Improving Arabic Relational Aggregated Search Using a Hybrid RAG and AraBERT Framework

Sara Saad Sedhom[1], Ahmed Younes[2], Islam Elkabani[3], Ashraf Elsayed[4]
Faculty of Science-Department of Mathematics and Computer Science, Alexandria University, Alexandria 21511, Egypt[1, 2, 3, 4]
Faculty of Computer Science and Engineering, Alamein International University, Alamein 51718, Egypt[2, 4]
School of Information Technology, University of Cincinnati, Ohio, USA[3]

***Abstract*—Aggregated search is challenged by data heterogeneity, redundancy, and irrelevant information, particularly in Arabic because of its rich morphology and dialectal diversity. This study proposes an AI-driven framework to enhance Arabic aggregated search by integrating Retrieval-Augmented Generation (RAG) with semantic clustering. The framework employs RAG to retrieve relevant information from multiple search verticals, including web pages, images, news articles, and videos, and to generate contextually enriched textual representations. These representations are encoded with AraBERT to produce 768-dimensional semantic embeddings, which are then compressed into compact latent feature vectors through a stacked autoencoder. K-Means clustering is then applied to organize the compressed representations into semantically coherent topic groups. The proposed framework was evaluated on nine Arabic datasets spanning five application domains: education, sports, information technology, healthy food, and weather. Experimental results demonstrate that the proposed approach increases the average silhouette score from 0.59 to 0.68, representing a 15% improvement over clustering based on raw aggregated search results. Ablation experiments further indicate that both Retrieval-Augmented Generation and stacked autoencoder–based feature compression independently contribute to the observed performance gains. These findings demonstrate the effectiveness of integrating Retrieval-Augmented Generation with semantic representation learning to improve the organization and clustering of Arabic aggregated search results.**

***Keywords*—Relational Aggregated Search (RAS); information retrieval; Natural Language Processing (NLP); Retrieval-Augmented Generation (RAG); AraBERT embedding; feature extraction; stacked autoencoders; K-Means**

I. Introduction

Conventional search engines, such as Google and Bing, typically return ranked lists of hyperlinks, requiring users to examine multiple webpages and manually integrate the retrieved information. Although this approach is adequate for straightforward information needs, it becomes increasingly inefficient when relevant information is distributed across different content types and specialized search services.

Aggregated search addresses this limitation by combining results from multiple verticals, including web pages, images, news articles, and videos, within a unified search interface [1]. Presenting heterogeneous information in a single results page improves information accessibility and reduces the need to navigate between multiple search services [2]. Nevertheless, the retrieved results often differ considerably in structure, quality, and semantic content. Duplicate entries, fragmented snippets, irrelevant information, and limited contextual information frequently reduce the effectiveness of subsequent analysis and organization. These challenges are particularly pronounced for Arabic content, where rich morphology, orthographic variations, and dialectal diversity further complicate semantic interpretation.

Recent advances in large language models (LLMs) have significantly improved the ability of Natural Language Processing (NLP) systems to understand and generate human language. Despite their impressive capabilities, LLMs rely primarily on knowledge acquired during pretraining and therefore cannot reliably incorporate newly available or domain-specific information without access to external knowledge sources [3]–[5]. This limitation has motivated the development of Retrieval-Augmented Generation (RAG), which combines information retrieval with neural text generation. Instead of generating responses solely from model parameters, RAG retrieves relevant documents from an external knowledge base and uses the retrieved evidence to generate contextually grounded textual representations [6]–[9]. Consequently, the generated content is generally more informative, coherent, and factually supported than text produced without retrieval augmentation.

The quality of the RAG system is closely related to the semantic representations used throughout the retrieval pipeline. For Arabic, AraBERT has become one of the most effective pretrained language models because it is specifically designed to capture the linguistic characteristics of Arabic, including its rich morphology and orthographic variations [10], [11]. The resulting contextual embeddings provide meaningful semantic representations that are well suited for downstream information retrieval and clustering tasks. However, the high dimensionality of these embeddings may reduce clustering performance because of the curse of dimensionality. Stacked autoencoders address this challenge by learning compact latent representations that preserve the underlying semantic information while reducing redundancy and noise [12]. These compressed representations provide a more suitable feature space for clustering algorithms.

Among existing clustering techniques, K-Means remains an attractive choice because of its computational efficiency, scalability, and ability to partition semantically similar instances into coherent groups [13], [14]. When combined with high-quality semantic representations, K-Means provides an effective

mechanism for organizing heterogeneous search results into meaningful topical categories.

Motivated by these challenges, this study proposes a unified framework for aggregated Arabic search that integrates Retrieval-Augmented Generation, AraBERT embeddings, stacked autoencoders, and K-Means clustering into a single processing pipeline. The framework begins by collecting relational aggregated search results across multiple search verticals. The retrieved content is subsequently enriched using RAG to transform fragmented search snippets into coherent contextual representations. The generated text is then preprocessed and encoded with AraBERT, after which a stacked autoencoder learns compact semantic representations to reduce dimensionality. Finally, K-Means clustering is applied to organize the compressed representations into semantically coherent topic groups. Experimental evaluation on nine Arabic datasets covering education, sports, information technology, healthy food, and weather demonstrates that the proposed framework consistently outperforms raw aggregated search results and alternative approaches, resulting in significant improvements in clustering quality.

The remainder is organized as follows: Section II reviews related work. Section III outlines the challenges and summarizes our contributions. Section IV describes our methodology. Section V presents the experiment setup. Section VI presents results and discussion. Section VII presents the conclusion and future work.

II. Related Work

Natural Language Processing systems have traditionally relied on either retrieval-based or generative paradigms, each characterized by distinct strengths and inherent limitations. Retrieval-based approaches, including conventional search engines, efficiently identify relevant documents from large-scale collections but are limited to presenting existing information without synthesizing new knowledge [15]. In contrast, transformer-based generative models have substantially advanced natural language understanding and text generation, producing fluent and contextually coherent responses across a broad spectrum of applications [16]. Despite these capabilities, their reliance on parametric knowledge often leads to hallucinations and factual inconsistencies when external or recently acquired information is required.

Retrieval-Augmented Generation (RAG) was introduced to overcome these limitations by integrating information retrieval with neural text generation. Early retrieval-enhanced systems, such as DrQA, combined document retrieval with extractive answer generation, although their ability to synthesize information remained constrained [17]. Subsequent research demonstrated that retrieval and generation could be jointly optimized within a unified architecture. REALM demonstrated the feasibility of incorporating differentiable retrieval into language model pretraining [18], while the RAG framework proposed by Lewis et al. integrated dense passage retrieval with transformer-based generation, significantly improving factual grounding and contextual relevance in knowledge-intensive tasks [19].

Recent advances have focused primarily on strengthening the retrieval component of RAG systems. RAPTOR introduced hierarchical document organization to facilitate multi-level reasoning over complex knowledge sources [20]. Similarly, METRAG employed hierarchical summarization to improve the quality and informativeness of the contextual evidence provided to the language model [21]. Retrieval-aware fine-tuning strategies, such as RAFT, further enhanced model robustness by enabling language models to distinguish relevant evidence from distracting or irrelevant retrieved documents during training [22].

The application of RAG to Arabic Natural Language Processing has attracted growing research interest. Recent investigations have systematically evaluated chunking strategies, embedding models, reranking techniques, and large language models for Arabic retrieval tasks [23]. Their findings indicate that sentence-aware chunking, modern multilingual embedding models such as BGE-M3 and Multilingual-E5-large, and retrieval reranking consistently improve retrieval effectiveness and the factual reliability of generated responses.

Additional studies have examined semantic embedding models specifically for Arabic lexical retrieval [24]. Comparative evaluations involving AraBERT, CAMeLBERT, Multilingual-E5, and related models demonstrated that sentence-level semantic representations substantially outperform word-level embeddings. Furthermore, comparative assessments of generation models, including GPT-4, Gemini-1.5, Aya-8B, and AceGPT-13B, confirmed that generation quality remains an important factor influencing the overall performance of Arabic RAG systems.

Despite these advances, existing studies have primarily concentrated on improving retrieval accuracy and response generation, while comparatively little attention has been devoted to the semantic organization of heterogeneous aggregated search results. In particular, the integration of Retrieval-Augmented Generation with Arabic aggregated search across multiple search verticals, including web pages, images, news articles, and videos, remains largely unexplored. Furthermore, although previous studies have extensively evaluated embedding models for retrieval tasks, the use of RAG-enriched textual representations as input to a semantic clustering framework has not been investigated. The proposed framework addresses these research gaps by integrating Retrieval-Augmented Generation, AraBERT semantic embeddings, stacked autoencoders for nonlinear feature compression, and K-Means clustering into a unified framework for the semantic organization of aggregated Arabic search results.

III. Challenges and Contributions

The rapid expansion of Arabic digital content has intensified the demand for intelligent information retrieval systems capable of effectively discovering, integrating, and organizing information from diverse sources. Conventional search engines, such as Google and Bing, primarily employ keyword-based retrieval strategies.

Although effective for simple information needs, these approaches often fail to capture the underlying semantic intent of user queries. This limitation is particularly pronounced in Arabic because of its rich morphology, dialectal diversity, and orthographic variations, all of which complicate semantic matching and reduce retrieval effectiveness.

Aggregated search extends traditional retrieval by integrating results from multiple search verticals, including web pages, images, news articles, and videos, into a unified interface. Despite improved information accessibility, the aggregated output is inherently heterogeneous, often containing duplicate entries, irrelevant content, fragmented snippets, and insufficient context. These characteristics reduce the suitability of the retrieved content for downstream semantic analysis and clustering. Consequently, effective semantic enrichment and content refinement are essential before meaningful organization can be achieved.

Retrieval-Augmented Generation (RAG) provides an effective mechanism for enriching aggregated search results by integrating document retrieval with neural text generation. Instead of relying solely on fragmented search snippets, RAG retrieves the most relevant evidence from multiple search verticals and generates coherent, contextually grounded textual representations. By incorporating external knowledge during generation, RAG improves the relevance, completeness, and semantic consistency of the resulting content, thereby producing richer representations for subsequent analytical tasks.

Representing Arabic text remains challenging due to the language's complex morphological structure. Individual words may contain multiple prefixes and suffixes, while orthographic variations and diacritics can substantially alter lexical meaning. Conventional embedding models often struggle to preserve these linguistic characteristics, limiting their ability to capture semantic similarity accurately. AraBERT addresses these challenges through Arabic-specific pretraining and a customized WordPiece tokenization strategy designed to model Arabic's linguistic properties more effectively. Consequently, AraBERT generates contextualized semantic representations that better preserve lexical and semantic relationships, providing a robust foundation for downstream clustering.

The main steps of the proposed work can be summarized as follows:

- Collect relational aggregated search (RAS) results from multiple verticals, represented as text.
- Enrich the collected data using RAG to produce cleaner, more accurate datasets.
- Preprocess the RAG-generated output (remove diacritics and normalize characters).
- Generate 768-dimensional semantic vectors using AraBERT.
- Compress the embedding using a stacked autoencoder to extract the most informative features.
- Cluster the compressed features using K-Means to identify coherent topic groups.

The performance of the proposed framework is evaluated by comparing it with several alternative approaches:

- The author's previous work [25]: the same pipeline but without RAG (RAS + AraBERT + SAE + KM).
- RAG + AraBERT + PCA + KM: replaces SAE with PCA.
- RAG + AraBERT + KM: no feature extraction at all.
- RAG + Aravec + KM: replace AraBERT with Aravec.

The comparative analysis shows that the proposed method outperforms all of them.

IV. METHODOLOGY

The proposed framework comprises two principal modules: Data Collection (Fig. 1) and Data Enrichment (Fig. 2). The enrichment module consists of five sequential stages: Retrieval-Augmented Generation (RAG), text preprocessing, AraBERT-based semantic embedding, stacked autoencoder (SAE) feature extraction, and K-Means clustering. Unlike conventional aggregated search frameworks, the proposed approach introduces an intelligent semantic enrichment stage between the collection of Relational Aggregated Search (RAS) results and the clustering process. This intermediate stage employs RAG to improve the semantic quality of the retrieved information and utilizes a stacked autoencoder to learn compact latent representations that better preserve semantic relationships among search results.

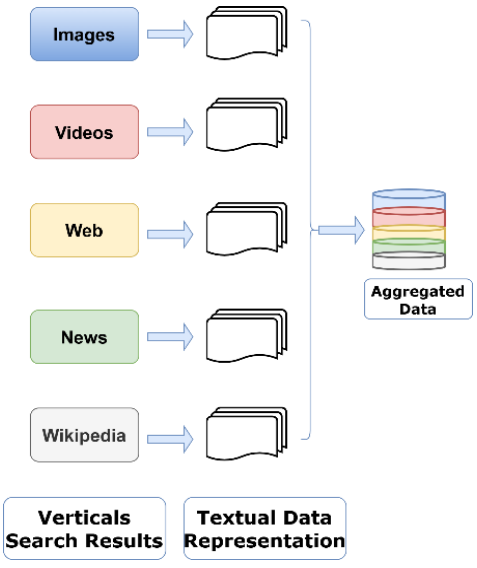

Fig. 1. Aggregated data from the aggregated search.

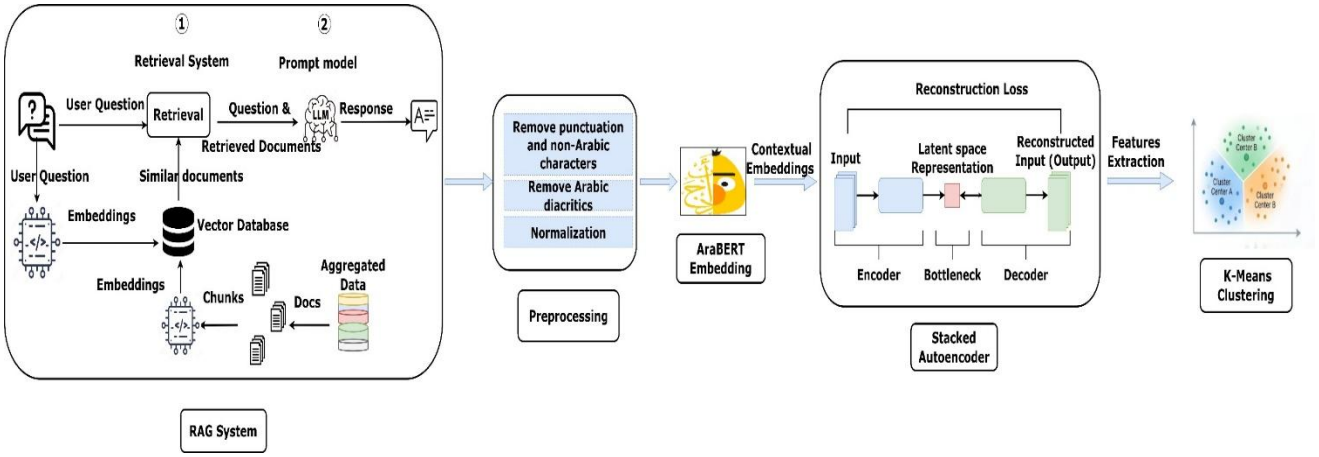

Fig. 2. The architecture of our proposed system comprises five steps: RAG system, pre-processing, results embedding (AraBERT), feature extraction (stacked autoencoder), and clustering algorithm (K-Means).

A. Aggregated Data Collection

Relational Aggregated Search (RAS) results are collected using publicly available application programming interfaces (APIs), including Google Search, Bing Search, YouTube Search, and Wikipedia. For each query, the top 100 results are retrieved from each search vertical, including web pages, images, news articles, and videos. Because each vertical returns information in a different format, all retrieved records are transformed into a unified textual representation consisting of the title, snippet, and URL. This standardized representation facilitates subsequent processing and semantic analysis. The complete data collection procedure is summarized in Algorithm 1. The resulting corpus serves as the input to the proposed data enrichment module.

Algorithm 1: Aggregated Data

Input: User Query Q
Output: Aggregated Data Text_Rep

Step 1: Query Verticals

```
Verticals = [ "web", "image", "video", "news"]
Text_Rep=""

FOR EACH V IN Verticals:

  Step 2: Retrieve the top 100 results
  results = SERP_API.search(engine=vertical, query=Q, num=100)

  Step 3: Process each result

  FOR EACH result IN results:
      // Extract core text components
      Title   = Doc.get ("title", "No Title")
      Snippet = Doc.get ("snippet", "")
      Link = Doc.get ("link", "No URL")

      // Skip results that have no usable text content
      IF Snippet IS EMPTY And Title IS EMPTY:
        CONTINUE

      Step 4: Build the text block
      Document_Block += "TITLE: " + Title + "\n"
      Document_Block += "CONTENT: " + Snippet + "\n"
      Document_Block += "Link: " + Link + "\n"
      Document_Block += "---" + "\n"

      // Append to the aggregate representation
      Text_Representation += Document_Block

  END FOR
END FOR

RETURN Text_Rep
```

B. Retrieval-Augmented Generation for Data Enrichment

The collected search results frequently contain duplicate entries, irrelevant information, fragmented snippets, and insufficient contextual information, all of which adversely affect clustering performance. To address these limitations, Retrieval-Augmented Generation (RAG) is employed as a semantic enrichment step prior to clustering. Rather than processing raw search snippets directly, the RAG framework retrieves the most relevant contextual information and generates coherent, contextually grounded textual representations. This enrichment process substantially reduces noise and redundancy while improving the completeness, consistency, and semantic relevance of the generated content, thereby providing higher-quality input for subsequent representation learning and clustering.

1) Building the vector database: To enable efficient semantic retrieval, the textual corpus is transformed into a vector database through the following stages, as illustrated in Fig. 3.

a) Chunking: The documents are segmented using a recursive text-splitting strategy that preserves the natural linguistic structure of the text by considering paragraph

boundaries, sentence boundaries, punctuation, and whitespace. This strategy generates semantically coherent text segments while maintaining contextual continuity.

b) Semantic embedding: Each text segment is encoded into a 384-dimensional dense vector using the sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2 model. This multilingual embedding model was selected because it provides effective semantic representations for Arabic while maintaining high computational efficiency, making it well suited for large-scale semantic retrieval.

c) Vector storage: The generated embeddings, together with their corresponding text segments, are stored in Chroma, an open-source vector database that supports persistent storage and efficient cosine similarity search. Chroma was selected for its lightweight architecture, local deployment capability, and seamless integration with the LangChain framework, enabling efficient retrieval without additional infrastructure.

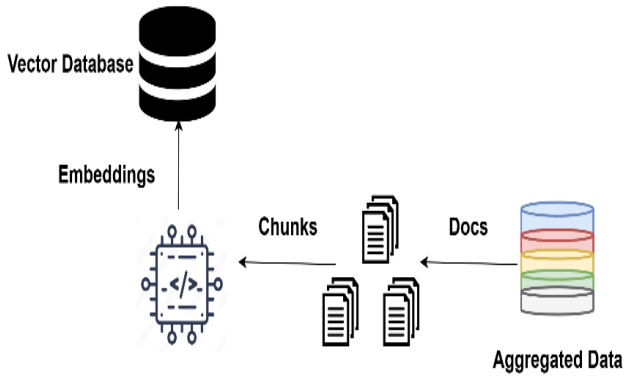

Fig. 3. Building a knowledge base.

2) Retrieval and generation workflow: For each Arabic user query, the Retrieval-Augmented Generation (RAG) pipeline performs five sequential stages, as illustrated in Fig. 4 and Algorithm 2.

a) Query representation: The input query is encoded using the same multilingual embedding model employed during vector database construction, thereby ensuring that queries and indexed documents reside within the same semantic embedding space.

b) Semantic retrieval: The generated query embedding is used to perform a similarity search within the Chroma vector database. The top ten semantically relevant text segments are retrieved based on cosine similarity.

c) Prompt construction: The retrieved text segments are concatenated to form a unified contextual representation, which is subsequently combined with the original query using a predefined prompt template. This enriched prompt provides the language model with the contextual evidence required to generate a grounded response.

d) Response generation: The constructed prompt is processed using the Google Gemini 1.5 Flash large language model to generate coherent, contextually grounded Arabic responses based on the retrieved evidence.

e) Dataset enrichment: Multiple semantically diverse responses are generated for each query and incorporated into the enriched dataset. These responses provide higher-quality textual representations for subsequent semantic embedding and clustering.

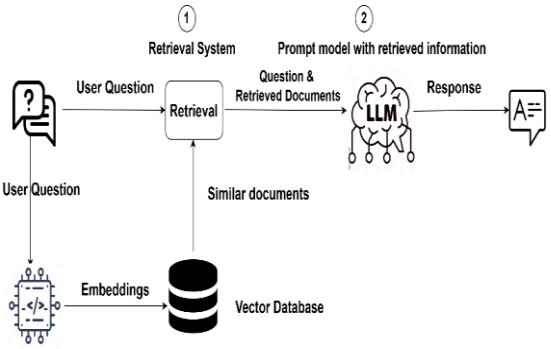

Fig. 4. RAG pipelines

Algorithm 2: RAG Framework

Input: User Question Q, Aggregated_Data
Output: Response
Step 1: Build Vector DB
FOR EACH document IN Aggregated_Data:
  Split document into chunks (C)
  Doc_Vecs = EMBEDDING_MODEL(C)
  Store (Doc_Vecs, C) in VectorDB
END FOR
Step 2: Similarity Ranking
Question_Vec = EMBEDDNG_MODEL(Q)
Scores = CALCULATE_COSINE_SIMILARITY(Question_Vec, Doc_Vecs)
Ranked_Docs = SORT (Doc_Vecs, BY = Scores, DESC)
Top_K_Context = Ranked_Docs[0:10]
Step 3: Prompt
Prompt_Template = "
You are an AI assistant for Arabic aggregated search.
Answer the user's question based only on the context provided.
Provide a single paragraph in Arabic. Do not add introductions, commentary, or information outside the context.
{Context}
Question:
{question}
Answer:
"
Final_Prompot = FILL (Prompt_Template, Context= Top_K_Context, question=Q)
Step 4: LLM Generation
Response = LLM_GENERATE(Final_Prompt)

3) Text preprocessing: The RAG-generated responses undergo a standardized preprocessing pipeline before semantic embedding. The preprocessing procedure includes:

- Removal of Arabic diacritics (Fatha, Damma, Kasra, Sukoon, and Shadda)
- Normalization of URLs, email addresses, and numerical expressions.

- Removal of Tatweel (elongation characters)
- Elimination of non-Arabic characters.
- Normalization of Alif variants (ا → آ، أ، إ)

Stop-word removal is not performed because AraBERT relies on function words (prepositions, conjunctions) to understand syntactic structure. The same preprocessing pipeline is applied to all documents to ensure consistent representation throughout the framework.

4) Semantic representation using AraBERT: Semantic representation is performed using the AraBERTv2-base model, a transformer-based language model pretrained specifically for Arabic [11]. Each preprocessed document is tokenized using the AraBERT WordPiece tokenizer. Documents exceeding the maximum sequence length are truncated, whereas shorter sequences are padded to a fixed length of 512 tokens.

The implementation is based on the Hugging Face Transformers library aubmindlab[1] using the aubmindlab/bert-base-arabertv02 pretrained model. Contextual document representations are obtained from the final hidden state of the [CLS] token, resulting in a 768-dimensional semantic embedding for each document. These embeddings capture the overall contextual meaning of the document and serve as input to the subsequent feature-extraction stage.

5) Stacked Autoencoder (SAE): To reduce the dimensionality of the AraBERT semantic embeddings, a stacked autoencoder (SAE) is employed as a nonlinear feature extraction model. The SAE learns a compact latent representation by encoding the original 768-dimensional embedding into a lower-dimensional feature space, then reconstructing the input from this compressed representation. Successful reconstruction indicates that the latent representation preserves the most informative semantic characteristics while eliminating redundant information.

As illustrated in Fig. 5, the proposed SAE architecture consists of three principal components:

- Encoder: Progressively transforms the high-dimensional input embeddings into a compact latent representation.
- Latent Space (Bottleneck): Represents the compressed semantic feature vector used for the subsequent clustering stage.
- Decoder: Reconstructs the original embedding from the latent representation.

Model training is performed by minimizing the reconstruction error between input embeddings and their reconstructions using the Mean Squared Error (MSE) loss. The Rectified Linear Unit (ReLU) activation function is adopted throughout the hidden layers because of its computational efficiency and its ability to alleviate the vanishing gradient problem. A linear activation function is employed in the output layer to reconstruct continuous embedding values. Network parameters are optimized through backpropagation until convergence.

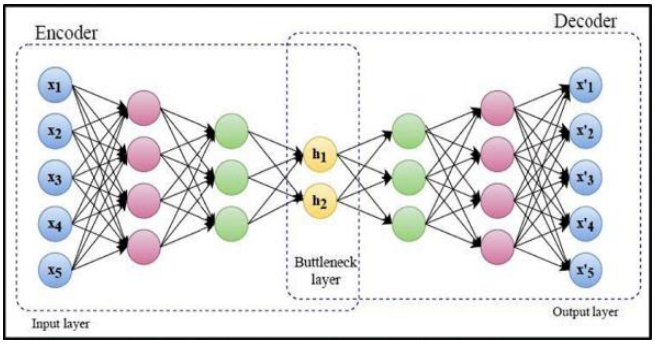

Fig. 5. Autoencoder architecture [26]

Because the experimental datasets differ in size, topic distribution, and semantic complexity, a single stacked autoencoder architecture was not suitable for all datasets. Therefore, several candidate architectures and hyperparameter configurations were evaluated empirically for each dataset. The final architecture was selected based on its validation performance, considering both the lowest reconstruction loss and the highest clustering quality, as measured by the Silhouette coefficient. This adaptive selection strategy balances feature compression with clustering performance while reducing the risk of underfitting or overfitting across datasets. The final configurations adopted in this study are summarized in Table I.

TABLE I. HYPERPARAMETERS FOR STACKED AUTOENCODER

Parameter	Optimized Value	Description
Input dimensions	768	AraBERT's dimensions
Hidden dims	(512, 256, 128, 64, 32, 16, 2) / (512, 256, 128, 64)	Hierarchical stages of feature compression.
Activation Function	ReLU	A non-linear function that is used between stacked layers.
Learning Rate (lr)	0.001	Step size for the Adam optimizer.
Batch Size	64 / 32	Number of samples processed per iteration.
Max_Epochs	100	Maximum training iterations

In the proposed architecture, for certain datasets, the encoder progressively reduces the original 768-dimensional AraBERT embeddings through successive hidden layers with 512, 256, 128, 64, 32, and 16 neurons before generating the final two-dimensional latent representation for clustering. This adaptive architecture enables effective feature compression while preserving the semantic structure required for accurate clustering.

During the decoding stage, the two-dimensional latent representation is progressively reconstructed through a sequence of hidden layers with 16, 32, 64, 128, 256, and 512 neurons. The final output layer restores the original 768-dimensional feature space, producing a reconstructed embedding corresponding to the input representation. The reconstruction quality is quantified

1 https://huggingface.co/aubmindlab/

by computing the Mean Squared Error (MSE) between the original and reconstructed embeddings, which serves as the optimization objective during training.

Model optimization is performed using the Adam optimizer with an initial learning rate of 0.001. Depending on the size and characteristics of each dataset, a batch size of either 32 or 64 is adopted. Training is conducted for up to 100 epochs, with an early stopping strategy employed to improve generalization and prevent overfitting. Specifically, training is terminated if the validation loss does not improve for 10 consecutive epochs, and the model parameters corresponding to the lowest validation loss are retained.

6) K-Means clustering: The compressed latent representations generated by the stacked autoencoder are subsequently partitioned into semantically coherent groups using K-Means clustering. For each dataset with T predefined target topics, the number of clusters is set to K = T + 1. The additional cluster is designated as "Other" and is intended to accommodate documents that cannot be reliably assigned to any predefined topic category, thereby improving the semantic purity and separation of the primary clusters. The K values adopted for each dataset are summarized in Table II.

TABLE II. DATASETS USED IN OUR EXPERIMENTS

Dataset	Target Topics	Dataset Size	K (clusters)	Clusters Label
D1	Education, Sports	383	3	Education, Sports, Other
D2	Education, IT	445	3	Education, IT, Other
D3	Sports, IT	458	3	Sports, IT, Other
D4	Education, Sports, IT	643	4	Education, Sports, IT, Other
D5	Weather, Healthy Food	439	3	Weather, Healthy Food, Other
D6	Sports, Healthy Food	426	3	Sports, Healthy Food, Other
D7	Healthy Food, Weather, Education	624	4	Healthy Food, Weather, Education, Other
D8	Sports, Healthy Food, Weather, Education	822	5	Sports, Healthy Food, Weather, Education, Other
D9	Education, Sports, Weather, IT	854	5	Education, Sports, Weather, IT, Other

K-Means was selected because the proposed stacked autoencoder transforms high-dimensional AraBERT embeddings into a compact latent feature space characterized by high intra-cluster cohesion and enhanced inter-cluster discrimination. Such representations are particularly well-suited to the centroid-based optimization strategy employed by K-Means, enabling effective partitioning of semantically similar documents while maintaining low computational complexity. Furthermore, because the semantic categories are known in advance, the value of K can be specified directly, without requiring additional cluster-estimation procedures. To improve the stability and reproducibility of the clustering results, K-Means was initialized using the scikit-learn library in Python, with a fixed random seed (random_state=42) to ensure reproducibility of the experimental results.

V. EXPERIMENTAL SETUP

This section describes the experimental configuration, dataset construction process, and evaluation criteria used to assess the effectiveness of the proposed framework. All experiments were implemented in Python using the PyTorch deep learning framework and executed on a 64-bit Windows 10 workstation equipped with an Intel Core i7 processor and 16 GB of RAM, as well as on Google Colab with GPU acceleration.

A. Data Collection and Dataset Construction

Relational aggregated search results were collected from eight search verticals using the Google Search APIs (Web, News, Images, and Videos) and Bing Search APIs (Web, News, Images, and Videos), using the SERP API [2] and the Wikipedia API. Five representative Arabic topics were selected for the experiments: التعليم (Education), الرياضة (Sports), تكنولوجيا المعلومات (Information Technology), الغذاء الصحي (Healthy Food), and الطقس (Weather). For each topic, the top 100 search results were retrieved from every search vertical.

To ensure consistency across heterogeneous sources, each retrieved record was transformed into a standardized textual representation comprising the document title, descriptive snippet, and source URL.

The collected search results were subsequently processed using the proposed Retrieval-Augmented Generation (RAG) framework. For each topic, multiple Arabic information-seeking questions were formulated to retrieve relevant contextual evidence and generate coherent Arabic paragraphs. These generated responses constituted the enriched corpus used throughout the experimental evaluation.

Nine experimental datasets, denoted D1–D9, were constructed by combining different pairs and groups of topic-specific documents to evaluate the proposed framework under varying levels of semantic diversity. The composition of each dataset is summarized in Table II.

B. Evaluation Metrics

The effectiveness of the proposed framework was evaluated using two widely adopted internal clustering validation measures: the Silhouette Coefficient [27] and the Davies–Bouldin Index (DBI) [28].

The Silhouette Coefficient quantifies both cluster cohesion and inter-cluster separation, producing values within the range [-1, 1]:

$$\text{Silhouette Score} = (b - a) / \max (a , b) \quad (1)$$

where, (a) denotes the average distance between a sample and all other samples within the same cluster, while (b) represents the average distance between that sample and observations belonging to neighboring clusters. Higher Silhouette values indicate stronger cluster cohesion and better

2 https://serpapi.com/

separation, whereas negative values suggest potential misclassification.

The Davies–Bouldin Index (DBI) evaluates clustering quality by jointly considering intra-cluster compactness and inter-cluster separation. It is computed as:

$$DB = \frac{1}{k}\sum_{i=1}^{k} \max\left(\frac{\Delta(X_i)+\Delta(X_j)}{\delta(X_i,X_j)}\right) \quad (2)$$

where, $(\Delta(X_i))$ represents the average intra-cluster distance within cluster (X_i), and $(\delta(X_i, X_j))$ denotes the distance between clusters (X_i) and (X_j). Lower DBI values indicate compact, well-separated clusters and therefore correspond to superior clustering performance.

VI. Results and Discussion

The proposed framework was evaluated using nine experimental datasets representing different topic combinations and varying levels of semantic complexity. The analysis was conducted in two stages. First, the overall clustering performance of the proposed framework was examined. Second, comparative experiments were performed to quantify the contribution of each major component.

A. Performance Analysis of the Proposed Framework

Table III summarizes the clustering performance of the proposed RAG + AraBERT + SAE + K-Means framework across all experimental datasets. The obtained Silhouette coefficients consistently exceed 0.65, indicating strong intra-cluster cohesion and clear inter-cluster separation. These results demonstrate that the proposed semantic representation effectively preserves topic-specific characteristics while maintaining sufficient discrimination between different semantic categories.

TABLE III. Clustering Evaluation of the Proposed Method

Dataset	Silhouette Score	Davies-Bouldin Index
D1	0.72	0.32
D2	0.70	0.42
D3	0.68	0.49
D4	0.68	0.38
D5	0.66	0.49
D6	0.70	0.45
D7	0.65	0.45
D8	0.66	0.42
D9	0.69	0.37

The proposed framework also exhibits stable performance as the semantic complexity of the datasets increases. Datasets D8 and D9, which comprise four distinct topics partitioned into five clusters, achieved Silhouette scores of 0.66 and 0.69, respectively. These findings indicate that the compressed latent representations generated by the stacked autoencoder remain highly discriminative even when multiple topics are simultaneously considered.

The Davies–Bouldin Index further supports these observations. Across all datasets, DBI values remain below 0.50, indicating compact cluster formation and substantial separation among neighboring clusters. Collectively, these results demonstrate that the proposed framework consistently produces well-defined semantic clusters under varying experimental conditions.

B. Comparative Evaluation

To evaluate the effectiveness of the proposed feature extraction strategy, the stacked autoencoder was compared with Principal Component Analysis (PCA), a widely adopted linear dimensionality reduction technique. Clustering performance was assessed using the Silhouette Coefficient and the Davies–Bouldin Index, where higher Silhouette values and lower DBI values indicate superior clustering quality.

As shown in Table IV, the proposed RAG + AraBERT + SAE + K-Means framework consistently outperforms the RAG + AraBERT + PCA + K-Means configuration across all evaluation metrics. The nonlinear feature-learning capability of the stacked autoencoder preserves complex semantic relationships that cannot be adequately captured by linear projection, resulting in more compact and better-separated clusters.

TABLE IV. Comparison of Alternative Methods in Terms of Silhouette Score and Davies-Bouldin Index.

Method	Avg. Silhouette	Avg. DBI
RAG + AraVec + KM	0.15	2.12
RAG + AraBERT + KM	0.18	1.99
RAG + AraBERT + PCA + KM	0.44	0.99
RAG + AraBERT + SAE + KM (Ours)	**0.68**	**0.42**

The contribution of nonlinear feature compression is further illustrated by comparing AraBERT embeddings before and after dimensionality reduction. Without the stacked autoencoder, AraBERT achieved a Silhouette score of 0.18, representing a modest improvement over AraVec (0.15). Incorporating the stacked autoencoder increased the average Silhouette coefficient to 0.68, demonstrating that nonlinear representation learning substantially enhances the discriminative structure of the embedding space before clustering.

The effectiveness of the proposed Retrieval-Augmented Generation (RAG) stage was further evaluated by comparing the current framework with the authors' previous approach (baseline), Aggregated + AraBERT + SAE + K-Means [25], which operated directly on raw aggregated search results without semantic enrichment. The comparison presented in Table V demonstrates that introducing RAG consistently improves clustering quality across all nine datasets. The average Silhouette coefficient increased from 0.59 to 0.68, corresponding to an improvement of approximately 15%.

The performance gain is particularly evident for datasets containing semantically related topics, such as D2, where lexical overlap increases the difficulty of cluster separation. By transforming fragmented search snippets into coherent, context-aware representations, the RAG module reduces semantic ambiguity and strengthens topic-specific representations prior to embedding. Consequently, documents belonging to the same

semantic category become more tightly grouped, while the separation between different topic categories increases. This behavior is maintained even for the most challenging datasets containing three or more topic categories (e.g., D4, D7, D8, and D9), demonstrating the robustness of the proposed framework as semantic complexity increases.

TABLE V. COMPARISON OF SILHOUETTE SCORES OF K-MEANS WITH OUR PREVIOUS METHOD.

Dataset	RAS + AraBERT + SAE + KM (baseline)	RAG + AraBERT + SAE + KM (proposed)
D1	0.67	**0.72**
D2	0.59	**0.70**
D3	0.62	**0.68**
D4	0.59	**0.68**
D5	0.63	**0.66**
D6	0.65	**0.70**
D7	0.55	**0.65**
D8	0.54	**0.66**
D9	0.55	**0.69**

These improvements can be attributed to the semantic enrichment performed by the RAG module. Rather than relying directly on heterogeneous aggregated search snippets, the framework generates coherent, contextually grounded textual representations that provide richer semantic information for subsequent embedding and clustering. The resulting representations contain less redundancy, reduced contextual ambiguity, and improved semantic consistency, thereby enabling the downstream feature extraction and clustering stages to operate on higher-quality input data.

To determine whether the observed performance gains were statistically significant, both the paired t-test and the Wilcoxon signed-rank test were performed using the Silhouette coefficients obtained across the nine datasets. The proposed framework achieved an average Silhouette coefficient of 0.682 ± 0.023, compared with 0.599 ± 0.047 for the baseline approach. The paired t-test confirmed that the improvement was statistically significant ($t = 6.68$, $p = 0.00016$), while the Wilcoxon signed-rank test produced a similarly significant result ($p = 0.0039$).

Hypothesis testing was subsequently conducted using a significance level of $\alpha = 0.05$. The null hypothesis (H_0) assumed that no statistically significant difference exists between the baseline aggregated search framework and the proposed RAG-enhanced framework. The alternative hypothesis (H_1) stated that the proposed framework yields superior clustering performance. Because the p-values obtained from both statistical tests were substantially lower than the predefined significance level, the null hypothesis was rejected in favor of the alternative hypothesis. These findings provide strong statistical evidence that the improvements achieved by the proposed framework are unlikely to have occurred by chance and instead result from integrating Retrieval-Augmented Generation with semantic representation learning.

VII. CONCLUSION AND FUTURE WORK

This study presented an intelligent framework for enhancing Arabic aggregated search by integrating Retrieval-Augmented Generation (RAG), contextual semantic representation, nonlinear feature learning, and unsupervised clustering. By enriching heterogeneous search results before semantic analysis, the proposed framework effectively reduced contextual ambiguity and redundancy while improving the semantic quality of the retrieved information. The enriched representations were subsequently encoded using AraBERT, compressed through a stacked autoencoder, and organized into semantically coherent topic clusters using the K-Means algorithm. Experimental evaluation on nine Arabic datasets spanning five application domains demonstrated the effectiveness and robustness of the proposed framework. Compared with clustering based on raw aggregated search results, the proposed approach increased the average Silhouette coefficient from 0.59 to 0.68, corresponding to an improvement of approximately 15%. The framework consistently achieved superior clustering performance across all experimental datasets and outperformed alternative dimensionality reduction and embedding approaches, including Principal Component Analysis (PCA), direct clustering of AraBERT embeddings, and AraVec-based representations. Statistical analysis further confirmed that these improvements were significant, demonstrating the effectiveness of integrating Retrieval-Augmented Generation with semantic representation learning for Arabic aggregated search.

Future research will investigate more advanced Retrieval-Augmented Generation architectures, including Agentic RAG, Corrective RAG (CRAG), Modular RAG, and Graph RAG, to further improve retrieval accuracy, reasoning capability, and factual consistency. Additional research directions include evaluating stronger multilingual retrieval models, incorporating retrieval reranking mechanisms, and extending the proposed framework to larger and more diverse Arabic corpora to assess its scalability and generalization across broader information retrieval applications.

REFERENCES

[1] A. Kopliku, K. Pinel-Sauvagnat, and M. Boughanem, “Aggregated search: A new information retrieval paradigm," ACM Comput. Surv., vol. 46, no. 3, pp. 1-31, 2014.

[2] S. Achsas, “Improving relational aggregated search from big data sources using stacked autoencoders,” Cognit. Syst. Res., vol. 51, pp. 61–71, 2018.

[3] Q. Luo, Q. Shao, Z. Zhang, B. Zhang, and J. Shi, “SCAT_LoRA: self-consistency adversarial training in large language models through low-rank adaptations,” Eng. Res. Express, vol. 7, no. 2, p. 025265, 2025.

[4] A. Mishra and A. Gupta, “Retrieval Augmented Generation (RAG) Model,” Int. J. Res., vol. 6, no. 6, pp. 4690–4693, 2025.

[5] N. Kandpal, H. Deng, A. Roberts, E. Wallace, and C. Raffel, “Large language models struggle to learn long-tail knowledge,” in Proc. Int. Conf. Mach. Learn., pp. 15696–15707, 2023.

[6] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, “Retrieval-augmented generation for large language models: A survey,” arXiv preprint arXiv:2312.10997, 2023.

[7] G. P. Wellawatte, H. Guo, M. Lederbauer, A. Borisova, M. Hart, M. Brucka, and P. Schwaller, “ChemLit-QA: a human evaluated dataset for chemistry RAG tasks,” Mach. Learn.: Sci. Technol., vol. 6, no. 2, p. 020601, 2025.

[8] I. O. William and M. Altamimi, “Text Embedding Implementation Using Retrieval Augmented Generation (RAG) Model Combined With Large Language Model,” Int. J. Adv. Nat. Sci. Eng. Res., vol. 5, no. 5, 2024.

[9] A. Kimothi, "A Simple Guide to Retrieval Augmented Generation," Manning Publications, 2025.

[10] B. Wang, A. Wang, F. Chen, Y. Wang, and C. C. J. Kuo, "Evaluating word embedding models: Methods and experimental results," APSIPA Trans. Signal Inf. Process., vol. 8, p. e19, 2019.

[11] W. Antoun, F. Baly, and H. Hajj, "Arabert: Transformer-based model for arabic language understanding," in Proc. 4th Workshop Open-Source Arabic Corpora Process. Tools, pp. 9–15, 2020.

[12] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P. A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," J. Mach. Learn. Res., vol. 11, no. 12, pp. 3371–3408, 2010.

[13] A. K. Jain, "Data clustering: 50 years beyond K-means," Pattern Recognit. Lett., vol. 31, no. 8, pp. 651–666, 2010.

[14] R. Xu and D. Wunsch, "Survey of clustering algorithms," IEEE Trans. Neural Networks, vol. 16, no. 3, pp. 645–678, 2005.

[15] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," Commun. ACM, vol. 18, no. 11, pp. 613–620, 1975.

[16] 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, vol. 30, 2017.

[17] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," in Proc. 55th Annu. Meeting Assoc. Comput. Linguistics (Vol. 1: Long Papers), pp. 1870–1879, 2017.

[18] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "Retrieval augmented language model pre-training," in Proc. Int. Conf. Mach. Learn., pp. 3929–3938, 2020.

[19] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschle, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.

[20] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, "Raptor: Recursive abstractive processing for tree-organized retrieval," in Proc. 12th Int. Conf. Learn. Represent., 2024.

[21] C. Gan, D. Yang, B. Hu, H. Zhang, S. Li, Z. Liu, C. Wang, and J. Zhou, "Similarity is not all you need: Endowing retrieval augmented generation with multi layered thoughts," arXiv preprint arXiv:2405.19893, 2024.

[22] T. Zhang, S. G. Patil, N. Jain, S. Shen, M. Zaharia, I. Stoica, and J. E. Gonzalez, "Raft: Adapting language model to domain specific rag," arXiv preprint arXiv:2403.10131, 2024.

[23] J. Alsubhi, M. D. Alahmadi, A. Alhusayni, I. Aldailami, I. Hamdine, A. Shabana, L. Al-Malki, and S. Khayyat, "Optimizing rag pipelines for arabic: A systematic analysis of core components," arXiv preprint arXiv:2506.06339, 2025.

[24] R. Al-Rasheed, A. Al Muaddi, H. Aljasim, R. Al-Matham, M. Alhoshan, A. Al Wazrah, and A. AlOsaimy, "Evaluating RAG Pipelines for Arabic Lexical Information Retrieval: A Comparative Study of Embedding and Generation Models," in Proc. 1st Workshop NLP Lang. Using Arabic Script, pp. 155–164, 2025.

[25] S. S. Soliman, "Deep Learning-Based Approach for Improving Relational Aggregated Search," arXiv preprint arXiv:2510.00966, 2025.

[26] S. Hosseini and Z. A. Varzaneh, "Deep text clustering using stacked AutoEncoder," Multimedia Tools Appl., vol. 81, no. 8, pp. 10861–10881, 2022.

[27] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," J. Comput. Appl. Math., vol. 20, pp. 53–65, 1987.

[28] D. L. Davies and D. W. Bouldin, "A cluster separation measure," IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.