Arabic Books Search Engine

Yousef Nasser Mohamed   
Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
ynaser.4251@gmail.com

Mahmoud Mohamed Mostafa  
Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
mahmoud2m2ali@gmail.com

Prof. Dr.Howida A. Shedeed Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
dr\_howida@cis.asu.edu.eg Mohamed Ahmed Elsayed  
Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
muhammedahmedelsayeed@gmail.com

Kerolos Ashraf Fouad  
Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
kero.ashraf212@gmail.com

TA.Nouran El-Said Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
noran.elsayed@cis.asu.edu.eg Mohamed Sayed Zakaria   
Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
mohamedzakaria7694@gmail.com

Abd Elrahman Mohamed Ahmed  
Scientfic Computing Department *Faculty of Computer and Information Sciences, Ain Shams University*Cairo, Egypt  
abdoawaad239@gmail.com

*Abstract*— In today's digital age, the exponential growth of Arabic textual content has created a pressing need for intelligent and efficient search systems tailored to the unique characteristics of the Arabic language. This project presents an Arabic Book Search Engine that leverages Natural Language Processing (NLP) techniques to enhance information retrieval accuracy and relevance for Arabic texts.

Unlike traditional keyword-based search engines, our system applies advanced preprocessing, including normalization, tokenization, stemming, and vectorization methods to understand the semantic context of user queries. By using TF-IDF and other vector-based models, the engine retrieves and ranks Arabic book descriptions based on linguistic relevance,

With an average similarity score of 85% between user queries and retrieved results, the system demonstrates high accuracy in matching user intent. The application is developed as a user-friendly mobile platform, providing users with fast and relevant access to books. This project aims to bridge the gap between Arabic language complexities and modern search technologies, ultimately improving access to Arabic literary resources.

# Introduction

The dramatic growth of Arabic digital libraries, academic repositories, and online bookstores has amplified the need for advanced search systems that can handle the linguistic complexity and semantic richness of the Arabic language. Yet, most existing Arabic search engines still depend heavily on **basic keyword-based retrieval methods**, which are inherently limited in understanding the **actual intent** or **contextual meaning** behind a user's query.

Arabic presents unique challenges for information retrieval due to its:

* **Morphological complexity**: Root-based word derivation causes significant variation in surface forms.
* **Ambiguity and synonymy**: Multiple words can express the same concept depending on dialect, context, or domain ("انطلاق“,"حرية")
* **Flexible syntax**: The position of words in a sentence can vary without changing the meaning.
* **Lack of robust Arabic NLP tools**: Compared to English, fewer open-source libraries exist for effective Arabic language processing.

As a result, users often experience **low-quality search results**, especially when their queries are complex, abstract, or phrased differently from the book metadata.

Our Solution: A **Hybrid Semantic-Keyword** Search Engine

To address these limitations, we developed a **hybrid Arabic book search engine** that intelligently combines the **efficiency of keyword-based retrieval** with the **depth of semantic search** powered by machine learning. This hybrid strategy ensures **accurate**, **meaningful**, and **context-aware** search results for Arabic readers.

The system operates in two main phases:

* Phase 1: keyword-Based searching
  + This phase significantly narrows the search space by **filtering results within related genres or topics**, which improves both speed and relevance.
  + If the query is short or contains rare keywords, this phase ensures that **exact or high-frequency term matches** are prioritized.
* Phase 2: Semantic Similarity searching
  + In this phase, we compute **contextualized embeddings** for the user query and the most similar in book space within certain genres.
* Phase 3: **Retrieval**.
  + **Cosine similarity** is computed between the query and each document embedding to re-rank the results based on semantic closeness
  + This ensures that books are not only matched by words but by **concepts**, **themes**, and **intent**.

# Related Works

***Arabic Information Retrieval (IR) remains a challenging task due to the complex morphology, lack of standard orthography, and limited availability of annotated corpora. Recent research has shifted from traditional keyword-based search models to hybrid and deep learning-based approaches that incorporate semantic understanding and contextual embeddings.***

### Kassimi and Essayad [1] conducted a comprehensive study on various Arabic IR models and their effectiveness when combined with advanced natural language processing techniques. Their paper evaluated several lemmatization techniques—including Farasa, ISRI, Qalsadi, and a custom-built dictionary—on Arabic corpora. They concluded that ISRI and Farasa offered better mean average precision (MAP) scores in preprocessing Arabic text. Additionally, the authors compared traditional vector-based models such as BM25 and TF-IDF against neural embedding techniques like Word2Vec and FastText. Among them, BM25 consistently outperformed others, but the integration of multilingual BERT into the IR pipeline provided the highest overall accuracy, reaching 89% on large-scale Arabic datasets. Their work emphasized the importance of combining keyword-based methods with semantic models like BERT to enhance retrieval effectiveness in morphologically complex languages like Arabic.

### In a follow-up study, Kassimi, Abdellatif, and Essayad [2] introduced a mono-lingual semantic search engine that integrates keyword search with context-aware semantic similarity using pre-trained language models. They conducted rigorous experiments comparing different transformer-based architectures, such as AraBERT, MARBERT, CAMeLBERT, QARiB, and multilingual models like MiniLM and MPNet. The results revealed that Arabic-specific models like QARiB and AraBERT outperformed general multilingual models in terms of semantic relevance, especially when combined with traditional retrieval methods such as BM25 for initial ranking and cross-encoders for reranking. Notably, the paraphrase-multilingual-mpnet-base-v2 model achieved the best performance across several MAP thresholds. This study demonstrated that hybrid architectures that combine semantic indexing with lexical retrieval strategies provide the best balance between relevance and performance for Arabic IR systems.

### El Hadi et al. [3] approached the semantic search problem from a distributed computing perspective, focusing on the scalability of semantic similarity measures in large Arabic corpora. Their system used a MapReduce programming model to compute semantic similarity scores between Arabic documents and queries. The process began with Arabic stemming and translation of terms into English using the Yandex API. The

### translated terms were then semantically compared using WordNet-based similarity metrics, particularly the Leacock–Chodorow measure. The authors also incorporated cosine similarity between TF-IDF vectors of queries and documents. Their findings showed that combining symbolic and semantic similarity computations significantly improved retrieval precision. Additionally, their use of parallel processing via MapReduce allowed their model to scale efficiently with large datasets, which is critical for real-world Arabic search engine deployment.

Collectively, these studies highlight the importance of integrating traditional IR models with modern NLP techniques to improve semantic understanding in Arabic search systems. While Kassimi et al. [1], [2] focused on the effectiveness of transformer-based models and lemmatization strategies, El Hadi et al. [3] introduced a scalable semantic similarity framework suitable for large-scale deployments. These foundational works form the basis of our own approach, which builds on their insights by incorporating contextual embeddings, domain-specific lemmatization, and a semantic-enhanced indexing pipeline tailored to Arabic books.

# proposed method

A diagram of a company

AI-generated content may be incorrect.The architecture of the proposed Arabic Books Search Engine is designed to support efficient and context-aware information retrieval using a hybrid approach that combines keyword-based filtering with semantic embedding models. The system is modular and consists of the following major components: Data Preprocessing Module, Indexing and Embedding Module, Query Processing Pipeline, and Similarity Matching & Ranking Engine. An overview of the architecture is shown in [*Figure 1*]

Figure 1: System Architecture

## preprocessing:

pre-processing step is crucial for information retrieval applications. It aims to identify the optimal form of the term to achieve the best retrieval performance. In our system, we first introduce methods for cleaning each sentence in the corpus removing noise and many unnecessary words. Second, we introduce a tokenization method to break a stream of text into tokens. Finally, we introduce our lexical database for the Arabic language aiming to overcome the challenge of Arabic language lemmatizing and stemming.

In the preprocessing step, not much semantic meaning is captured by the keyword-based model. Word embedding is a new technique in which each word or phrase is mapped to an N-dimensional vector. Therefore, a simple tokenizer corpus works better compared to more complex preprocessing techniques such as lemmatization or stemming First, we are losing valuable information that would help deep learning. Second, most embeddings have preprocessed their corpus in their way. However, it is better to improve our data quality by cleaning up the corpus to remove some punctuation marks to take into account the context of the word.

## Indexing :

Keyword indexing is the process of extracting important terms from documents and mapping them to the documents where they appear. This creates an **inverted index** — a data structure that allows fast lookup of documents by keyword. During indexing, documents are preprocessed (tokenized, cleaned, and filtered), and keywords are stored with references to their corresponding document IDs or genres.At query time, extracted keywords are matched against this index to quickly **filter relevant documents or genres**, enabling fast retrieval. If no match is found, the system falls back to **embedding-based semantic search** to ensure robust performance even with unseen or uncommon query terms.

## Embeddings:

There are three main approaches to text representation and retrieval:

### **TF-IDF** is a simple, fast statistical method that uses word frequency to represent text but lacks understanding of context or meaning.

### **Keras-based models** (like DNNs, CNNs, LSTMs) can learn patterns from text and improve relevance scoring, but they require labeled data and training time.

### **Transformer-based-embeddings**(e.g.,BERT, AraBERT,all-MiniLM-L6-v2) provide deep semantic understanding by converting text into dense vectors. These are ideal for capturing meaning and handling complex or unseen queries.

Each method serves a different purpose: TF-IDF for keyword matching, Keras for learning from data, and transformers for semantic search.

## Retrieval

Semantic search engines compare user queries with each text from a set of documents, calculate a similarity score for each comparison, and get the n-top results based on the similarity scores. To provide an effective search engine over a large corpus

*Cosine similarity measures how similar two vectors are by calculating the cosine of the angle between them. It's commonly used to compare text embeddings. [Equation 1]*

***Cosine Similarity=∣∣A∣∣⋅∣∣B∣∣A⋅B​***

Equation 1

*Values range from* ***-1 (opposite)*** *to* ***1 (identical)****, with* ***0*** *meaning no similarity.*

*To compute* ***average similarity*** *,calculate the cosine similarity between the query vector and each document vector, then take the mean of all scores. This helps identify how semantically close the query is to a group of documents.*

# User Interface and Experimental Results:

## Datasets:

*We constructed a high-quality Arabic book dataset by ethically scraping publicly available content from online sources for academic use. Each of the ~31,000 entries includes key fields such as title, author, category, description, rating (if available), and cover image URL. The data was cleaned, normalized, and stored in both CSV and XLSX formats for seamless integration with NLP models.  
The dataset covers a wide range of 22 main categories, ensuring broad topical diversity. All entries include meaningful Arabic descriptions, enabling effective keyword-based search and semantic similarity tasks. This custom dataset was foundational to our system, supporting every stage from preprocessing and embedding training to retrieval and evaluation.*

## Experimental Setup and Model Evaluation:

### Key-word based search:

*Keyword-based search matches user queries to documents by identifying overlapping terms. It relies on an inverted index that maps keywords to relevant documents, making it fast and effective—especially for* ***short queries*** *with clear, specific terms.*

*However, this approach has notable limitations:*

* *It struggles with* ***long queries****, where important concepts may be spread across multiple words or phrased differently.*
* *It fails to understand* ***semantic meaning*** *or* ***user intent****, treating words literally without recognizing synonyms, context, or phrasing variations*

A diagram of a word

AI-generated content may be incorrect.*As a result, keyword-based search may return incomplete or irrelevant results when the user's language differs from the indexed terms, making it less suitable for complex or conversational queries*.

Figure 2:Inverted Index Construction

### Embedding-Based Search:

*Embedding-based search represents queries and documents as dense semantic vectors using pretrained language models. It captures the* ***meaning and context*** *of text, enabling accurate matching even when the query and documents use different words or phrasing. This makes it highly effective for* ***long or complex queries****, where user intent is better expressed.  
However, its effectiveness drops with* ***short queries****, which often lack sufficient context or representative information. In such cases, the embedding may be vague or ambiguous, leading to reduced retrieval accuracy. Additionally, embedding-based methods are computationally heavier, requiring vector storage and similarity computations, which can impact performance at scale.*

(Average Similarity Scores for top 10 retrieved book)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | AraBert | all-MiniLM-L6-v2 | TF-IDF | BOW |  |  |  |
| long | 91% | 90% | 68% | 64% |  |  |  |
| short | 79% | 75% | 64% | 55% |  |  |  |

Table 1: Average Similarity Scores

### Hybrid Search Approach:

*To balance speed and semantic accuracy, we adopted a hybrid search strategy that combines keyword-based and embedding-based methods. For* ***short queries****, the system first checks for keyword matches using an inverted index. If matching terms are found, the system performs* ***fast keyword-based retrieval****, which is more efficient and accurate for concise queries where keywords are clear and direct.*

*If no keywords are found, or if the query is* ***long and context-rich****, the system falls back to* ***embedding-based search****, which leverages dense vector representations to capture deeper semantic meaning and user intent [Table 2]. This allows the system to retrieve relevant results even when the query and documents use different vocabulary or phrasing.*

*By combining both techniques, the hybrid approach maximizes retrieval quality: it maintains high efficiency for short queries while ensuring robust semantic understanding for longer, more expressive inputs.*

| **Search Method** | **Avg. Similarity Score** | **Retrieval Speed** | **Performance Summary** |
| --- | --- | --- | --- |
| **Keyword-Based Search** | - | Fast | Efficient for short queries, poor semantic understanding |
| **Embedding-Based Search** | 0.85 | Slower | Accurate for long queries, captures user intent |
| **Hybrid Search** | **0.90** | Balanced | Combines both: fast when possible, accurate when needed |

Table 2: Performance

*Conclusion*

This work presents a hybrid search engine for Arabic books that intelligently combines keyword-based search with semantic embedding techniques, achieving significant improvements in retrieval accuracy and relevance over traditional approaches. By integrating inverted indexes for fast lexical matching and incorporating transformer-based embeddings for deep semantic understanding, our system effectively addresses the challenges posed by the morphological richness, synonym, and syntactic complexity characteristic of Arabic text.

Key contributions and findings include:

* Enhanced Retrieval Performance
* Scalable and Efficient Architecture
* User-Centered Relevance
* Broader Implications for Arabic NLP

*Future Work*

* Feedback-Driven Refinement: Integrate user relevance feedback to continuously fine-tune ranking models.
* Full-Text Support: Extend indexing and embedding beyond descriptions to include complete book content.
* Multilingual Capability: Support cross-lingual searches by embedding Arabic and foreign-language texts into a shared vector space.
* Production-scale Deployment: Optimize system components for deployment in real-world environments, focusing on efficiency and scalability.

##### References

1. M. A. Kassimi and A. Essayad, “BERT Representation for Arabic Information Retrieval: A Comparative Study,” *Multimedia Research*, vol. 6, no. 3, pp. 1–12, 2023. doi: [10.46253/j.mr.v6i3.a1](https://doi.org/10.46253/j.mr.v6i3.a1).
2. M. A. Kassimi, H. Abdellatif, and A. Essayad, “Mono-Lingual Search Engine: Combining Keywords with Context for Semantic Search Engine,” in *Advances in Intelligent System and Smart Technologies*, Lecture Notes in Networks and Systems, vol. 826, Springer, 2024, pp. 353–361. doi: [10.1007/978-3-031-47672-3\_34](https://doi.org/10.1007/978-3-031-47672-3_34).
3. A. El Hadi, Y. Madani, R. El Ayachi, and M. Erritali, “A New Semantic Similarity Approach for Improving the Results of an Arabic Search Engine,” *Procedia Computer Science*, vol. 151, pp. 1170–1175, 2019. doi: <https://doi.org/10.1016/j.procs.2019.04.167>.
4. Darwish et al. (2021) provide a broad overview, including Arabic IR with discussions on stemming, diacritization, embeddings, and semantic search.
5. A full pipeline including Arabic text normalization, TF–IDF vectorization, inverted-index construction, and a Flask search API.
6. Compares TF‑IDF against word embeddings for sentence-level similarity in Arabic, analyzing various stemmers’ impact on retrieval performance.
7. Proposes TF‑IDF feature selection for Arabic text classification—a key component in IR pipelines.
8. Abderrahim et al. (2018) apply word embedding similarity techniques to boost Arabic web search accuracy.