A PROPOSED DISTRIBUTED ARCHITECTURE FOR SEARCHING SEMANTICALLY ON A LARGE DATASET OF HACKING NEWS

Ngoc Long Do[1], *The Hung Nguyen*[1,*],
Trung Dung Nguyen[1], *Xuan Duc Le*[1], *Chi Thanh Nguyen*[2],
Quoc Khanh Nguyen[3], *Thi Bich Van Pham*[3]

Abstract

In this paper, we propose a distributed architecture to support semantic search on large-scale datasets of online technology news. The solution combines knowledge graph modeling, natural language processing techniques, and distributed processing on Apache Spark. The paper presents: (1) A distributed architecture that stores a large-scale semantic news dataset using resource description framework model; (2) A pipeline for extracting knowledge from text using natural language processing (NLP) tools such as dependency parsing and named entity recognition; (3) A distributed search engine that uses keyword expansion and graph reasoning to return semantically related results. The experimental results show that the proposed model improves the semantic search capabilities on large-scale data compared to traditional keyword-based search methods.

Index terms

Distributed system; knowledge graph; semantic search; natural language processing; hackernews; large language models.

1. Introduction

Semantic search represents a significant advancement in information retrieval by focusing on understanding the meaning behind user queries to deliver more relevant results. Unlike traditional keyword-based systems, semantic search comprehends the context, intent, and semantics of the query, leading to more accurate responses [1], [2]. This approach leverages advanced NLP techniques [3], [4] and large language

[1]Institute 486, Command 86
[2]Institute of Information Technology and Electronics, Academy of Military Science and Technology
[3]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author, email: hungnguyenthe1211@gmail.com

DOI: 10.56651/lqdtu.jst.v14.n01.1040.ict

models [5]–[7], which have been instrumental in recent improvements in the field [8]–[11].

The capabilities of large language models, such as Bidirectional Encoder Representations from Transformers (BERT) [4] and Generative Pre-Training (GPT) [5], have greatly enhanced the precision of human language processing. These models excel at capturing nuanced word relationships, making them ideal for semantic search engines. BERT's bidirectional context understanding, for instance, improves search result relevance and accuracy. More recent models, like Large Language Model Meta AI [7], continue to advance semantic embeddings and contextual awareness.

In practical applications, semantic search has demonstrated superiority across various domains, including healthcare, e-commerce, and legal research [12]. Generalist embedding models often outperform domain-specific models in clinical searches by effectively handling diverse queries, challenging the traditional reliance on specialized models. This evolution underscores the potential of semantic search to transform information retrieval in both specialized and general contexts.

Strong cybersecurity defenses are essential to safeguarding sensitive data and upholding user confidence as cyber threats change [13], [14]. Creating efficient information security search tools is crucial to quickly detecting, evaluating, and reducing possible threats. These tools provide security professionals with timely and relevant information so they can stay ahead of emerging risks.

Different terminologies are used while discussing information security and hacking [13]. Terms such as infosec, network security, data breaches, and penetration testing frequently overlap. Even when distinct phrases are used, semantic search can recognize and match synonyms to provide comprehensive search results. Users may not know exactly what terms to use, which can lead to unsuccessful keyword-based searches. Furthermore, semantic search can classify content more efficiently by considering underlying subjects [14].

While semantic search significantly enhances traditional methods, deploying a large-scale semantic search system presents challenges, including data volume, processing speed, scalability, and resource management. Centralized systems struggle with millions of documents, leading to slow responses and inefficient data processing, and they often encounter hardware limitations that prevent scaling.

In the paper, we address these challenges by employing a distributed architecture in which each node handles a subset of the data. Our approach includes preprocessing the data, indexing across multiple nodes, performing semantic search on partitions in parallel, and aggregating the results using a reranking model. This design significantly improves speed and scalability. The main contributions of the paper are as follows:

- Integrate reranking models into centralized semantic search systems to improve accuracy.
- Propose a scalable distributed architecture for semantic search that reduces indexing

time while maintaining performance.
- Conduct a comprehensive evaluation comparing reranking strategies and search architectures.

2. Related work

2.1. Keyword search and semantic search for information retrieval

Information retrieval has advanced significantly with developments from traditional keyword methods like TF-IDF and BM25 [2] to sophisticated Transformer-based models such as BERT [4]. Initially, keyword-based systems determined document relevance by word frequency but lacked context sensitivity, missing synonyms and related terms.

The introduction of neural networks like word2vec added depth by embedding words in a semantic vector space, yet context remained a challenge [1], [6]. This was addressed by Transformer models [6], which enhance contextual understanding through self-attention mechanisms. BERT, a notable Transformer model, comprehensively captures text relationships by processing text bidirectionally, thus refining query and document relevance.

Furthermore, BERT variants like SBERT [15] and ColBERT [16] have been developed to enhance semantic search further, offering improvements in textual similarity tasks and large-scale retrieval efficiencies. The BGE-M3 model by the Beijing Academy of Artificial Intelligence [17] exemplifies the next step in the evolution of semantic search models. It can support more than 100 working languages, leading to new state-of-the-art performances on multi-lingual and cross-lingual retrieval tasks.

These advancements have considerably elevated the accuracy and relevance of semantic search, making it integral to modern information retrieval.

2.2. Centralized semantic search

Developments in machine learning and NLP have greatly benefitted centralized semantic search engines. For these systems to convert text into meaningful vector representations that capture semantic linkages, they usually rely on strong embedding models. Centralized approaches often employ sophisticated indexing techniques to manage large volumes of data.

For instance, Elasticsearch, an open-source search engine, integrates deep learning models to facilitate semantic search. Techniques like deep hashing have been employed to efficiently manage and search extensive datasets [18], as seen in projects like ElasticHash. These centralized systems are capable of providing high accuracy and relevance, making them suitable for various applications, including e-commerce and healthcare.

Another centralized system is the Semantic Product Search [19], which uses a neural network architecture to generate embeddings for queries and products. This system optimizes search processes through average pooling and batch normalization, demonstrating the effectiveness of advanced embedding models in real-world applications.

2.3. Distributed semantic search

Distributed semantic search systems address the challenges of scaling and efficiency by partitioning data and processing tasks across multiple nodes. This approach not only improves performance but also ensures that large datasets can be managed effectively.

A distributed framework for semantic trajectory [20] exemplifies this approach by efficiently handling semantic trajectory similarity joins. It uses specialized indices for textual, temporal, and spatial data, employing effective pruning techniques to manage large-scale data.

DiskANN [21] is another useful contribution in the distributed semantic search domain. It focuses on vector search at web scale, using approximate nearest neighbor search to handle billions of vectors with high accuracy and low latency. DiskANN leverages auxiliary SSD storage to achieve a lower memory footprint while maintaining high search performance, demonstrating the scalability and efficiency required for modern semantic search engines.

BioASQ, an annual biomedical semantic indexing and question-answering challenge, has highlighted the importance of distributed systems in managing large-scale biomedical data. The challenge tasks systems with indexing and retrieving relevant biomedical documents from vast datasets, showcasing the capabilities and advancements in distributed semantic search techniques.

2.4. Challenges of search systems

Scalability is a key feature of systems that need to accommodate large amounts of concurrent requests [22]. Designing centralized semantic search systems presents several challenges, primarily related to scalability, performance, resource management, fault tolerance, data synchronization, security, and relevance. As data volume grows, managing millions of documents can lead to processing and storage bottlenecks [23]. This happens because the system’s infrastructure may struggle to handle the increased load, resulting in slow data retrieval and storage access times. Additionally, indexing large datasets is time-consuming, as the process requires significant computational resources to create and update indexes efficiently, often exceeding the capacity of traditional storage solutions.

High latency is also a significant problem [23], especially when using models with vector lengths of 1024 like the BGE family [24] and GPT, which are computationally demanding, particularly for sophisticated queries needing deep semantic understanding.

It is essential to manage computational and storage resources efficiently, which calls for high-performance technology that can be expensive and difficult to maintain.

Because centralized systems are similarly susceptible to single points of failure, strong fault tolerance techniques like automated failover and data replication are required to guarantee high availability.

It is difficult to guarantee data consistency between system components because synchronization errors or delays can produce inaccurate or out-of-date search results. Robust security measures, including encryption, access control, and frequent security audits, are necessary when handling huge volumes of sensitive data in order to prevent unwanted access and guarantee compliance with privacy laws [23].

In summary, both centralized and distributed semantic search systems have made significant strides in recent years. Centralized systems benefit from powerful models and sophisticated indexing techniques to ensure high accuracy and relevance. In contrast, distributed systems excel in scalability and efficiency, addressing the challenges posed by large datasets. Together, these approaches provide comprehensive solutions for various semantic search applications.

2.5. Aggregating techniques from multiple search results

Reciprocal Rank Fusion (RRF) is a widely used method for combining the results of multiple search systems. This technique is particularly effective in information retrieval tasks where aggregating multiple ranked lists can enhance the final search results. RRF assigns a score to each document based on its rank across multiple result lists.

The formula for calculating the RRF score for a document d is given by:

$$\text{RRF}(d) = \sum_{i=1}^{N} \frac{1}{k + \text{rank}_i(d)} \tag{1}$$

where,

- N is the number of different ranked lists;
- $\text{rank}_i(d)$ is the rank of document d in the i-th ranked list;
- k is a constant, typically set to 60 to avoid overly favoring top-ranked documents.

In research [25], RRF plays a crucial role as a technique for combining search results from multiple sources. Specifically, the author employed RRF to enhance the performance of the query system by merging query results from various models or data sources. RRF aids in the model's ability to effectively retrieve and integrate information from diverse document sources, thereby improving the accuracy and efficiency of the Retrieval-Augmented Generation (RAG) system.

Additionally, the author in [26] highlighted the advantages of hybrid approaches such as RRF in information retrieval competitions. In recent research on document ranking and information retrieval, several methods have been proposed to improve the effectiveness of relevance fusion in search systems. The RRF method has been widely used as a baseline in various studies due to its simplicity and effectiveness.

For example, in the TREC tasks, RRF outperformed other fusion methods like Condorcet and CombMNZ across several collections (TREC Robust, TREC 3, TREC 5, and TREC 9) [25], [26], demonstrating its robustness in various retrieval scenarios. The Mean Average Precision (MAP) scores of RRF, Best Individual, Condorcet, and CombMNZ for different TREC collections are summarized in Table 1 and Table 2. Table 3 shows that RRF outperforms all individual ranking methods.

Table 1. Pilot results. Effect of k on MAP for RRF of 30 model system results on TREC topics 351–400. Results of best model system and competing fusion methods shown for comparison

k	0	10	20	30	40	50	60	70	80	90	100	500
MAP	.2072	.2123	.2134	.2139	.2138	.2144	**.2145**	.2146	.2147	.2145	.2142	.2098

Table 2. MAP scores for the fusion of submitted runs in TREC 3, TREC 5, TREC 9, and the TREC 2004 Robust track

Collection	**RRF**	**Best Individual**	**Condorcet**	**CombMNZ**
TREC Robust	0.3686	0.3586	0.3652	0.3575
TREC 3	0.4350	0.4226	0.4256	0.4381
TREC 5	0.3394	0.3165	0.3213	0.3237
TREC 9	0.2830	0.2801	0.2750	0.2671

Table 3. MAP scores for individual models and fusion results on the LETOR 3 dataset. Differences and p-values included

Method	$\mathbf{MAP_{method}}$	$\mathbf{MAP_{RRF} - MAP_{method}}$	**p-value**
RRF	0.6051 (0.58 – 0.63)	–	–
Condorcet	0.5917 (0.56 – 0.62)	0.0134 (0.00 – 0.02)	0.004
CombMNZ	0.6107 (0.58 – 0.64)	-0.0056 (-0.01 – 0)	0.200
ListNet	0.5846 (0.56 – 0.61)	0.0205 (0.01 – 0.03)	0.001
LGD	0.5837 (0.56 – 0.61)	0.0214 (0.01 – 0.04)	0.003
AdaRank-MAP	0.5778 (0.55 – 0.61)	0.0273 (0.01 – 0.04)	0.000
RankSVM	0.5737 (0.55 – 0.60)	0.0314 (0.02 – 0.04)	0.000
RankBoost	0.5622 (0.53 – 0.59)	0.0429 (0.03 – 0.06)	0.000

The *BAAI/bge-reranker-base* [24] model is a state-of-the-art reranker designed to enhance the precision of search results in semantic search systems. The model utilizes the BERT architecture, known for its deep bidirectional transformers, which capture contextual information from both directions in a sentence. This makes BERT particularly effective for tasks requiring nuanced understanding of text.

By using Cross-Attention mechanisms, the model compares the query with each document, assigning a relevance score based on the contextual match between them. The model is fine-tuned on a large dataset of query-document pairs. This training process ensures that the model learns to rank relevant documents higher than non-relevant ones.

Mean reciprocal rank and normalized discounted mumulative gain, two important performance metrics, show notable improvements when using the *bge-reranker-base* model. The model performs well in mean reciprocal rank, efficiently placing the most pertinent documents at the head of the list. Additionally, it receives excellent normalized discounted cumulative gain scores, demonstrating its tremendous relevance all the way up to the top of the ranking list. These measurements confirm that the model can improve relevancy and precision in search results.

3. Proposed method

In the paper, we propose a scalable distributed architecture for semantic search. It is possible to split up large datasets into smaller clusters, and then combine the search results from these clusters. Our proposed method makes it possible to execute the indexing and retrieval process in parallel. A deep model is then used to aggregate and rerank the search results that have been retrieved. With this method, the processing time can be greatly decreased while the search system's precision is maintained.

3.1. Adding rerank model to sort retrieval result

Our proposed method involves developing a robust, large-scalable semantic search system by integrating a re-ranking model into the search pipeline and utilizing a distributed architecture to reduce indexing time. Initially, a centralized semantic search model is implemented using a pre-trained sentence-transformer model to encode both queries and documents into dense vector representations. The cosine similarity between these vectors is calculated to retrieve an initial set of *First_K* documents. This traditional retrieval process is illustrated in Figure 1.

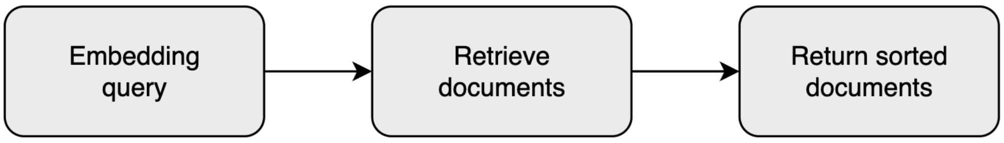

Fig. 1. The traditional search process without the rerank stage.

In the paper, a cross-encoder re-ranking model *BAAI/bge-reranker-base* [24] is added to the pipeline to improve accuracy. It processes the initial results to refine their ranking based on deeper semantic understanding. Figure 2 illustrates our rerank stage in the entire proposed architecture. In this approach, the *BAAI/bge-reranker-base* deep model is integrated to rerank the retrieved results before returning the sorted documents.

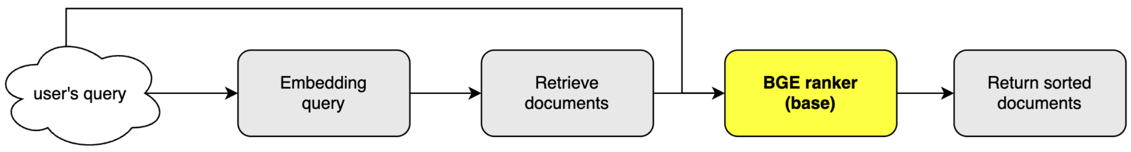

Fig. 2. The proposed search process includes the addition of a BGE base deep rerank model.

3.2. Scalable distributed architecture for semantic search

To efficiently handle large datasets, we propose a distributed architecture. The dataset is partitioned into smaller clusters, enabling parallel processing and significantly reducing indexing time. Each cluster processes queries and documents independently, retrieving the initial *First_K* results. These results are then aggregated and re-ranked using RRF and *bge-reranker-base*. RRF assigns higher scores to documents that appear in multiple rankings from different clusters, while *bge-reranker-base* applies a neural re-ranking model to improve relevance based on semantic similarity between the query and the retrieved documents. This approach ensures high accuracy and relevance in the final results. The distributed design not only facilitates efficient scalability but also maintains high system accuracy, making it suitable for real-world, large-scale applications.

Algorithm 1: Indexing process of distributed semantic search system

```
Input: Dataset $D$; Number of clusters $N$
Output: $N$ clusters with indexed data
Split $D$ into $N$ parts: $\{D_1, D_2, \dots, D_N\}$;
foreach cluster $C_i$ in $\{C_1, C_2, \dots, C_N\}$ do
    $index\_name \leftarrow$ "large_" + str($N$) + "_" + str($i$);
    $init\_connection(cluster\_name)$;
    foreach document in $D_i$ do
        $document\_embedding \leftarrow calculate\_embedding(document)$;
        if document_embedding does not exist then
            insert embedding into $index\_name$ of cluster;
        end
    end
end
```

Algorithm 1 provides the indexing process of a distributed semantic search system. The input of Algorithm 1 includes the dataset D and number of clusters N. The algorithm enables parallel indexing by assigning each sub-dataset D_i to cluster C_j and performing the embedding and insertion processes concurrently.

Figure 3 illustrates our distributed architecture for semantic search. The process begins with preparing the database, often referred to as indexing. Depending on the system's

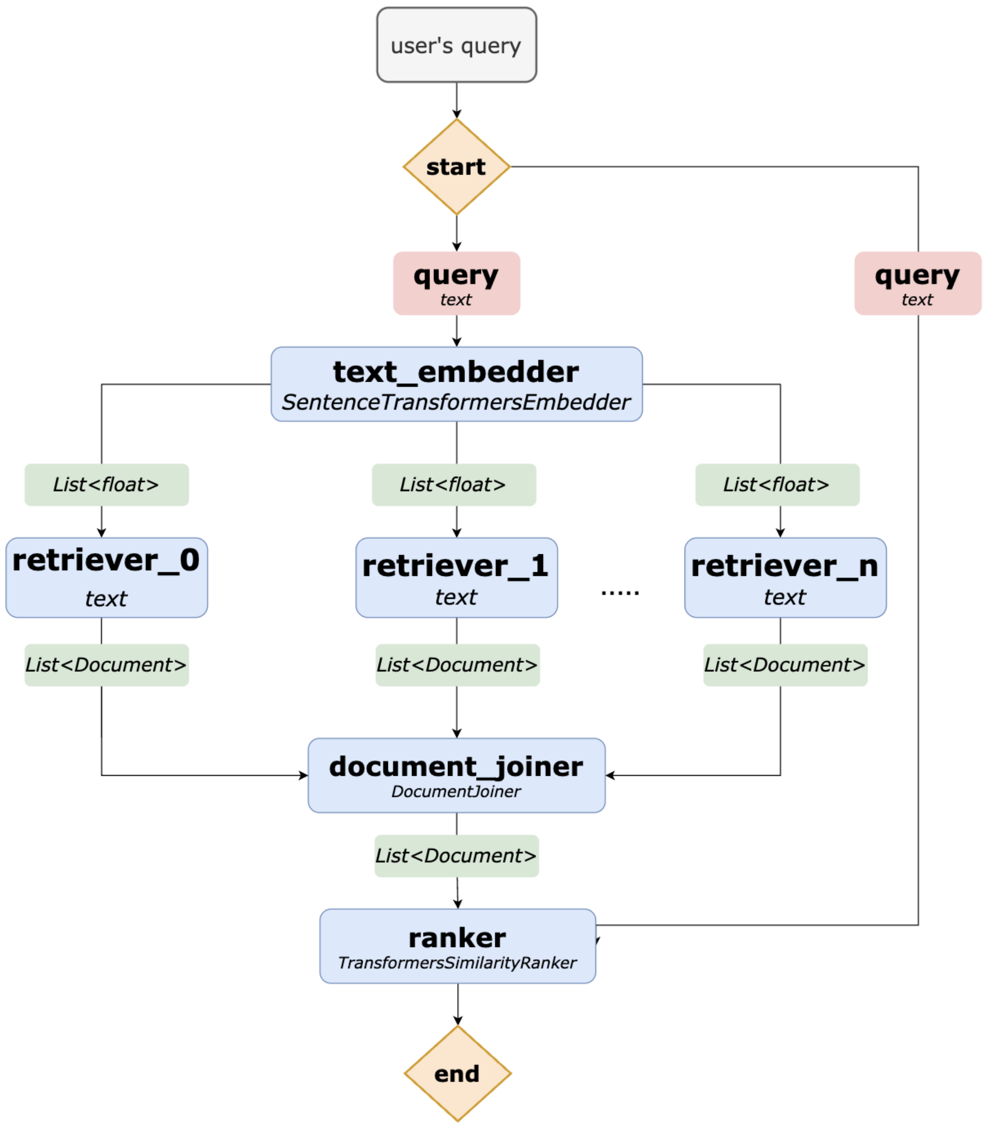

Fig. 3. Distributed architecture for semantic search.

resources and the dataset size, we determine the number of clusters (denoted as n). The dataset is then split into n parts, and embeddings are generated using a sentence-transformer model before being stored in the database server. The proposed search process that consists of the following phases:

- **Phase 1 – Query embedding:** We use the pre-trained model *BAAI/bge-large-en-v1.5* to embed the query. This model is chosen for its efficiency and effectiveness in generating dense vector representations.
- **Phase 2 – Parallel document retrieval:** The query's embedding is

simultaneously fed into retriever components 1 to n. Each component returns a result set of *First_K* documents.
- **Phase 3 – Document joining and re-ranking:** The results from n retrievers are combined, resulting in $n\times$*First_K* documents. These are re-ranked using the BGE base model, which computes a relevance score between the query and each document.
- **Phase 4 – Result generation:** The documents with the highest relevance scores are returned as final results.

Algorithm 2: Search process of distributed semantic search system

```
Input: $N$-cluster with indexed data; user's query; rerank method
Output: Result list $R$
1  $r \leftarrow$ list[];
2  $query\_embedding \leftarrow calculate\_embedding(user\_query)$;
3  foreach cluster $C_i$ in $\{C_1, C_2, ..., C_n\}$ do
4      $index\_name \leftarrow$ "large_" + str($N$) + "_" + str($i$);
5      $res \leftarrow search(index\_name, query\_embedding)$ $r \leftarrow r \cup res$
6  end
7  if mode = "RRF" then
8      $R \leftarrow RRF(r)$;
9  end
10 else
11     $R \leftarrow BGE\_ranker(r, user\_query)$
12 end
13 return $R$;
```

Algorithm 2 presents the search process of a distributed semantic search system. The input includes N-cluster indexed data, the user query *user_query*, and the rerank method. The algorithm performs parallel document retrieval across clusters and reranks the combined results using either the RRF method or a deep re-ranking model (*BAAI/bge-reranker-base*). This ensures both efficiency and high relevance of the returned documents.

3.3. Key features of the architecture

- **Parallel retrieval:** The query embedding is distributed across multiple retrievers, each handling a subset of the dataset. This parallelization improves efficiency, as each retriever only needs to work on a smaller portion of the data.
- **Centralized ranking:** Although retrieval is parallelized, the joining and ranking of documents happen centrally on a server, allowing for a unified ranking of results.
- **Optimized hardware allocation:** Each server in the architecture is configured with hardware suited to its role. Retrieval servers are equipped with large storage capacities and GPUs for fast vector-based search, while the embedding and ranking

servers prioritize GPU and memory resources to handle computationally expensive tasks like document ranking and merging. This division of labor between servers maximizes overall system performance and scalability.

4. Experimental design

This section details the experimental design used to evaluate the performance of the centralized and distributed semantic search architecture. The design focuses on data preprocessing, sentence-transformer model selection, indexing, query evaluation, and statistical analysis.

4.1. Data collection and preprocessing

Data source: This study makes use of the training set of *kerinin/hackernews-stories*, which comprises 313,317 stories from Hacker News [27]. Hacker News is a well-known social news website that focuses on computer science and business. The dataset contains an extensive compilation of stories and comments from this website.

Preprocessing: We preprocessed the dataset to guarantee data quality by eliminating entries that were incorrect due to scraping errors. The filtering process involved removing entries that were missing information or had faulty data. The final cleaned dataset included 278,072 records, which were formatted for indexing into Qdrant.

Figure 4 and Figure 5 show the word length distributions of the *Title* (Question) and *Text* (Answer) fields. These visualizations guide the parameter design for *split_size* and *chunk_size*.

4.2. Model selection

The BGE series features three English models: *bge-small-en-v1.5*, *bge-base-en-v1.5*, and *bge-large-en-v1.5*. These models are tailored for processing English text and provide options balancing efficiency and effectiveness. We selected the most robust variant, *bge-large-en-v1.5*, which outputs 1,024-dimensional vector embeddings.

4.3. Indexing into Qdrant vector database

In the experiments, 10 Qdrant database instances were set up to serve as the backend for indexing and retrieving documents. The cleaned dataset was indexed using the Haystack framework. Each document (story) was split with *chunk_size* = 500 words and *overlap_size* = 32 words, then encoded using the selected sentence-transformer model to generate dense vector representations. These embeddings were stored in Qdrant to support efficient similarity-based search.

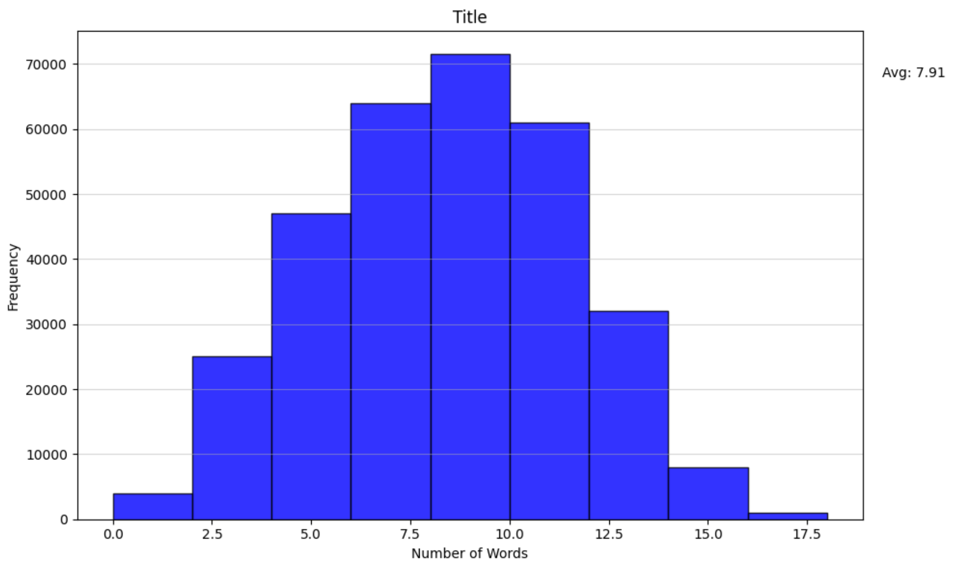

Fig. 4. Word count distribution of the Title (question) field.

4.4. Evaluation method

Query selection: A random sample of 1,000 questions was extracted from the dataset to evaluate the search pipeline. This sampling ensures diverse query distribution to test the robustness and generalizability of the models.

Evaluation metrics:

- **Indexing time:** Time required to complete dataset indexing, indicating preprocessing and vectorization complexity.
- **Query time:** Time required to process 1,000 queries, repeated over 100 trials. We report mean and standard deviation.
- **Top-N accuracy:** Accuracy of top-1, top-5, and top-10 results, evaluated over 30 runs. A result is correct if the expected document appears within the top-N.

4.5. Implementation

Ten identical Qdrant instances were deployed on different ports (6301–6310). Each instance stores a partitioned subset of the dataset. For example, with $n = 3$, each instance stores one-third of the full data, indexed with distinct names.

All experiments were executed on a supercomputer with:

- CPU: Intel Core i7
- RAM: 32GB
- GPU: RTX 4090 24GB
- OS: Ubuntu 20.04

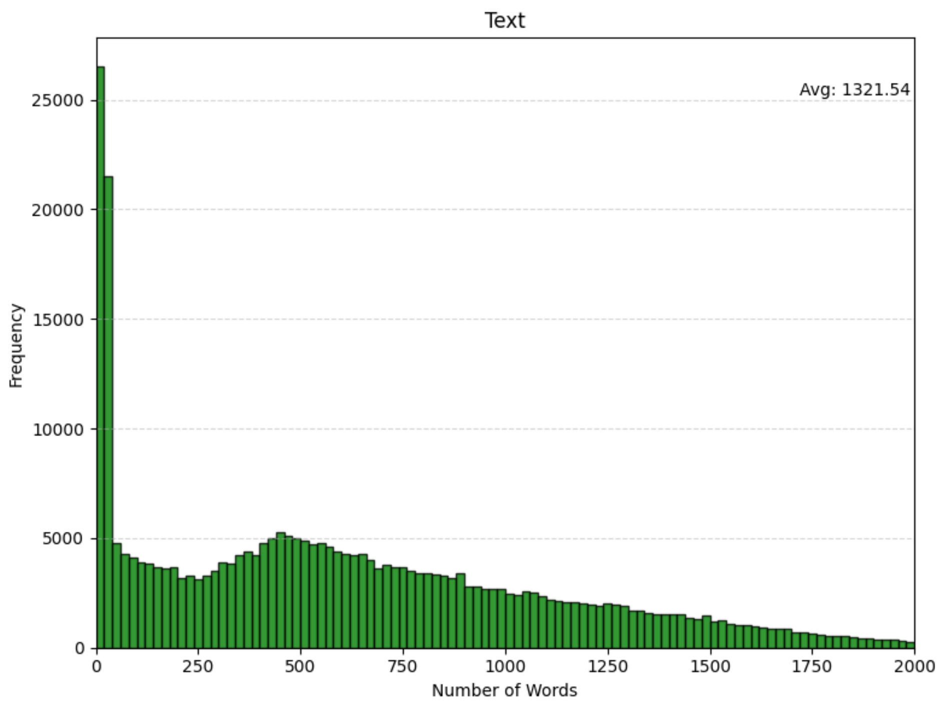

Fig. 5. Word count distribution of the Text (answer) field.

5. Result and discussion

As an unsupervised method, RRF is really fast and easy to implement. It has been found that RRF outperforms many other reranking methods [28]. Recent research highlights RRF's role in ranking aggregation. For instance, RAG-Fusion [25] integrates RAG with RRF by generating multiple queries, using reciprocal scores for reranking, and merging the resulting documents. This approach has been shown to deliver accurate and comprehensive responses by capturing the original query's context from various perspectives. Additionally, in the consumer health search domain, where precision is critical, RRF has been shown to effectively merge results from multiple retrieval models and handle variations in queries, making it a suitable choice for applications requiring high accuracy [28].

In summary, the paper aims to compare a state-of-the-art neural-network based [24] ranking method with RRF-an efficient statistics-based approach—within the proposed distributed architecture. The methodology is evaluated according to three criteria:

1) Indexing time of the distributed scheme.
2) Accuracy comparison between the deep-model reranking technique and RRF.
3) Query-processing time for both reranking methods.

In the semantic search systems, the indexing time occupies a considerable processing effort. Our proposed distributed approach enables to process the indexing in parallel.

Table 4 illustrates the indexing time required for different quantities of clusters. As the number of clusters increased, the indexing time decreased linearly. This tenfold decrease in indexing time demonstrates the effectiveness of the distributed architecture in handling large datasets.

The outcomes show how our method has advanced when used to big datasets. The suggested method makes it possible to drastically reduce how long it takes for semantic search systems to index. In the following section, we demonstrate that our distributed architecture preserves the accuracy and performance of the overall search system. In terms of accuracy (%), Table 5 compares the RRF, which has the parameter $k = 60$, with the proposed methodology integrating the reranking deep model [24]. The table displays the values as mean $\pm$ standard deviation.

Table 4. Indexing time based on the number of clusters (hours)

Number of Clusters	Indexing Time (hours)
1	2.72
3	0.89
5	0.54
7	0.38
10	0.27

Table 5. Top-N accuracy of RRF ($k = 60$) and the proposed deep rerank-based system on different First_K and N-cluster values

N-cluster	First_K	Top 1 (%)		Top 5 (%)		Top 10 (%)	
		RRF	Deep model	RRF	Deep model	RRF	Deep model
1	1	42.8±1.7	42.6±1.4	42.8±1.7	42.6±1.4	42.8±1.7	42.6±1.4
	3	42.7±1.5	46.6±1.6	57.9±1.5	58.4±1.9	57.9±1.5	58.4±1.9
	10	43.0±1.4	47.7±1.7	63.1±1.4	66.0±1.7	68.4±1.3	69.0±1.6
	20	43.0±1.4	47.2±1.6	63.1±1.5	66.5±1.2	68.6±1.4	70.7±1.3
3	1	20.7±1.5	45.6±1.5	55.4±1.7	45.6±1.5	55.4±1.7	45.6±1.5
	3	19.5±0.9	47.7±1.7	60.1±1.4	65.4±1.5	67.6±1.4	65.4±1.5
	10	14.1±1.0	47.2±1.3	52.6±1.4	67.2±1.3	67.4±1.2	71.9±1.2
	20	9.8±0.8	47.0±1.8	39.2±1.8	67.5±1.7	62.1±1.3	72.5±1.4
10	1	10.4±1.0	46.6±1.5	34.1±1.3	64.1±1.4	66.1±1.3	66.2±1.4
	3	8.5±0.9	**47.8±1.8**	25.7±1.2	**68.2±1.7**	56.7±1.3	72.7±1.6
	10	5.2±0.7	46.9±1.4	10.0±1.1	68.0±1.5	22.8±1.4	**73.3±1.7**
	20	3.2±0.4	46.0±1.5	5.4±0.8	67.0±1.6	22.8±1.4	72.5±1.4

As can be seen, our distributed approach outperforms the centralized architecture with one cluster in all Top-1, Top-5, and Top-10 setups. With 10 clusters, the method achieves the highest accuracy at 73.3% when retrieving the top-10 documents. The distributed method additionally outperforms the RRF method with the same number of 10 clusters, reaching higher accuracy at 47.8% and 68.2% in Top-1 and Top-5, respectively. The proposed deep model surpasses the RRF method in all setups of our distributed architectures with 3 and 10 clusters. Furthermore, the proposed deep model also obtains superior accuracy in the centralized architecture with one cluster across all Top-N scenarios; specifically, in the Top-10, its accuracy reaches the highest level at 70.7%.

It is evident that, regarding Top-N metrics, accuracy rises significantly as more documents are retrieved, particularly in Top-5 and Top-10. Another intriguing finding is that our proposed deep model for reranking shows increasing accuracy as more initial documents are retrieved. On the other hand, for RRF, lower accuracy is observed as more initial documents are included. This is clearly seen in the case where the number of retrieved documents increases from 1 to 20, with the RRF accuracy dropping from 20.7% to 9.8% in the 3-cluster setup. These findings indicate that our proposed deep model is both increasingly beneficial and stable when handling more initial documents. Compared to both the centralized architecture and the RRF approach, our distributed method achieves superior accuracy and maintains stability.

In addition to the accuracy evaluation, query-processing times for RRF and the proposed deep model are compared. This comparison gives a clearer picture of the consistent performance of our distributed architecture. Table 6 presents the findings, with the initial document counts ranging from 1 to 20. The processing times are reported in seconds as mean $\pm$ standard deviation.

Table 6. The comparison results among models through running 30 times with 1,000 random queries.

First_K	**Time (second)**	
	RRF	**Deep model**
1	10.52 $\pm$ 0.61	15.77 $\pm$ 0.07
3	10.37 $\pm$ 0.08	24.91 $\pm$ 0.13
10	10.81 $\pm$ 0.06	55.47 $\pm$ 0.12
20	11.70 $\pm$ 0.09	95.71 $\pm$ 0.16

The results in Table 6 show that, although RRF's processing time is shorter, the deep model's processing time remains within an acceptable range. It is noteworthy that our suggested deep model manages to outperform the RRF technique in terms of stability and accuracy, completing 1,000 queries in approximately 96 seconds. The effectiveness of our distributed architecture with a deep model for reranking is seen in Figure 6. It is evident that the deep model is stable and convergent at the architecture of 10 clusters, with accuracy over 70% for all *First-K* configurations. The RRF approach, on the other hand, may operate quickly but exhibits large fluctuations and a noticeably declining accuracy. There are three phases in the document retrieval: text embedding, retrieving, and joining and ranking. For the retrieval time, we use the maximum query time from across the n clusters. The Table 7 presents the experimental results.

In terms of time comparison, we observe that increasing the number of clusters reduces retrieval time for the same *First-K* value due to a smaller search space. However, the time required for joining and ranking increases significantly as the total number of documents grows. The recorded time excludes the duration spent on data transmission between servers. This ensures that the evaluation focuses solely on the internal processing efficiency without being influenced by network latency or communication overhead.

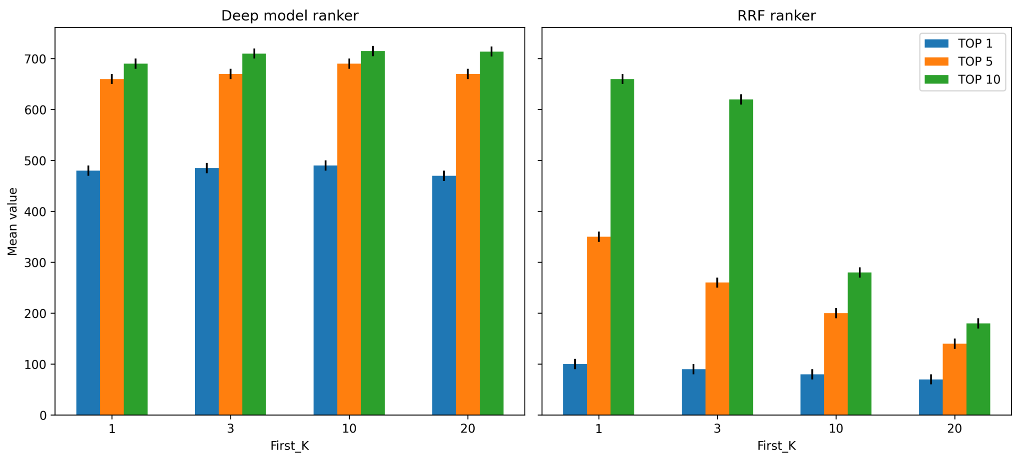

Fig. 6. Accurate comparison between RRF ranker (k = 60) and deep model ranker in case number of clusters = 10.

Table 7. The time comparison (in seconds) across N-clusters for each phase, running 30 times with 1,000 random queries.

N-cluster	First_K	Total docs	Retrieval time	Ranking time (Deep-model)	Ranking time (RRF)
1	1	1	5.72	5.6	–
1	3	3	5.93	14.1	–
1	10	10	6.36	44.16	–
3	1	3	5.04	13.45	0.01
3	3	9	5.39	38.99	0.02
3	10	30	6.01	122.47	0.05
10	1	10	4.65	42.65	0.02
10	3	30	5.02	121.88	0.07
10	10	100	5.53	381.41	0.37

In summary, our distributed approach perform more efficiently in terms of indexing processing and accuracy than the standard centerized architecture, as demonstrated by the findings. Furthermore, in comparison to the RRF approach, the integrated deep model outperformed in both *Top-N* and *First-K* cluster configurations. Furthermore, the distributed approach's deep model maintains stability and takes a reasonable amount of time for each query.

6. Conclusion

In the paper, we propose a distributed architecture, where each node manages a piece of the data, to overcome these issues. Preprocessing the data, indexing across numerous nodes, conducting parallel semantic search on these divisions, and aggregating the outcomes using a ranking deep model are some of the crucial processes in our technique. We may process data in parallel by dividing the dataset among several nodes, which greatly improves the search process's speed and scalability. We assess RRF and our combined deep model for merging cluster-level search results. Various cluster counts were used in the experiments, which gave a comprehensive assessment of the precision and processing speed of our suggested design.

The results demonstrate that our distributed method outperforms the industry standard centerized design in terms of indexing processing and accuracy. In addition, the integrated deep model performed better than the RRF method across the board for both *Top-N* and *First-K* cluster configurations. Furthermore, our distributed approach's deep model keeps stability and responds to queries in a fair period of time. To further evaluate this method's usefulness, future study will investigate integrating more models and extending it to different fields.

References

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *International Conference on Learning Representations*, 2013. DOI: 10.48550/arXiv.1301.3781

[2] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[3] D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Pearson, 2021.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. DOI: 10.48550/arXiv.1810.04805

[5] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018, OpenAI.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008. DOI: 10.48550/arXiv.1706.03762

[7] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "LLaMA: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023. DOI: 10.48550/arXiv.2302.13971

[8] T. Kenter, A. Borisov, and M. de Rijke, "Siamese CBOW: Optimizing word embeddings for sentence representations," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 941–951. DOI: 10.18653/v1/P16-1089

[9] S. Reddy, D. Chen, and C. D. Manning, "CoQA: A conversational question answering challenge," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 249–266, 2019. DOI: 10.1162/tacl_a_00266

[10] B. Nye, J. J. Li, R. Patel, Y. Yang, I. Marshall, A. Nenkova, and B. Wallace, "A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 2, 2018, pp. 197–203. DOI: 10.18653/v1/P18-1019

[11] J. Guo, Y. Fan, Q. Ai, and W. B. Croft, "A deep relevance matching model for ad-hoc retrieval," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2016, pp. 2251–2259. DOI: 10.1145/2983323.2983769

[12] J. Prater, "AI-Powered semantic search: Everything you need to know," Sep. 25, 2023. [Online]. Available: https://www.graft.com/blog/the-future-is-semantic-transforming-search-in-the-age-of-ai.

[13] N. Sun, M. Ding, J. Jiang, W. Xu, X. Mo, Y. Tai, and J. Zhang, "Cyber threat intelligence mining for proactive cybersecurity defense: A survey and new perspectives," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, pp. 1748–1774, 2023. DOI: 10.1109/COMST.2023.3273282

[14] M. Thakur, "Cyber security threats and countermeasures in digital age," *Journal of Applied Science and Education (JASE)*, vol. 4, no. 1, pp. 1–20, 2024. DOI: 10.54060/a2zjournals.jase.42

[15] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410

[16] O. Khattab and M. Zaharia, "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 39–48. DOI: 10.1145/3397271.3401075

[17] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation," *arXiv preprint arXiv:2402.03216*, 2024. DOI: 10.18653/v1/2024.findings-acl.137

[18] N. Korfhage, M. Mühling, and B. Freisleben, "ElasticHash: Semantic image similarity search by deep hashing with Elasticsearch," in *Computer Analysis of Images and Patterns: 19th International Conference, CAIP 2021*. Springer International Publishing, 2021, pp. 14–23. DOI: 10.1007/978-3-030-89131-2_2

[19] P. Nigam, Y. Song, V. Mohan, V. Lakshman, W. Ding, A. Shingavi, C. H. Teo, H. Gu, and B. Yin, "Semantic Product Search," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019, pp. 2876–2885. DOI: 10.1145/3292500.3330759

[20] R. Tian, J. Li, W. Zhang, and F. Wang, "A distributed framework for large-scale semantic trajectory similarity join," *Multimedia Tools and Applications*, vol. 83, no. 6, pp. 16 205–16 229, 2024. DOI: 10.1007/s11042-023-15236-w

[21] S. Gollapudi, N. Karia, V. Sivashankar, R. Krishnaswamy, N. Begwani, S. Raz, H. V. Simhadri, Y. Lin, Y. Zhang, N. Mahapatro, P. Srinivasan, and A. Singh, "Filtered-DiskANN: graph algorithms for approximate nearest neighbor search with filters," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 3406–3416. DOI: 10.1145/3543507.3583552

[22] D. Talia, "A view of programming scalable data analysis: from clouds to exascale," *Journal of Cloud Computing*, vol. 8, no. 1, p. 127, 2019. DOI: 10.1186/s13677-019-0127-x

[23] P. Jatkiewicz and S. Okrój, "Differences in performance, scalability, and cost of using microservice and monolithic architecture," in *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, 2023, pp. 1038–1041. DOI: 10.1145/3555776.3578725

[24] S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, D. Lian, and J.-Y. Nie, "C-Pack: packed resources for general Chinese embeddings," *arXiv preprint arXiv:2309.07597*, 2023. DOI: 10.48550/arXiv.2309.07597

[25] Z. Rackauckas, "RAG-Fusion: A new take on retrieval-augmented generation," *arXiv preprint arXiv:2402.03367*, 2024. DOI: 10.48550/arXiv.2402.03367

[26] H. Yahnoosh, "Relevance scoring in hybrid search using reciprocal rank fusion (RRF)," 2024. [Online]. Available: https://learn.microsoft.com/en-us/azure/search/hybrid-search-ranking.

[27] R. Michael, "kerinin/hackernews-stories," 2023. [Online]. Available: https://huggingface.co/datasets/kerinin/hackernews-stories.

[28] G. M. Di Nunzio, S. Marchesin, and F. Vezzani, "A study on reciprocal ranking fusion in consumer health search IMS UniPD at CLEF eHealth 2020 Task 2," in *Working Notes of CLEF 2020 – Conference and Labs of the Evaluation Forum*, vol. 2696, 2020.

Manuscript received 31-12-2024; Accepted 12-06-2025.

■

Ngoc Long Do graduated from Le Quy Don Technical University in Information Technology in 2023. He is currently pursuing research in computer vision and natural language processing, with a particular emphasis on real-world security and intelligence applications. His primary work in computer vision focuses on the development of smart surveillance systems, including facial recognition, face authentication, and motion detection technologies. In the field of natural language processing, his research delves into semantic search, keyword-based retrieval, and hybrid search methods that combine structured and unstructured information. He has also contributed to the domain of information extraction from text, with publications addressing techniques to automate and enhance understanding of large-scale textual data. His broader interests span the integration of AI into practical systems that require both image and language intelligence. E-mail: longdn2k@gmail.com.

The Hung Nguyen received his Bachelor of Engineering degree in Information Technology from Le Quy Don Technical University, Vietnam, in 2010. He later pursued graduate studies in Australia, earning his Master of Science in Computer Science in 2018 and subsequently completing his Ph.D. in the same field at the University of New South Wales, Canberra, in 2023. His research is primarily centered on distributed computing systems and edge artificial intelligence (AI), with a strong emphasis on enabling intelligent computation within resource-constrained environments. He is particularly interested in optimizing system architectures for decentralized AI model inference, along with enhancing data communication efficiency and fault tolerance in large-scale distributed infrastructures. His broader research interests also encompass system orchestration, federated learning, and AI-driven collaboration between edge and cloud computing platforms. E-mail: hungnguyenthe1211@gmai.com

Trung Dung Nguyen received his Ph.D. in the field of Mathematical Foundations for Information Technology in 2019. He obtained both his Master's and Bachelor of Engineering degrees in Information Technology from Le Quy Don Technical University, Vietnam, in 2002 and 2008, respectively. His current research focuses on the intersection of cybersecurity, natural language processing, and computer vision, with an emphasis on applying theoretical insights to real-world applications. In the domain of natural language processing, his work centers around semantic search techniques, enabling systems to understand and retrieve information based on meaning. His broader research aims to enhance the intelligence and robustness of automated systems by integrating multi-modal data understanding and improving interpretability in AI-based decision-making. E-mail: ntdtoanud2011@gmail.com

Xuan Duc Le received his Ph.D. degree from Russian Technological University, Russian Federation, in 2012. His primary research interests lie in the areas of command and control automation systems, secure data communication, and comprehensive information security. A significant part of his work involves building secure and reliable communication frameworks that ensure the integrity and confidentiality of transmitted data, especially within distributed and high-risk operational systems. His research also actively explores the application of advanced cryptographic techniques to further enhance data protection. In addition, he has contributed to the integration of natural language processing for command interpretation and decision support in automated systems. His interdisciplinary approach bridges system engineering, cybersecurity, and AI technologies to improve the efficiency and resilience of modern command and control architectures. E-mail: lexuanducvn@gmail.com

Chi Thanh Nguyen received his Ph.D. in Information Science and Control Engineering from Nagaoka University of Technology, Japan, in 2012. His recent research focuses on applying deep learning to high-impact challenges in medical diagnostics and natural language processing. The central focus of his research lies in medical image analysis, with publications introducing innovative frameworks for the diagnosis of coronary artery disease using SPECT imaging techniques, segmenting polyps in colonoscopies, and assessing post-thyroidectomy tissue. In the field of natural language processing, his publications explore advancements for the Vietnamese language, including fine-tuning large-scale speech recognition models and developing legal question-answering systems for educational institutions. His broader research also covers topics in human action recognition and biometric signature verification for physical rehabilitation and biometric signature verification. E-mail: thanhnc@ioit.ai.vn

Quoc Khanh Nguyen is the head of the Software Engineering Department at the Institute of Information and Communication Technology, Le Quy Don Technical University. He graduated with both a Bachelor's and a Master's degree in Software Engineering and Computer Networks from the Moscow State Institute of Radio-engineering, Electronics and Automation in 2009. He received his Ph.D. degree in Software for Computers, Complexes, and Networks from the Belarusian State University of Informatics and Radioelectronics in 2014. His research interests span software engineering, service-oriented architecture, web service composition, computer vision, and natural language processing. In recent years, he has focused on the development of integrated systems that combine multiple AI components and services, aiming to support intelligent decision-making in real-time applications. These systems emphasize interoperability, scalability, and modularity. E-mail: khanh29bk@mta.edu.vn

Thi Bich Van Pham received her B.Sc. in Information Technology and M.Sc. in Information Systems from Le Quy Don Technical University in 2010 and 2014, respectively. She earned her Ph.D. in Software Engineering from the University of New South Wales, Canberra, Australia, in 2020. Her research is primarily focused on software quality assurance, software metrics, and cybersecurity, with a growing interest in the application of artificial intelligence (AI) techniques to improve software reliability and security. Her work also involves developing intelligent systems for source code analysis, leveraging static and dynamic analysis techniques to assess code quality, maintainability, and compliance with best practices. By integrating AI into software engineering workflows, she seeks to enhance the robustness and security of modern software systems, particularly in mission-critical and large-scale applications. E-mail: vanptb@lqdtu.edu.vn

KIẾN TRÚC PHÂN TÁN ĐỀ XUẤT CHO TÌM KIẾM NGỮ NGHĨA TRÊN TẬP DỮ LIỆU LỚN VỀ TIN TỨC HACK

Đỗ Ngọc Long, Nguyễn Thế Hùng, Nguyễn Trung Dũng, Lê Xuân Đức,
Nguyễn Chí Thành, Nguyễn Quốc Khánh, Phạm Thị Bích Vân

Tóm tắt

Bài báo đề xuất một kiến trúc phân tán hiệu quả để xử lý bài toán tìm kiếm ngữ nghĩa quy mô lớn. Nhóm tác giả tích hợp mô hình mạng học sâu cho phép xếp hạng lại trong hệ thống tìm kiếm để cải thiện độ chính xác của kết quả. Với cách tiếp cận của nhóm tác giả, các tập dữ liệu lớn có thể được chia thành các cụm nhỏ hơn. Phương pháp này làm giảm đáng kể thời gian xử lý chỉ mục. Bộ dữ liệu tin tức hack hơn 300 nghìn bản ghi được sử dụng để đánh giá kỹ thuật phân tán được đề xuất. Một so sánh giữa mô hình mạng học sâu và phương pháp tổng hợp xếp hạng đối ứng thông thường (RRF) để hợp nhất các kết quả tìm kiếm từ các cụm được cung cấp. Ngoài ra, nhóm tác giả so sánh phương pháp đề xuất với kiến trúc tập trung. Các thử nghiệm đã được thực hiện trên số lượng cụm khác nhau và đưa ra đánh giá kỹ lưỡng về độ chính xác cũng như thời gian xử lý của kiến trúc được đề xuất của chúng tôi. Kết quả cho thấy về mặt xử lý lập chỉ mục và độ chính xác, kỹ thuật phân tán của chúng tôi vượt trội hơn kiến trúc tập trung tiêu chuẩn và mô hình học sâu được tích hợp hoạt động hiệu quả hơn phương pháp RRF trên mọi cấu hình.

Từ khóa

Hệ thống phân tán; tìm kiếm ngữ nghĩa; mô hình ngôn ngữ lớn; xử lý ngôn ngữ tự nhiên; tin tức hack.