áp dụng linh hoạt
   TextRank có thể áp dụng cho nhiều ngôn ngữ mà không cần thay đổi cấu trúc thuật toán.
   Ngoài ra, phần tính độ tương đồng có thể được thay thế bằng:

   - TF-IDF
   - word embedding
   - sentence embedding

   Điều này giúp mô hình mở rộng tốt tùy điều kiện và yêu cầu chất lượng.

4. Tạo tóm tắt dựa trên thông tin quan trọng nhất
   Thông qua PageRank, TextRank đánh giá mức độ "nổi bật" của từng câu dựa trên mối quan hệ toàn cục trong văn bản.
   Kết quả giúp mô hình chọn được các câu có tính bao quát cao, chứa thông tin quan trọng.

2.6.2 Hạn chế

1. Không hiểu ngữ nghĩa sâu
   TextRank dựa trên thông tin bề mặt (tần suất từ, vector đơn giản, độ tương đồng), nên không hiểu được ngữ nghĩa sâu hoặc nội dung suy luận.
   Khi văn bản phức tạp, câu chứa thông tin chính nhưng diễn đạt khác biệt có thể không được chọn.

2. Phụ thuộc mạnh vào chất lượng tách câu
   Nếu bước tách câu sai hoặc văn bản chứa câu quá dài, TextRank sẽ đánh giá mối quan hệ sai lệch, làm giảm chất lượng tóm tắt.

3. Tóm tắt dạng trích xuất (extractive)
   TextRank chỉ chọn lại câu gốc, không tạo câu mới.
   Vì vậy:

   - tóm tắt đôi khi dài hơn cần thiết
   - có thể thiếu mạch lạc
   - không gộp thông tin nằm rải rác trong nhiều câu

4. Chất lượng phụ thuộc vào cách biểu diễn vector câu
   Nếu sử dụng Bag-of-Words hoặc TF-IDF:

   - không xét đến ngữ cảnh
   - từ đồng nghĩa được xem như không liên quan

   Điều này làm giảm độ chính xác khi văn bản đa dạng về cách diễn đạt.

5. Không phù hợp với văn bản rất dài
   Số lượng câu tăng ß ma trận tương đồng tăng theo n^2ß thời gian và bộ nhớ tăng mạnh.
   Đối với tài liệu hàng nghìn câu, TextRank hoạt động kém hiệu quả hơn các mô hình embedding hoặc transformer.

2.7 Kết quả thực nghiệm

Đánh giá chất lượng tóm tắt của mô hình TextRank trên tập *test.csv*

Sau khi áp dụng thuật toán TextRank để tóm tắt toàn bộ tập dữ liệu và so sánh với các bản tóm tắt chuẩn, năm chỉ số đánh giá được sử dụng gồm ROUGE-1, ROUGE-2, ROUGE-L, BERTScore và Cosine Similarity. Kết quả tổng hợp được trình bày như sau:

- ROUGE-1 (F1): 0.5717
- ROUGE-2 (F1): 0.5390
- ROUGE-L (F1): 0.4528
- BERTScore (F1): 0.7953
- Cosine Similarity: 0.7394

2.7.1 Hiệu quả của TextRank theo ROUGE

Ba chỉ số ROUGE đều đạt mức trung bình khá, đặc biệt ROUGE-1 và ROUGE-2 lần lượt khoảng 0.57 và 0.54. Điều này cho thấy TextRank khôi phục được phần lớn thông tin chính xuất hiện trong bản tóm tắt chuẩn. Việc ROUGE-2 đạt giá trị tương đối cao là dấu hiệu cho thấy các câu được mô hình chọn chứa nhiều cụm từ quan trọng.

Tuy nhiên, ROUGE-L chỉ đạt 0.45 phản ánh một hạn chế cố hữu của phương pháp trích xuất: mô hình chưa tái tạo được cấu trúc liền mạch của bản tóm tắt chuẩn. Điều này là hợp lý vì TextRank chỉ lựa chọn các câu theo thứ hạng mà không thực hiện tổ chức lại bố cục.

2.7.2 Sự tương đồng ngữ nghĩa theo BERTScore

Điểm BERTScore F1 đạt 0.7953, cao hơn đáng kể so với các chỉ số ROUGE. Điều này chứng minh rằng, dù TextRank không sinh câu mới, nội dung tóm tắt vẫn duy trì được ý nghĩa cốt lõi so với tóm tắt chuẩn.

Các câu được chọn có mức độ ngữ nghĩa rất gần với nội dung quan trọng của văn bản gốc, thể hiện khả năng "giữ chủ đề" của thuật toán.

2.7.3 Đánh giá theo Cosine Similarity

Giá trị Cosine Similarity là 0.7394, tương đương mức độ tương đồng tương đối cao về mặt phân bố từ khóa. Chỉ số này chứng minh rằng các bản tóm tắt tạo bởi TextRank chứa phần lớn từ khóa quan trọng tương tự bản chuẩn.

2.7.4 Kết luận chung

Tổng thể, TextRank cho thấy hiệu quả khá tốt trên tập dữ liệu *test.csv*:

- Thuật toán bắt được thông tin quan trọng, thể hiện ở điểm ROUGE-1 và ROUGE-2.
- Giữ được tính tương đồng ngữ nghĩa cao (BERTScore gần 0.8).
- Có mức độ tương đồng từ khóa ổn định (Cosine 0.74).
- Hạn chế chủ yếu nằm ở cấu trúc tổ chức câu, dẫn đến điểm ROUGE-L thấp hơn.

Kết quả này phù hợp với bản chất của mô hình trích xuất: hiệu quả trong việc chọn câu quan trọng nhưng không thể diễn đạt lại hoặc rút gọn linh hoạt như các mô hình tóm tắt trừu tượng hiện đại.

Nhìn chung, TextRank là một phương pháp đơn giản, không cần huấn luyện, nhưng vẫn mang lại chất lượng tóm tắt tốt và ổn định trên dữ liệu thực tế.

Chương 3

BARTpho

3.1 Mô tả cách tiếp cận

BartPho là một mô hình tóm tắt tiếng Việt dựa trên kiến trúc BART (Bidirectional and Auto-Regressive Transformers). Mô hình áp dụng cách tiếp cận Seq2Seq gồm Encoder → Decoder, cho phép mô hình hiểu văn bản đầu vào và sinh ra bản tóm tắt ngắn gọn, tự nhiên.

Sử dựng mô hình đã train sẵn 20GB dữ liệu tiếng Việt, mô hình đã hiểu sâu đặc trưng của tiếng Việt như dấu câu, từ ghép, cấu trúc câu, cần thực hiện chạy lại với tập dữ liệu cần tóm tắt để chỉnh sửa phù hợp với yêu cầu bài toán.

3.1.1 Kiến trúc và khung tiếp cận

BARTpho sử dụng kiến trúc và sơ đồ huấn luyện trước của mô hình BART

- BART là gì? BART là một bộ tự mã hóa khử nhiễu (denoising autoencoder) được xây dựng dưới dạng mô hình Seq2Seq. Nó tích hợp Transformer hai chiều (Bidirectional, giống như bộ mã hóa của BERT) và Transformer tự hồi quy (Auto-Regressive, trái sang phải, giống như bộ giải mã của GPT).
- Kiến trúc: Cả hai phiên bản của BARTpho đều sử dụng kiến trúc "lớn" ("large" architecture), với 12 lớp (layer) trong cả bộ mã hóa và bộ giải mã.
- Chức năng kích hoạt: BARTpho sử dụng hàm kích hoạt GeLU (Gaussian Error Linear Units) thay vì ReLU và khởi tạo tham số từ N(0,0.02).
- Chuẩn hóa: BARTpho cũng bổ sung một lớp chuẩn hóa lớp (layer-normalization) trên cả bộ mã hóa và bộ giải mã, theo mô hình mBART.

3.1.2 Quá trình huấn luận trước (Pre-training)

BARTpho được huấn luyện theo cơ chế denoising autoencoder, trong đó mô hình học cách phục hồi câu gốc từ các phiên bản đã bị nhiễu (noised input) như xóa từ, tráo vị trí, che một đoạn liên tục hoặc xáo trộn câu. Quá trình này buộc mô hình không chỉ ghi nhớ từ vựng, mà còn phải nắm bắt cấu trúc ngữ pháp, quan hệ ngữ nghĩa và mạch văn ở cấp độ sâu hơn. Nhờ vậy, BARTpho hình thành năng lực mạnh trong việc tạo câu mới có ý nghĩa đầy đủ, không chỉ sao chép lại văn bản nguồn.

3.1.3 Các phiên bản của BARTpho

Phiên bản	Kiểu đầu vào	Số lượng tham số	Tokenizer/Từ vựng	Tính hiệu quả
BARTpho-syllable	Cấp độ âm tiết	396M	Sử dụng mô hình Sentence-Piece từ RoBERTa, với 40K loại phụ âm tiết, bao gồm cả các phụ âm ít xuất hiện.	Hiệu quả hơn mBART.
BARTpho-word	Cấp độ từ	420M	Tokenizer RoBERTa với 64K subword, áp dụng Byte Pair Encoding (BPE) giúp tối ưu hóa khả năng biểu diễn từ.	Hiệu quả nhất, vượt trội hơn các phiên bản syllable-level và mBART.

3.1.4 Ứng dụng

Tóm tắt văn bản tiếng Việt (Text Summarization)
Khôi phục chữ viết hoa và dấu câu (Capitalization and Punctuation Restoration)

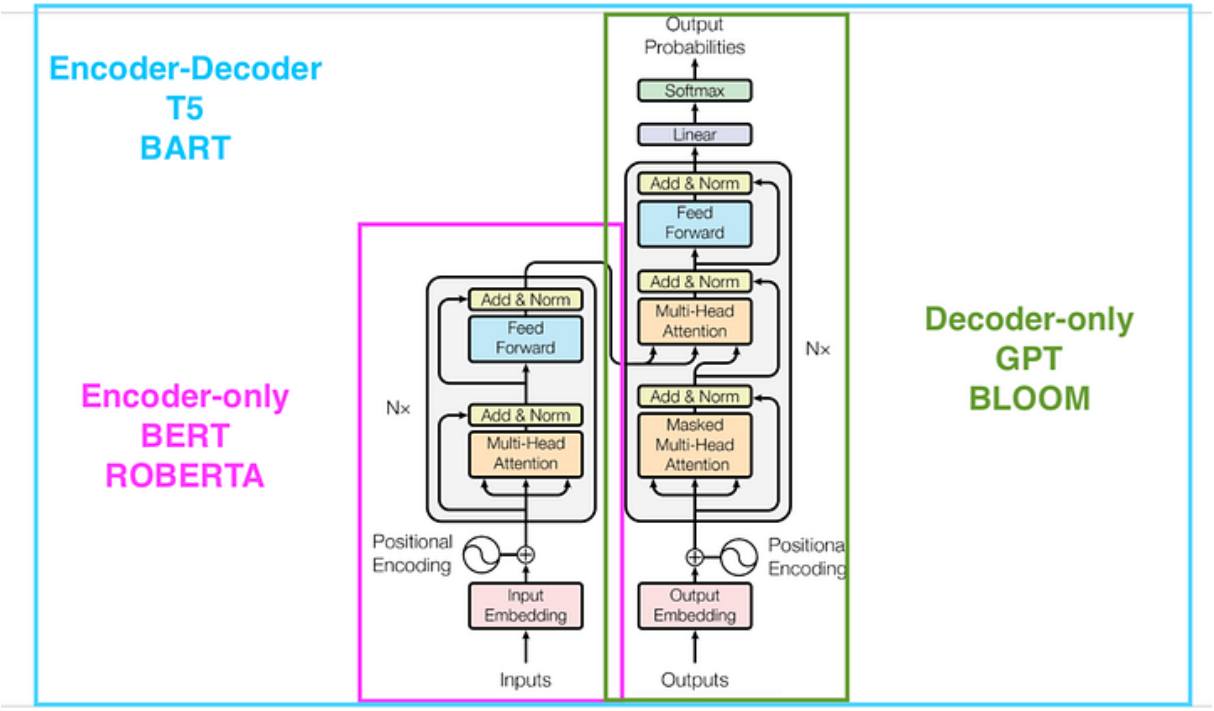
3.2 Cơ sở lý thuyết

Cơ sở lý thuyết của mô hình BART (Bidirectional and Auto-Regressive Transformers) nằm ở việc nó là một bộ tự mã hóa khử nhiễu (denoising autoencoder) được huấn luyện trước (pre-trained) theo kiến trúc chuỗi-tới-chuỗi (Sequence-to-Sequence - Seq2Seq).

BART được thiết kế để cung cấp một phương pháp huấn luyện tự giám sát tổng quát và linh hoạt hơn so với các mô hình trước đây, cho phép nó xử lý hiệu quả cả các tác vụ tạo sinh ngôn ngữ (generation) và hiểu ngôn ngữ (comprehension).

3.2.1 Kiến trúc Seq2Seq khái quát hóa

BART sử dụng kiến trúc Transformer chuỗi-tới-chuỗi tiêu chuẩn. Về mặt lý thuyết, kiến trúc này được coi là sự khái quát hóa (generalization) của các sơ đồ huấn luyện trước khác:



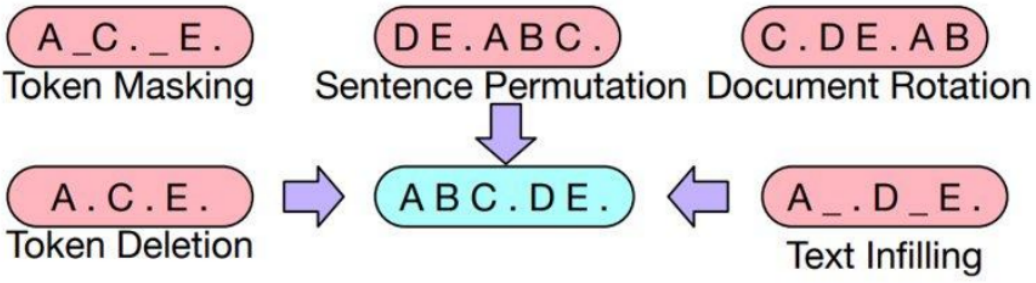
- Bộ Mã hóa Hai chiều (Bidirectional Encoder): Bộ mã hóa của BART tương đương với BERT (Bidirectional Encoder Representations from Transformers). Nhiệm vụ của nó là mã hóa đầu vào bị làm nhiễu loạn bằng cách điều kiện hóa đồng thời trên ngữ cảnh cả bên trái và bên phải ở tất cả các lớp. Khả năng hai chiều này là cần thiết cho các tác vụ hiểu văn bản.
- Bộ Giải mã Tự hồi quy (Auto-Regressive Decoder): Bộ giải mã của BART hoạt động theo cơ chế tự hồi quy từ trái sang phải (left-to-right autoregressive), tương tự như GPT. Nhiệm vụ của nó là tái tạo lại tài liệu gốc từ đầu ra của bộ mã hóa. Đáng chú ý, mỗi lớp của bộ giải mã thực hiện cơ chế chú ý chéo (cross-attention) trên lớp ẩn cuối cùng của bộ mã hóa.

Sự kết hợp này cho phép BART linh hoạt hơn: bộ mã hóa hiểu ngữ cảnh sâu rộng, còn bộ giải mã tạo điều kiện cho việc tinh chỉnh trực tiếp (direct fine-tuning) trên các tác vụ tạo sinh chuỗi.

3.2.2 Phương pháp huấn luyện trước khử nhiễu (Denoising Pre-training)

Cơ sở lý thuyết cốt lõi của BART là một bộ tự mã hóa khử nhiễu được tối ưu hóa thông qua tổn thất tái tạo (reconstruction loss):

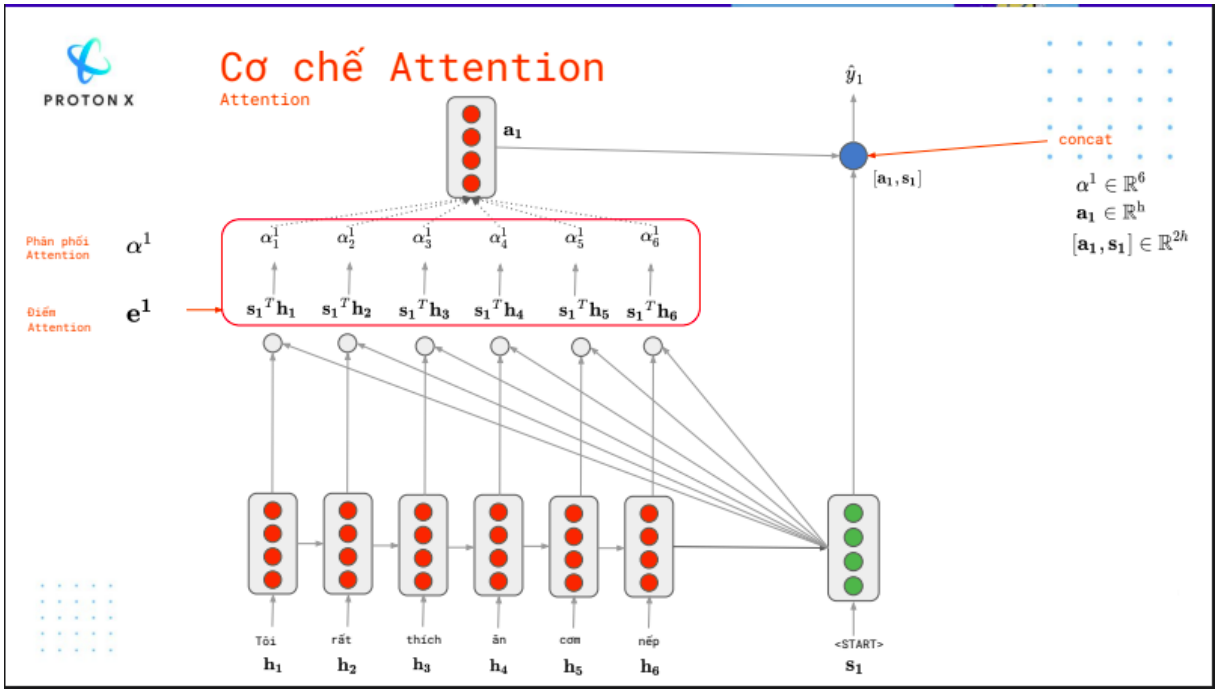
- Làm nhiễu Văn bản (Text Corruption): Văn bản đầu vào được làm hỏng bằng cách áp dụng một hàm gây nhiễu tùy ý (arbitrary noising function). BART được thiết kế để linh hoạt với nhiều loại nhiễu khác nhau, ngay cả khi chúng thay đổi độ dài của chuỗi.
- Tối ưu hóa Tổn thất Tái tạo (Reconstruction Loss): Mô hình Seq2Seq được huấn luyện để khôi phục lại văn bản gốc, bằng cách tối ưu hóa tổn thất entropy chéo (cross-entropy) giữa đầu ra của bộ giải mã và tài liệu gốc ban đầu.



Trong quá trình huấn luyện, BART sử dụng hai kỹ thuật gây nhiễu chính, được chứng minh là hiệu quả nhất:

- Điền khuyết Văn bản (Text Infilling): Nhiều khoảng văn bản (text spans), với độ dài được lấy mẫu từ phân phối Poisson (thường với λ=3 hoặc 3.5 trong BARTpho), được thay thế bằng một token [MASK] duy nhất. Cơ sở lý thuyết: Kỹ thuật này dạy mô hình phải dự đoán số lượng token bị thiếu trong một khoảng trống, giúp mô hình lý luận về độ dài chuỗi tổng thể.
- Hoán vị Câu (Sentence Permutation): Tài liệu được chia thành các câu và các câu này bị xáo trộn theo thứ tự ngẫu nhiên. Cơ sở lý thuyết: Kỹ thuật này buộc mô hình phải học cách sắp xếp các đơn vị ngôn ngữ lớn hơn (câu) một cách hợp lý.
- Các kĩ thuật gây nhiễu khác: Che từ ngẫu nhiên (Token Masking), Xóa từ (Token Deletion), Xoay tài liệu (Document Rotation)

3.2.3 Cơ chế self-attention



Self-Attention là cơ chế cho phép mô hình Transformer xác định mức độ quan trọng giữa các từ trong cùng một câu. Thay vì xử lý từng từ độc lập, Self-Attention giúp mỗi từ "nhìn" toàn bộ câu để hiểu ngữ cảnh. Với mỗi từ, mô hình tạo ra ba vector: $Query(Q), Key(K) vValue(V)$. Đầu tiên, mô hình tính điểm tương quan giữa Query của một từ với Key của tất cả các từ còn lại để xác định mức độ liên quan. Các điểm này sau đó được chuẩn hóa bằng hàm softmax để tạo thành trọng số chú ý. Cuối cùng, đầu ra của từ được tính bằng cách lấy tổng có trọng số của các vector Value.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

3.2.4 Cross-attention

Trong kiến trúc Encoder–Decoder của BARTpho, cơ chế Cross-Attention giữ vai trò trung tâm trong quá trình sinh tóm tắt. Khác với Self-Attention, vốn chỉ xử lý thông tin bên trong một chuỗi, Cross-Attention cho phép decoder truy cập trực tiếp vào các biểu diễn ngữ nghĩa do encoder tạo ra từ văn bản nguồn. Điều này giúp mô hình không chỉ dựa vào các từ đã sinh trước đó, mà còn liên tục "tham chiếu" lại nội dung gốc trong suốt quá trình giải mã.

Đối với nhiệm vụ tóm tắt văn bản, mô hình cần rút gọn thông tin từ một đoạn văn dài thành một câu hoặc đoạn ngắn, nhưng vẫn đảm bảo giữ nguyên ý chính và không được làm sai lệch nội dung. Cross-Attention hỗ trợ mục tiêu này thông qua các chức năng sau:

- Lựa chọn thông tin quan trọng: Tại mỗi bước sinh, decoder tính toán trọng số chú ý đối với từng vector ẩn của encoder. Các trọng số này giúp mô hình xác định phần nào của văn bản gốc đang liên quan nhất tới từ cần sinh tiếp theo.
- Đảm bảo tính mạch lạc và đúng ngữ nghĩa: Nhờ liên tục đối chiếu với biểu diễn của encoder, mô hình có thể duy trì dòng chảy thông tin nhất quán, hạn chế việc sinh ra những câu không ăn khớp với nội dung nguồn.
- Giảm hiện tượng "ảo giác mô hình" (hallucination): Vì decoder luôn tham chiếu thông tin thật từ input, mô hình giảm khả năng bịa thêm chi tiết không có trong văn bản ban đầu.
- Tái cấu trúc nội dung theo hướng cô đọng: Cross-Attention hỗ trợ mô hình trích xuất những phần cốt lõi và loại bỏ chi tiết dư thừa, từ đó xây dựng bản tóm tắt ngắn gọn nhưng bao quát.

Nhờ những đặc điểm trên, Cross-Attention được xem là cơ chế không thể thiếu trong các mô hình tóm tắt hiện đại, giúp kết nối chặt chẽ giữa nội dung gốc và văn bản tóm tắt được sinh ra.

3.3 Huấn luyện mô hình

3.3.1 Dữ liệu

Bộ dữ liệu sử dụng trong nghiên cứu gồm hai thành phần chính: văn bản đầu vào (text) và bản tóm tắt tương ứng (summary). Mỗi mẫu dữ liệu là một cặp text–summary, trong đó phần text chứa nội dung đầy đủ của đoạn văn, còn summary là phiên bản rút gọn do con người biên soạn. Phần text thường có độ dài lớn hơn đáng kể so với summary, phù hợp với nhiệm vụ tóm tắt trích rút và tóm tắt sinh.

Toàn bộ dữ liệu được chia thành ba tập: train, validation và test. Tập train được sử dụng để mô hình học các mẫu ngôn ngữ và quy luật tóm tắt; tập validation dùng để điều chỉnh siêu tham số và theo dõi quá trình huấn luyện; trong khi tập test dùng để đánh giá khách quan hiệu suất của mô hình trên dữ liệu chưa từng gặp. Việc phân chia này đảm bảo quá trình huấn luyện diễn ra ổn định và kết quả đánh giá phản ánh đúng khả năng tổng quát hóa của mô hình.

3.3.2 Tokenization

Trong quá trình tiền xử lý, mô hình được thiết lập giới hạn độ dài chuỗi đầu vào là 1024 token và chuỗi đầu ra là 256 token. Việc áp dụng giới hạn này là cần thiết nhằm đảm bảo hiệu quả tính toán và chất lượng sinh văn bản.

- Các mô hình Transformer có độ phức tạp tính toán tỉ lệ với bình phương độ dài chuỗi. Nếu không giới hạn số token, chi phí xử lý sẽ tăng rất nhanh, gây tiêu tốn tài nguyên và dễ dẫn đến lỗi tràn bộ nhớ. Ngưỡng 1024 token giúp mô hình tiếp nhận đủ lượng ngữ cảnh cần thiết trong khi vẫn phù hợp với khả năng xử lý của phần cứng.
- Nhiệm vụ tóm tắt yêu cầu đầu ra ngắn gọn và tập trung vào thông tin chính. Do đó, việc giới hạn chuỗi sinh ở mức 256 token giúp mô hình tránh tạo văn bản dài dòng, đồng thời buộc mô hình ưu tiên chọn lọc các nội dung quan trọng.
- Việc đặt độ dài cố định giúp chuẩn hóa dữ liệu đầu vào, đảm bảo sự nhất quán giữa các mẫu huấn luyện. Những văn bản vượt quá giới hạn sẽ được cắt hoặc chia nhỏ, còn văn bản ngắn sẽ được padding, từ đó giúp quá trình huấn luyện ổn định hơn.

3.3.3 Huấn luyện

Việc huấn luyện sử dụng lớp Seq2SeqTrainer cùng các tham số quan trọng như: batch size 2 (kết hợp gradient accumulation), 4 epochs, learning rate 3e-5, warmup 500 bước và tối ưu trọng số bằng weight decay. Quá trình sinh đầu ra trong lúc đánh giá kích hoạt predict_with_generate và sử dụng beam search 4 beams để tăng chất lượng tóm tắt. Mô hình được đánh giá sau mỗi epoch bằng thang điểm ROUGE.

Sau khi huấn luyện hoàn tất, mô hình và tokenizer được lưu lại vào thư mục output để phục vụ suy luận và triển khai.

3.3.4 Kiểm thử mô hình

Sau khi hoàn tất quá trình huấn luyện, mô hình được kiểm thử trên tập dữ liệu test nhằm đánh giá khả năng tổng quát hóa và chất lượng sinh tóm tắt trên dữ liệu chưa từng xuất hiện trong quá trình huấn luyện.

Quy trình kiểm thử

Quy trình kiểm thử được triển khai theo các bước sau

1. Nạp mô hình và tokenizer đã fine-tuned
   Mô hình BARTpho đã được lưu trong thư mục outputs/bartpho-finetuned được tải lại cùng bộ tokenizer tương ứng, sau đó chuyển sang thiết bị tính toán (GPU nếu khả dụng).

2. Sinh tóm tắt cho từng văn bản đầu vào
   Với mỗi mẫu văn bản trong tập test, mô hình sinh ra bản tóm tắt bằng phương pháp beam search với 4 beams nhằm tối ưu chất lượng đầu ra.
   Các tham số sinh (max_length, min_length, num_beams) được thiết lập nhất quán với pha đánh giá trong huấn luyện.

3. So sánh tóm tắt sinh với nhãn tham chiếu
   Mỗi bản tóm tắt dự đoán được đối chiếu với bản tóm tắt chuẩn (summary) theo ba nhóm chỉ số:

   - ROUGE (ROUGE-1, ROUGE-2, ROUGE-L): đo mức độ trùng lặp về từ/ngữ giữa dự đoán và tham chiếu.
   - BERTScore (F1): đo mức độ tương đồng ngữ nghĩa dựa trên embedding ngữ cảnh.
   - Cosine Similarity dựa trên TF-IDF: đánh giá mức độ gần nhau về mặt không gian từ vựng giữa văn bản gốc và bản tóm tắt sinh.

4. Lưu kết quả kiểm thử
   Tất cả các bản tóm tắt dự đoán cùng điểm số đánh giá được lưu vào tệp 'data/evaluate/output_predicted_evaluated.csv' bao gồm các cột:

   - summary_pred: tóm tắt mô hình sinh ra
   - ROUGE-1, ROUGE-2, ROUGE-L
   - BERTScore_F1
   - Cosine_Similarit

Mục tiêu của kiểm thử

Việc kiểm thử nhằm:

- Đánh giá khả năng mô hình giữ lại thông tin quan trọng (ROUGE).
- Đo lường mức độ chính xác về mặt ý nghĩa và ngữ nghĩa (BERTScore).
- Quan sát mức độ liên hệ giữa văn bản gốc và tóm tắt (Cosine Similarity).
- Xác định liệu mô hình có bị hiện tượng overfitting hay không thông qua việc so sánh kết quả test với validation.

Kết luận từ kiểm thử

Kết quả từ tệp đánh giá giúp nhìn nhận toàn diện hiệu quả của mô hình trong việc:

- Chọn lọc và rút gọn thông tin quan trọng.
- Sinh văn bản mạch lạc, hợp lý và có tính bao phủ nội dung.
- Duy trì sự tương đồng ngữ nghĩa với văn bản gốc theo cả mức độ từ vựng (lexical) và mức độ ngữ nghĩa (semantic).

3.4 Đánh giá mô hình

3.4.1 Đánh giá mô hình thực tế

Đánh giá bằng ROUGE

Chỉ số	Giá trị	Nhận xét
ROUGE-1	0.6456	Mức trùng lặp từ đơn cao, chứng tỏ mô hình giữ lại được nhiều từ khóa quan trọng.
ROUGE-2	0.4110	Khả năng bảo toàn các cụm từ ngắn (bigrams) ở mức tốt, giúp tóm tắt có tính mạch lạc.
ROUGE-3	0.4224	Phản ánh mô hình duy trì được cấu trúc câu và trật tự thông tin tương đối tốt.

Nhìn chung, các chỉ số ROUGE đều đạt mức khá cao, cho thấy mô hình có khả năng nắm bắt nội dung quan trọng và tạo câu tương đồng với bản tham chiếu.

3.4.2 Đánh giá bằng BERTScore

BERTScore F1 = 0.7326 cho thấy mức độ tương đồng ngữ nghĩa cao giữa bản tóm tắt sinh và bản tóm tắt chuẩn.

Điểm số >0.70 chứng tỏ mô hình không chỉ sao chép từ vựng mà còn hiểu được nội dung và sinh câu có ý nghĩa gần với bản gốc.

Đánh giá bằng Cosine Similarity (TF-IDF)

Cosine Similarity = 0.6128 → Mức độ tương đồng nội dung giữa văn bản gốc và tóm tắt được giữ ở mức trung bình – khá.

Giá trị ≈ 0.6 cho thấy tóm tắt vẫn bao phủ phần lõi thông tin của văn bản đầu vào, nhưng không quá trùng lặp — điều này phù hợp với đặc trưng của tóm tắt abstractive.

Nhận xét tổng quan

Các chỉ số ROUGE cao → mô hình giữ lại nội dung quan trọng và tạo văn bản có cấu trúc tốt.

BERTScore cao → mô hình hiểu và bảo toàn ý nghĩa, không chỉ sao chép từ.

Cosine Similarity tốt → tóm tắt vẫn liên quan chặt chẽ đến văn bản gốc.

Kết luận chung

Mô hình BARTpho fine-tuned thể hiện hiệu suất mạnh mẽ, đặc biệt ở các khía cạnh:

- Nắm bắt nội dung trọng tâm
- Giữ được tính mạch lạc và tự nhiên của câu
- Bảo toàn ngữ nghĩa
- Tạo ra các bản tóm tắt hợp lý và nhất quán với văn bản gốc

Chương 4

So sánh , Đánh giá kết quả thực nghiệm

4.1 Phương pháp đánh giá

4.1.1 Đánh giá dựa trên trùng khớp từ vựng – ROUGE

Các chỉ số ROUGE (Recall-Oriented Understudy for Gisting Evaluation) được sử dụng phổ biến trong đánh giá tóm tắt văn bản.

- ROUGE-1: đo mức trùng lặp từ đơn giữa bản tóm tắt mô hình sinh và bản tóm tắt tham chiếu
  Phản ánh khả năng giữ lại các từ khóa quan trọng.
- ROUGE-2: đo mức trùng lặp bigram (2 từ liên tiếp).
  Phản ánh mức độ bảo toàn các cụm từ có ý nghĩa cục bộ.
- ROUGE-L: đo độ dài chuỗi con chung dài nhất (Longest Common Subsequence).
  Đánh giá mức độ mạch lạc tổng thể của câu sinh ra.

Ưu điểm: đơn giản, phổ biến, dễ so sánh.
Hạn chế: không đo được ý nghĩa sâu, nhạy với sự thay đổi từ ngữ không quan trọng.

4.1.2 Đánh giá dựa trên ngữ nghĩa – BERTScore

BERTScore sử dụng embedding ngữ nghĩa từ mô hình ngôn ngữ (như BERT, mBERT hoặc PhoBERT) để so sánh từng token trong hai câu dựa trên cosine similarity.

- Đầu ra gồm Precision, Recall và F1, trong đó F1 được dùng để đánh giá tổng quát.
- Dùng tốt cho tóm tắt vì hai câu có thể khác từ ngữ nhưng vẫn giữ nghĩa tương đồng.

Ưu điểm: bắt được ngữ nghĩa sâu, đánh giá đúng các câu diễn đạt lại (paraphrasing).
Hạn chế: tính toán nặng, phụ thuộc vào chất lượng mô hình embedding.

4.1.3 Đánh giá mức độ bao phủ nội dung – Cosine Similarity (TF-IDF)

Phương pháp này chuyển văn bản gốc và tóm tắt mô hình sinh thành vector TF-IDF, sau đó tính độ tương đồng giữa hai vector:

$$Similarity = \frac{AB}{||A||||B||}$$

- Cho biết tóm tắt có giữ được phần nội dung nào trong văn bản gốc hay không.
- Giá trị dao động từ 0 đến 1, càng cao càng giống nhau.

Ưu điểm: đánh giá dựa trên "vùng nội dung", không cần bản tóm tắt tham chiếu.
Hạn chế: chỉ phản ánh mức trùng lặp từ theo trọng số, không đại diện ngữ nghĩa sâu.

4.2 So sánh

Bảng kết quả:

Mô hình	**LSA**	**KL_Sum**	**TextRank**	**BARTpho**
ROUGE-1	0.4	0.5972	0.5717	0.6456
ROUGE-2	<0.25	0.5268	0.5390	0.4110
ROUGE-L	0.334	0.5196	0.4528	0.4224
BERTScore	0.7	0.7565	0.7953	0.7326
Cosine Similarity	0.4830	0.8896	0.7394	0.6128

Đánh giả tổng hợp:

Mô hình	**Điểm mạnh**	**Điểm yếu**	**Khi nào nên dùng**
LSA	Dễ triển khai, chạy nhanh	Chất lượng kém nhất, không giữ ngữ nghĩa	Tóm tắt tin tức, văn bản ngắn
TextRank	BERTScore cao → giữ nghĩa tốt, ổn định	ROUGE-L, ROUGE-2 không cao nhất do mô hình trừu tượng hơn	Tóm tắt trích rút chính xác nội dung, báo cáo kỹ thuật
KL-sum	Cosine cao nhất, ROUGE tốt	Dễ trùng lặp hơi "thô"	Tối ưu nội dung, tóm tắt sinh tự nhiên, văn phong mượt giống người viết
BARTpho	ROUGE-1 cao nhất, văn phong tự nhiên	Phụ thuộc fine-tune, ROUGE-2 thấp	Tạo bản tóm tắt sinh tự nhiên, mượt như người viết

Chương 5

Kết luận và hướng phát triển

5.1 Kết luận

Trong đề tài này, nhóm đã nghiên cứu và triển khai bốn phương pháp tóm tắt văn bản gồm LSA, KL-Sum, TextRank và BARTpho, đại diện cho hai hướng tiếp cận chính: trích rút (extractive) và sinh (abstractive).

Quy trình thực nghiệm bao gồm: tiền xử lý văn bản, huấn luyện mô hình (đối với BARTpho), sinh tóm tắt, và đánh giá kết quả bằng các thước đo ROUGE, BERTScore và Cosine Similarity. Kết quả cho thấy:

- LSA có chất lượng thấp nhất do hạn chế về khả năng nắm bắt ngữ nghĩa.
- KL-Sum đạt ROUGE-L và Cosine Similarity cao nhất, cho thấy mô hình bám sát nội dung nguyên bản.
- TextRank đạt BERTScore cao nhất, thể hiện khả năng giữ ngữ nghĩa tổng thể rất tốt và ổn định.
- BARTpho đạt ROUGE-1 cao nhất và tạo ra bản tóm tắt mượt mà, tự nhiên hơn, phù hợp cho các ứng dụng thực tế cần văn phong giống con người.

Tổng thể, mô hình BARTpho thể hiện hiệu quả tốt về chất lượng và độ tự nhiên khi tóm tắt, trong khi KL-Sum và TextRank phù hợp với các bài toán yêu cầu bám sát câu gốc. Việc so sánh nhiều phương pháp giúp đánh giá toàn diện ưu – nhược điểm của từng mô hình và đưa ra lựa chọn phù hợp tùy mục đích sử dụng.

5.2 Phương hướng phát triển

1. Mở rộng và nâng cao chất lượng dữ liệu huấn luyện
    - Thu thập thêm tập dữ liệu văn bản tiếng Việt lớn và đa dạng hơn (tin tức, khoa học, báo cáo, pháp luật...).
    - Làm sạch dữ liệu tốt hơn: loại nhiễu, chuẩn hóa dấu câu, tách câu.
    - Tăng kích thước dữ liệu fine-tune cho BARTpho để mô hình tạo bản tóm tắt mượt và chính xác hơn.
2. Kết hợp nhiều mô hình (Hybrid Summarization)

3. Phát triển mô hình kết hợp trích rút và sinh

    - Dùng TextRank hoặc KL-Sum chọn câu quan trọng.
    - Dùng BARTpho để diễn đạt lại những câu đó thành bản tóm tắt tự nhiên hơn. Giải pháp này vừa đảm bảo độ chính xác, vừa có tính mượt mà

Đề tài đã hoàn thành mục tiêu xây dựng, triển khai và đánh giá hệ thống tóm tắt văn bản tiếng Việt dựa trên nhiều mô hình khác nhau.

Kết quả thực nghiệm cho thấy không có mô hình "tốt nhất tuyệt đối", mà mỗi mô hình phù hợp cho từng mục đích riêng:

- BARTpho: tóm tắt tự nhiên, phù hợp sản phẩm thực tế.
- TextRank & KL-Sum: trích rút ổn định, độ chính xác nội dung cao.
- LSA: chỉ phù hợp cho mục đích học thuật và minh họa.

Những kết quả đạt được là cơ sở quan trọng giúp phát triển các hệ thống tóm tắt mạnh hơn trong tương lai và có thể được mở rộng thành một ứng dụng hỗ trợ đọc – phân tích văn bản tiếng Việt hiệu quả.

Nguồn tham khảo

1. OpenGenus IQ. *KL-Sum algorithm for text summarization.*
2. SciELO. Artículo: S1405-55462023000401203.
3. Dataiku. *Text Summarization Plugin Documentation.*
4. 4. Brigham Young University. *Generating Extractive Sentiment Summaries for Natural Language User Queries on Products.*
5. *Self-Attention và Multi-head Sefl-Attention trong Transformers.*
6. *BERT, GPT and BART: a short comparison.*
7. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.*
8. *BARTpho: Pre-trained Sequence-to-Sequence Models for Vietnamese.*
9. *VL-GPT: A Generative Pre-trained Transformer for Vision and Language Understanding and Generation.*
10. *TextRank: Bringing Order into Texts — Rada Mihalcea & Paul Tarau (2004).*
11. *Graph-Based Natural Language Processing and Information Retrieval – Mihalcea & Radev (2011)*