

PROJECT 2A: PHÂN CỤM NỘI DUNG BÀI BÁO TRÊN TRANG BÁO DÂN TRÍ

1st Mai Thanh Phuc, 2nd Hoang Thi Yen Nhi, 3rd Tran Trong Thanh, and Le Nhat Tung
HUTECH University, Vietnam
{Mai Thanh Phuc, Hoang Thi Yen Nhi, Tran Trong Thanh}@hutech.edu.vn, and lenhattung@hutech.edu.vn

Tóm tắt nội dung

Trong bối cảnh dữ liệu văn bản tiếng Việt ngày càng phong phú, việc khai thác, phân tích và nhóm các văn bản có nội dung tương đồng đóng vai trò quan trọng trong nhiều ứng dụng của khoa học dữ liệu và xử lý ngôn ngữ tự nhiên. Nghiên cứu này tập trung vào bài toán **phân cụm văn bản tiếng Việt**, nhằm khám phá cấu trúc tiềm ẩn và mối quan hệ ngữ nghĩa giữa các tài liệu mà không cần nhãn dữ liệu.

Dữ liệu được biểu diễn thông qua ba phương pháp chính gồm **TF-IDF**, **PhoBERT-base** và **PhoBERT-large** nhằm nắm bắt cả đặc trưng từ vựng và ngữ nghĩa. Để giảm nhiễu và tối ưu hiệu quả tính toán, **kỹ thuật giảm chiều UMAP** được áp dụng trước khi tiến hành phân cụm.

Trên nền các biểu diễn này, năm thuật toán phân cụm được triển khai gồm **Hierarchical Clustering**, **K-Means**, **DBSCAN**, **HDBSCAN** và **Spectral Clustering**. Mỗi thuật toán được lựa chọn nhằm phản ánh những cách tiếp cận khác nhau trong việc xác định ranh giới và mật độ cụm dữ liệu.

Nghiên cứu hướng đến việc so sánh, trực quan hóa và đánh giá cấu trúc cụm từ các mô hình này, qua đó làm rõ khả năng ứng dụng của các kỹ thuật phân cụm trong việc tổ chức và phân tích dữ liệu văn bản tiếng Việt trên quy mô lớn.

Index Terms

I. GIỚI THIỆU

Trong thời đại số, lượng dữ liệu văn bản tiếng Việt tăng nhanh trên mạng xã hội, báo điện tử và các nền tảng trực tuyến, đặt ra thách thức lớn cho các hệ thống xử lý ngôn ngữ tự nhiên (NLP). Trong bối cảnh đó, **phân cụm văn bản** trở thành hướng tiếp cận quan trọng, giúp nhóm các văn bản có nội dung tương đồng mà không cần dữ liệu gán nhãn. Phương pháp này không chỉ hỗ trợ khám phá cấu trúc dữ liệu mà còn ứng dụng trong gợi ý nội dung, phát hiện chủ đề, tóm tắt và truy xuất thông tin. Tuy nhiên, đặc trưng ngôn ngữ phức tạp và đa nghĩa của tiếng Việt khiến việc xây dựng mô hình phân cụm hiệu quả vẫn là một thách thức đáng kể.

Để giải quyết vấn đề này, nghiên cứu tập trung vào việc **kết hợp nhiều phương pháp biểu diễn và thuật toán phân cụm khác nhau** nhằm tìm ra cách tiếp cận tối ưu cho dữ liệu văn bản tiếng Việt. Cụ thể, ba phương pháp biểu diễn được sử dụng gồm:

- **TF-IDF (Term Frequency–Inverse Document Frequency)** [1]: mô hình thống kê truyền thống giúp thể hiện mức độ quan trọng của từ trong từng văn bản.
- **PhoBERT-base** và **PhoBERT-large** [2]: hai mô hình ngôn ngữ dựa trên kiến trúc Transformer [3], được huấn luyện đặc biệt cho tiếng Việt, có khả năng nắm bắt ngữ nghĩa và ngữ cảnh sâu sắc hơn.

Sau khi biểu diễn dữ liệu, kỹ thuật **UMAP (Uniform Manifold Approximation and Projection)** [4] được áp dụng nhằm giảm chiều dữ liệu, loại bỏ nhiễu và giữ lại cấu trúc không gian quan trọng, giúp các thuật toán phân cụm hoạt động hiệu quả hơn.

Trên các dữ liệu đã giảm chiều, năm thuật toán phân cụm được triển khai:

- **Hierarchical Clustering**: nhóm dữ liệu theo cấu trúc phân cấp, trực quan hóa bằng cây dendrogram.
- **K-Means**: thuật toán phân cụm phổ biến dựa trên khoảng cách trung bình giữa các điểm [1].
- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** [5]: phát hiện các cụm có hình dạng bất kỳ dựa trên mật độ điểm dữ liệu.
- **HDBSCAN (Hierarchical DBSCAN)** [6]: mở rộng từ DBSCAN, tự động xác định số cụm tối ưu và xử lý nhiễu tốt hơn.
- **Spectral Clustering** [7]: sử dụng giá trị riêng của ma trận tương đồng để phát hiện cấu trúc phi tuyến trong dữ liệu.

Nghiên cứu hướng đến việc phân tích và đánh giá hiệu quả của các phương pháp này trên cùng một tập dữ liệu, thông qua các chỉ số đo lường phổ biến như **Silhouette Score**, **Davies–Bouldin Index** và **Calinski–Harabasz Index**. Ngoài ra, các kỹ thuật trực quan hóa như UMAP và biểu đồ dendrogram được sử dụng để hỗ trợ diễn giải kết quả.

Thông qua việc kết hợp đa dạng các mô hình biểu diễn và thuật toán phân cụm, nghiên cứu này kỳ vọng mang lại cái nhìn toàn diện về hiệu quả của các phương pháp phân cụm trên dữ liệu văn bản tiếng Việt, từ đó hỗ trợ cho các nghiên cứu và ứng dụng thực tế trong lĩnh vực khai phá văn bản và xử lý ngôn ngữ tự nhiên.

II. NGHIÊN CỨU LIÊN QUAN

A. Các phương pháp biểu diễn dữ liệu văn bản

Các nghiên cứu ban đầu chủ yếu dựa trên mô hình **Bag-of-Words (BoW)** và **TF-IDF** [1], cho phép biểu diễn văn bản dưới dạng vector không gian từ vựng. Tuy nhiên, các mô hình này không thể nắm bắt ngữ nghĩa và mối quan hệ ngữ cảnh giữa các từ. Sự ra đời của các mô hình học sâu đã mở ra hướng tiếp cận mới với **Word2Vec** [8], **GloVe** [9], và **BERT** [3]. Đối với tiếng Việt, **PhoBERT** [2] là mô hình đầu tiên được huấn luyện đặc thù cho ngôn ngữ này.

B. Các thuật toán phân cụm văn bản

Các phương pháp truyền thống như **K-Means** và **Hierarchical Clustering** được áp dụng rộng rãi nhờ tính đơn giản và hiệu quả. Trong khi đó, các phương pháp dựa trên mật độ như **DBSCAN** [5] và **HDBSCAN** [6] được phát triển để phát hiện các cụm có hình dạng bất kỳ và xử lý nhiễu. **Spectral Clustering** [7] là hướng tiếp cận khác, sử dụng phân tích phổ để phát hiện cấu trúc phi tuyến trong dữ liệu.

C. Ứng dụng trong ngôn ngữ tiếng Việt

PhoBERT [2] đã được chứng minh là hiệu quả trong các tác vụ NLP tiếng Việt như phân loại cảm xúc, tóm tắt văn bản và phân cụm chủ đề. Ngoài ra, các kỹ thuật giảm chiều dữ liệu như **UMAP** [4] giúp cải thiện tốc độ tính toán và khả năng trực quan hóa cụm dữ liệu.

D. Tổng kết

Việc kết hợp giữa các mô hình biểu diễn ngữ nghĩa hiện đại và thuật toán phân cụm tiên tiến đang là xu hướng nghiên cứu mới trong khai phá văn bản tiếng Việt. Nghiên cứu này kế thừa các hướng tiếp cận đó, đồng thời so sánh hiệu quả của nhiều thuật toán trên cùng tập dữ liệu, nhằm đánh giá khả năng phân cụm tối ưu cho dữ liệu tiếng Việt.

III. PHƯƠNG PHÁP NGHIÊN CỨU

Phần này trình bày quy trình nghiên cứu được áp dụng trong đề tài, bao gồm các bước chính: thu thập dữ liệu, tiền xử lý văn bản, biểu diễn đặc trưng (embedding), giảm chiều dữ liệu, phân cụm bằng nhiều thuật toán khác nhau và đánh giá kết quả. Mỗi bước được mô tả chi tiết nhằm đảm bảo tính minh bạch và khả năng tái lập của quy trình.

A. Thu thập dữ liệu

Dữ liệu được thu thập từ **báo điện tử Dân Trí** — một trong những nguồn tin tức phổ biến và đa dạng chủ đề nhất tại Việt Nam. Mục tiêu là thu thập tập hợp các bài viết thuộc nhiều chuyên mục khác nhau (chính trị, kinh tế, giáo dục, sức khỏe, thể thao, giải trí, v.v.), nhằm đảm bảo dữ liệu có độ bao phủ nội dung rộng, phục vụ tốt cho quá trình phân cụm chủ đề.

1) *Công cụ và ngôn ngữ lập trình:* Việc thu thập dữ liệu được thực hiện bằng ngôn ngữ **Python**, sử dụng hai thư viện chính là:

- **Selenium**: mô phỏng thao tác người dùng trên trình duyệt, cho phép cuộn trang và tải động nội dung.
- **BeautifulSoup4**: dùng để trích xuất dữ liệu HTML (tiêu đề, ngày đăng, tác giả, nội dung) từ từng bài viết.

Ngoài ra, thư viện **pandas** được sử dụng để xử lý dữ liệu dạng bảng và lưu trữ kết quả cào dưới dạng **CSV**, giúp thuận tiện cho các bước tiền xử lý sau này.

2) *Quy trình cào dữ liệu:* Quá trình thu thập được chia làm hai giai đoạn:

a) (1) *Cào danh sách bài viết theo chuyên mục:* Trình duyệt Chrome được chạy ở chế độ **headless** để tăng tốc độ và tiết kiệm tài nguyên. Mỗi chuyên mục trên trang <https://dantri.com.vn> được tự động duyệt qua tối đa 15 trang (cuộn 10 lần mỗi trang) để thu thập các liên kết bài viết, tiêu đề, chuyên mục và thời điểm cào. Hệ thống tự động loại bỏ trùng lặp bằng cách đối chiếu với dữ liệu đã có, sau đó lưu kết quả vào tệp **CSV** kèm timestamp.

`dantri_new_unique_2025-10-12_00-50-08.csv`

b) (2) *Cào nội dung chi tiết từng bài báo.*: Ở giai đoạn thứ hai, mỗi liên kết thu được từ bước trên được truy cập riêng lẻ bằng thư viện `requests`. Dữ liệu chi tiết được trích xuất gồm:

- **Tiêu đề bài viết (title)**
- **Ngày đăng (pub_date)**
- **Tác giả (author)**
- **Nội dung (content)** – gồm toàn bộ phần thân bài dưới thẻ HTML `<p>` trong vùng `singular-content`.

Để đảm bảo tính ổn định, chương trình thực hiện:

- Tạm dừng ngẫu nhiên (`time.sleep(random.uniform(0.5, 1.2))`) giữa các yêu cầu để tránh bị chặn IP.
- Bỏ qua các bài không tải được hoặc thiếu nội dung chính.
- Gộp dữ liệu đã cào mới với danh sách link cũ bằng `pandas.merge()` theo khóa “link”.

Sau khi hoàn tất, toàn bộ dữ liệu được lưu lại trong một tệp CSV tổng hợp, ví dụ:

```
dantri_crawl_full_2025-10-12_01-40-30.csv
```

c) (3) *Quy mô dữ liệu.*: Tổng cộng có **7.580 bài viết** được thu thập, trải rộng trên hơn 20 chuyên mục, bao gồm cả các chủ đề phổ biến như thời sự, thể thao, sức khỏe, giải trí và kinh doanh. Mỗi bản ghi trong dữ liệu gồm các trường:

```
[category, title, link, pub_date, author, content, crawl_date]
```

3) *Đánh giá chất lượng dữ liệu.*: Sau khi cào xong, dữ liệu được kiểm tra và làm sạch sơ bộ nhằm loại bỏ:

- Các bài viết trùng lặp hoặc bị lỗi khi tải nội dung.
- Những bài có phần nội dung quá ngắn (dưới 30 ký tự).
- Các bài không có tiêu đề hoặc ngày đăng.

Quy trình này giúp đảm bảo dữ liệu đầu vào có chất lượng tốt, phục vụ hiệu quả cho các bước tiền xử lý ngôn ngữ và biểu diễn đặc trưng tiếp theo.

B. Tiền xử lý dữ liệu

Sau khi thu thập được dữ liệu thô từ báo điện tử Dân Trí, bước tiếp theo là tiến hành **tiền xử lý dữ liệu văn bản** nhằm loại bỏ nhiễu, chuẩn hóa ngôn ngữ và chuyển đổi nội dung thành dạng có thể xử lý bằng mô hình học máy. Toàn bộ quá trình được thực hiện bằng ngôn ngữ **Python**, sử dụng các thư viện chính như `pandas`, `re`, `underthesea`, `tqdm` và `matplotlib`.

1) *Đọc và lựa chọn dữ liệu đầu vào.*: Từ tệp dữ liệu gốc `dantri_crawl_full.csv`, chỉ hai trường quan trọng được giữ lại là:

- **category**: chuyên mục của bài viết.
- **content**: phần nội dung chi tiết của bài báo.

Những bản ghi thiếu nội dung hoặc rỗng được loại bỏ. Sau khi lọc, dữ liệu còn lại khoảng 7.500 bài viết.

2) *Làm sạch văn bản.*: Mỗi bài viết được xử lý thông qua hàm `clean_text()` với các bước cụ thể như sau:

- 1) **Chuyển toàn bộ văn bản về chữ thường** nhằm giảm sự phân biệt không cần thiết giữa chữ hoa và chữ thường.
- 2) **Loại bỏ liên kết URL** bằng biểu thức chính quy để tránh nhiễu từ các đường dẫn trong bài viết.
- 3) **Xóa ký tự đặc biệt, số, emoji và dấu câu**, chỉ giữ lại các ký tự chữ cái và khoảng trắng.
- 4) **Chuẩn hóa khoảng trắng** giúp loại bỏ các dấu cách thừa.
- 5) **Tách từ tiếng Việt (word tokenization)** bằng thư viện `Underthesea` ở định dạng văn bản (`format="text"`), giúp xác định rõ ràng ranh giới giữa các từ đơn và từ ghép.

Kết quả của bước này được lưu vào cột mới `clean_text` trong bảng dữ liệu.

3) *Loại bỏ từ dừng (Stopwords)*: Danh sách **stopwords tiếng Việt** được tải từ tệp `vietnamese-stopwords.txt`, bao gồm các từ phổ biến nhưng không mang ý nghĩa phân biệt chủ đề (ví dụ: “là”, “và”, “những”, “các”, “được”, “của”, “một”). Hàm `remove_stopwords()` được áp dụng để loại bỏ toàn bộ các từ này ra khỏi mỗi câu, giúp giảm nhiễu và tăng tính tập trung cho mô hình biểu diễn từ.

4) *Loại bỏ trùng lặp và câu ngắn.*: Sau khi làm sạch, dữ liệu được xử lý để loại bỏ:

- **Các văn bản trùng nội dung**: sử dụng hàm `drop_duplicates()` dựa trên cột `clean_text`.
- **Các văn bản quá ngắn**: loại bỏ những dòng có ít hơn 8 từ (`len(str(x).split()) < 8`) nhằm đảm bảo mỗi mẫu dữ liệu có đủ ngữ cảnh phục vụ cho việc trích xuất đặc trưng.

Sau khi lọc, tập dữ liệu còn lại **7.278 bài viết** hợp lệ.

5) *Lưu kết quả tiền xử lý.*: Toàn bộ dữ liệu sau khi làm sạch được lưu vào tệp:

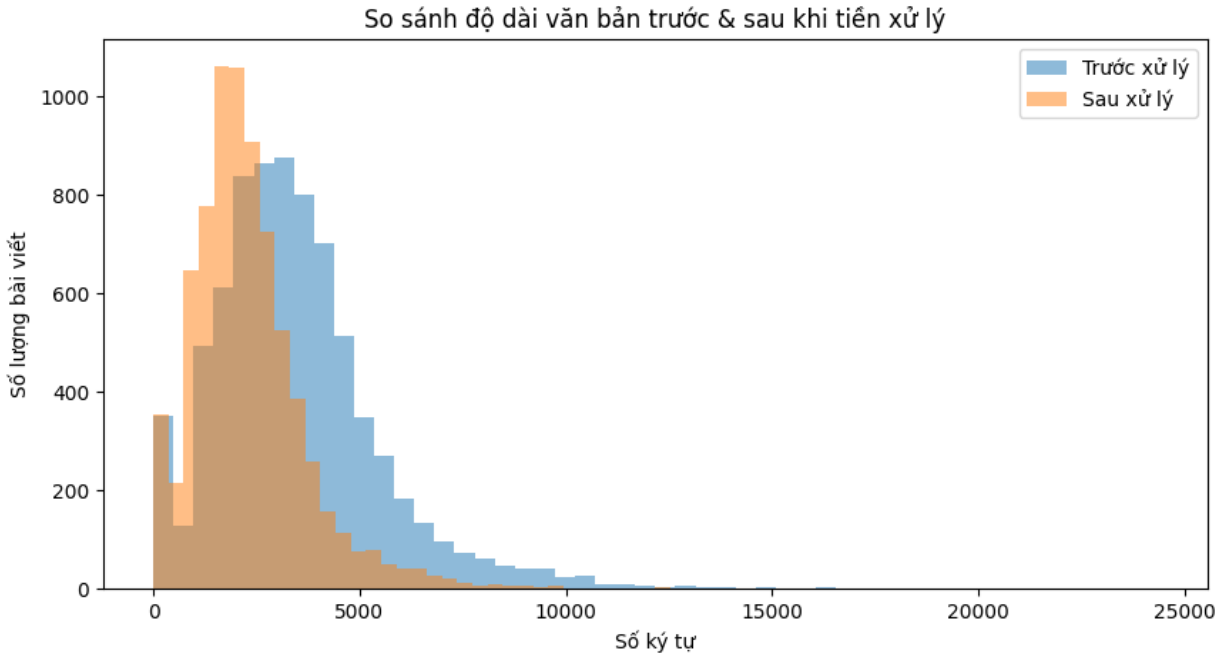
```
/content/drive/MyDrive/project2/project2a/dantri_preprocessed.csv
```

Tập này sẽ là đầu vào cho bước biểu diễn đặc trưng văn bản (embedding).

6) So sánh trước và sau khi tiền xử lý: Để minh chứng cho hiệu quả của bước tiền xử lý, hệ thống tiến hành so sánh trực tiếp dữ liệu trước và sau khi xử lý:

- Hiện thị song song 5 mẫu văn bản đầu tiên để quan sát thay đổi.
- Tính và so sánh **độ dài trung bình** (tính theo số ký tự) giữa dữ liệu gốc và dữ liệu đã làm sạch.
- Trực quan hóa phân bố độ dài văn bản bằng biểu đồ **histogram**, thể hiện sự khác biệt rõ rệt về độ dài và mức độ nhiễu.

Hình 1 minh họa biểu đồ phân bố độ dài văn bản trước và sau khi tiền xử lý, cho thấy phần lớn các bài viết sau xử lý ngắn gọn, nhất quán và ít nhiễu hơn so với dữ liệu gốc.



Hình 1. So sánh phân bố độ dài văn bản trước và sau tiền xử lý

C. Biểu diễn đặc trưng văn bản (Embedding)

Sau khi dữ liệu văn bản được tiền xử lý, bước tiếp theo là chuyển đổi các đoạn văn thành dạng số học mà máy tính có thể hiểu và xử lý được. Quá trình này được gọi là **biểu diễn đặc trưng** (text embedding). Trong nghiên cứu này, ba phương pháp embedding được áp dụng gồm:

- 1) TF-IDF (Term Frequency–Inverse Document Frequency)
- 2) PhoBERT-base
- 3) PhoBERT-large

Mỗi phương pháp thể hiện một cách nhìn khác nhau về ngôn ngữ: từ thống kê tần suất đến ngữ nghĩa sâu dựa trên mô hình ngôn ngữ Transformer.

1. TF-IDF (Term Frequency–Inverse Document Frequency)

TF-IDF là phương pháp biểu diễn văn bản truyền thống, dựa trên thống kê tần suất xuất hiện của các từ trong tập tài liệu [1]. Phương pháp này giúp đánh giá mức độ quan trọng của một từ trong một văn bản so với toàn bộ tập văn bản.

Với mỗi từ t trong văn bản d , trọng số TF-IDF được tính như sau:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

trong đó:

$$\text{TF}(t, d) = \frac{f_{t,d}}{\sum_k f_{k,d}} \quad \text{và} \quad \text{IDF}(t) = \log \frac{N}{n_t + 1}$$

Với:

- $f_{t,d}$: số lần xuất hiện của từ t trong văn bản d .
- N : tổng số văn bản trong tập dữ liệu.
- n_t : số văn bản chứa từ t .

Nhờ đó, các từ phổ biến (như “và”, “là”, “của”) sẽ có giá trị IDF thấp, trong khi những từ đặc trưng cho chủ đề (như “kính tế”, “thể thao”, “giáo dục”) có giá trị cao hơn. Kết quả cuối cùng là một vector thưa (sparse vector), trong đó mỗi chiều tương ứng với một từ trong từ điển của tập dữ liệu.

Trong nghiên cứu này, mô hình TF-IDF được huấn luyện bằng thư viện `scikit-learn`, với các tham số mặc định:

- `max_features = 5000`
- `ngram_range = (1, 2)` để xét cả từ đơn và cụm hai từ.
- `sublinear_tf = True` để giảm ảnh hưởng của từ quá phổ biến.

Kết quả embedding được lưu dưới dạng tệp `X_tfidf.npz` để sử dụng trong các thuật toán phân cụm.

2. PhoBERT-base và PhoBERT-large

PhoBERT là mô hình ngôn ngữ tiếng Việt được huấn luyện dựa trên kiến trúc **RoBERTa** [10], kế thừa từ mô hình **BERT (Bidirectional Encoder Representations from Transformers)** [3]. PhoBERT được phát triển bởi nhóm nghiên cứu của Nguyễn và cộng sự [2] và hiện là một trong những mô hình embedding mạnh nhất cho tiếng Việt.

a) *Kiến trúc Transformer*:: BERT và PhoBERT đều dựa trên cơ chế **Transformer encoder** [11], với thành phần chính là **cơ chế tự chú ý (self-attention)**. Công thức tính toán trọng số chú ý giữa hai từ i và j trong một câu như sau:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

trong đó:

- Q, K, V : lần lượt là ma trận truy vấn (query), khóa (key) và giá trị (value) của các từ.
- d_k : số chiều của không gian khóa (key dimension).

Cơ chế này cho phép mô hình hiểu mối quan hệ giữa các từ trong ngữ cảnh song song, thay vì chỉ tuần tự như RNN hay LSTM.

b) *PhoBERT-base và PhoBERT-large*::

- **PhoBERT-base**: gồm 12 lớp Transformer, kích thước ẩn 768, khoảng 135 triệu tham số.
- **PhoBERT-large**: gồm 24 lớp Transformer, kích thước ẩn 1024, khoảng 370 triệu tham số.

Cả hai phiên bản đều được huấn luyện trước (pre-trained) trên khoảng 20GB dữ liệu tiếng Việt từ Wikipedia, VNESEcorpus và báo chí, sử dụng chiến lược **Masked Language Modeling (MLM)** tương tự BERT.

c) *Trích xuất embedding*:: Trong nghiên cứu này, các câu được mã hóa bằng tokenizer của PhoBERT, sau đó truyền qua mô hình để lấy đầu ra `last_hidden_state`. Vector biểu diễn của mỗi văn bản được tính bằng **trung bình cộng (mean pooling)** của các vector từ:

$$\vec{v}_d = \frac{1}{n} \sum_{i=1}^n \vec{h}_i$$

với \vec{h}_i là vector của từ thứ i trong câu. Kết quả embedding của từng mô hình được lưu dưới dạng:

- `embeddings_phobert_base.npy`
- `embeddings_phobert_large.npy`

d) *So sánh TF-IDF và PhoBERT*::

- TF-IDF phản ánh tần suất xuất hiện của từ, thích hợp cho dữ liệu lớn và dễ tính toán.
- PhoBERT nắm bắt ngữ nghĩa và ngữ cảnh, giúp mô hình hiểu sâu hơn về mối quan hệ giữa các từ.

Do đó, việc kết hợp cả hai hướng tiếp cận — thống kê và ngữ nghĩa — giúp đánh giá toàn diện hiệu quả phân cụm văn bản tiếng Việt.

3. Tổng kết

Ba phương pháp embedding trên đại diện cho ba cấp độ biểu diễn văn bản:

- 1) **TF-IDF**: Biểu diễn nông, dựa vào tần suất thống kê.
- 2) **PhoBERT-base**: Biểu diễn ngữ nghĩa tầm trung, cân bằng giữa hiệu quả và tốc độ.
- 3) **PhoBERT-large**: Biểu diễn ngữ nghĩa sâu, cho kết quả chính xác hơn nhưng chi phí tính toán cao hơn.

Kết quả embedding của ba mô hình sẽ được giảm chiều bằng **UMAP** trong bước tiếp theo để phục vụ cho các thuật toán phân cụm.

D. Giảm chiều dữ liệu bằng UMAP

Sau khi biểu diễn văn bản thành các vector đặc trưng có chiều cao từ các mô hình TF-IDF, PhoBERT-base và PhoBERT-large, bước tiếp theo là tiến hành **giảm chiều dữ liệu** để rút gọn không gian đặc trưng, giảm chi phí tính toán và loại bỏ nhiễu trước khi phân cụm. Trong nghiên cứu này, phương pháp **Uniform Manifold Approximation and Projection (UMAP)** [4] được lựa chọn nhờ khả năng bảo toàn cấu trúc cục bộ của dữ liệu tốt và tốc độ xử lý nhanh hơn so với các phương pháp truyền thống như t-SNE.

1) *Cơ sở lý thuyết của UMAP*: UMAP là một kỹ thuật học đa tạp (manifold learning) dựa trên lý thuyết tô pô và hình học Riemann. Phương pháp này giả định rằng dữ liệu quan sát trong không gian cao chiều thực chất nằm trên một đa tạp (manifold) có số chiều thấp hơn. Mục tiêu của UMAP là tìm ánh xạ:

$$f: \mathbb{R}^D \rightarrow \mathbb{R}^d, \quad d \ll D$$

sao cho cấu trúc lân cận (local neighborhood) giữa các điểm dữ liệu trong không gian gốc được giữ nguyên tối đa trong không gian giảm chiều.

2) *Nguyên lý hoạt động*: Thuật toán UMAP bao gồm ba giai đoạn chính:

a) *Xây dựng đồ thị lân cận mờ (fuzzy simplicial complex)*: Mỗi điểm dữ liệu x_i được nối với k điểm gần nhất của nó, hình thành một đồ thị trọng số. Xác suất kết nối giữa hai điểm x_i và x_j được định nghĩa như sau:

$$p_{ij} = \exp\left(-\frac{\max(0, d(x_i, x_j) - \rho_i)}{\sigma_i}\right)$$

trong đó $d(x_i, x_j)$ là khoảng cách giữa hai điểm, ρ_i là khoảng cách tối thiểu để đảm bảo tính liên thông, và σ_i là hệ số tỷ lệ cục bộ.

b) *Tối ưu hóa trong không gian giảm chiều*: Trong không gian thấp hơn, mô hình khởi tạo ngẫu nhiên các điểm y_i và xác định xác suất kết nối tương ứng:

$$q_{ij} = \frac{1}{1 + a \cdot d(y_i, y_j)^{2b}}$$

Hàm mất mát của UMAP là tổng entropy chéo giữa hai phân bố p_{ij} và q_{ij} :

$$C = \sum_{i \neq j} \left[p_{ij} \log \frac{p_{ij}}{q_{ij}} + (1 - p_{ij}) \log \frac{1 - p_{ij}}{1 - q_{ij}} \right]$$

Mục tiêu là tối thiểu hóa C thông qua thuật toán tối ưu gradient ngẫu nhiên (SGD).

c) *Tạo không gian biểu diễn mới*: Sau khi hội tụ, ta thu được tập điểm $\{y_i\}$ trong không gian \mathbb{R}^d thể hiện cấu trúc dữ liệu gốc dưới dạng cô đọng và dễ trực quan hóa hơn.

3) *Lý do lựa chọn UMAP*: So với các kỹ thuật giảm chiều khác như PCA hay t-SNE, UMAP có các ưu điểm nổi bật:

- **Bảo toàn tốt cấu trúc cục bộ** của dữ liệu, giúp các điểm gần nhau trong không gian gốc vẫn giữ khoảng cách gần trong không gian mới.
- **Hiệu quả tính toán cao**, có thể áp dụng trên hàng chục nghìn mẫu dữ liệu mà vẫn đảm bảo tốc độ xử lý.
- **Tái sử dụng mô hình được huấn luyện**, hỗ trợ thao tác `fit-transform` trên tập dữ liệu mới.
- **Kết hợp tốt với các phương pháp phân cụm** nhờ khả năng tạo không gian đặc trưng mượt mà và ít nhiễu.

4) *Ứng dụng trong nghiên cứu*: Trong nghiên cứu này, UMAP được sử dụng để giảm chiều dữ liệu của ba loại embedding:

- **TF-IDF**: giảm từ 3.000 chiều xuống 50 chiều.
- **PhoBERT-base**: giảm từ 768 chiều xuống 50 chiều.
- **PhoBERT-large**: giảm từ 1.024 chiều xuống 50 chiều.

Việc giảm chiều này giúp tối ưu hóa hiệu năng cho các thuật toán phân cụm (K-Means, Hierarchical, DBSCAN, HDBSCAN, Spectral Clustering) được trình bày trong các phần tiếp theo, đồng thời hỗ trợ trực quan hóa dữ liệu dễ dàng hơn trong không gian hai hoặc ba chiều.

E. Xác định số cụm tối ưu bằng phương pháp Elbow và chỉ số Silhouette

Trước khi áp dụng các thuật toán phân cụm, cần xác định số lượng cụm (K) phù hợp để đảm bảo kết quả phân nhóm đạt được độ chính xác và ổn định cao nhất. Trong nghiên cứu này, hai phương pháp phổ biến được sử dụng để xác định giá trị K tối ưu gồm:

- Phương pháp **Elbow (Khủy tay)** [12].
- Chỉ số **Silhouette** [13].

1) *Phương pháp Elbow (Khuyết tay)*: Phương pháp Elbow được đề xuất nhằm tìm ra số cụm K sao cho việc tăng thêm cụm không mang lại nhiều cải thiện đáng kể về độ nén dữ liệu. Ý tưởng chính là quan sát **tổng sai số bình phương trong cụm** (Within-Cluster Sum of Squares – WCSS) theo số cụm K [12].

Giả sử có tập dữ liệu $X = \{x_1, x_2, \dots, x_n\}$, sau khi được chia thành K cụm, ta tính WCSS như sau:

$$WCSS = \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

trong đó:

- C_k là tập các điểm thuộc cụm thứ k ,
- μ_k là trung tâm của cụm k ,
- $\|x_i - \mu_k\|^2$ là bình phương khoảng cách Euclidean giữa điểm dữ liệu và trung tâm cụm.

Khi tăng K , giá trị WCSS luôn giảm, vì các cụm nhỏ hơn mô tả dữ liệu chi tiết hơn. Tuy nhiên, sau một điểm nhất định, việc tăng K chỉ làm giảm WCSS không đáng kể — điểm gấp khúc (elbow point) tại đó được xem là giá trị K tối ưu.

a) *Quy trình thực hiện trong nghiên cứu*:

- Sử dụng thuật toán K-Means để tính WCSS cho từng giá trị K trong khoảng [2, 30].
- Biểu diễn đồ thị $WCSS$ theo K .
- Quan sát điểm “gãy khuỷu tay” để xác định K thích hợp cho từng embedding (TF-IDF, PhoBERT-base, PhoBERT-large).

2) *Chỉ số Silhouette*: Chỉ số Silhouette được sử dụng để đánh giá **độ tách biệt và độ đồng nhất** của các cụm sau khi phân nhóm [13]. Giá trị của chỉ số này nằm trong khoảng từ -1 đến 1 , trong đó:

- $s_i \approx 1$: điểm dữ liệu được gán đúng cụm.
- $s_i \approx 0$: điểm nằm giữa ranh giới hai cụm.
- $s_i < 0$: điểm bị gán sai cụm.

Công thức tính chỉ số Silhouette cho mỗi điểm dữ liệu i như sau:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

với:

- a_i : khoảng cách trung bình giữa điểm i và các điểm khác trong cùng cụm.
- b_i : khoảng cách trung bình nhỏ nhất giữa điểm i và các điểm thuộc cụm gần nhất khác.

Giá trị trung bình của tất cả s_i trong tập dữ liệu được gọi là **Silhouette Score**, đại diện cho chất lượng phân cụm toàn cục:

$$S = \frac{1}{n} \sum_{i=1}^n s_i$$

Quy trình thực hiện trong nghiên cứu:

- Với mỗi embedding sau khi giảm chiều bằng UMAP, tiến hành huấn luyện mô hình K-Means với K từ 2 đến 30.
- Tính Silhouette Score tương ứng với từng giá trị K .
- Lựa chọn K có chỉ số Silhouette cao nhất làm số cụm tối ưu.

3) *So sánh giữa hai phương pháp*: Hai phương pháp Elbow và Silhouette bổ sung cho nhau trong việc xác định số cụm hợp lý:

- Elbow mang tính **trực quan**, dễ nhận biết điểm giảm hiệu quả WCSS.
- Silhouette mang tính **định lượng**, phản ánh trực tiếp độ tách biệt giữa các cụm.

Do đó, trong nghiên cứu này, cả hai phương pháp được sử dụng song song nhằm đảm bảo lựa chọn giá trị K tối ưu một cách khách quan cho từng mô hình embedding.

F. Phân cụm văn bản

Sau khi dữ liệu văn bản được tiền xử lý, chuyển đổi thành vector embedding và giảm chiều bằng UMAP, bước tiếp theo là tiến hành **phân cụm (clustering)** để nhóm các bài viết có nội dung tương đồng. Phân cụm giúp khám phá cấu trúc tiềm ẩn của dữ liệu mà không cần nhãn trước (unsupervised learning). Trong nghiên cứu này, năm phương pháp phân cụm được áp dụng gồm: **K-Means**, **Hierarchical Clustering**, **DBSCAN**, **HDBSCAN** và **Spectral Clustering**.

1) *K-Means Clustering*: Thuật toán K-Means [14], [15] là phương pháp phân cụm phổ biến nhất do tính đơn giản và hiệu quả. Mục tiêu của K-Means là chia tập dữ liệu $X = \{x_1, x_2, \dots, x_n\}$ thành K cụm sao cho **tổng bình phương khoảng cách giữa các điểm và tâm cụm** là nhỏ nhất:

$$J = \sum_{k=1}^K \sum_{x_i \in C_k} \|x_i - \mu_k\|^2$$

trong đó:

- C_k : tập điểm thuộc cụm thứ k ,
- μ_k : trung tâm của cụm k ,
- $\|x_i - \mu_k\|^2$: khoảng cách Euclidean giữa điểm x_i và tâm cụm.

Quy trình K-Means gồm ba bước chính:

- Khởi tạo ngẫu nhiên K tâm cụm.
- Gán mỗi điểm dữ liệu vào cụm gần nhất.
- Cập nhật lại tâm cụm bằng trung bình các điểm trong cụm.

Thuật toán lặp lại đến khi hội tụ (các tâm cụm không thay đổi đáng kể). K-Means hoạt động hiệu quả trên dữ liệu có dạng cầu (spherical clusters), được dùng rộng rãi để nhóm văn bản theo chủ đề.

Ứng dụng trong nghiên cứu: Phương pháp này được sử dụng sau khi giảm chiều embedding (TF-IDF, PhoBERT-base, PhoBERT-large) để phân nhóm nội dung văn bản. Số cụm K được xác định thông qua phương pháp Elbow và chỉ số Silhouette.

2) *Hierarchical Clustering*: Phân cụm phân cấp (Hierarchical Clustering) [16], [17] là phương pháp tạo cấu trúc cây (dendrogram) biểu diễn mối quan hệ giữa các cụm dữ liệu. Có hai hướng tiếp cận:

- **Agglomerative (Bottom-up)**: mỗi điểm ban đầu là một cụm riêng biệt, sau đó các cụm gần nhau dần được gộp lại.
- **Divisive (Top-down)**: bắt đầu từ một cụm lớn, sau đó tách dần thành nhiều cụm nhỏ hơn.

Khoảng cách giữa hai cụm có thể được tính bằng các phương pháp:

- *Single linkage*: khoảng cách nhỏ nhất giữa hai điểm thuộc hai cụm.
- *Complete linkage*: khoảng cách lớn nhất giữa hai điểm thuộc hai cụm.
- *Average linkage*: khoảng cách trung bình giữa tất cả cặp điểm.
- *Ward linkage*: dựa trên phương sai để giảm tổng bình phương sai số.

Kết quả được thể hiện dưới dạng dendrogram, giúp trực quan hóa quá trình hình thành cụm. Hierarchical Clustering phù hợp khi cần quan sát cấu trúc phân cấp của dữ liệu mà không cần xác định trước số cụm.

Ứng dụng trong nghiên cứu: Phương pháp Ward linkage được áp dụng để phân cụm embedding sau UMAP, đồng thời vẽ dendrogram cho từng mô hình nhằm quan sát sự phân nhánh tự nhiên của văn bản.

3) *DBSCAN (Density-Based Spatial Clustering of Applications with Noise)*: DBSCAN [5] là thuật toán phân cụm dựa trên mật độ, không yêu cầu xác định trước số cụm. Ý tưởng là các cụm được hình thành từ các điểm dữ liệu có mật độ cao, trong khi các điểm nằm trong vùng thưa thớt được xem là nhiễu.

Hai tham số chính của DBSCAN:

- ε (epsilon): bán kính lân cận.
- *MinPts*: số điểm tối thiểu trong vùng ε để hình thành một cụm.

Các điểm được phân loại như sau:

- **Core point**: có ít nhất *MinPts* điểm trong vùng ε .
- **Border point**: nằm trong vùng ε của điểm lõi, nhưng không đủ mật độ.
- **Noise point**: không thuộc bất kỳ cụm nào.

Thuật toán lặp qua từng điểm, mở rộng cụm từ các điểm lõi, và loại bỏ điểm nhiễu. DBSCAN đặc biệt hữu ích cho dữ liệu có hình dạng cụm phức tạp (phi cầu) và chứa nhiễu.

Ứng dụng trong nghiên cứu: DBSCAN được dùng để phân cụm các văn bản trong không gian embedding sau khi giảm chiều bằng UMAP, nhằm phát hiện các nhóm nội dung tự nhiên mà không cần xác định trước K .

4) *HDBSCAN (Hierarchical DBSCAN)*: HDBSCAN [6], [18] là phần mở rộng của DBSCAN, kết hợp giữa ý tưởng mật độ và phân cấp. Khác với DBSCAN, HDBSCAN không cần chọn một giá trị cố định cho ε , mà xây dựng **cấu trúc phân cấp theo mật độ** (density hierarchy) và trích xuất cụm tối ưu dựa trên độ ổn định.

Các bước chính:

- Xây dựng đồ thị khoảng cách có trọng số bằng khoảng cách tương hỗ (mutual reachability distance).
- Sinh cây phân cấp (hierarchical tree) bằng cách giảm dần ngưỡng mật độ.
- Cắt cây để chọn ra các cụm ổn định nhất dựa trên chỉ số *cluster stability*.

HDBSCAN có khả năng:

- Xử lý dữ liệu có mật độ không đồng đều.
- Phát hiện và đánh dấu điểm nhiễu rõ ràng.
- Tự động xác định số lượng cụm.

Ứng dụng trong nghiên cứu: Phương pháp này được sử dụng nhằm cải thiện hạn chế của DBSCAN trong việc chọn tham số ε , giúp phân tách các cụm văn bản tự nhiên và ổn định hơn.

5) *Spectral Clustering*: Spectral Clustering [7], [19] là thuật toán phân cụm dựa trên lý thuyết đồ thị. Thay vì làm việc trực tiếp trên không gian đặc trưng, Spectral Clustering biểu diễn dữ liệu dưới dạng **đồ thị kề** $G = (V, E)$, trong đó mỗi đỉnh tương ứng với một điểm dữ liệu, và các cạnh biểu diễn mức độ tương đồng.

Ma trận Laplacian được định nghĩa như:

$$L = D - W$$

với:

- W : ma trận trọng số biểu diễn độ tương đồng giữa các điểm.
- D : ma trận đường chéo chứa tổng trọng số của mỗi đỉnh, $D_{ii} = \sum_j W_{ij}$.

Sau đó, Spectral Clustering tìm các **vector riêng** (eigenvectors) tương ứng với các giá trị riêng nhỏ nhất của L , rồi áp dụng K-Means trên không gian các vector này để phân cụm.

Ứng dụng trong nghiên cứu: Spectral Clustering được chọn nhờ khả năng phát hiện các cụm có ranh giới phi tuyến, giúp mô hình hóa tốt mối quan hệ ngữ nghĩa phức tạp giữa các văn bản trong không gian embedding.

G. Chỉ số đánh giá hiệu quả phân cụm

Sau khi thực hiện phân cụm văn bản bằng các thuật toán khác nhau, cần có những chỉ số đánh giá khách quan để xác định chất lượng và độ hợp lý của kết quả. Các chỉ số này giúp đo mức độ **đồng nhất trong cụm (cohesion)**, **tách biệt giữa các cụm (separation)**, và **tỷ lệ nhiễu (noise)**. Trong nghiên cứu này, bốn chỉ số được sử dụng: **Silhouette Score**, **Davies–Bouldin Index**, **Calinski–Harabasz Index** và **Noise Ratio**.

1. Silhouette Score

Silhouette Score [13] là một chỉ số phổ biến để đánh giá hiệu quả phân cụm mà không cần nhãn dữ liệu. Nó phản ánh mức độ một điểm dữ liệu x_i phù hợp với cụm mà nó được gán, thông qua hai giá trị:

$$a(i) = \frac{1}{|C_i| - 1} \sum_{x_j \in C_i, j \neq i} d(x_i, x_j)$$

$$b(i) = \min_{k \neq i} \frac{1}{|C_k|} \sum_{x_j \in C_k} d(x_i, x_j)$$

Trong đó:

- $a(i)$: khoảng cách trung bình giữa x_i và các điểm trong cùng cụm (độ gắn kết nội cụm);
- $b(i)$: khoảng cách trung bình nhỏ nhất từ x_i đến các cụm khác (độ tách biệt liên cụm).

Giá trị Silhouette của điểm i được tính theo:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

với $s(i) \in [-1, 1]$. Trung bình tất cả các điểm:

$$S = \frac{1}{n} \sum_{i=1}^n s(i)$$

Diễn giải:

- $S \approx 1$: các cụm tách biệt tốt, dữ liệu phân bố hợp lý.
- $S \approx 0$: biên giới giữa các cụm chồng lấn.
- $S < 0$: nhiều điểm bị gán sai cụm.

Ưu điểm: dễ tính toán, không yêu cầu nhãn. **Nhược điểm:** không phản ánh tốt khi dữ liệu có mật độ cụm không đồng đều.

2. Davies–Bouldin Index (DBI)

Davies–Bouldin Index [20] đo độ tương đồng giữa các cụm dựa trên tỉ lệ giữa độ phân tán trong cụm và khoảng cách giữa các cụm. Đối với hai cụm C_i và C_j :

$$R_{ij} = \frac{S_i + S_j}{M_{ij}}$$

với:

- $S_i = \frac{1}{|C_i|} \sum_{x \in C_i} \|x - \mu_i\|$: độ phân tán trung bình trong cụm i ;
- $M_{ij} = \|\mu_i - \mu_j\|$: khoảng cách giữa tâm cụm i và j .

Chỉ số tổng thể:

$$DBI = \frac{1}{K} \sum_{i=1}^K \max_{j \neq i} R_{ij}$$

Diễn giải:

- DBI thấp \rightarrow cụm tách biệt tốt, phân tán nhỏ \rightarrow chất lượng cao.
- DBI cao \rightarrow cụm chồng lấn, biên giới không rõ.

Ưu điểm: tính toán đơn giản, hiệu quả cho dữ liệu có cấu trúc cầu. **Nhược điểm:** không phản ánh tốt khi cụm có mật độ khác nhau.

3. Calinski–Harabasz Index (CH Index)

Chỉ số Calinski–Harabasz [21] (còn gọi là Variance Ratio Criterion) đo tỷ lệ giữa phân tán giữa cụm và trong cụm:

$$CH = \frac{\text{Tr}(B_k)/(K-1)}{\text{Tr}(W_k)/(N-K)}$$

trong đó:

- B_k : tổng phương sai giữa các cụm;
- W_k : tổng phương sai trong từng cụm;
- N : số điểm dữ liệu; K : số cụm.

Diễn giải:

- CH càng cao \rightarrow các cụm càng tách biệt và đồng nhất nội bộ.
- CH thấp \rightarrow cụm phân tán hoặc chồng lấn.

Ưu điểm: đơn giản, tính nhanh, trực quan. **Nhược điểm:** không phù hợp với cụm có kích thước quá chênh lệch.

4. Noise Ratio

Trong các thuật toán dựa trên mật độ như DBSCAN và HDBSCAN, không phải tất cả các điểm đều thuộc về cụm — một số điểm được gán nhãn là **nhiều (noise)** nếu chúng nằm ngoài vùng mật độ đủ cao. Tỷ lệ điểm nhiễu phản ánh khả năng mô hình xác định đúng các vùng dữ liệu có ý nghĩa.

Giả sử có tổng số N điểm dữ liệu, trong đó N_{noise} là số điểm bị xem là nhiễu, thì:

$$\text{Noise Ratio} = \frac{N_{noise}}{N} \times 100\%$$

Diễn giải:

- Tỷ lệ nhiễu thấp (1–5%) \rightarrow cụm ổn định, ít điểm bị loại bỏ.
- Tỷ lệ nhiễu vừa phải (5–15%) \rightarrow hợp lý khi dữ liệu có vùng mật độ khác nhau.
- Tỷ lệ nhiễu cao (>20%) \rightarrow mô hình quá nghiêm ngặt hoặc $\epsilon, min_samples$ chưa phù hợp.

Ưu điểm: giúp đánh giá khả năng lọc nhiễu của thuật toán. **Nhược điểm:** không áp dụng cho các thuật toán yêu cầu số cụm cố định (như K-Means, Spectral Clustering).

5. Tổng kết

Bảng I
TỔNG HỢP CÁC CHỈ SỐ ĐÁNH GIÁ CHẤT LƯỢNG PHÂN CỤM

Chỉ số	Ý nghĩa đo lường	Xu hướng tốt
Silhouette Score	Mức độ đồng nhất và tách biệt cụm	Cao hơn tốt hơn
Davies–Bouldin Index	Độ tương đồng giữa các cụm	Thấp hơn tốt hơn
Calinski–Harabasz Index	Tỷ lệ phân tán giữa và trong cụm	Cao hơn tốt hơn
Noise Ratio	Tỷ lệ điểm bị xem là nhiễu	Thấp hơn tốt hơn

IV. THIẾT LẬP THÍ NGHIỆM

A. Môi trường thực nghiệm

Toàn bộ quá trình thực nghiệm được thực hiện trên máy tính cá nhân với cấu hình như sau:

Bảng II
CẤU HÌNH MÔI TRƯỜNG THỰC NGHIỆM

Thành phần	Thông tin
Hệ điều hành	Windows 11 64-bit
Bộ xử lý (CPU)	Intel Core i7-12700H (14 nhân, 20 luồng)
RAM	16 GB DDR5
Ngôn ngữ lập trình	Python 3.13.2
IDE	PyCharm Community Edition 2022.2.3

Thư viện chính sử dụng:

Trong quá trình cài đặt và thực nghiệm, các thư viện Python sau được sử dụng để thực hiện các bước tiền xử lý, trích xuất đặc trưng, giảm chiều, phân cụm và đánh giá mô hình:

- **pandas, numpy**: hỗ trợ xử lý, làm sạch và thao tác dữ liệu dạng bảng.
- **underthesea**: dùng để tách từ và chuẩn hoá văn bản tiếng Việt.
- **scikit-learn**: cung cấp các công cụ cho TF-IDF, KMeans, Hierarchical, Spectral Clustering, cùng các chỉ số đánh giá như Silhouette, Davies–Bouldin và Calinski–Harabasz.
- **hdbscan**: triển khai thuật toán HDBSCAN giúp phát hiện cụm có mật độ thay đổi.
- **umap-learn**: dùng để giảm chiều dữ liệu và trực quan hóa không gian biểu diễn.
- **transformers (Hugging Face)**: sử dụng các mô hình ngôn ngữ như PhoBERT để sinh vector biểu diễn câu.
- **matplotlib, seaborn**: phục vụ trực quan hoá dữ liệu và kết quả phân cụm.
- **tqdm**: tạo thanh tiến trình trong quá trình xử lý và huấn luyện, giúp dễ theo dõi tiến độ.

B. Dữ liệu đầu vào

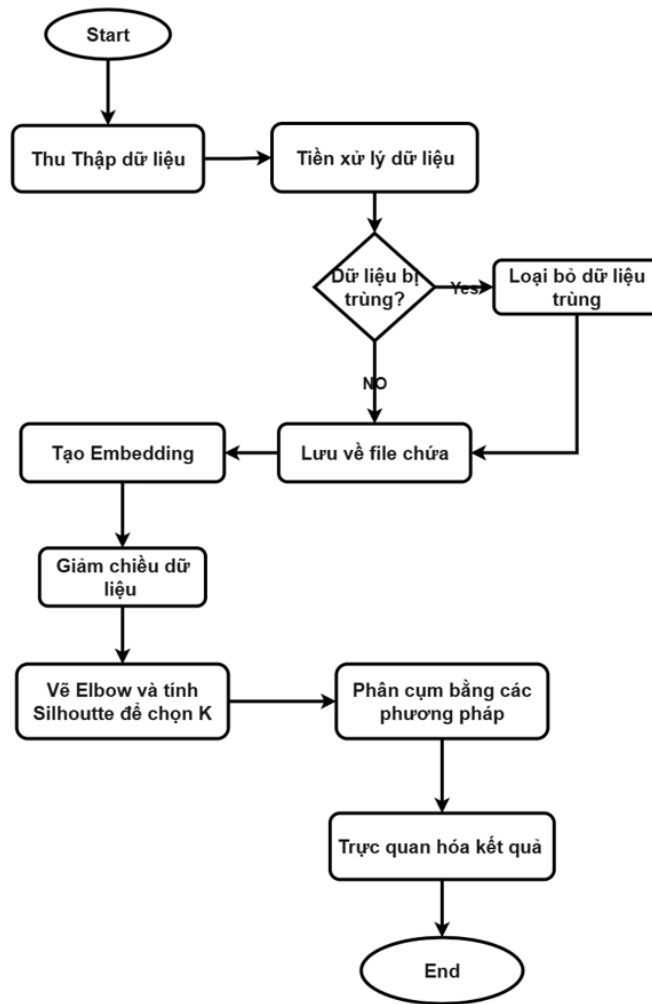
Dữ liệu đầu vào của nghiên cứu được thu thập từ trang báo điện tử **Dân Trí** — một trong những nguồn tin tức phổ biến và có lượng bài viết lớn tại Việt Nam. Mục tiêu của việc thu thập dữ liệu là xây dựng một tập văn bản đa dạng về chủ đề nhằm phục vụ cho các thuật toán phân cụm văn bản tiếng Việt.

Mô tả dữ liệu:

- Tổng số bài viết sau khi thu thập là **7.580 bài**.
- Sau khi tiến hành các bước tiền xử lý như loại bỏ trùng lặp, lọc các bài quá ngắn (ít hơn 8 từ), và loại bỏ bài không có nội dung, tập dữ liệu còn lại gồm **7.278 bài viết hợp lệ**.
- Mỗi dòng dữ liệu biểu diễn một văn bản (một bài báo) — là đối tượng đầu vào cho quá trình trích xuất đặc trưng (embedding) và phân cụm.

C. Thuật toán và quy trình thực hiện

Quy trình thực hiện được thiết kế nhằm xây dựng một hệ thống phân cụm văn bản tiếng Việt có khả năng hoạt động ổn định và khách quan trên nhiều dạng biểu diễn khác nhau. Mục tiêu chính là so sánh hiệu quả của các mô hình biểu diễn (embedding) và các thuật toán phân cụm phổ biến trong việc phát hiện các nhóm chủ đề tiềm ẩn từ dữ liệu báo điện tử. Toàn bộ quy trình được minh họa trong Hình 2.



Hình 2. Quy trình tổng quát của hệ thống phân cụm văn bản tiếng Việt

(1) Giai đoạn tiền xử lý dữ liệu:

- Dữ liệu văn bản thô được thu thập từ trang **Dân Trí** qua công cụ tự động sử dụng Selenium.
- Các bước tiền xử lý gồm: chuyển chữ thường, loại bỏ ký tự đặc biệt và URL, tách từ bằng thư viện *Underthesea*, loại bỏ stopwords và các câu có độ dài dưới 8 từ.
- Dữ liệu sau xử lý được lưu dưới dạng `clean_text`, đảm bảo tính chuẩn hóa cho bước biểu diễn đặc trưng.

(2) Giai đoạn trích xuất đặc trưng (Embedding):

- Ba mô hình embedding được áp dụng để biểu diễn văn bản thành vector số:
 - **TF-IDF**: biểu diễn thống kê dựa trên tần suất xuất hiện của từ trong tập dữ liệu.
 - **PhoBERT-base** và **PhoBERT-large**: hai biến thể của mô hình ngôn ngữ tiền huấn luyện (pre-trained language model) trên kho ngữ liệu tiếng Việt, giúp biểu diễn ngữ nghĩa sâu hơn.
- Kết quả của giai đoạn này là ba ma trận embedding tương ứng có kích thước:

$$TF-IDF \in \mathbb{R}^{7278 \times 3000}, \quad PhoBERT-base \in \mathbb{R}^{7278 \times 768}, \quad PhoBERT-large \in \mathbb{R}^{7278 \times 1024}.$$

(3) Giai đoạn giảm chiều dữ liệu:

- Để tối ưu hoá tốc độ tính toán và trực quan hoá, mô hình **UMAP (Uniform Manifold Approximation and Projection)** được sử dụng để giảm chiều các vector embedding về không gian 50 chiều.
- UMAP giúp bảo toàn cấu trúc cục bộ của dữ liệu tốt hơn so với PCA, đồng thời hỗ trợ hiệu quả cho các thuật toán dựa trên khoảng cách như K-Means hoặc DBSCAN.

(4) Giai đoạn phân cụm:

- Trên các tập embedding đã giảm chiều, năm thuật toán phân cụm được triển khai:

- **K-Means**: thuật toán dựa trên khoảng cách Euclidean, phân chia dữ liệu thành K cụm cố định.
- **Hierarchical Clustering**: phân cấp dữ liệu thành cây dendrogram, cho phép quan sát cấu trúc cụm con.
- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)**: phân cụm dựa trên mật độ, có khả năng phát hiện điểm nhiễu.
- **HDBSCAN (Hierarchical DBSCAN)**: mở rộng của DBSCAN, tự động xác định số cụm tối ưu và tách biệt tốt vùng nhiễu.
- **Spectral Clustering**: sử dụng phân rã trị riêng của ma trận Laplacian để phân tách cụm theo cấu trúc đồ thị.
- Tham số K đối với các thuật toán yêu cầu số cụm cố định được lựa chọn dựa trên kết quả của hai phương pháp đánh giá: **Elbow Method** và **Silhouette Analysis**.

(5) Giai đoạn đánh giá và trực quan hoá:

- Chất lượng phân cụm được đánh giá thông qua các chỉ số: **Silhouette Score**, **Davies–Bouldin Index**, **Calinski–Harabasz Index** và **Noise Ratio**.
- Kết quả được trực quan hoá bằng các biểu đồ:
 - Biểu đồ **Elbow** và **Silhouette** để chọn giá trị K tối ưu.
 - Biểu đồ **UMAP 2D** thể hiện phân bố cụm trong không gian hai chiều.
 - Biểu đồ phân bố **Noise** giúp nhận biết tỷ lệ điểm nhiễu trong các thuật toán dựa trên mật độ.

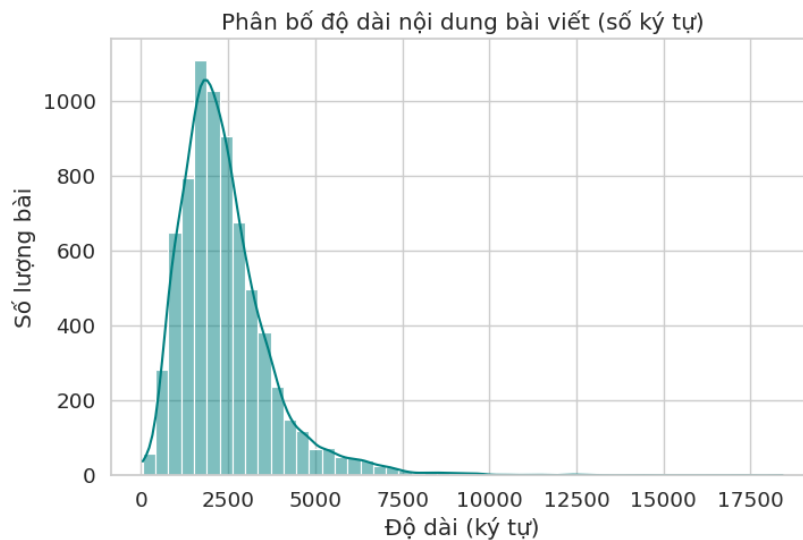
V. KẾT QUẢ VÀ THẢO LUẬN

A. Tổng quan dữ liệu

Tập dữ liệu sau khi tiền xử lý gồm **7.278 bài viết** được thu thập từ báo điện tử **Dân Trí**. Mỗi bài viết bao gồm ba trường thông tin chính: `category`, `content`, và `clean_text`. Trong đó:

- `category`: chuyên mục của bài viết (ví dụ: Thời sự, Kinh doanh, Giáo dục, v.v.);
- `content`: nội dung gốc được trích xuất từ trang web;
- `clean_text`: nội dung sau khi đã tiền xử lý (chuyển chữ thường, loại bỏ ký tự đặc biệt, tách từ và loại bỏ stopwords).

Dữ liệu không chứa giá trị thiếu ở bất kỳ cột nào. Độ dài trung bình của các bài viết là khoảng **2.402 ký tự**, cho thấy độ đa dạng vừa phải về độ dài nội dung, phù hợp cho bài toán biểu diễn văn bản và phân cụm.



Hình 3. Phân bố độ dài nội dung bài viết (số ký tự)

Nhìn chung, phân bố độ dài có dạng *lệch phải* (right-skewed), trong đó phần lớn bài viết có độ dài từ 1.000–3.000 ký tự. Điều này phản ánh đặc trưng của các bài báo trực tuyến — nội dung ngắn gọn, tập trung và dễ xử lý khi biểu diễn bằng vector hoá ngôn ngữ.

B. Biểu diễn dữ liệu và giảm chiều

Sau khi tiền xử lý, dữ liệu văn bản được biểu diễn dưới ba dạng vector embedding khác nhau nhằm phục vụ cho quá trình phân cụm. Ba mô hình embedding được sử dụng gồm: **TF-IDF**, **PhoBERT-base** và **PhoBERT-large**. Các mô hình này được lưu trữ dưới dạng tệp và kiểm tra lại trước khi đưa vào giai đoạn giảm chiều.

1. Kết quả embedding:

- **TF-IDF**: được lưu trong tệp `X_tfidf.npz`, có kích thước (7278, 3000). Ma trận ở dạng CSR sparse matrix với mật độ chỉ khoảng 4.45%, thể hiện sự thưa thớt của không gian đặc trưng từ vựng. Ví dụ, vector tại dòng thứ 6 có dạng rút gọn:

`[0, 0, 0, 0.0347, 0, 0, 0, 0, ...]`

- **PhoBERT-base**: lưu trong tệp `embeddings_phobert_base.npy`, kích thước (7278, 768). Đây là mô hình ngôn ngữ tiền huấn luyện được huấn luyện trên kho dữ liệu tiếng Việt lớn, cho phép biểu diễn ngữ nghĩa theo ngữ cảnh.
- **PhoBERT-large**: lưu trong tệp `embeddings_phobert_large.npy`. Đây là phiên bản lớn hơn của PhoBERT, kích thước (7278, 1024). Giúp mô hình hóa ngữ nghĩa phức tạp hơn nhờ không gian đặc trưng có chiều cao hơn.

2. Giảm chiều bằng UMAP:

Để tối ưu hóa thời gian tính toán và thuận tiện cho trực quan hoá, cả ba embedding đều được giảm chiều bằng mô hình **UMAP (Uniform Manifold Approximation and Projection)** từ kích thước gốc xuống còn 50 chiều. UMAP được lựa chọn thay vì PCA do khả năng bảo toàn tốt hơn cấu trúc cục bộ và phi tuyến của dữ liệu trong không gian nhúng.

Quá trình giảm chiều được thực hiện tuần tự cho ba mô hình:

- **TF-IDF (UMAP)**: kích thước sau giảm (7278, 50), lưu tại tệp `X_tfidf_umap.npy`;
- **PhoBERT-base (UMAP)**: kích thước (7278, 50), lưu tại tệp `embeddings_phobert_base_umap.npy`;
- **PhoBERT-large (UMAP)**: kích thước (7278, 50), lưu tại tệp `embeddings_phobert_large_umap.npy`.

Kết quả cho thấy cả ba tập vector sau giảm chiều đều có phân bố ổn định và cấu trúc dữ liệu rõ ràng hơn trong không gian 50 chiều, hỗ trợ tốt cho các bước phân cụm tiếp theo.

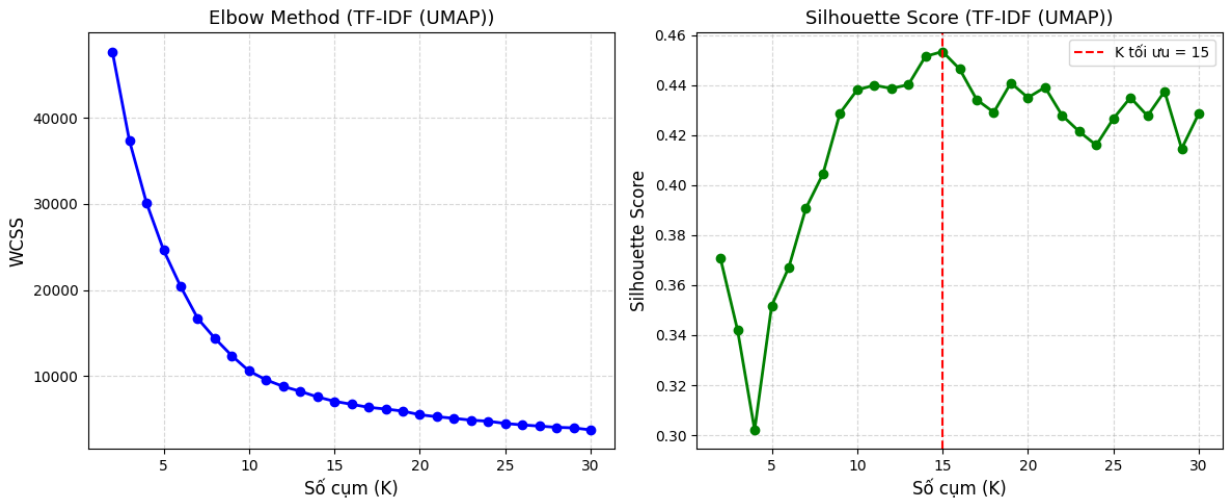
C. Xác định số cụm bằng Elbow và Silhouette

Để lựa chọn số cụm K tối ưu cho các mô hình phân cụm, hai phương pháp phổ biến được áp dụng là **Elbow Method** và **Silhouette Score**. Mỗi phương pháp cung cấp một góc nhìn khác nhau về cấu trúc của dữ liệu sau khi giảm chiều bằng UMAP.

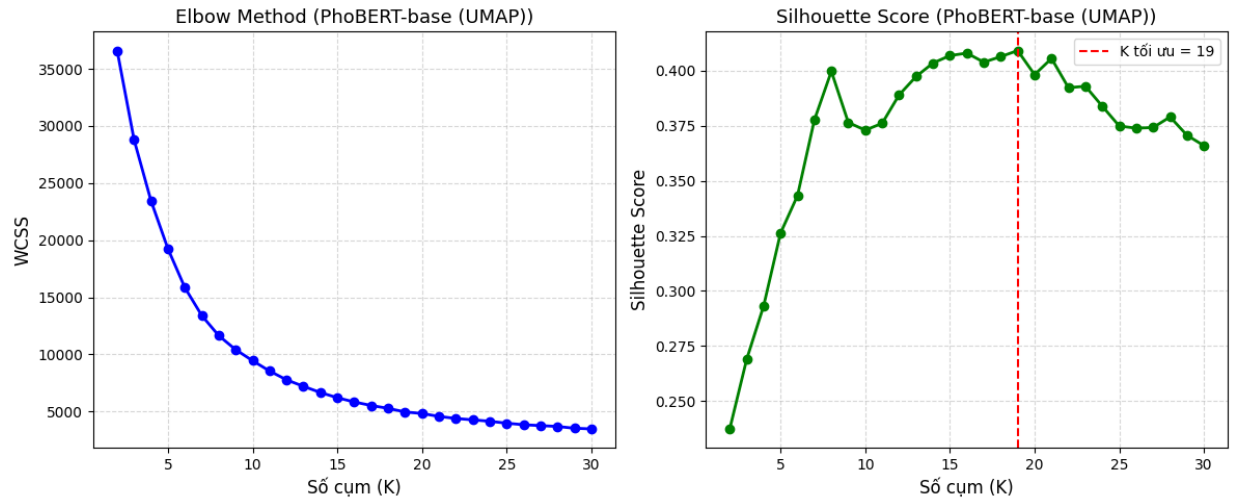
Kết quả lựa chọn số cụm: Dựa trên hai phương pháp trên, số cụm tối ưu được xác định như sau:

- **TF-IDF (UMAP)**: $K = 15$
- **PhoBERT-base (UMAP)**: $K = 19$
- **PhoBERT-large (UMAP)**: $K = 8$

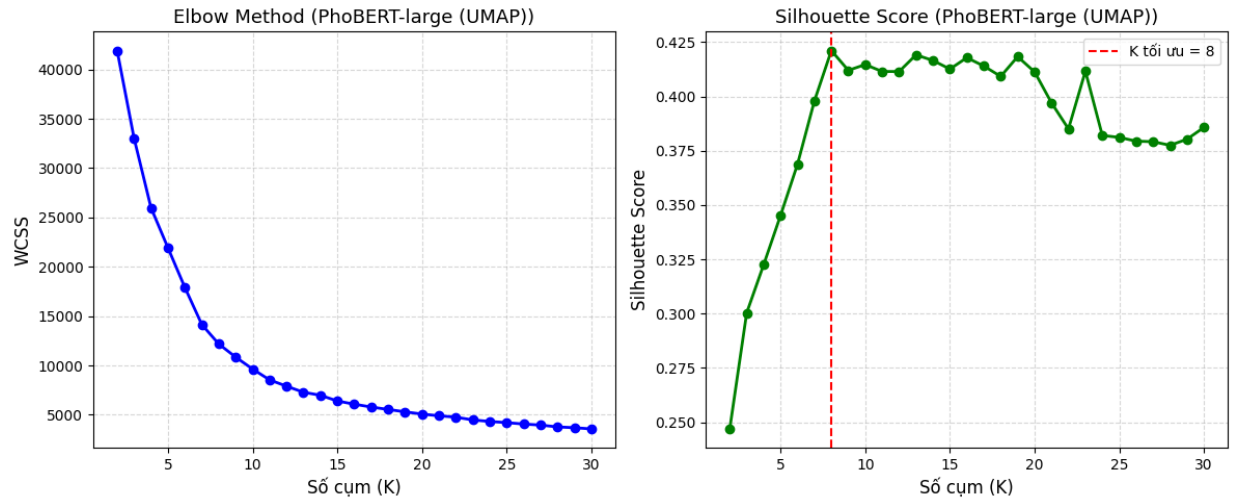
Các biểu đồ Elbow và Silhouette của ba mô hình embedding được trình bày dưới đây:



Hình 4. Elbow và Silhouette cho TF-IDF (UMAP), $K = 15$



Hình 5. Elbow và Silhouette cho PhoBERT-base (UMAP), $K = 19$



Hình 6. Elbow và Silhouette cho PhoBERT-large (UMAP), $K = 8$

Nhìn chung, đường cong WCSS cho thấy các điểm gấp rõ ràng tương ứng với giá trị K tối ưu, trong khi biểu đồ Silhouette khẳng định chất lượng phân cụm ổn định ở các giá trị trên. Điều này cho phép lựa chọn số cụm phù hợp cho từng mô hình embedding trước khi tiến hành các thuật toán phân cụm khác nhau.

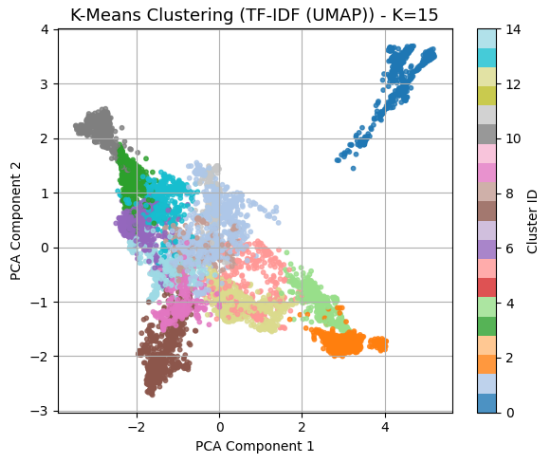
D. Phân cụm

1. Hierarchical Clustering

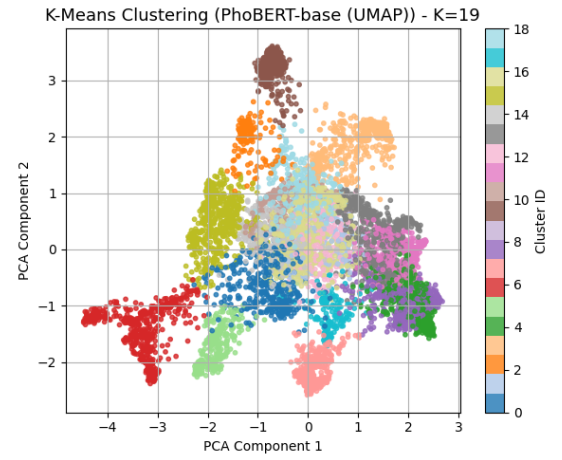
Thuật toán **Hierarchical Clustering** được áp dụng trên ba mô hình embedding (TF-IDF, PhoBERT-base và PhoBERT-large) sau khi giảm chiều bằng UMAP. Phương pháp này giúp biểu diễn mối quan hệ giữa các điểm dữ liệu thông qua cấu trúc cây phân cấp (dendrogram), trong đó các cụm nhỏ được gộp dần thành cụm lớn hơn dựa trên khoảng cách giữa chúng.

Ba chỉ số đánh giá chính được sử dụng gồm:

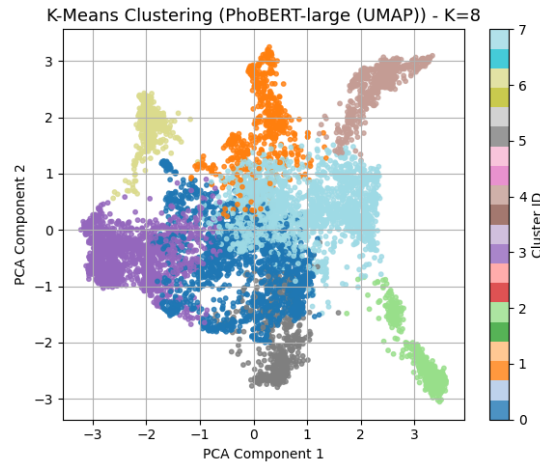
- **Silhouette Score**: đo mức độ tương đồng của một điểm với các điểm trong cùng cụm so với các cụm khác. Giá trị càng cao (gần 1) thì cụm càng tách biệt và gắn kết tốt.
- **Calinski-Harabasz Index**: phản ánh tỷ lệ giữa phương sai giữa các cụm và phương sai trong cụm. Giá trị lớn thể hiện cụm được phân tách rõ ràng.
- **Davies-Bouldin Index**: đo mức độ chồng lấn giữa các cụm. Giá trị càng thấp thì chất lượng phân cụm càng tốt.



Hình 10. Phân cụm KMeans với TF-IDF (UMAP), $K = 15$



Hình 11. Phân cụm KMeans với PhoBERT-base (UMAP), $K = 19$



Hình 12. Phân cụm KMeans với PhoBERT-large (UMAP), $K = 8$

Nhìn chung, phương pháp KMeans cho kết quả tốt hơn so với Hierarchical, với các chỉ số Silhouette và Calinski-Harabasz cao hơn rõ rệt. Điều này cho thấy dữ liệu tin tức sau khi biểu diễn bằng embedding và giảm chiều bằng UMAP có thể được gom cụm khá hiệu quả bằng thuật toán KMeans.

3. DBSCAN

Phương pháp **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** phát hiện cụm dựa trên mật độ điểm dữ liệu mà không cần xác định trước số cụm. Thuật toán dựa trên hai tham số: epsilon (bán kính lân cận) và min samples (số điểm tối thiểu để tạo cụm). DBSCAN đặc biệt hiệu quả trong việc nhận diện cụm có hình dạng phức tạp và loại bỏ các điểm nhiễu.

Bảng V
ĐÁNH GIÁ CHẤT LƯỢNG PHÂN CỤM BẰNG DBSCAN

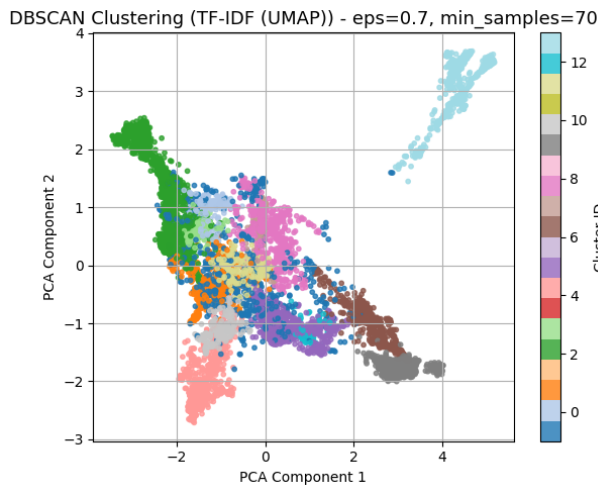
Embedding	eps	min_samples	Silhouette	Davies-Bouldin	Calinski-Harabasz
TF-IDF (UMAP)	0.7	70	0.3120	1.2622	2480.17
PhoBERT-base (UMAP)	0.6	60	0.2574	1.1541	1866.32
PhoBERT-large (UMAP)	0.7	70	0.2262	1.3783	2189.11

Phân tích kết quả:

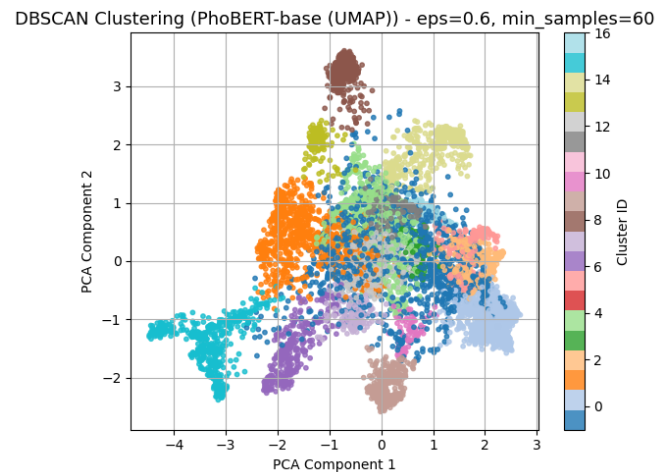
- **TF-IDF (UMAP)** đạt Silhouette Score cao nhất (0.3120), cho thấy cấu trúc cụm tương đối rõ ràng so với hai embedding còn lại. Tuy nhiên, chỉ số Davies-Bouldin là 1.2622 cho thấy vẫn tồn tại mức độ chồng lấn nhất định giữa các cụm. Chỉ số Calinski-Harabasz đạt 2480.17 phản ánh sự phân tách vừa phải giữa các cụm.

- **PhoBERT-base (UMAP)** có Silhouette Score 0.2574 và Davies–Bouldin 1.1541 — thấp hơn TF-IDF, thể hiện mức phân cụm kém tách biệt hơn. Dù vậy, giá trị Calinski–Harabasz (1866.32) cho thấy cấu trúc cụm vẫn tương đối ổn định khi xét về mật độ điểm.
- **PhoBERT-large (UMAP)** đạt Silhouette Score thấp nhất (0.2262), phản ánh sự phân tán và chồng lấn cao giữa các cụm. Davies–Bouldin đạt 1.3783 — lớn nhất trong ba mô hình — chứng tỏ ranh giới giữa các cụm mờ hơn, tuy nhiên vẫn thể hiện khả năng gom cụm ở mức chấp nhận được trong không gian ngữ nghĩa sâu hơn.

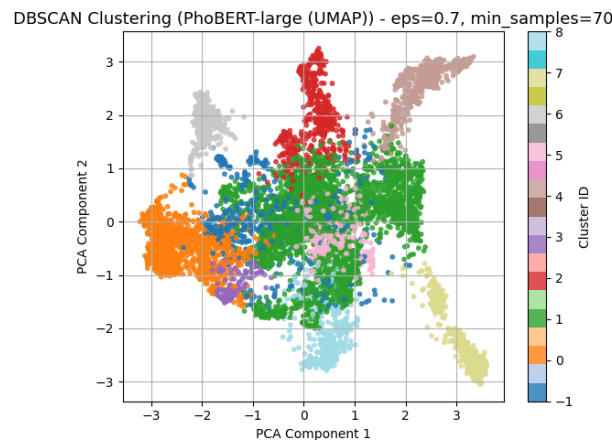
Trực quan hoá kết quả phân cụm:



Hình 13. DBSCAN với TF-IDF (UMAP), $\epsilon=0.7$, $\min_samples=70$



Hình 14. DBSCAN với PhoBERT-base (UMAP), $\epsilon=0.6$, $\min_samples=60$



Hình 15. DBSCAN với PhoBERT-large (UMAP), $\epsilon=0.7$, $\min_samples=70$

Tổng thể, phương pháp DBSCAN cho phép xác định các vùng dữ liệu dày đặc mà không cần xác định trước số cụm, đồng thời tự động nhận diện các điểm nhiễu.

4. HDBSCAN

Phương pháp **HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise)** mở rộng từ DBSCAN, hoạt động dựa trên nguyên lý phân cụm theo mật độ kết hợp phân cấp. Thay vì cố định ϵ như DBSCAN, HDBSCAN tự động xác định cấu trúc cụm thông qua cây mật độ, giúp thích nghi tốt hơn với dữ liệu có mật độ không đồng nhất và giảm phụ thuộc tham số.

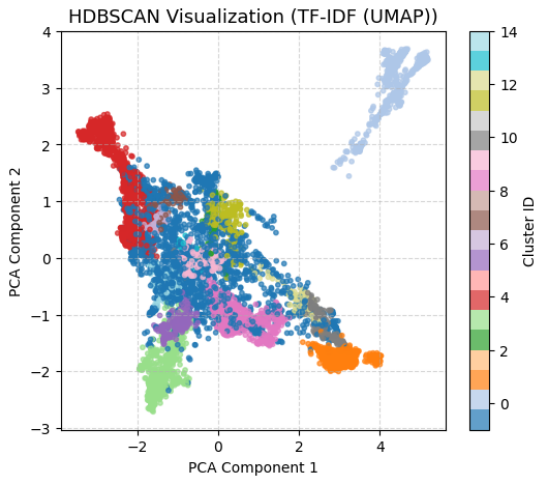
Bảng VI
ĐÁNH GIÁ CHẤT LƯỢNG PHÂN CỤM BẰNG HDBSCAN

Embedding	Số cụm	Silhouette	Davies–Bouldin	Calinski–Harabasz	Noise (%)
TF-IDF (UMAP)	15	0.4727	0.7646	2721.27	29.06
PhoBERT-base (UMAP)	15	0.4487	0.7098	2501.67	30.87
PhoBERT-large (UMAP)	14	0.4867	0.6693	3092.03	33.86

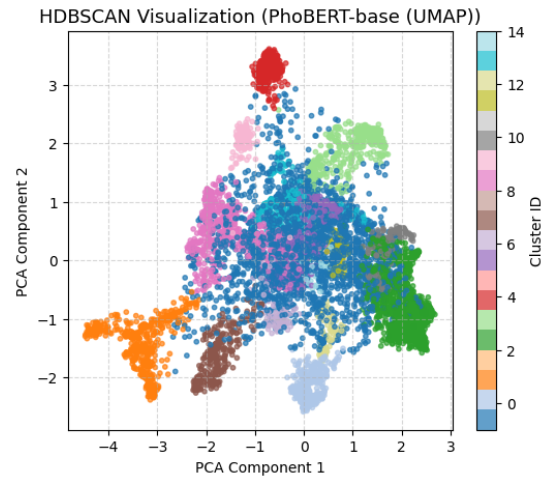
Phân tích kết quả:

- **TF-IDF (UMAP)** đạt Silhouette Score 0.4727 và Davies–Bouldin 0.7646, cho thấy khả năng phân tách cụm khá tốt với ranh giới cụm rõ ràng hơn so với DBSCAN. Tuy nhiên, tỷ lệ điểm nhiễu khoảng 29.06% phản ánh rằng vẫn còn nhiều văn bản chưa thuộc cụm nào cụ thể, thường do các bài có nội dung ngắn hoặc khác biệt chủ đề.
- **PhoBERT-base (UMAP)** có Silhouette Score 0.4487 — thấp hơn một chút so với TF-IDF, cho thấy mức độ đồng nhất giữa các điểm trong cụm giảm nhẹ. Dù vậy, Davies–Bouldin 0.7098 thấp hơn, chứng tỏ các cụm có xu hướng gọn hơn. Noise chiếm khoảng 30.87%.
- **PhoBERT-large (UMAP)** đạt Silhouette cao nhất (0.4867) và Davies–Bouldin thấp nhất (0.6693), cho thấy khả năng gom cụm hiệu quả nhất trong ba mô hình. Tuy nhiên, tỷ lệ điểm nhiễu tăng lên 33.86%, thể hiện sự đánh đổi giữa việc phân cụm rõ ràng và việc loại bỏ các điểm không đủ mật độ.

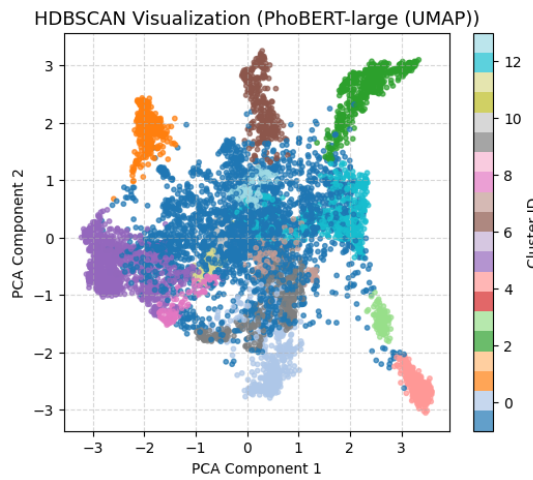
Trực quan hoá kết quả phân cụm:



Hình 16. Phân cụm HDBSCAN với TF-IDF (UMAP)



Hình 17. Phân cụm HDBSCAN với PhoBERT-base (UMAP)



Hình 18. Phân cụm HDBSCAN với PhoBERT-large (UMAP)

5. Spectral Clustering

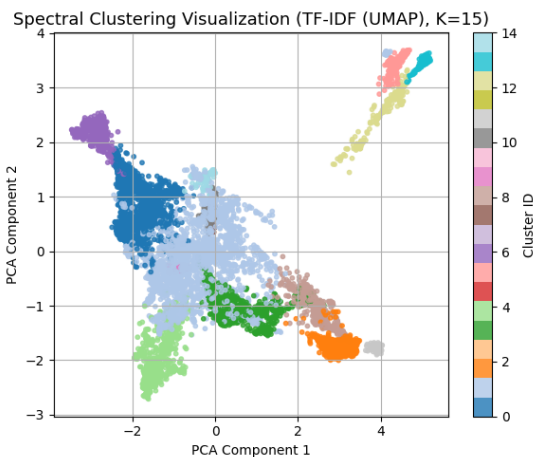
Spectral Clustering dựa trên lý thuyết phổ của ma trận kề, sử dụng các trị riêng và vector riêng để giảm chiều và xác định cấu trúc cụm. Thuật toán này hiệu quả với dữ liệu phi tuyến, giúp nhận diện các cụm phức tạp mà KMeans khó phát hiện.

Bảng VII
ĐÁNH GIÁ CHẤT LƯỢNG PHÂN CỤM BẰNG SPECTRAL CLUSTERING

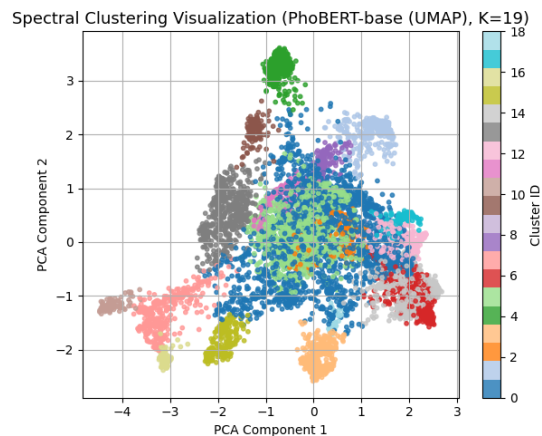
Embedding	K	Silhouette	Davies–Bouldin	Calinski–Harabasz
TF-IDF (UMAP)	15	0.3120	0.8360	2259.86
PhoBERT-base (UMAP)	19	0.2581	0.9197	1861.72
PhoBERT-large (UMAP)	8	0.3466	0.7688	2416.71

Phân tích kết quả:

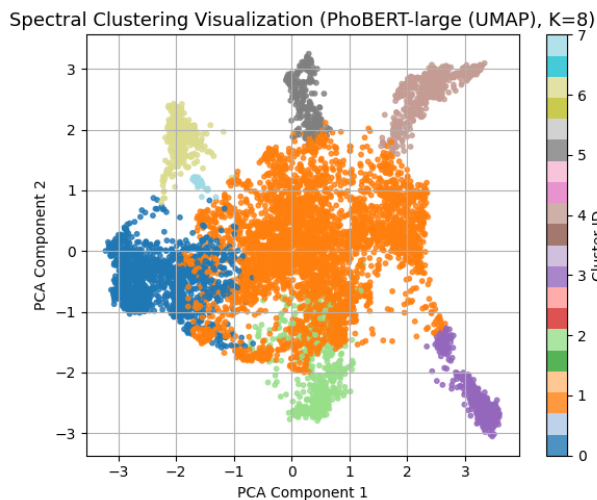
- **TF-IDF (UMAP)** đạt Silhouette Score 0.3120, cao hơn PhoBERT-base, thể hiện sự tách biệt cụm tương đối ổn định. Chỉ số Davies–Bouldin 0.8360 cho thấy mức độ chồng lấn thấp, tuy nhiên Calinski–Harabasz 2259.86 phản ánh cụm chưa thực sự đậm đặc.
- **PhoBERT-base (UMAP)** cho kết quả thấp nhất với Silhouette 0.2581 và Davies–Bouldin 0.9197, cho thấy cụm ít tách biệt và phân bố rời rạc hơn. Giá trị Calinski–Harabasz thấp nhất (1861.72) củng cố nhận định này.
- **PhoBERT-large (UMAP)** có Silhouette cao nhất (0.3466) và Davies–Bouldin thấp nhất (0.7688), chứng tỏ mô hình này phân tách cụm rõ hơn và đồng nhất hơn trong không gian embedding. Calinski–Harabasz đạt 2416.71, cao nhất trong ba mô hình.



Hình 19. Spectral Clustering với TF-IDF (UMAP), $K = 15$



Hình 20. Spectral Clustering với PhoBERT-base (UMAP), $K = 19$



Hình 21. Spectral Clustering với PhoBERT-large (UMAP), $K = 8$

Nhìn chung, Spectral Clustering cho thấy khả năng tách cụm khá rõ, đặc biệt khi kết hợp với embedding PhoBERT-large. Tuy không đạt giá trị Silhouette cao như HDBSCAN, nhưng phương pháp này tạo ra cấu trúc phân cụm ổn định và trực quan trong không gian giảm chiều UMAP.

VI. KẾT LUẬN

Sau quá trình thực nghiệm với năm thuật toán phân cụm khác nhau — **Hierarchical**, **KMeans**, **DBSCAN**, **HDBSCAN** và **Spectral Clustering** — trên ba dạng biểu diễn văn bản gồm **TF-IDF**, **PhoBERT-base** và **PhoBERT-large**, các kết quả đánh giá cho thấy sự khác biệt rõ rệt về khả năng nhận diện cấu trúc cụm của từng mô hình.

1. Bảng tổng hợp kết quả phân cụm

Bảng VIII
KẾT QUẢ PHÂN CỤM HIERARCHICAL

Embedding	Silhouette	Davies–Bouldin	Calinski–Harabasz
TF-IDF (UMAP)	0.0326	5.8083	50.69
PhoBERT-base (UMAP)	0.0436	3.8068	108.60
PhoBERT-large (UMAP)	0.0278	3.3361	215.67

Bảng IX
KẾT QUẢ PHÂN CỤM KMEANS

Embedding	K	Silhouette	Davies–Bouldin	Calinski–Harabasz
TF-IDF (UMAP)	15	0.4534	0.8917	4710.54
PhoBERT-base (UMAP)	19	0.4090	0.8814	3531.94
PhoBERT-large (UMAP)	8	0.4211	0.8683	3797.48

Bảng X
KẾT QUẢ PHÂN CỤM DBSCAN

Embedding	ϵ	min_samples	Silhouette	Davies–Bouldin
TF-IDF (UMAP)	0.7	70	0.3120	1.2622
PhoBERT-base (UMAP)	0.6	60	0.2573	1.1541
PhoBERT-large (UMAP)	0.7	70	0.2262	1.3783

Bảng XI
KẾT QUẢ PHÂN CỤM HDBSCAN

Embedding	Số cụm	Silhouette	Davies–Bouldin	Calinski–Harabasz	Noise (%)
TF-IDF (UMAP)	15	0.4727	0.7646	2721.27	29.06
PhoBERT-base (UMAP)	15	0.4487	0.7098	2501.67	30.87
PhoBERT-large (UMAP)	14	0.4867	0.6693	3092.03	33.85

Bảng XII
KẾT QUẢ PHÂN CỤM SPECTRAL CLUSTERING

Embedding	K	Silhouette	Davies–Bouldin	Calinski–Harabasz
TF-IDF (UMAP)	15	0.3120	0.8359	2259.85
PhoBERT-base (UMAP)	19	0.2581	0.9197	1861.72
PhoBERT-large (UMAP)	8	0.3466	0.7687	2416.70

2. Phân tích và đánh giá

Dựa trên các chỉ số đánh giá:

- **Silhouette Score:** thể hiện độ tách biệt và gắn kết giữa các cụm. Giá trị càng cao, cụm càng rõ ràng.
- **Davies–Bouldin Index:** càng thấp thì chất lượng phân cụm càng tốt.
- **Calinski–Harabasz Index:** càng lớn, cấu trúc phân cụm càng rõ ràng.

Kết quả cho thấy:

- **HDBSCAN** đạt hiệu suất cao nhất với *Silhouette* trung bình khoảng **0.47** và *Davies–Bouldin* thấp nhất (**0.66–0.76**), chứng tỏ khả năng phát hiện cụm tự nhiên và loại bỏ nhiễu hiệu quả.
- **KMeans** cũng thể hiện tốt, đặc biệt ở TF-IDF với *Silhouette* = 0.45, nhưng vẫn phụ thuộc vào việc chọn *K* tối ưu.
- **Hierarchical** và **Spectral Clustering** cho kết quả trung bình, phù hợp khi số cụm nhỏ và dữ liệu tuyến tính.
- **DBSCAN** thể hiện ổn định nhưng nhạy cảm với các tham số *eps* và *min_samples*.

3. So sánh giữa các biểu diễn Embedding

- **PhoBERT-large** cho kết quả tốt nhất trong hầu hết các thuật toán, đặc biệt với HDBSCAN (*Silhouette* = 0.4867, *Davies–Bouldin* = 0.6693), nhờ khả năng biểu diễn ngữ nghĩa sâu.
- **PhoBERT-base** xếp thứ hai, cho hiệu suất khá ổn định.
- **TF-IDF** dù đơn giản nhưng vẫn đạt kết quả cạnh tranh, đặc biệt ở KMeans và HDBSCAN.

4. Kết luận tổng quát

Tổng thể, mô hình **HDBSCAN kết hợp PhoBERT-large (UMAP)** được đánh giá là hiệu quả nhất cho bài toán phân cụm văn bản tiếng Việt nhờ khả năng:

- Tự động xác định số cụm mà không cần tham số K .
- Xử lý tốt dữ liệu phi tuyến, mật độ không đồng nhất.
- Loại bỏ nhiễu tự nhiên trong dữ liệu.

Ngược lại, **Hierarchical Clustering** là phương pháp kém hiệu quả nhất trong thực nghiệm, do *Silhouette* thấp và mức tách biệt cụm yếu. Điều này chứng tỏ các phương pháp hiện đại như HDBSCAN và KMeans khi kết hợp với embedding ngữ nghĩa (PhoBERT, UMAP) có khả năng mô hình hóa văn bản tiếng Việt tốt hơn đáng kể so với các kỹ thuật cổ điển.

TÀI LIỆU

- [1] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.
- [2] D. Q. Nguyen and A. T. Nguyen, "Phobert: Pre-trained language models for vietnamese," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 1037–1042.
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.
- [4] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.
- [5] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, 1996, pp. 226–231.
- [6] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," *Advances in Knowledge Discovery and Data Mining*, pp. 160–172, 2013.
- [7] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*, vol. 14, 2002, pp. 849–856.
- [8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
- [9] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
- [10] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," in *arXiv preprint arXiv:1907.11692*, 2019.
- [11] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.
- [12] T. M. Kodinariya and P. R. Makwana, "Review on determining number of cluster in k-means clustering," *International Journal of Advance Research in Computer Science and Management Studies*, vol. 1, no. 6, pp. 90–95, 2013.
- [13] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.
- [14] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.
- [15] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, 1967, pp. 281–297.
- [16] D. Müllner, "Modern hierarchical, agglomerative clustering algorithms," in *arXiv preprint arXiv:1109.2378*, 2011.
- [17] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.
- [18] L. McInnes, J. Healy, and S. Astels, "hdbscan: Hierarchical density based clustering," in *Journal of Open Source Software*, vol. 2, no. 11, 2017, p. 205.
- [19] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.
- [20] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 2, pp. 224–227, 1979.
- [21] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics*, vol. 3, no. 1, pp. 1–27, 1974.