

Cơ sở của Hệ thống Truy xuất(Retrieval system)

- Cơ chế **tìm kiếm, truy xuất dữ liệu** liên quan đến một truy vấn người dùng (text, hình ảnh, v.v.).
- Các hệ thống như **search engines, document retrieval, question answering...**
- nói về những hệ thống AI như **ChatGPT + tìm kiếm dữ liệu**, thì "Hệ thống Truy xuất" ở đây ám chỉ kiến trúc **Retrieval-Augmented Generation**, một dạng mô hình kết hợp:
 - **Retrieval module** (phần truy xuất)
 - **Generation module** (phần sinh nội dung)
- Giới thiệu
- Truy vấn Thông tin (Information Retrieval - IR) - Nền tảng của Hệ thống Tìm kiếm và AI
- Tokenization (Tách Từ) - Nền Tảng Xử Lý Ngôn Ngữ Tự Nhiên (NLP)

Giới thiệu

Bạn tò mò về "phép thuật" đằng sau những kết quả tìm kiếm? Phần này sẽ bật mí tất cả. Chúng ta sẽ học nguyên lý cơ bản của hệ thống truy vấn và hiểu cách các công cụ tìm kiếm hoạt động.

Bạn sẽ nắm vững những kỹ thuật then chốt—những kỹ năng tưởng chừng chỉ dành cho chuyên gia.

1. Tokenization & Tiền Xử Lý Dữ Liệu

- **Tokenization:** Bước đầu tiên để xử lý dữ liệu văn bản.
- Thực hành các kỹ thuật tiền xử lý, đảm bảo dữ liệu sẵn sàng cho phân tích.

2. Xây Dựng Các Loại Hệ Thống Truy Vấn

- **Hệ thống Boolean:** Sử dụng AND, OR, NOT.
- **Mô hình Không Gian Vector (VSM):** Ứng dụng TF-IDF.
- **Mô hình Xác Suất Truy Vấn.**

3. Truy Vấn & Xếp Hạng Kết Quả

- Yếu tố then chốt để trả về kết quả tìm kiếm chất lượng cao.

4. Kỹ Năng Lập Trình Thực Tế

- Tokenize và tiền xử lý văn bản.
- Xây dựng & truy vấn **inverted index** (chỉ mục ngược).
- Áp dụng các mô hình truy vấn.

Bạn sẽ đạt được

- Thành thạo tokenization & tiền xử lý.
- Hiểu sâu các mô hình: Boolean, Vector Space, Xác suất.
- Tự tin xây dựng và truy vấn inverted index.
- Kinh nghiệm lập trình ứng dụng ngay.

Truy vấn Thông tin (Information Retrieval - IR) - Nền tảng của Hệ thống Tìm kiếm và AI

IR là gì? Tại sao nó quan trọng?

Trong bài này, bạn sẽ hiểu rõ **Information Retrieval (IR)** và tầm quan trọng của nó trong thời đại AI và Big Data.

IR là quá trình tìm kiếm thông tin liên quan trong một tập dữ liệu lớn dựa trên truy vấn của người dùng.

- Khi bạn tìm kiếm trên Google, IR chính là công nghệ đằng sau việc trả về các trang web phù hợp.
- Ví dụ: Bạn gõ "*VHTsoft công ty công nghệ*", hệ thống sẽ quét qua hàng tỷ trang, lọc và trả về kết quả chính xác nhất.

Các thành phần chính của IR

1. Indexing (Lập chỉ mục)

- **Xây dựng một "danh mục" thông minh** bằng cách phân tách văn bản thành các từ khóa (token) và lưu trữ chúng dưới dạng dễ tìm kiếm.
- **Giống như thư viện:** Mỗi cuốn sách được gắn nhãn (tag) để tra cứu nhanh.

2. Querying (Truy vấn)

- **Tìm kiếm thông tin dựa trên đầu vào người dùng.**

- **Ví dụ:** Bạn hỏi trợ lý ảo "*Cách triển khai RAG*", nó sẽ tra cứu chỉ mục và trả về tài liệu phù hợp.

3. Ranking (Xếp hạng)

- **Sắp xếp kết quả theo độ liên quan**, đảm bảo thông tin hữu ích nhất hiển thị đầu tiên.
- **Giống như thủ thư** đặt sách phù hợp nhất lên trên cùng khi bạn hỏi về một chủ đề.

Cách hoạt động của hệ thống IR

1. **Thu thập dữ liệu** (Crawling) – Quét website, tài liệu, database.
2. **Tiền xử lý** (Tokenization, loại bỏ stopwords, stemming).
3. **Lập chỉ mục** (Xây dựng inverted index để tìm kiếm nhanh).
4. **Xử lý truy vấn** (Phân tích câu hỏi người dùng).
5. **Xếp hạng kết quả** (Dùng TF-IDF, BM25, hoặc AI để đánh giá độ phù hợp).

Ứng dụng thực tế của IR

- **Công cụ tìm kiếm** (Google, Bing).
- **Thư viện số** (Google Scholar, PDF databases).
- **E-commerce** (Tìm kiếm sản phẩm trên Amazon, Shopee).
- **Mạng xã hội** (Facebook, Twitter search).
- **Trợ lý ảo** (Siri, Alexa dùng IR để trả lời câu hỏi).

Vai trò của IR trong AI & Data Science

- **Là nền tảng của RAG (Retrieval-Augmented Generation)**, giúp AI truy xuất thông tin chính xác trước khi trả lời.
- **Xử lý ngôn ngữ tự nhiên (NLP)**: IR giúp chatbot, search engine hiểu ngữ cảnh tốt hơn.
- **Big Data**: IR tối ưu hóa việc tìm kiếm trong dataset khổng lồ.

“Không có IR, AI sẽ chỉ là một cỗ máy 'đoán mò' thay vì đưa ra câu trả lời chính xác.”

IR không chỉ là công nghệ cốt lõi của Google mà còn là "**xương sống**" của các hệ thống AI hiện đại. Hiểu IR giúp bạn xây dựng công cụ tìm kiếm thông minh, chatbot chính xác và hệ thống phân tích dữ liệu mạnh mẽ.

Tác giả: **Đỗ Ngọc Tú**
Công Ty Phần Mềm **VHTSoft**

Tokenization (Tách Từ) - Nền Tảng Xử Lý Ngôn Ngữ Tự Nhiên (NLP)

Tokenization là gì?

Tokenization là quá trình chia nhỏ văn bản thành các đơn vị nhỏ hơn như từ, cụm từ hoặc câu, gọi là **token**.

Ví dụ:

“VHTSoft is a technology company”

→ **Tokens:** ["VHTSoft", "is", "a", "technology", "company"]

Tưởng tượng tokenization giống như việc **cắt một cuốn sách thành từng từ riêng lẻ hoặc tách các câu**—đây là bước cực kỳ quan trọng trong NLP và Hệ thống Truy vấn Thông tin (IR).

Tại sao Tokenization quan trọng?

1. Đơn giản hóa xử lý văn bản

- Biến dữ liệu thô thành các phần nhỏ, dễ phân tích.

2. Hỗ trợ đánh chỉ mục (indexing)

- Giúp tìm kiếm và truy xuất thông tin nhanh hơn (vd: search engine).

3. Làm sạch dữ liệu

- Loại bỏ stopwords (từ không quan trọng như "a", "the"), chuẩn hóa văn bản.

4. Giúp AI hiểu ngữ nghĩa

- Là đầu vào cho các mô hình NLP như BERT, GPT.

1. Word Tokenization (Tách từ)

- Chia văn bản thành các từ riêng lẻ.
- Ví dụ:

“Học máy rất thú vị” → ["Học", "máy", "rất", "thú", "vị"]

2. Sentence Tokenization (Tách câu)

- Chia văn bản thành các câu.
- Ví dụ:

“Tôi thích AI. Tôi học NLP.” → ["Tôi thích AI.", "Tôi học NLP."]

3. Character Tokenization (Tách ký tự)

- Chia văn bản thành từng ký tự.
- Ví dụ:

“AI” → ["A", "I"]

Thách Thức Khi Tokenization

- **Xử lý dấu câu:** Dấu chấm, phẩy có nên là token riêng?
- **Từ ghép:** "ice-cream", "mother-in-law"—nên tách hay giữ nguyên?
- **Ký tự đặc biệt:** Emoji, hashtag (#AI), URL.
- **Đa ngôn ngữ:**
 - Tiếng Việt: "Xin chào" → ["Xin", "chào"]
 - Tiếng Anh: "Hello" → ["Hello"]
 - Tiếng Nhật: *"[]" " (không có khoảng cách giữa từ).

Triển Khai Tokenization Trong Python

Sử dụng thư viện **NLTK** (Natural Language Toolkit):

```
import nltk
nltk.download('punkt') # Tải dữ liệu tokenizer

# Word Tokenization
from nltk.tokenize import word_tokenize
text = "VHTSoft is a technology company"
tokens = word_tokenize(text)
print("Word Tokens:", tokens) # Output: ['VHTSoft', 'is', 'a', 'technology', 'company']

# Sentence Tokenization
from nltk.tokenize import sent_tokenize
text = "I love AI. I study NLP."
sentences = sent_tokenize(text)
```

```
print("Sentence Tokens:", sentences) # Output: ['I love AI.', 'I study NLP.']
```

Tiền Xử Lý Văn Bản(TextPreprocessing)

Sau khi **tokenization**, bước tiếp theo là **chuẩn hóa văn bản** bằng cách:

1. **Chuyển thành chữ thường** (lowercase)
2. **Loại bỏ các token không phải chữ và số** (non-alphanumeric)

Triển Khai Preprocessing Trong Python

```
import re

from nltk.tokenize import word_tokenize

def preprocess(text):
    # 1. Chuyển thành chữ thường
    text = text.lower()

    # 2. Tokenization
    tokens = word_tokenize(text)

    # 3. Loại bỏ token không phải chữ/số (giữ lại từ có dấu)
    tokens = [token for token in tokens if re.match(r'^[a-z0-9áàãåăąäääääâêëèēęěēēēēōóòôõöőűűüúũůurúrűýÿŷđ]+$', token)]

    return tokens

# Ví dụ các tài liệu
documents = [
    "VHTSoft is a TECHNOLOGY company!",
    "AI, Machine Learning & NLP are COOL.",
    "Xử lý ngôn ngữ tự nhiên (NLP) rất quan trọng!"
]

# Tiền xử lý từng tài liệu
preprocessed_docs = [' '.join(preprocess(doc)) for doc in documents]

# Kết quả
for i, doc in enumerate(preprocessed_docs):
    print(f"Document {i+1}: {doc}")
```

Kết Quả Sẽ Là:

Document 1: vhtsoft is a technology company

Document 2: ai machine learning nlp are cool

Document 3: xử lý ngôn ngữ tự nhiên nlp rất quan trọng

Giải Thích:

- `text.lower()`: Chuyển tất cả thành chữ thường để đồng nhất hóa.
- `re.match()`: Chỉ giữ lại token chứa chữ cái (kể cả tiếng Việt), số.
- `' '.join()`: Ghép các token lại thành câu sau khi xử lý.

Mẹo Thực Tế Để Tokenization Hiệu Quả

Dù đơn giản, **tokenization** là bước cực kỳ quan trọng để phân tích dữ liệu văn bản và tạo nền tảng cho các tác vụ NLP nâng cao. Dưới đây là những lời khuyên thiết thực:

1. Tiền Xử Lý Văn Bản (Preprocess Text)

- **Chuẩn hóa dữ liệu** trước khi tokenize:
 - Chuyển thành chữ thường (`lowercase`).
 - Loại bỏ ký tự đặc biệt (như `!?, @`), nhưng giữ lại từ có dấu (tiếng Việt).
 - Xử lý viết tắt (vd: "ko" → "không").
- **Ví dụ:**

```
text = "Tokenization là BƯỚC ĐẦU!!!"  
text = text.lower() # "tokenization là bước đầu!!!"
```

2. Xử Lý từ không mang nhiều ý nghĩa(Stopwords) (Handle Stopwords)

- **Stopwords** là những từ ít mang nghĩa (vd: "và", "là", "the").
- Nên loại bỏ chúng để giảm nhiễu, nhưng **cẩn thận với ngữ cảnh**:
 - Tiếng Việt: "**không** tốt" → Nếu xóa "không", nghĩa đảo ngược!
- **Cách làm:**
- **Tùy chỉnh danh sách stopwords để giữ lại từ như:**

```
stopwords = [...] # danh sách từ dừng mặc định  
important_words = ['không', 'chưa', 'chẳng', 'chả', 'đừng']  
  
# Loại bỏ các từ phủ định khỏi stopwords
```


stopwords = [word for word in stopwords if word not in important_words]

- **Dùng mô hình học sâu hiểu ngữ cảnh (BERT tiếng Việt)**

Mô hình như **PhoBERT**, **viBERT**, hoặc **VietAI-BERT** đã được huấn luyện để **hiểu từ phủ định** theo ngữ cảnh, không cần xử lý thủ công.

Khi Nào Dùng Tokenization?

- Xây dựng chatbot, search engine.
- Phân tích cảm xúc (sentiment analysis).
- Xử lý dữ liệu trước khi đưa vào AI model.

Tác giả: **Đỗ Ngọc Tú**
Công Ty Phần Mềm **VHTSoft**