DOI: 10.31276/VJST.2026.3849 *Khoa học tự nhiên* | *Khoa học máy tính và thông tin; Khoa học tính toán*

Phân tích so sánh các mô hình tìm kiếm dựa trên từ khóa và ngữ nghĩa cho dữ liệu báo điện tử tiếng Việt

Vũ Thảo Nguyên[1*], Mai Hoàng Bách[2], Đỗ Hà Duy Phong[3]

Trường Khoa học Máy tính và Dữ liệu, Đại học Công nghệ Nanyang, 50 Nanyang Avenue, Singapore

Trường THPT Chuyên Hà Nội - Amsterdam, 1 Hoàng Minh Giám, phường Yên Hoà, Hà Nội, Việt Nam

Trường THPT Yên Hòa, 251 Nguyễn Khang, phường Yên Hoà, Hà Nội, Việt Nam

Ngày nhận bài 18/2/2026; ngày chuyển phản biện 20/2/2026; ngày nhận phản biện 3/3/2026; ngày chấp nhận đăng 16/3/2026

Tóm tắt:

Công việc thường xuyên khi duyệt trang web là tìm kiếm nội dung các trang theo từ khóa, chẳng hạn tìm các trang trong máy tìm kiếm của Google. Việc tìm kiếm văn bản là quá trình tìm những tư liệu liên quan nhất tới câu hỏi của người dùng. Bài báo đề cập quy trình xử lý văn bản tiếng Việt từ trang web. Các văn bản được xử lý theo ngôn ngữ học, chuẩn bị cho việc tìm kiếm theo từ khóa và ngữ nghĩa. Các mô hình tìm kiếm logic (bool), mô hình TF-IDF, Word2Vec và BERT được nêu ra để so sánh. Sau các phân tích, bài báo đề cập mô hình tìm kiếm BM25 và tìm kiếm lai. Văn bản sử dụng là trang báo điện tử VnExpress vào tháng 1/2026. Sau các thực nghiệm trên môi trường Google Colab với ngôn ngữ Python, bài báo khuyến cáo sử dụng mô hình tìm kiếm lai, kết hợp BM25 và BERT/PhoBERT để tăng hiệu quả tìm kiếm.

Từ khóa: **BERT, BM25, TF-IDF, tìm kiếm ngữ nghĩa, tìm kiếm văn bản, Word2Vec.**

Chỉ số phân loại: **1.2, 1.8**

Comparative analysis of keyword-based and semantic search models for Vietnamese online news data

Thao Nguyen Vu[1*], Hoang Bach Mai[2], Ha Duy Phong Do[3]

College of Computing and Data Science, Nanyang Technological University, 50 Nanyang Avenue, Singapore

Hanoi - Amsterdam High School for the Gifted, 1 Hoang Minh Giam Street, Yen Hoa Ward, Hanoi, Vietnam

Yen Hoa High School, 251 Nguyen Khang Street, Yen Hoa Ward, Hanoi, Vietnam

Received 18 February 2026; revised 3 March 2026; accepted 16 March 2026

*Tác giả liên hệ: Email: vuthaonguyen26052005@gmail.com

Abstract:

A common activity in web browsing involves retrieving webpage content based on specified keywords, such as querying pages through widely used search engines like Google. In this context, text retrieval is understood as the systematic process of identifying, filtering, and selecting documents that are most relevant to a user's information need or query. This article focuses on the processing of Vietnamese textual data collected from online sources. The texts undergo linguistic preprocessing steps to ensure they are suitable for both keyword-based and semantic search tasks. Several established retrieval models are examined and compared, including the Boolean (BOOL) model, TF-IDF, Word2Vec, and BERT. Following this comparative analysis, the study further explores the BM25 ranking model as well as hybrid search approaches that integrate multiple techniques. The dataset utilized in this research consists of articles from the VnExpress online newspaper, specifically collected from January 2026. Experimental evaluations are conducted in the Google Colab environment using Python programming tools. Based on the obtained results, the study recommends adopting a hybrid search model that combines BM25 with advanced language models such as BERT or PhoBERT in order to significantly enhance overall search effectiveness and retrieval performance.

Keywords: **BERT, BM25, TF-IDF, semantic search, text search, Word2Vec.**

Classification numbers: **1.2, 1.8**

1. Đặt vấn đề

Công việc thường xuyên khi duyệt trang web là tìm kiếm nội dung các trang theo từ khóa, chẳng hạn tìm các trang trong máy tìm kiếm của Google. Việc tìm kiếm văn bản là quá trình tìm những tư liệu liên quan nhất tới câu hỏi của người dùng. Trong trang web, người ta mong muốn nhận được kết quả chính xác và nhanh chóng từ các siêu văn bản và siêu liên kết. Nhìn lại các hệ thống tìm kiếm, có mô hình sử dụng thuật toán so sánh logic bool; sau đó có các mô hình học máy.

Văn bản được xử lý theo ngôn ngữ học về tách từ, loại trừ từ dừng, tạo từ điển… là các đầu vào cho mô hình tìm kiếm theo từ khóa hay ngữ nghĩa [1, 2]. Phần 2 sau đây sẽ trình bày về bộ dữ liệu tự động thu thập được. Bài báo sẽ tải tự động nhiều văn bản từ trang tin chính thống *vnexpress.net*, chuyên mục Thời sự, vào ngày 23/1/2026. Vấn đề ở đây là bộ dữ liệu sẽ được phân tích thống kê, thứ nhất để hiểu rõ về bản chất của dữ liệu; thứ hai là chuẩn bị để có thể áp dụng các mô hình toán học và trí tuệ nhân tạo để xử lý dữ liệu.

Các mô hình tìm kiếm chính xác theo từ khóa như mô hình bool và sử dụng trọng số như term frequency-inverse document frequency (TF-IDF) [1, 3], mô hình tìm kiếm

theo ngữ nghĩa Word2Vec [4, 5] là các mô hình tìm kiếm thông dụng, như bài học cho học sinh.

Việc tìm kiếm trên văn bản với công cụ trí tuệ nhân tạo như mô hình bidirectional encoder representations from transformers (BERT) hay PhoBERT [6, 7] tạo khả năng tìm kiếm mạnh hơn đối với các câu hỏi có ngữ nghĩa phức tạp. Những mô hình tìm kiếm này cũng là đề tài nghiên cứu cho sinh viên đại học.

Phần 3 đề cập bốn mô hình tìm kiếm dữ liệu. Dựa trên các văn bản tải được, các mô hình tìm kiếm văn bản sau được dùng.

- Tìm kiếm Bool, với các toán tử logic để xác định tài liệu liên quan.

- Tìm kiếm với TF-IDF, dựa trên tần suất từ và phân biệt tầm quan trọng của các thuật ngữ.

- Tìm kiếm Word2Vec, sử dụng từ ngữ để bắt mối quan hệ ngữ nghĩa.

- Tìm kiếm BERT, và các mô hình *Transformer* khác, để học ngữ cảnh hai chiều của từ trong câu, nâng cao khả năng truy vấn ngữ nghĩa phức tạp.

Các mô hình này được sử dụng phổ biến trong các bài toán xử lý ngôn ngữ tự nhiên, đặc biệt là trong các hệ thống truy hồi thông tin văn bản, để tìm kiếm các đoạn văn bản theo các từ khóa.

Bài báo tìm hiểu mô hình best matching 25 (BM25) [8] và tìm kiếm lai (hybrid search) [9]. Việc tìm hiểu các mô hình này nhằm so sánh với bốn mô hình tìm kiếm nói trên. Sau đó nhóm nghiên cứu khuyến nghị sử dụng mô hình tìm kiếm lai đối với các văn bản tiếng Việt trên các trang web.

2. Tự động thu thập dữ liệu và xử lý sơ bộ dữ liệu

2.1. Thu thập dữ liệu

Các đoạn văn bản được tải nhờ chương trình Python. Chương trình sử dụng hàm mẫu install requests beautifulsoup4 pandas lxml, rồi thực hiện đoạn chương trình sau để lấy dữ liệu. Trước hết cần thực hiện câu lệnh import requests; from bs4 import BeautifulSoup.

Đoạn chương trình sau tự động tiếp cận từ trang web, bằng hàm extract_article(url), rồi duyệt trang web https://vnexpress.net/thoi-su.

Việc dò tìm (*crawl*) cho phép (i) duyệt từng trang; (ii) lấy nội dung, các liên kết, siêu dữ liệu, mà không cần thực hiện thủ công. Nếu không sử dụng công cụ này, người ta chỉ tìm được những địa chỉ tài nguyên (*URL*) đã biết trước.

```
#Crawl theo NGÀY + CHUYÊN MỤC
TARGET_DATE = datetime(2026, 1, 23).date()
CATEGORY_URL = "https://vnexpress.net/thoi-su"
links = get_links_by_category(CATEGORY_URL, max_pages=10)
data = []
for link in links:
 try:
 title, publish_date, content = extract_article(link)
if publish_date == TARGET_DATE:
data.append({
 "date": publish_date,
 "category": "thoi-su",
 "title": title,
 "url": link,
 "content": content
})
time.sleep(1)
 except Exception as e:
 print("Lỗi:", link, e)
```

Kết quả thu được như sau:

```
Mounted at /content/drive
```

	date	category	title	url	content
0	2026-01-23	thoi-su	Người hâm mộ vỡ òa khi Việt Nam thắng Hàn Quốc...	https://vnexpress.net/nguoi-ham-mo-vo-oa-khi-v...	Hàng nghìn người hâm mộ trên phố Nguyễn Huệ hồ...
1	2026-01-23	thoi-su	10.000 quả pháo hoa thắp sáng bầu trời Hà Nội	https://vnexpress.net/10-000-qua-phao-hoa-thap...	Chào mừng thành công Đại hội đại biểu toàn quố...
2	2026-01-23	thoi-su	6 ôtô tông liên hoàn trên cao tốc TP HCM - Tru...	https://vnexpress.net/6-oto-tong-lien-hoan-tre...	Hơn 18h, ôtô 7 chỗ đang chạy trên cao tốc TP H...
3	2026-01-23	thoi-su	Chương trình nghệ thuật và pháo hoa chào mừng ...	https://vnexpress.net/chuong-trinh-nghe-thuat-...	Theo Sở Văn hóa và Thể thao Hà Nội, chương trì...
4	2026-01-23	thoi-su	Tổng Bí thư: 'Tháo gỡ mọi điểm nghẽn, khơi thô...	https://vnexpress.net/tong-bi-thu-chu-tri-hop-...	Ngay sau lễ bế mạc, Tổng Bí thư Tô Lâm; Thường...

Hình 1. Các văn bản từ trang web được tự động tải về.

Nội dung ở dạng các đoạn văn bản được thể hiện qua một vài câu ngắn như trong hình 1. Để xem nội dung đoạn văn bản, như hình 2, người ta nháy vào tùy chọn phù hợp.

index	date	category	title	url	content
0	2026-01-23	thoi-su	Người hâm mộ vỡ òa khi Việt Nam thắng Hàn Quốc ở loạt luân lưu	https://vnexpress.net/nguoi-ham-mo-vo-oa-khi-viet-nam-thang-han-quoc-o-loat-luan-luu-5009323.html	Hàng nghìn người hâm mộ trên phố Nguyễn Huệ hồi hộp rồi vỡ òa, nhảy mừng khi Thanh Nhàn hạ Jaeyun bằng cú sút vào góc trái, giúp Việt Nam đánh bại Hàn Quốc 7–6 ở loạt sút luân lưu, sau khi hai đội hòa 2–2 trong 120 phút ở trận tranh hạng ba U23 châu Á. Hàng nghìn người hâm mộ trên phố Nguyễn Huệ hồi hộp rồi vỡ òa, nhảy mừng khi Thanh Nhàn hạ Jaeyun bằng cú sút vào góc trái, giúp Việt Nam đánh bại Hàn Quốc 7–6 ở loạt sút luân lưu, sau khi hai đội hòa 2–2 trong 120 phút ở trận tranh hạng ba U23 châu Á. Cảm xúc của đông đảo cổ động viên lên đến cao trào sau khi trải qua 120 phút đầy căng thẳng, đặc biệt kể từ thời điểm Việt Nam bị Hàn Quốc gỡ hòa 2–2 ở những giây cuối trận và phải thi đấu thêm 30 phút hiệp phụ trong thế bị đối phương ép sân dữ dội do chơi thiếu người. Cảm xúc của đông đảo cổ động viên lên đến cao trào sau khi trải qua 120 phút đầy căng thẳng, đặc biệt kể từ thời điểm Việt Nam bị Hàn Quốc gỡ hòa 2–2 ở những giây cuối trận và phải thi đấu thêm 30 phút hiệp phụ trong thế bị đối phương ép sân dữ dội do chơi thiếu người. Trong bối cảnh khung thành của thủ môn Văn Bình liên tục bị vây hãm, các cổ động viên chỉ biết chắp tay cầu nguyện rồi thở phào sau mỗi pha đội nhà giải vây thành công. Trong bối cảnh khung thành của thủ môn Văn Bình liên tục bị vây hãm, các cổ động viên chỉ biết chắp tay cầu nguyện rồi thở phào sau mỗi pha đội nhà giải vây thành công. Trong bối cảnh khung thành của thủ môn Văn Bình liên tục bị vây hãm, các cổ động viên chỉ biết chắp tay cầu nguyện rồi thở phào sau mỗi pha đội nhà giải vây thành công. Trong 90 phút chính thức, các cổ động viên trên phố đi bộ Nguyễn Huệ hai lần rơi vào cảm giác tiếc nuối khi thấy trò HLV Kim Sang-sik bị Hàn Quốc gỡ hòa. Đáng tiếc nhất là khoảnh khắc đội tuyển Việt Nam đánh rơi chiến thắng ở những giây cuối trận, khi Minha cài đè tốt rồi dứt điểm đánh bại thủ môn Cao Văn Bình, gỡ hòa 2–2. Phút 68,

Hình 2. Văn bản đầy đủ.

2.2. Tách từ, loại bỏ từ dừng trong văn bản

Đối với bài toán ngôn ngữ học, các đoạn văn bản sẽ được tách từ. Tuy nhiên quá trình làm tinh dữ liệu ở đây thực hiện các nhiệm vụ: (i) làm sạch; (ii) loại bỏ các từ dừng; (iii) tách từ; và (iv) tạo từ điển với n-gram. Bài báo chọn 2-gram đối với các thuật ngữ tìm kiếm. Tách từ tiếng Việt (*tokenization*) có mục đích giải quyết vấn đề đa âm tiết của tiếng Việt, không thể tách từ bằng khoảng trắng. Hàm *underthesea.word_tokenize* tách đúng từ ghép. Việc tách từ là nền tảng cho xử lý ngôn ngữ Việt. Việc loại bỏ từ dừng (*stopwords* tiếng Việt) nhằm loại các từ xuất hiện rất nhiều, ít mang nội dung. Chỉ giữ lại các từ mang thông tin. Việc này là quan trọng cho mô hình TF-IDF, tìm kiếm văn bản… Bài này sử dụng tập các từ dừng là "và", "là", "của", "có", "cho", "được", "trong", "một", "các", "với", "đã", "này", "đó", "những", "khi", "từ", "theo", "đến", "về", "như", "ra", "thì", "sẽ", "bị" (hình 3).

	tokens_filtered	bigrams
0	[hàng, nghìn, người, hâm_mộ, trên, phố, nguyễn...	[hàng nghìn, nghìn người, người hâm_mộ, hâm_mộ...
1	[chào_mừng, thành_công, đại_hội, đại_biểu, toà...	[chào_mừng thành_công, thành_công đại_hội, đại...
2	[hơn, h, ôtô, chỗ, đang, chạy, trên, cao_tốc, ...	[hơn h, h ôtô, ôtô chỗ, chỗ đang, đang chạy, c...
3	[sở, văn_hóa, thể_thao, hà_nội, chương_trình, ...	[sở văn_hóa, văn_hóa thể_thao, thể_thao hà_nội...
4	[ngay, sau, lễ, bế_mạc, tổng_bí_thư, tô_lâm, t...	[ngay sau, sau lễ, lễ bế_mạc, bế_mạc tổng_bí_t...

Hình 3. Văn bản được làm sạch, loại bỏ từ dừng và tách từ.

Việc tạo 2-gram (*bigram*) là chỉ ra các cặp hai từ liên tiếp trong văn bản. Nó giúp duy trì ngữ cảnh, hiệu quả hơn 1-gram (*unigram*) trong: tìm kiếm, phân loại văn bản, phân tích chủ đề.

Trong bài toán xử lý văn bản, từ điển trong Python có vai trò quan trọng đối với toàn bộ quá trình xử lý. Từ điển là ánh xạ từ một thuật ngữ (*term*) sang thông tin mô tả. Ở đây từ điển term là tập tất cả các bigram không trùng lặp. Mỗi term là một khái niệm gồm hai từ. Từ điển term dùng cho: vec tơ hóa văn bản, lập chỉ mục tìm kiếm, TF-IDF bigram. Từ các văn bản, chương trình tạo nên 4906 term (hình 4).

```
# Tạo từ điển các term
from itertools import chain

TERM_DICTIONARY = sorted(
    set(chain.from_iterable(df["bigrams"]))
)

len(TERM_DICTIONARY)
```

```
4906
```

```
TERM_DICTIONARY[:40]
```

```
['ai lui_tới',
 'an_chánh thấp',
 'an_ninh an_toàn',
 'an_ninh khu_vực',
 'an_ninh nhân_dân',
 'an_ninh năm',
 'an_ninh quốc_phòng',
 'an_ninh thượng_cách',
 'an_ninh thượng_nay',
 'an_ninh trước',
 'an_ninh đối_ngoại',
 'an_ninh_mạng chiến_tranh',
 'an_toàn ổn_định',
 'an_độ hà_tĩnh_độ',
 'anh lâm',
 'anh nguyễn_thanh_hùng',
 'anh nói',
 'anh vương_thanh_tuấn',
 'anh_dũng bùi_minh_quang',
 'anh_dũng nguyễn_minh_quang',
```

Hình 4. Tạo từ điển phục vụ cho tìm kiếm.

Minh họa ánh xạ term tới nội dung: từ điển term chứa tất cả các bigram duy nhất được tìm thấy trong tập dữ liệu. Người ta có thể dùng một term từ từ điển này để tìm kiếm các văn bản gốc chứa nó. Đoạn chương trình sau dùng hàm *search_term* để lấy tất cả các hàng trong khung dữ liệu mà cột bigrams của chúng chứa term đã cho (hình 5).

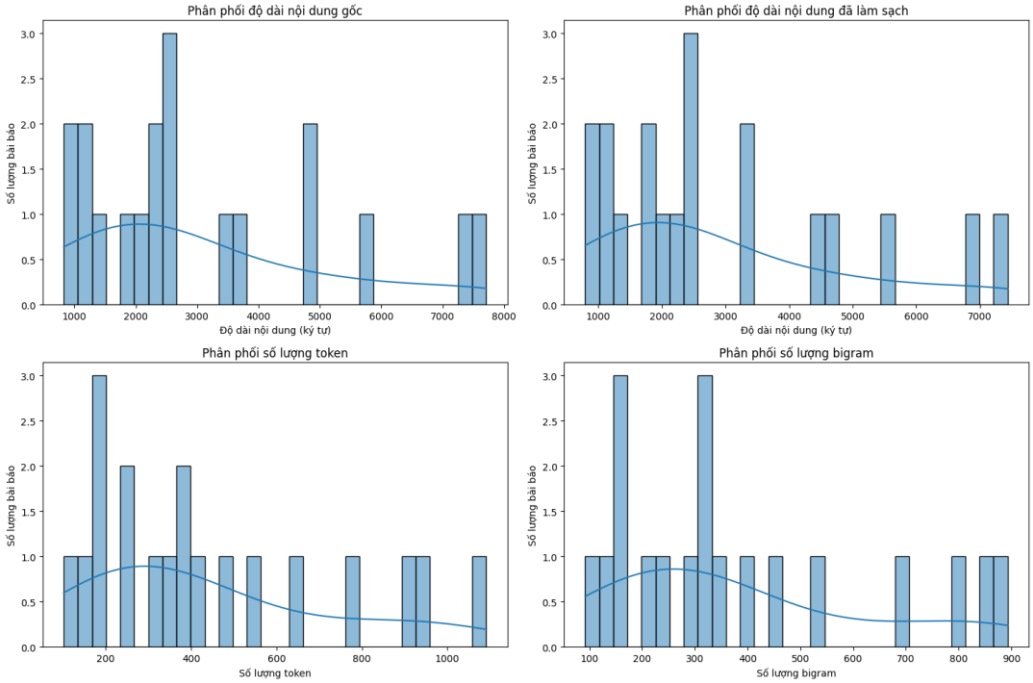
```
selected_term = "ban thường_trực"

# Sử dụng hàm search_term đã định nghĩa trước đó

result_mapping = search_term(selected_term, df)

print(f"Nội dung chứa term '{selected_term}':")

display(result_mapping)
```

```
Nội dung chứa term 'ban thường_trực':
```

	content
8	Chiều 23/1, tại phiên bế mạcĐại hội Đảng XIV, ...
10	Chiều 23/1, sau 5 ngày làm việc, Đại hội Đảng ...
14	Tối 22/1, danh sách Ban Chấp hành Trung ương Đ...
16	200 Ủy viên Trung ương khóa XIV sẽ tiến hành c...

Hình 5. Nội dung chứa một term.

Các đặc trưng văn bản được thể hiện nhờ đoạn chương trình với nhiệm vụ (i) thiết lập kích thước hình vẽ; (ii) vẽ biểu đồ phân phối độ dài nội dung gốc; (iii) vẽ biểu đồ phân phối độ dài nội dung đã làm sạch; (iv) vẽ biểu đồ phân phối số lượng token.



Hình 6. Các đặc trưng của văn bản.

Đến cuối phần 2, tư liệu gồm các đoạn văn bản. Nó được xem như bộ dữ liệu. Bộ dữ liệu này đã được làm sạch, tiện cho việc áp các mô hình toán học hay mô hình học máy để xử lý văn bản (hình 6). Phần 3 sau đây thực hiện việc tìm kiếm văn bản theo từ khóa, tức theo term.

2.3. Đặc trưng ngôn ngữ của tiếng Việt

Tiếng Việt là một ngôn ngữ khó với các mô hình tìm kiếm truyền thống như TF-IDF hay BM25 do cấu trúc ngữ pháp và đặc điểm chữ viết đặc thù. Do vậy đối với các mô hình này cho tiếng Việt, người ta cần đặc biệt lưu ý các thách thức sau:

- Bài toán tách từ: Đây là rào cản lớn nhất. Tiếng Anh phân tách từ bằng khoảng trắng, còn tiếng Việt dùng khoảng trắng để phân tách tiếng (syllable). Nếu không có bước tách từ tốt, chỉ số TF-IDF sẽ bị tính sai vì các từ ghép bị xẻ nhỏ thành các đơn từ vô nghĩa.

+ Từ đồng âm khác nghĩa và đa nghĩa: Tiếng Việt cực kỳ phong phú về ngữ nghĩa dựa trên ngữ cảnh.

+ Hiện tượng từ mượn và viết tắt: người ta sử dụng đan xen rất nhiều từ mượn từ tiếng nước ngoài và từ viết tắt trong văn phong hàng ngày.

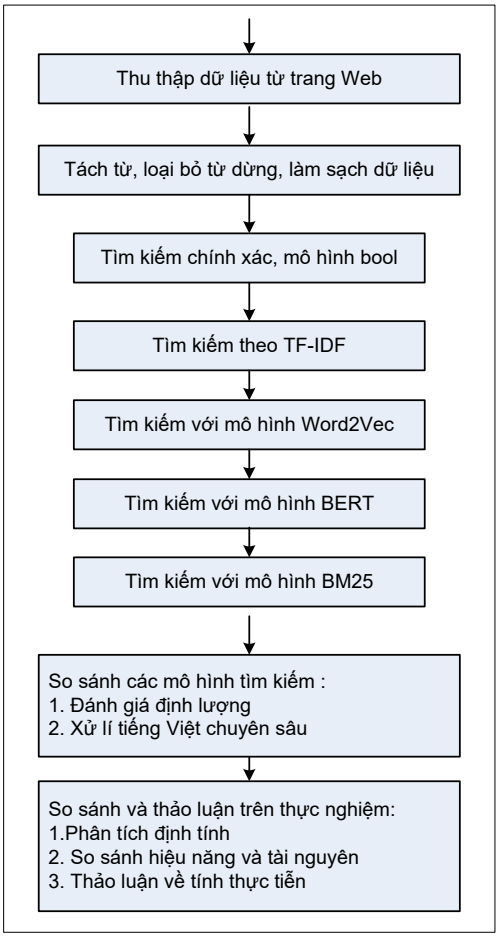
+ Dấu câu và biến thể telex: Dấu thanh trong tiếng Việt thay đổi hoàn toàn nghĩa của từ.

3. Các mô hình tìm kiếm

Phần này trình bày một số mô hình tìm kiếm trên các tệp văn bản thu được từ web. Tìm kiếm văn bản theo term là phương pháp tìm kiếm trong đó đơn vị trung tâm là term. Hệ thống sẽ xác định văn bản nào có chứa term được truy vấn, từ đó trả về kết quả phù hợp. Trong xử lý văn bản, term là (i) một từ đơn; hay (ii) một từ ghép. Cũng có thể xem tìm kiếm văn bản theo term là quá trình tìm kiếm các tài liệu dựa trên sự xuất hiện, tần suất hoặc trọng số của các term trong tài liệu.

Chương trình Python chạy trong Colab minh họa quá trình tìm kiếm. Google Colab (*Colaboratory*) là một môi trường lập trình Python chạy trên nền tảng đám mây do Google cung cấp, cho phép người dùng viết, chạy và chia sẻ chương trình, còn gọi là *Jupyter Notebook*, trực tiếp trên trình duyệt. Bản chất của Google Colab về mặt kỹ thuật là (i) dùng Jupyter Notebook mở rộng; (ii) chạy trên máy chủ của Google; (iii) truy cập qua trình duyệt. Người dùng chỉ cần tài khoản Google. Chương trình Python ở đây gồm các đoạn chương trình nhỏ, như các đoạn chương trình đã liệt kê trong phần 2.

Các phần sau sẽ thực hiện các hoạt động như trình bày trong hình 7.



Hình 7. Sơ đồ các hoạt động phân tích so sánh các mô hình tìm kiếm dựa trên từ khóa và ngữ nghĩa.

3.1. Tìm kiếm chính xác

Tìm kiếm chính xác, hay tìm kiếm bool [1, 3] là cách truyền thống; nó sử dụng các toán tử logic như AND, OR, NOT trong biểu thức câu hỏi. Thuật toán xác định các tài liệu thỏa mãn điều kiện hỏi, chính xác về mặt cú pháp.

Cách này không đánh giá tầm quan trọng của từ, không xử lý ngữ nghĩa và yêu cầu cú pháp chính xác. Đánh giá về tiếp cận này (i) ưu điểm: tốc độ tìm kiếm nhanh, dễ cài đặt; (ii) nhược điểm: bỏ qua ngữ nghĩa, khó mở rộng tìm kiếm linh hoạt.

Thí dụ sau tìm thuật ngữ 2-gram, tức bigram, trong các đoạn văn bản. Nó (i) kiểm tra bigram có xuất hiện trong văn bản hay không; (ii) trả về các đoạn chứa thuật ngữ đó; (iii) đây là tìm kiếm bool dựa trên n-gram.

Đoạn chương trình sau tìm kiếm theo 2-gram. Kết quả thu được là nội dung bài báo 16, "200 ủy viên trung ương…".

```
# Tìm kiếm theo term (2-gram)
def search_term(term, df):
 result = df[df["bigrams"].apply(lambda x: term in x)]
return result[["content"]]
search_term("ban bộ_ngành", df).head()
```

Việc tìm kiếm theo nhiều từ khóa là tương tự, như trong đoạn chương trình sau tìm văn bản có hai từ khóa (hình 8).

```
# Tìm nhiều TERM (OR search)
def search_terms(terms, df):
return df[df["bigrams"].apply(
lambda x: any(t in x for t in terms)
 )][["content"]]
search_terms(["ban thường_trực", "an_toàn ổn_định"], df).head()
```

	content
8	Chiều 23/1, tại phiên bế mạcĐại hội Đảng XIV, ...
10	Chiều 23/1, sau 5 ngày làm việc, Đại hội Đảng ...
14	Tối 22/1, danh sách Ban Chấp hành Trung ương Đ...
16	200 Ủy viên Trung ương khóa XIV sẽ tiến hành c...

Hình 8. Kết quả tìm kiếm theo nhiều từ khóa.

Mô hình tìm kiếm bool cho kết quả là các văn bản có đúng các từ khóa. Yêu cầu này là khá chặt chẽ. Để thu được các kết quả tìm kiếm rộng hơn, người ta có thể sử dụng

các mô hình khác. Các mô hình này được sử dụng làm cơ sở so sánh với mô hình Boolean truyền thống. Chúng là ba mô hình được trình bày sau đây.

Ý tưởng so sánh định lượng thể hiện trong ba mô hình sau đây là (i) với một câu hỏi tìm kiếm, người ta cần biểu diễn câu hỏi thành một vec tơ; (ii) biểu diễn mỗi văn bản thành một vec tơ; (iii) tính độ tương tự cosine; (iv) khi được nhiều độ tương tự cosine, người ta chọn ra một số văn bản gần nhất với câu hỏi theo độ đo cosine. Tiêu chí định lượng ở đây là chọn ra các văn bản nào gần với câu hỏi về vec tơ.

Độ tương tự cosine là thước đo dùng để đánh giá mức độ giống nhau giữa hai vec tơ dựa trên góc giữa chúng, không phụ thuộc vào độ dài vec tơ. Độ đo giữa hai vec tơ a và b được tính bằng (tích vô hướng của a và b)/ ((độ dài chuẩn của vec tơ a) * (độ dài chuẩn của vec tơ b)).

3.2. Tìm kiếm với mô hình TF-IDF

Mô hình TF-IDF [1] là một trong những mô hình tìm kiếm dựa trên đặc trưng văn bản. TF-IDF tạo ra vec tơ đặc trưng số cho văn bản, trong đó: (i) mỗi chiều là một term; (ii) giá trị mỗi chiều bằng mức độ quan trọng của term. Như vậy văn bản được chuyển thành vec tơ TF-IDF.

Người ta dùng TF đo tần suất từ trong tài liệu, IDF đo độ hiếm của từ trong toàn tập tư liệu. Với thuật ngữ t và tư liệu d, TF(t, d) = (số lần xuất hiện t trong d)/(tổng số các từ trong văn bản d); IDF(t) = log(tổng số các văn bản trong tập tư liệu)/ (số văn bản trong tư liệu có thuật ngữ t xuất hiện); TF-IDF(t, d) = TF(t, d) x IDF(t).

Mô hình cho vec tơ đặc trưng dùng để đo độ tương đồng giữa (i) câu hỏi; và (ii) tư liệu. Mô hình này xử lý tốt trọng số từ và làm việc với dữ liệu thưa.

Bài báo sử dụng Python trong Colab tìm kiếm bằng TF-IDF, với từ khóa "nhân_dân", gồm (i) ghép token thành chuỗi; (ii) viết hàm tìm kiếm. Kết quả thu được như trong hình 9.

...

	content	score
4	Ngay sau lễ bế mạc, Tổng Bí thư Tô Lâm; Thường...	0.103344
10	Chiều 23/1, sau 5 ngày làm việc, Đại hội Đảng ...	0.095474
11	Trước đó một ngày, Tổng Bí thư Tô Lâm cùng 9 Ủ...	0.083761
16	200 Ủy viên Trung ương khóa XIV sẽ tiến hành c...	0.056583
8	Chiều 23/1, tại phiên bế mạcĐại hội Đảng XIV, ...	0.033351

Hình 9. Kết quả tìm kiếm "nhân dân" với mô hình TF-IDF.

3.3. Tìm kiếm với mô hình Word2Vec

Mô hình Word2Vec [4, 5] sử dụng học máy, nhúng từ, không giám sát, ánh xạ từ vào không gian vec tơ liên tục có chiều thấp hơn. Word2Vec là mô hình biểu diễn từ dựa

trên ngữ nghĩa, trong đó mỗi từ (*word*) được ánh xạ thành một vec tơ trong không gian liên tục, phản ánh ý nghĩa và mối quan hệ ngữ cảnh giữa các từ. Trong tìm kiếm văn bản, Word2Vec giúp tìm theo nghĩa, không chỉ theo từ khóa.

Bản chất của mô hình Word2Vec là (i) học vec tơ từ ngữ cảnh xuất hiện; (ii) các từ có ngữ cảnh giống nhau thì vec tơ tương ứng sẽ gần nhau; (iii) mỗi từ ứng với một vec tơ. Mô hình này khác với mô hình TF-IDF ở chỗ (i) TF-IDF dựa trên tần suất; (ii) Word2Vec dựa trên ngữ nghĩa.

Mô hình Word2Vec có hai kiến trúc chính (i) CBOW (*continuous bag-of-words*) cho phép dự đoán từ trung tâm từ ngữ cảnh; nhanh, phù hợp dữ liệu lớn; (ii) *Skip-gram*. Mỗi từ được biểu diễn bằng một vec tơ số. Các vec tơ này giữ thông tin ngữ nghĩa. Người cho phép dự đoán ngữ cảnh với từ trung tâm; tốt cho từ hiếm, nên hay được dùng trong tìm kiếm.

Người ta sử dụng độ đo cosine để so sánh các vec tơ. Mô hình Word2Vec mở rộng khả năng truy vấn qua việc (i) so sánh ngữ nghĩa; (ii) không chỉ xem xét trùng từ. Trước hết cần huấn luyện mô hình.

Đoạn chương trình sau tìm kiếm và hỏi với từ khóa "nhân_dân". Bài báo (i) sử dụng gensim.models trong Python để gọi Word2Vec; (ii) mô tả hàm search_word2vec(query, df, model, doc_vectors, top_k=5); (iii) hỏi bằng search_word2vec("nhân_dân", df, w2v_model, doc_vectors_w2v). Kết quả thu được như trong hình 10.

...

	content	score
11	Trước đó một ngày, Tổng Bí thư Tô Lâm cùng 9 Ủ...	0.874639
16	200 Ủy viên Trung ương khóa XIV sẽ tiến hành c...	0.873273
4	Ngay sau lễ bế mạc, Tổng Bí thư Tô Lâm; Thường...	0.872994
10	Chiều 23/1, sau 5 ngày làm việc, Đại hội Đảng ...	0.871743
6	Tại phiên bế mạcĐại hội Đảng XIV, Chủ tịch Quố...	0.871255

Hình 10. Kết quả tìm kiếm "nhân dân" với mô hình Word2Vec.

3.4. Tìm kiếm với mô hình BERT

BERT [6, 7] là một mô hình ngôn ngữ dựa trên kiến trúc Transformer, học biểu diễn hai chiều của từ trong câu. Không giống mô hình Word2Vec chỉ học từ đơn lẻ trong ngữ cảnh cục bộ, mô hình BERT học ngữ cảnh phong phú: (i) Mỗi từ được mã hóa phụ thuộc vào toàn bộ ngữ cảnh trái-phải; (ii) Có thể tinh chỉnh để thực hiện các nhiệm vụ như phân loại, xếp hạng…

Về đặc trưng của mô hình này, người ta thấy (i) BERT là mô hình ngôn ngữ sâu dựa trên Transformer, cho phép hiểu ngữ cảnh hai chiều của văn bản. Trong tìm kiếm văn bản, BERT giúp hệ thống hiểu ý định truy vấn và ngữ nghĩa sâu của tài liệu, vượt xa các mô

hình dựa trên term; (ii) mỗi từ khóa, hay term, được biểu diễn bằng vec tơ phụ thuộc ngữ cảnh; (iii) cùng một từ có thể là vec tơ khác nhau trong ngữ cảnh khác nhau; (iv) câu hỏi và tư liệu được so khớp theo nghĩa, không chỉ theo từ khóa. Vậy sử dụng BERT cho phép chuyển từ việc tìm kiếm theo term sang tìm kiếm theo ngữ nghĩa.

Đặc trưng cơ bản của BERT được thống kê như sau:

- Ngữ cảnh hai chiều: hiểu từ dựa trên cả bên trái và bên phải; giải quyết vấn đề đa nghĩa của một term. Do vậy mô hình BERT hơn hẳn mô hình Word2Vec;

- Biểu diễn theo ngữ cảnh: vector biểu diễn của term phụ thuộc vào ngữ cảnh câu. Điều này quan trọng đối với việc hỏi tự nhiên, câu hỏi dài;

- Hiểu ý định câu hỏi: hiểu cấu trúc ngữ pháp; hiểu mối quan hệ giữa các từ

- Khớp nghĩa ở mức câu / đoạn: so sánh câu hỏi với tư liệu ở mức câu văn, mức đoạn văn; sử dụng độ đo cosine. Như vậy việc tìm kiếm không nhất thiết trùng từ khóa;

- Không cần lấy đặc trưng các văn bản một cách thủ công: không yêu cầu TF, IDF, từ dừng…; BERT tự học đặc trưng. Như vậy sẽ bỏ quan bước tiền xử lý văn bản.

Đoạn chương trình sau mô tả hàm tìm kiếm theo mô hình BERT trong Python và kết quả thu được như hình 11, với câu hỏi là search_bert("nhân_dân", df, bert_embeddings, bert_model).

	content	score
12	Khoảng 9h20, xe ben do anh Vương Thanh Tuấn (2...	0.397721
7	Sáng 23/1, hội nghị lần thứ nhất Ban Chấp hành...	0.380154
11	Trước đó một ngày, Tổng Bí thư Tô Lâm cùng 9 Ủ...	0.366257
4	Ngay sau lễ bế mạc, Tổng Bí thư Tô Lâm; Thường...	0.358633
1	Chào mừng thành công Đại hội đại biểu toàn quố...	0.312465

Hình 11. Kết quả tìm kiếm "nhân dân" với mô hình BERT.

3.5. Mô hình cải tiến của TF-IDF, BM25, và mô hình tìm kiếm lai

Mô hình TF-IDF tìm kiếm dựa trên từ khóa. Nó đánh giá tầm quan trọng của một từ dựa trên: (i) TF: Từ xuất hiện càng nhiều trong văn bản thì càng quan trọng; (ii) IDF: Từ xuất hiện ở quá nhiều văn bản khác nhau thì càng ít giá trị.

BM25 là mô hình kế thừa và hiện là tiêu chuẩn cho tìm kiếm văn bản cổ điển [8]. Nó cải tiến TF-IDF ở hai điểm (i) Bão hòa tần suất từ: BM25 hiểu rằng nếu tìm "iPhone", một văn bản chứa từ này 20 lần không nhất thiết phải tốt gấp đôi văn bản chứa 10 lần. Sau một ngưỡng nhất định, điểm số sẽ đi ngang; (ii) Độ dài văn bản : Nó "phạt" các văn bản quá dài nhưng loãng và ưu tiên các văn bản ngắn nhưng súc tích, chứa đúng từ khóa.

Nói về mô hình tìm kiếm lai [9]: mặc dù BM25 tốt trong việc khớp chính xác từ khóa, nó lại "mù" về mặt ngữ nghĩa. Nếu người ta tìm "đồ uống có cồn" mà văn bản chỉ có

chữ "bia", BM25 sẽ bỏ qua. Đó là lý do sử dụng mô hình này. Tìm kiếm lai kết hợp (i) tìm kiếm dense vector: sử dụng các mô hình AI (như BERT) để hiểu ngữ nghĩa. Nó tìm kiếm dựa trên "ý nghĩa" thay vì "mặt chữ"; (ii) tìm kiếm sparse, BM25: để các từ khóa chuyên ngành, mã sản phẩm hoặc tên riêng được khớp chính xác tuyệt đối.

3.6. So sánh các mô hình tìm kiếm

Với tư liệu tải từ trang vnexpress.net ngày 23/1/2026 và thuật ngữ hỏi là "an_ninh quốc_phòng", chương trình sau sẽ so sánh định lượng bốn mô hình tìm kiếm. Bảng 1 so sánh các mô hình tìm kiếm. Mô hình bool có kết quả là 1, trong khi ba mô hình sau có kết quả là 5.

- Tìm kiếm bool: (i) văn bản chỉ được trả về nếu có đúng thuật ngữ hỏi; (ii) không có khái niệm “gần nghĩa”;

- Mô hình TF-IDF: (i) dựa trên trùng từ / cụm từ; (ii) không hiểu nghĩa;

- Mô hình Word2Vec: (i) dựa trên gần nghĩa từ vựng; (ii) vec tơ văn bản là trung bình của vec tơ từ;

- Mô hình BERT: (i) tìm kiếm theo ngữ nghĩa, hiểu toàn bộ ngữ cảnh; (ii) khi từ cùng ý nghĩa sẽ có vec tơ gần nhau.

Bảng 1. So sánh kết quả tìm kiếm theo bốn mô hình.

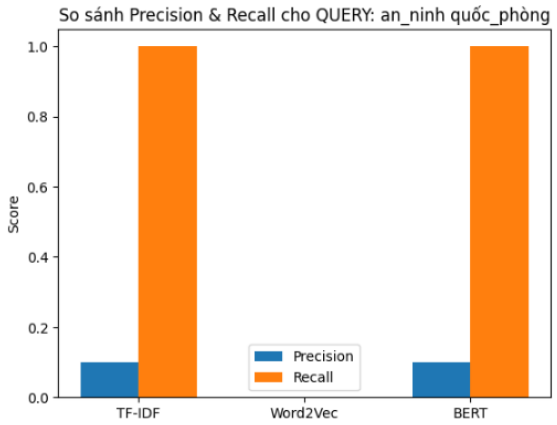
Mô hình tìm kiếm	Đoạn chương trình	Kết quả	Giải thích
Tìm kiếm bool	QUERY = "an_ninh quốc_phòng" def search_by_term(term, df): return df[df["bigrams"].apply(lambda x: term in x)][["content"]] result_term = search_by_term(QUERY, df) len(result_term)	1	Đây là baseline thấp nhất, Độ chính xác cao (precision), Độ phủ thấp (recall)
Tìm kiếm TF-IDF	result_tfidf = search_tfidf(QUERY, df, tfidf_vectorizer, tfidf_matrix) result_tfidf	5	Trả về văn bản có nhiều từ trùng; Không cần xuất hiện đúng TERM
Tìm kiếm Word2Vec	result_w2v = search_word2vec(QUERY, df, w2v_model, doc_vectors_w2v) result_w2v	5	Có thể trả về văn bản không chứa TERM; Nhưng nói về an ninh, quốc phòng, quốc gia

Tìm kiếm BERT	result_bert = search_bert(QUERY, df, bert_embeddings, bert_model) result_bert	5

Theo thực nghiệm trên dữ liệu bài báo sử dụng, đối với ba mô hình tìm kiếm sau, đánh giá về độ chính xác P và độ phủ R thể hiện như trong bảng 2.

Bảng 2. So sánh độ chính xác P và độ phủ R của ba mô hình tìm kiếm sau.

Mô hình tìm kiếm	Độ chính xác P	Độ phủ R
TF-IDF	Trung bình	Thấp
Word2Vec	Trung bình	Trung bình
BERT	Cao	Cao



Hình 12. Biểu đồ so sánh ba mô hình.

Theo lý thuyết, có nhận xét chung về hiệu quả của từng mô hình tìm kiếm theo hình 12 như sau [2]:

- Mô hình bool: (i) nhanh, đơn giản; (ii) không hiểu ngữ nghĩa, hỏi chưa linh hoạt.

- Mô hình TF-IDF: (i) đơn giản hơn mô hình Word2Vec hay mô hình BERT, cải thiện hơn mô hình bool; (ii) không mô hình hóa ngữ nghĩa.

- Mô hình Word2Vec: (i) hiểu được phần nào ngữ nghĩa; (ii) nhúng từ đơn, mất cấu trúc câu.

- Mô hình BERT: (i) hiểu ngữ cảnh hai chiều, hiệu quả cao; (ii) yêu cầu tính toán lớn, phức tạp khi triển khai.

3.6.1. Đánh giá định lượng

Thay vì chỉ nhận xét cảm tính, sau đây là các chỉ số đo lường chuẩn trong tìm kiếm thông tin:

- PrecisionK: Độ chính xác trong K kết quả đầu tiên;

- Recall: Độ phủ;

- Mean reciprocal rank (MRR): Vị trí của kết quả đúng đầu tiên;

- Thời gian phản hồi: So sánh tốc độ giữa TF-IDF (nhanh) và BERT (chậm, cần GPU).

Chương trình Python cho phép tính toán độ chính xác sau mỗi lần tìm kiếm theo các mô hình khác nhau.

```
ground_truth = {
 "an_ninh quốc_phòng": [10, 25, 42, 55], # ID hoặc Index của các bài báo liên quan
"chính_sách kinh_tế": [5, 12, 18],
"y_tế giáo_dục": [3, 7, 21, 33, 45]
}
def evaluate_model(search_func, queries, gt_data, df, **kwargs):
 results = []
 for query, true_ids in gt_data.items():
start_time = time.time()
# Lấy Top 10 kết quả từ mô hình
pred_df = search_func(query, df, **kwargs, top_k=10)
latency = (time.time() - start_time) * 1000 # miliseconds
# Lấy index của kết quả dự đoán
pred_ids = pred_df.index.tolist()
# Tính Precision@5
top_5 = pred_ids[:5]
hits_5 = len(set(top_5) & set(true_ids))
p_at_5 = hits_5/5
# Tính Recall@10
hits_10 = len(set(pred_ids) & set(true_ids))
recall_at_10 = hits_10 / len(true_ids)
# Tính Reciprocal Rank
rs = [1 if i in true_ids else 0 for i in pred_ids]
```

```
rank = np.where(np.array(rs) == 1)[0]
mrr = 1 / (rank[0] + 1) if len(rank) > 0 else 0
results.append({
"Query": query,
"P@5": p_at_5,
"Recall@10": recall_at_10,
"MRR": mrr,
"Latency (ms)": latency
})
return pd.DataFrame(results)
# 2. Chạy đánh giá cho các mô hình
# Đánh giá BM25
eval_bm25 = evaluate_model(search_bm25, ground_truth.keys(), ground_truth, df, bm25_model=bm25)
# Đánh giá BERT
eval_bert = evaluate_model(search_bert, ground_truth.keys(), ground_truth, df, embeddings=bert_embeddings, model=bert_model)
# 3. Tổng hợp kết quả trung bình
summary = pd.DataFrame({
"Metric": ["Avg P@5", "Avg Recall@10", "Avg MRR", "Avg Latency (ms)"],
"BM25": [eval_bm25["P@5"].mean(), eval_bm25["Recall@10"].mean(), eval_bm25["MRR"].mean(), eval_bm25["Latency (ms)"].mean()],
"BERT": [eval_bert["P@5"].mean(), eval_bert["Recall@10"].mean(), eval_bert["MRR"].mean(), eval_bert["Latency (ms)"].mean()]
})
print("BẢNG SO SÁNH ĐỊNH LƯỢNG")
display(summary)
```

Sau khi chạy chương trình, bài báo có nhận xét sau:

- Về độ chính xác: BM25 thường có giá trị P@5 cao hơn đối với các truy vấn ngắn, từ khóa cụ thể vì nó khớp chính xác thực thể. BERT có thể trả về các văn bản "có vẻ liên

quan" về chủ đề nhưng không chứa từ khóa, làm giảm P@5 trong một số trường hợp tìm kiếm thông tin thô.

- Về độ phủ: BERT vượt trội hoàn toàn. Trong khi BM25 có Recall = 0 nếu văn bản dùng từ đồng nghĩa (ví dụ: truy vấn "không quân" nhưng văn bản ghi "máy bay chiến đấu"), BERT vẫn tìm thấy nhờ không gian vec tơ ngữ nghĩa.

- Về hiệu năng: BM25 phản hồi gần như tức thì (<10 ms), trong khi BERT mất thời gian tính toán vec tơ truy vấn và so khớp Cosine trên không gian cao chiều (>100 ms nếu không có GPU).

3.6.2. Xử lý tiếng Việt chuyên sâu

Theo nhận xét về ngôn ngữ Việt trình bày trong phần trên, bài báo đề xuất

- Sử dụng PhoBERT: Thay vì BERT đa ngôn ngữ chung chung, hãy dùng PhoBERT. PhoBERT là phiên bản BERT được tối ưu riêng cho tiếng Việt, cho phép để nhúng văn bản. Điều này làm tăng tính chuyên sâu cho bài báo;

- Xử lý thực thể (NER): Nhận diện các tên riêng như "Bộ Công an", "Chính phủ" để ưu tiên trọng số khi tìm kiếm tin tức thời sự.

Chương trình sử dụng PhoBERT dưới đây sẽ tìm kiếm. kết quả tìm kiếm sẽ được so sánh với các mô hình tìm kiếm khác để thấy ưu điểm của PhoBERT đối với văn bản tiếng Việt.

```
# Hàm tìm kiếm PhoBERT
def search_phobert(query, df, embeddings, model, top_k=5):
query_vec = model.encode([query], convert_to_tensor=True)
scores = cosine_similarity(query_vec.cpu(), embeddings.cpu())[0]
df_result = df.copy()
df_result["score"] = scores
return df_result.sort_values("score", ascending=False)[["content", "score"]].head(top_k)
# Chạy tìm kiếm với PhoBERT
result_phobert = search_phobert(QUERY, df, phobert_embeddings, phobert_model)
print("Kết quả tìm kiếm bằng PhoBERT:")
display(result_phobert)
```

Để tích hợp mô hình BM25 bài báo dùng thư viện rank_bm25. Đây là thuật toán cải tiến dựa trên nền tảng của TF-IDF nhưng tối ưu hơn trong việc xử lý độ dài văn bản và sự bão hòa của tần suất từ.

Mô hình tìm kiếm lai được thực hiện theo hai bước. Bước thứ nhất sử dụng BM25 để lọc ra các ứng viên. Thí dụ đoạn chương trình sau lọc ra 10 ứng viên, với từ khóa "đại_hội".

```
# BM25 tìm 10 ứng viên
corpus = df["processed"].tolist()
tokenized_corpus = [doc.split() for doc in corpus]
bm25 = BM25Okapi(tokenized_corpus)
query = "đại_hội"
query = preprocess(query)
tokenized_query = query.split()
scores = bm25.get_scores(tokenized_query)
top10_idx = np.argsort(scores)[::-1][:10]
bm25_candidates = df.iloc[top10_idx].copy()
bm25_candidates
```

Bước thứ hai sử dụng mô hình tìm kiếm BERT hay PhoBERT để xếp hạng kết quả tìm kiếm. Đoạn chương trình sau sử dụng PhoBERT để xếp hạng.

```
phobert_scores = rerank(query,
bm25_candidates["processed"].tolist(),
phobert_tokenizer,
phobert_model)
bm25_candidates["phobert_score"] = phobert_scores
phobert_ranked = bm25_candidates.sort_values(by="phobert_score",
ascending=False)
phobert_ranked
```

Hiện với dữ liệu thử nghiệm không đủ lớn, sử dụng mô hình tìm kiếm BERT hay PhoBERT đều cho hiệu quả về độ chính xác P và độ phủ R như nhau.

3.7. So sánh và thảo luận trên những thực nghiệm

Dựa trên dữ liệu thực nghiệm thu thập từ *VnExpress* vào ngày 23/01/2026, bài báo tiến hành phân tích đối chiếu giữa các nhóm mô hình: (1) Nhóm so khớp từ khóa truyền

thống (Boolean, TF-IDF), (2) Nhóm cải tiến trọng số (BM25), và (3) Nhóm học máy ngữ nghĩa (Word2Vec, BERT).

3.7.1. Phân tích định tính các kết quả truy vấn

Với câu hỏi truy vấn "an_ninh quốc_phòng", kết quả từ các mô hình bộc lộ những đặc điểm khác biệt rõ rệt:

- Mô hình bool và TF-IDF: Cho kết quả rất hẹp. Nếu văn bản không chứa chính xác cụm từ ghép, mô hình bool sẽ bỏ sót hoàn toàn. TF-IDF cải thiện hơn bằng cách gán trọng số, nhưng vẫn bị nhiễu bởi các bài viết dài có tần suất từ lặp lại cao mà không thực sự liên quan đến nội dung chính.

- Mô hình BM25 là tốt: BM25 thể hiện sự vượt trội so với TF-IDF. Nhờ cơ chế bão hòa tần suất từ và điều chỉnh theo độ dài văn bản, BM25 ưu tiên các đoạn văn ngắn có chứa từ khóa cô đọng. Kết quả trả về có độ chính xác rất cao đối với các truy vấn chứa thực thể xác định (ví dụ: tên cơ quan, bộ ngành).

- Mô hình BERT: Khác biệt hoàn toàn với các mô hình trên, BERT có thể trả về các bài báo không chứa trực tiếp từ "an ninh" hay "quốc phòng" nhưng chứa các khái niệm liên quan như "tuần tra biên giới", "chủ quyền biển đảo" hoặc "hợp tác quân sự". Điều này chứng minh khả năng hiểu ngữ cảnh hai chiều vượt xa việc so khớp ký tự.

3.7.2. So sánh hiệu năng và tài nguyên

Bảng dưới đây tóm tắt các tiêu chí đánh giá giữa hai mô hình mạnh nhất đại diện cho hai tiếp cận (i) BM25; (ii) BERT.

Bảng 3. So sánh hiệu năng và tài nguyên.

Tiêu chí	Mô hình BM25 (Keyword-based)	Mô hình BERT (Semantic-based)
Cơ chế chính	Thống kê tần suất & độ dài văn bản	Nhúng ngữ cảnh (Contextual embedding)
Độ chính xác	Rất cao với từ khóa rõ ràng	Cao, đôi khi bị nhiễu ngữ nghĩa
Độ phủ	Thấp (chỉ tìm thấy bài có từ khóa)	Rất cao (tìm theo ý nghĩa tương đồng)
Tốc độ xử lý	Cực nhanh (mili giây)	Chậm (đòi hỏi GPU để tính toán vec tơ)
Ứng dụng tối ưu	Tìm tên riêng, mã luật, dữ liệu thô	Tìm ý tưởng, câu hỏi tự nhiên, chatbot

3.7.3. Thảo luận về tính thực tiễn

Kết quả thực nghiệm cho thấy không có mô hình nào là hoàn hảo tuyệt đối.

- BM25 là sự lựa chọn hàng đầu cho các hệ thống tìm kiếm trang web cần tốc độ phản hồi nhanh và sự minh bạch trong kết quả (người dùng thấy từ khóa họ gõ nằm trong kết quả).

- BERT cho thấy tìm kiếm thông minh hơn, đặc biệt hiệu quả khi người dùng nhập các câu hỏi dài hoặc không nhớ rõ thuật ngữ chính xác.

- Kết luận rút ra: Xu hướng tối ưu nhất cho các trang web thương mại điện tử hoặc báo chí hiện nay là sử dụng tìm kiếm lai: (i) trước hết dùng BM25 để lọc nhanh các văn bản chứa từ khóa; (ii) sau đó dùng BERT để xếp hạng lại các kết quả đó theo độ tương đồng ngữ nghĩa. Sự kết hợp này giúp tận dụng tốc độ của BM25 và thông minh của mô hình Transformer.

4. Kết luận

Bài báo đã hoàn thành mục tiêu nghiên cứu và thực nghiệm so sánh các mô hình tìm kiếm văn bản từ truyền thống đến hiện đại trên tập dữ liệu thực tế từ trang tin VnExpress (tháng 1/2026). Thông qua quá trình tự động thu thập, làm sạch và biểu diễn dữ liệu bằng các thuật toán từ khóa và ngữ nghĩa, nghiên cứu đã rút ra những kết luận sau:

- Về hiệu quả của các mô hình đơn lẻ: (i) Các mô hình truyền thống như bool và TF-IDF bộc lộ hạn chế lớn về độ phủ và khả năng xử lý biến thể ngôn ngữ; (ii) Mô hình BM25 chứng minh được vai trò quan trọng trong tìm kiếm từ khóa với khả năng cân bằng độ dài văn bản và tối ưu hóa trọng số thuật ngữ, mang lại tốc độ phản hồi cực nhanh; (iii) Mô hình BERT thể hiện sự vượt trội về khả năng hiểu ý định người dùng qua các truy vấn ngữ nghĩa phức tạp, vượt xa khả năng của mô hình nhúng từ cục bộ như Word2Vec.

- Khuyến cáo sau kết quả phân tích và thực nghiệm: Sử dụng mô hình tìm kiếm lai. Dựa trên các phân tích định lượng và định tính, bài báo nhận thấy mô hình tìm kiếm lai là giải pháp tối ưu nhất cho các hệ thống tìm kiếm trang web hiện nay. Mô hình này kết hợp sức mạnh của hai phần (i) truy vấn dữ liệu bằng BM25 để nhanh chóng sàng lọc các tài liệu chứa từ khóa chính xác từ kho dữ liệu lớn, đảm bảo hiệu năng và tính ổn định; (ii) xếp hạng nhờ BERT để đánh giá lại mức độ tương đồng ngữ nghĩa của các tài liệu đã lọc được, giúp đưa những kết quả phù hợp nhất về mặt nội dung lên vị trí đầu tiên.

- Giá trị thực tiễn và hướng phát triển: Kết quả tìm hiểu và thực nghiệm này có khả năng ứng dụng trực tiếp vào các nền tảng thương mại điện tử, hệ thống quản trị nội dung và công cụ phân tích thái độ người dùng mạng. Việc chuyển dịch từ tìm kiếm từ khóa đơn thuần sang tìm kiếm lai giúp doanh nghiệp giải quyết bài toán "hiểu đúng ý khách hàng" mà vẫn đảm bảo được tốc độ xử lý trên quy mô dữ liệu lớn.

Trong tương lai, nghiên cứu có thể mở rộng sang việc tích hợp các mô hình ngôn ngữ lớn (LLM) và kỹ thuật retrieval-augmented generation (RAG) để không chỉ tìm kiếm mà còn tự động tổng hợp câu trả lời từ dữ liệu web, kiến tạo nên các hệ thống hỗ trợ người dùng thông minh và toàn diện hơn. Các kết quả mà nhóm nghiên cứu đưa ra hiện dựa trên dữ liệu thử nghiệm nhỏ, của trang web vnexpress.net. Kết quả phân tích, đánh giá các mô hình tìm kiếm sẽ tốt hơn nếu thử nghiệm trên bộ dữ liệu chuẩn về văn bản tiếng Việt. Nhóm

nghiên cứu mới dừng ở mức tìm hiểu các mô hình để phân tích so sánh và đưa ra khuyến nghị sử dụng mô hình tìm kiếm lai.

TÀI LIỆU THAM KHẢO

[1] C.D. Manning, P. Raghavan, H. Schütze (2008), *Introduction to Information Retrieval*, Cambridge University Press, 189pp.

[2] D. Jurafsky, J.H. Martin (2023), *Speech and Language Processing* (3rd Ed., draft), Prentice Hall, https://web.stanford.edu/~jurafsky/slp3/, accessed 26 March 2026.

[3] G. Salton, M.J. McGill (1983), *Introduction to Modern Information Retrieval*, McGraw-Hill, 278pp.

[4] T. Mikolov, K. Chen, G. Corrado, Jet al. (2013), "Efficient estimation of word representations in vector space", *arXiv preprint*, DOI: 10.48550/arXiv.1301.3781.

[5] X. Liu (2020), "Learning semantic representations for textual similarity with Word2Vec and BERT", *IEEE Access*, **8**, pp.19125-19136.

[6] J. Devlin, M.W. Chang, K. Lee, et al. (2019), "BERT: pre-training of deep bidirectional transformers for language understanding", *Proceedings of The North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp.4171-4186.

[7] Y. Zhang, B. Wallace (2017), "A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification", *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp.253-263.

[8] S.E. Robertson, S. Walker, S. Jones, et al. (1994), "Okapi at TREC-3", *NIST Special Publication*, **500-225**, pp.109-126.

[9] L. Xia, Y. Sun, W.X. Zhao, et al. (2021), "Learning to hybridize sparse and dense models for sequential recommendation", *Proceedings of The World Wide Web Conference (WWW '21)*, pp.214-221.