Hybrid Layered Retrieval and Task-aware Embeddings for Efficient Persian RAG Systems

Toktam Zoughi[1,*], Ehsan Arianyan[2,*], Maryam Mahmoudi[2], Mahtab Aghdamifard[1] and Mohadese Nikoogoftar[1]

[1]*Department of Computer Engineering, Technical and Vocational University (TVU), Tehran, Iran*
[2]*Information Technology Faculty, ICT Research Institute, Tehran, Iran*
E-mail: t.zoughi@shariaty.ac.ir; ehsan_arianyan@itrc.ac.ir; mahmoudy@itrc.ac.ir; Aghdamifard.mahtab@gmail.com; nikomohdse@gmail.com
[*]*Corresponding Authors*

Received 30 October 2025; Accepted 10 December 2025

Abstract

This study introduces task-injected layered hybrid retrieval-augmented generation (TILHR-RAG), a framework specifically designed for Persian to address the scarcity of native-language resources and the limitations of English-centric approaches. The architecture combines task-aware query augmentation, a layered retrieval strategy, and a hybrid semantic–lexical retriever, all supported by a multi-stage pipeline that includes preprocessing, document chunking, question generation, and embedding. A novel mechanism for injecting task-specific vectors directs retrieval toward domain intent while preserving comparability across documents. The layered design operates in three stages: per-task frequently asked questions (FAQ) retrieval, hybrid document search using FAISS semantic similarity combined with BM25 keyword matching, and a fallback response generated by a large language model (LLM). This structure ensures both precision and robustness. Comprehensive experiments across five progressively refined configurations

Journal of Web Engineering, Vol. 25_5, 765–796.
doi: 10.13052/jwe1540-9589.2552

demonstrate that TILHR-RAG achieves the best balance among accuracy, efficiency, and scalability, reaching 89.67% semantic accuracy with moderate latency and memory consumption on NVIDIA A100 hardware. Further evaluations on low-resource graphics processing units (GPUs) confirm that accuracy remains stable under hardware constraints, although latency increases significantly. Moreover, multilingual E5 embedding models substantially improve retrieval and generation quality for Persian – outperforming ParsBERT and Sentence-BERT (SBERT) – by mitigating challenges such as orthographic variation and complex compound word structures. Taken together, these findings establish task-injected layered hybrid retrieval-augmented generation as a practical, reproducible, and resource-efficient blueprint for Persian question answering, advancing retrieval-augmented generation for low-resource languages without requiring costly large language model fine-tuning, while also offering adaptable strategies for broader multilingual applications.

Keywords: Retrieval-augmented generation (RAG), natural language processing, task-specific embeddings, hybrid retrieval, low-resource languages, layered retrieval architecture.

1 Introduction

In the era of data explosion, the vast volume of textual information generated across scientific, industrial, and social domains necessitates intelligent systems capable of precise processing and knowledge extraction. Large language models (LLMs), empowered by deep learning and massive datasets, have significantly advanced tasks such as text generation, question answering, and semantic analysis [1, 2]. However, when operating independently, LLMs remain constrained by their static internal knowledge, especially when dealing with domain-specific, up-to-date, or localized information. Retrieval-augmented generation (RAG) architectures have emerged to address these limitations by integrating information retrieval into the generative pipeline [3, 4]. RAG enables a language model to first retrieve relevant content from an external corpus and then generate answers grounded in retrieved evidence. This hybrid design improves accuracy, verifiability, and the ability to incorporate evolving knowledge [5, 6].

A RAG system typically begins by converting a user query into a vector representation in a high-dimensional semantic space. Using similarity metrics such as dense vector matching, the system retrieves the most semantically

relevant textual chunks [7, 8]. These documents are then preprocessed – through normalization, noise reduction, and smart trimming – and injected into the generative model as context for more precise responses. Unlike traditional models that rely solely on pre-trained knowledge, RAG systems base their answers on actual documents, making them especially effective for low-resource languages and specialized domains. In this context, Persian presents unique challenges due to its rich syntactic structure, semantic diversity, and limited computational resources.

This study proposes the design and implementation of a native Persian-language RAG-based chatbot system. The project began by collecting Persian textual data from credible scientific sources, followed by a thorough preprocessing phase including normalization, punctuation correction, and structural unification. Cleaned texts were split into semantically meaningful chunks, each of which was used to generate focused questions through the LLaMA model with tailored prompts. These questions were manually and automatically evaluated for quality. Next, embedding models compatible with Persian were used to encode both questions and chunks into dense vectors. Semantic similarity was measured, and tools such as t-SNE and heatmaps were applied to visualize and assess alignment quality. This multi-stage methodology demonstrated that response quality and retrieval precision are closely tied to preprocessing fidelity and prompt design. Overall, this research integrates computational linguistics, prompt engineering, and semantic retrieval to provide a viable foundation for building intelligent Persian-language systems. Beyond its academic contribution, the resulting infrastructure holds potential for developing chatbots, question-answering systems, and other AI tools in Persian.

The remainder of this paper is organized as follows. Section 2 reviews related work on retrieval-augmented generation and highlights the challenges of adapting such methods to Persian. This section also introduces the problem statement, defining the key gaps in current approaches and clarifying the specific objectives addressed by this study. Section 3 presents the proposed TILHR-RAG framework in detail, covering preprocessing, task classification, layered retrieval, hybrid scoring, and task-specific vector injection. Section 4 describes the experimental setup, datasets, evaluation metrics, and implementation details, and further reports and discusses the results, comparing multiple retrieval configurations and embedding backbones while analyzing performance across diverse query categories and hardware settings. Finally, Section 5 concludes the paper by summarizing the main findings, outlining limitations, and suggesting directions for future research.

2 Related Work

This section presents a review and synthesis of recent research on retrieval-augmented generation (RAG) architectures and text preprocessing methods, with a focus on their applicability to low-resource languages such as Persian and Arabic. The goal is to highlight advancements in RAG systems, analyze practical challenges, and assess the strengths and limitations of current natural language processing (NLP) approaches. Several foundational studies explore the deployment of RAG-based organizational question-answering systems [9–11]. The studies emphasize the shortcomings of traditional language models in high-stakes domains such as academia, where factual accuracy and traceability are essential. To address issues like hallucinations, the researchers propose combining LLMs' reasoning capabilities with the reliability of document-grounded retrieval. The systems use modular pipelines that preprocess organizational content – such as websites and FAQs – into small chunks, embed them using OpenAI's embedding models, and store them in a vector database [12–14]. During inference, relevant texts are retrieved and passed to the LLM for answer generation. Evaluation using both automated and human methods confirms that RAG-based answers are significantly more accurate and verifiable than those from standalone models. The studies also underscore the importance of modular design, transparency, ethical deployment, and mechanisms for feedback and source citation [15].

Other studies evaluate semantic search for Arabic RAG systems, addressing challenges posed by the language's complex morphology and limited structured resources. Using customer support transcripts and tailored queries, the researchers test multilingual and Arabic-specific encoders. Results demonstrate that advanced embedding models and deep learning architectures significantly improve semantic retrieval accuracy. Integration of these models into RAG pipelines enhances the quality of generated responses and highlights the need for custom benchmarks and embedding models for low-resource languages [16–18].

The other studies document the development of a RAG-powered virtual assistant for the University of São Paulo. The system's modular architecture supports independent retrieval and generation components, enables flexible updates without system downtime. The knowledge base includes official documents and institutional guidelines, and systems' performance is evaluated based on retrieval strategies, chunk size, and result integration methods [19, 20]. Challenges include misalignment between multilingual embeddings and semantic content, which the authors address through data diversification

and improved QA pair construction. Further researches introduce TextING, a novel approach for semantic embedding generation using graph neural networks (GNNs). By representing documents as semantic graphs, this method captures complex relationships between terms, even with unseen or domain-specific vocabulary. Evaluations show that TextING outperforms traditional methods, especially in generalization to new data [21, 22].

Complementary works focus on the evaluation of RAG systems via realistic datasets. Researchers criticize reliance on generic QA datasets, which often introduce imbalance and reduce accuracy. They propose classifications of question-context pairs – direct answer, summarization, inference, and unanswerable – to guide dataset creation aligned with system use cases [23, 24]. Additional methods such as complex prompt design, multi-step question generation with LLMs, and fine-tuning smaller models are recommended for improving data diversity and balance. Experimental results indicate that tailoring retrieval and generation strategies to specific question types substantially enhances overall system performance [18]. In conclusion, successful RAG architectures depend on the precise integration of retrieval and generation mechanisms, language-aware design, and access to high-quality datasets. For low-resource languages like Persian and Arabic, building native infrastructures, training specialized encoders, and developing localized benchmarks are crucial steps toward creating scalable, reliable, and accurate information systems suitable for organizational, academic, and commercial environments [25, 26].

2.1 Problem Statement

Persian-language institutional question answering (QA) presents a unique set of challenges that conventional English-centric retrieval and generation pipelines fail to address. The Persian language is characterized by rich morphology, frequent orthographic variation, compound word structures, and optional diacritics, all of which undermine the reliability of off-the-shelf tokenizers, embedding models, and lexical matchers. Moreover, the scarcity of high-quality, task-oriented retrieval-augmented generation (RAG) datasets and the limited availability of Persian-adapted retrieval components further complicate system development. In real-world production environments – such as research institutes and public administrations – precision is critical: systems must deliver evidence-grounded answers from heterogeneous document collections and curated FAQs. At the same time, practical deployments must achieve a balance between accuracy and efficiency, given

hardware constraints such as GPU memory and latency, while ensuring full reproducibility for ongoing maintenance and auditing.

Where q_i is the input query, A the set of candidate answers, and $\widehat{a_i}$ the selected response. Here, a denotes an individual candidate drawn from A, which may correspond to an FAQ entry, a document segment, or an LLM-generated output. The three scoring functions represent task-awaure alignment (S_{task}), semantic similarity (S_{FAISS}), and lexical matching (S_{BM25}), with tunable weights $\lambda_1, \lambda_2, \lambda_3$ balancing their contributions. This formulation encapsulates the paper's central objective: to design a system that integrates task-awareness, semantic embeddings, and lexical evidence in order to achieve reliable, scalable, and reproducible retrieval for Persian QA.

Accordingly, the work addresses four operational research questions: (a) How can compact, per-task synonym sets be used to construct task vectors that improve retrieval precision without degrading cross-document comparability? (b) What is an effective method for combining semantic nearest-neighbor scores and BM25 lexical scores to ensure robustness across heterogeneous Persian texts? (c) Which embedding backbones and large language model (LLM) back-ends best balance semantic accuracy, latency, and memory usage for deployment under real-world constraints? (d) How should indexing strategies (exact vs. IVF approximate) and batching methods be configured to scale to large corpora while maintaining reproducibility?

3 Methodology

The proposed methodology is designed to address the unique challenges of Persian text processing within a retrieval-augmented generation (RAG) framework by combining systematic preprocessing, structured retrieval pipelines, and task-aware query augmentation. Recognizing the decisive impact of data quality on RAG system performance, we developed a fully automated, end-to-end data processing framework that transforms heterogeneous Persian-language inputs into semantically coherent, analysis-ready datasets while ensuring semantic integrity, privacy compliance, and domain-specific normalization. The framework consists of four integrated modules. First, the universal file handling module ingests diverse input types, including individual files, directory structures, and compressed archives, applying filename sanitization, consistent renaming rules, and automated extraction routines to standardize the input layer. Second, the multimodal text extraction with OCR combines direct text parsing for formats such as PDF, DOCX, and TXT with optical character recognition for scanned or image-based

documents, while specialized routines capture textual content from embedded graphics such as SmartArt diagrams. Third, the Persian-specific text preprocessing core implements linguistic normalization tailored to Persian script, including tokenization, diacritic removal, elimination of markers, and paragraph reconstruction, thereby improving token-level accuracy and semantic fidelity. Fourth, the post-processing and chunk segmentation module removes redundant metadata and applies recursive splitting with controlled overlap to segment the text into semantically continuous chunks, each assigned a unique identifier for precise traceability. This framework outputs clean, structured Markdown or TXT files optimized for LLM-based RAG pipelines and is adaptable to other right-to-left scripts.

Following preprocessing, the pipeline applies a question–answer generation module to enrich the dataset, ensuring comprehensive coverage of semantic variations across tasks. For retrieval, we employ a layered architecture that progressively integrates FAQ matching, document-level semantic search, and fallback general reasoning, maximizing coverage and precision. Task-specific vectors are injected into query embeddings to guide the model toward domain-sensitive interpretations without sacrificing generalizability. Finally, a hybrid retriever that combines weighted semantic similarity with BM25 lexical scoring balances contextual alignment and keyword sensitivity, particularly valuable for heterogeneous Persian corpora. This methodology ensures reproducibility through deterministic parameterization, modular design, and standardized evaluation protocols, laying the foundation for scalable and accurate Persian RAG systems. In the following sections, we will describe each stage of this pipeline in detail, providing a step-by-step explanation of the processing framework and its components.

3.1 System Architecture and Processing Pipeline

The system architecture (Figure 1) establishes the overall design logic, component organization, and resource management strategies, ensuring scalability, stability, and adaptability. In the context of large language model-based NLP systems, this modular architecture enables independent development, testing, and updating of individual components – such as preprocessing, retrieval, embedding, and content generation – without requiring extensive reengineering. Such decoupling is particularly vital for resource-limited languages like Persian, where gradual improvements in accuracy, robustness, and response quality depend on efficient coordination between retrieval, generation, and knowledge base modules.

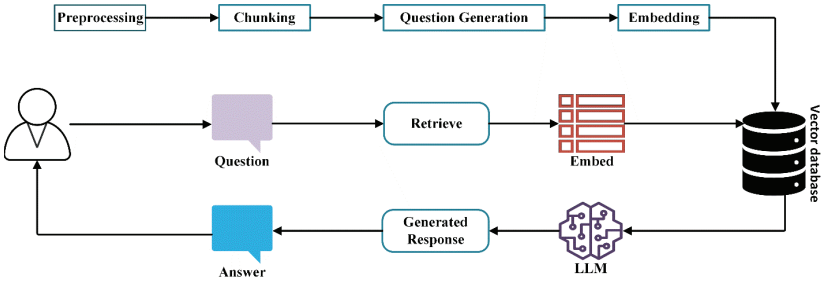


Figure 1 Architectural overview of the RAG model.

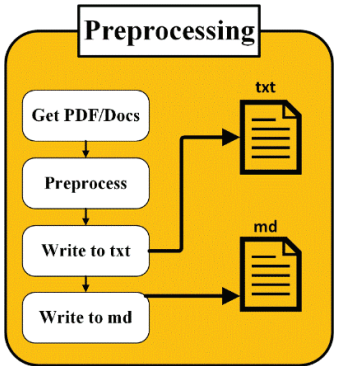


Figure 2 Block diagram of the preprocessing process.

As illustrated in Figure 2, the processing pipeline begins with preprocessing, where raw texts are extracted, cleaned, and standardized into structured formats (e.g., Markdown) for downstream tasks. The cleaned data are stored in a dedicated *CORPUS* directory to ensure orderly data flow and traceability.

Subsequent content chunking (Figure 3) divides large documents into semantically coherent segments using a recursive splitting algorithm with controlled overlap, producing CSV files (*chunks.csv*) stored in a directory.

As Figure 4 shows, from these chunks, the question generation module produces topic-explicit questions that serve as compressed semantic keys, enhancing retrieval performance by enriching the dataset with conceptual metadata.

Figure 5, represents in the embedding stage, both chunks and generated questions are transformed into high-dimensional semantic vectors using ParsBERT and multilingual-e5-base models, and stored in a FAISS vector database.

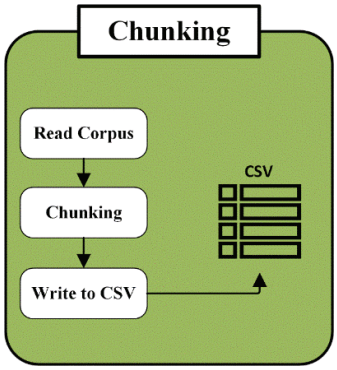


Figure 3 Content segmentation block diagram.

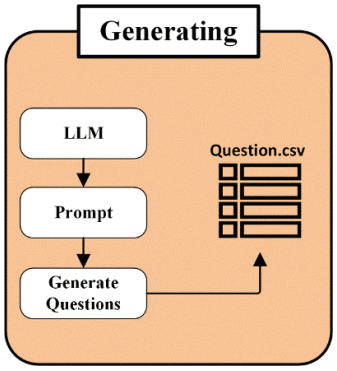


Figure 4 Question generation block diagram.

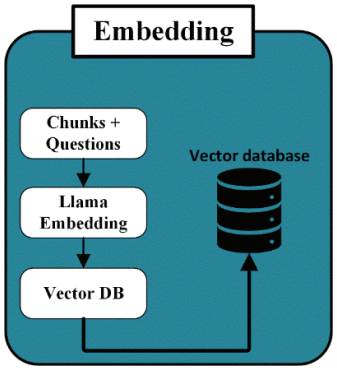


Figure 5 Block diagram of the embedding of chunks.

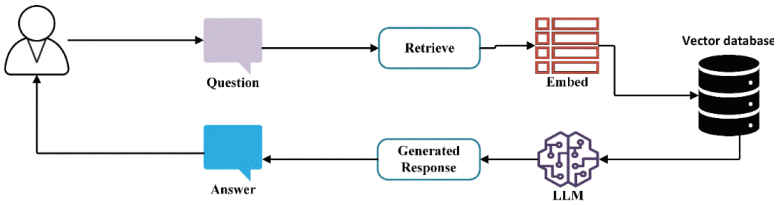


Figure 6 End-to-end retrieval-augmented generation pipeline linking query retrieval with LLM-based response generation.

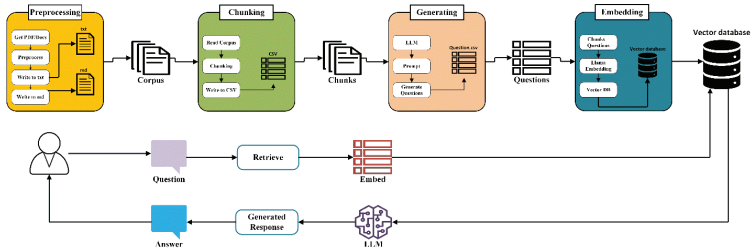


Figure 7 Workflow of the proposed Persian RAG system from preprocessing to LLM-based answer generation.

During retrieval (Figure 6), user queries are converted into semantic vectors and compared against the database using similarity metrics such as cosine similarity, returning a ranked list of relevant content segments.

The integrated system architecture (Figure 7) ensures efficient coordination between the retriever, the language model, and the document store, enabling accurate, scalable, and resource-efficient Persian-language RAG applications. The implementation uses Python 3.12, GPU acceleration (NVIDIA A100 and NVIDIA Geforce GTX 1080), and specialized libraries including *pandas*, *langchain*, *transformers*, *hazm*, *PyMuPDF*, *python-docx*, and *pytesseract*.

3.2 Automated Question and Answer Generation for Dataset Enrichment

To populate the RAG system with rich, high-quality Persian-language datasets, a dual-stage automated question generation process was implemented, directly integrated with the preprocessing and chunking pipeline.

Stage 1. Chunk-based question generation: Each chunk is processed to generate topic-aware, self-contained questions that explicitly reference the subject of the chunk. This eliminates referential ambiguity (e.g., replacing

"this service" with a specific term) and aligns query semantics with stored content. The resulting question–chunk pairs serve as supervised training data, enabling fine-grained mapping between queries and relevant segments.

Stage 2. Question–answer pair creation: Using the same source material, explicit QA pairs are generated with answers that are unambiguous, self-contained, and strictly derived from the original text. This evaluation dataset is used to measure retrieval and reasoning performance under realistic conditions.

The novelty of this stage lies in its alignment of training and evaluation domains, its topic-explicit query formulation, and its ability to produce large-scale, semantically rich datasets without manual annotation. Its modularity allows adaptation to other low-resource or right-to-left languages with minimal modifications.

This integrated methodology – spanning architecture design, automated preprocessing, intelligent segmentation, and dual-mode data generation – provides a scalable and resource-efficient foundation for Persian-language RAG systems. The combination of multimodal ingestion, linguistically informed processing, semantic enrichment, and high-performance retrieval ensures that the system can deliver accurate, verifiable, and contextually relevant responses, even in resource-constrained linguistic environments.

3.3 Three-layer Retrieval Architecture for Persian RAG Systems

The proposed system implements a three-layer retrieval-augmented generation (RAG) architecture tailored for Persian institutional documents and FAQ collections. The design ensures both high precision and robust fallback behavior by combining task-aware semantic embeddings, task-specific FAISS indices, BM25 lexical retrieval, and a general-purpose large language model (LLM). Scalability is achieved through FAISS inverted-file (IVF) indexing for large corpora, while computational efficiency is maintained using batched embedding, in-place L2 normalization, and runtime profiling of semantic evaluation.

Incoming queries are first subjected to lightweight preprocessing – including punctuation removal and whitespace normalization – to reduce surface-level variability. A task classification module, based on FAISS indices constructed from curated Persian synonym sets, determines whether a query belongs to a specific domain. Specifically, five task categories were defined, each supported by a manually curated list of synonymous expressions to capture orthographic and lexical variations in Persian: Executive Guidelines

(e.g., "regulations," "laws," "directives"), Postdoctoral Support (e.g., "postdoctoral," "postdoctoral stage"), Thesis Support (e.g., "thesis," "dissertation," "research manuscript"), Event Support (e.g., "conference," "workshop," "festival"), and Study Opportunity (e.g., "study opportunity," "scientific travel"). These synonym sets, stored as FAISS indices, enable the system to robustly classify queries despite orthographic inconsistencies or informal variations. If the top similarity score surpasses a predefined threshold, the query is routed to the corresponding task's FAQ entries; otherwise, it is processed against the entire FAQ collection. A margin-based disambiguation rule ensures robustness when scores between candidate tasks are close.

The retrieval process operates through three ordered layers:

Layer 1. Task-specific FAQ retrieval. Queries confidently classified into a task are searched against its dedicated FAISS FAQ index. If the similarity score exceeds the acceptance threshold, the retrieved FAQ entry is forwarded to the QA chain for grounded LLM generation.

Layer 2. Hybrid document retrieval. Queries unresolved at the FAQ layer are processed by a hybrid retriever that integrates FAISS-based semantic search with BM25 lexical ranking. Normalized scores are linearly combined (Equation (4)) to balance contextual alignment with keyword-level coverage, which is particularly valuable for heterogeneous Persian corpora.

Layer 3. General LLM fallback. If neither the FAQ nor the hybrid retriever produces a confident match, a persona-guided LLM generates a safe, general response.

Routing decisions and similarity scores are logged for ablation studies and reproducibility. This hierarchical structure guarantees rapid FAQ-based resolution when available, while hybrid retrieval and fallback LLM generation ensure resilience in ambiguous or out-of-distribution cases.

3.4 Task-augmented Query Embeddings for Improved Retrieval

To enhance retrieval accuracy for domain-specific queries, the system constructs embeddings for both the query and its associated task independently before combining them. Let $encode(q)$ denote the raw embedding of the query q, and $encode(task)$ the embedding of a task vector derived from curated synonym sets. Both embeddings are computed separately. The final task-augmented representation is obtained by taking their weighted sum:

$$q' = encode(q) + \alpha_{task} \cdot encode(task) \tag{1}$$

where α_{task} determines the relative influence of the task signal. To ensure stability and comparability under cosine similarity, the combined embedding is then normalized using L2 normalization:

$$q'' = \frac{q'}{\|q'\|_2} \tag{2}$$

This procedure ensures that the task signal is seamlessly integrated into the query representation while preserving global semantic alignment across embeddings. If all embeddings are maintained L2-normalized (via $faiss.normalize\_L2$), cosine similarity and inner product become equivalent; we explicitly state this assumption where it is used for efficiency. Task injection biases retrieval toward task-relevant semantics (mitigating orthographic variation and compound forms in Persian) while the final normalization step preserves global geometry and cross-task comparability.

3.5 Adaptive Indexing for Efficient and Scalable Retrieval

The indexing process is designed to be adaptive to corpus size while ensuring both efficiency and reproducibility. All embeddings are stored in float32 format and normalized in batches to minimize memory overhead during construction. For small to moderately sized corpora, the system employs the FAISS IndexFlatIP, which performs exact inner-product searches on normalized vectors, effectively equivalent to cosine similarity. When the corpus size exceeds the predefined threshold (default: 5000), the system automatically switches to IndexIVFFlat, an inverted file index that trains a coarse quantizer and delivers substantially faster search with only minimal approximation error.

Per-task FAQ indices are constructed separately to restrict search to semantically coherent subsets and reduce false positives. Document collections are chunked before encoding; for large collections chunk embeddings are indexed in parallel to accelerate build and query time. All indexing steps are scripted and reproducible, and the codebase allows switching index types through a single configuration parameter to support ablation experiments.

3.6 Hybrid Semantic–Lexical Retrieval for Robust Persian Queries

Persian corpora combine formal and informal registers with orthographic variants and domain terms. To remain robust, retrieval fuses semantic signals

(dense embeddings/FAISS) with lexical signals (BM25). The BM25 score is normalized by the maximum BM25 value across candidate documents, ensuring that lexical scores fall within a dynamic range comparable to semantic similarities. This hybrid design ensures that semantically aligned documents are prioritized while keyword-level matches serve as a safeguard for noisy or sparse queries. The weighting factor β allows practitioners to control the trade-off between semantic and lexical evidence depending on dataset characteristics. This formulation provides the mathematical foundation for the retrieval pipeline, balancing precision and recall across heterogeneous Persian corpora.

4 Experiments and Results

Accurate evaluation is essential for Persian text processing due to the language's complex morphology, punctuation conventions, and use of diacritics. In this study, we systematically compared five retrieval-based approaches: (1) base (semantic RAG), which relies solely on semantic search; (2) hybrid semantic–BM25 retrieval extends this by combining FAISS-based semantic embeddings with BM25 keyword scoring; (3) layered retrieval RAG, which employs a multi-stage retrieval pipeline; (4) TILR-RAG (task-injected layered retrieval RAG), which incorporates task-aware embeddings within the layered framework; and (5) TILHR-RAG (task-injected layered hybrid retrieval RAG), which extends the previous method by combining semantic retrieval with additional hybrid mechanisms. The evaluation process combined targeted manual review with automated preprocessing checks to ensure proper normalization, tokenization, and noise removal in Persian text. Randomly sampled outputs were compared against source content to detect errors, enabling iterative refinement of the preprocessing pipeline. Both quantitative and qualitative assessments confirmed that blending automated procedures with human oversight improves overall data quality and enhances the reliability of retrieval-augmented generation.

Table 1 presents a detailed overview of the preprocessing pipeline and its effectiveness across different stages of Persian text normalization and structuring. The results indicate that most steps achieved a success rate above 90%, demonstrating the robustness of the preprocessing framework. Character normalization, special character removal, and final cleanup consistently produced high-quality outputs, with success rates ranging from 95% to 100%. Sentence reconstruction and paragraph merging also performed well, ensuring surface coherence and readability, particularly in formal

Table 1 A summary of the results and performance of the preprocessing stages

No.	Preprocessing Stage	Items Reviewed	Performance Status	Success Rate (%)	Additional Notes
1	Character normalization	Removal of Arabic forms (e.g., ي, ك) – conversion of redundant symbols – correction of elongated characters	Successful	95%	Normalization was accurate in most cases, except for Persian numerals in some texts.
2	Removal of special characters and symbols	Removal of list markers (●, ■, ✓, ➤) – combined or extra spaces	Successful	100%	All common characters found in informal texts were successfully removed.
3	Tokenization of words	Separation of words with correct spacing – elimination of common word segmentation errors	Relatively successful	85%	Some compound words and phrasal verbs have been incorrectly split.
4	Sentence reconstruction after tokenization	Combining tokens into readable words – ensuring the surface coherence of the sentence	Successful	90%	In most cases, the sentence structure has been reconstructed in an understandable and natural manner.
5	Merging lines into paragraphs	Paragraph boundary detection – identifying empty lines – reconstructing textual structure	Successful	95%	It has produced highly satisfactory results for formal and structured administrative texts.
6	Final cleanup (strip + spacing)	Removing extra spaces – eliminating irregular spacing between tokens	Successful	95%	The final output is visually consistent and suitable for modeling purposes.

Table 2 Summary of the results and performance of the text segmentation and question generation pipeline

Component	Evaluation Criteria	Score	Absolute Error Reduction
Preprocessing	Accuracy of Persian text normalization	98%	32%
Question quality	Semantic alignment with chunks	89%	26%
Embeddings	Cosine similarity (question–answer pairs)	82%	ParsBERT showed the best performance.

administrative texts. The most challenging stages were word tokenization and sentence reconstruction, where compound words and phrasal verbs occasionally caused segmentation errors, leading to a slightly lower success rate of 85%. Nevertheless, the overall performance shows that the preprocessing module effectively prepares Persian text for downstream retrieval and generation tasks, striking a balance between linguistic accuracy and computational efficiency.

Results demonstrated that preprocessing accuracy exceeded 95% for normalization and cleanup, though challenges unique to Persian remained, including compound word handling, non-standard orthography, and the scarcity of specialized open-source tools. Iterative review cycles gradually improved system robustness. Visual analyses (e.g., t-SNE plots and similarity heatmaps) further revealed embedding weaknesses in handling long and complex Persian texts, providing insights for refining retrieval strategies.

Building on these observations, a systematic evaluation of the end-to-end pipeline was conducted to quantify its effectiveness across preprocessing, question generation, and embedding stages. Table 2 summarizes the performance of the text segmentation and question generation pipeline, highlighting strong results across preprocessing, question generation, and embedding evaluation. The preprocessing stage achieved the highest accuracy at 98%, reflecting the effectiveness of Persian text normalization and contributing to a substantial absolute error reduction of 32%. The quality of generated questions, measured by their semantic alignment with source chunks, reached 89%, demonstrating the system's ability to produce contextually consistent and meaningful queries with an error reduction of 26%. Finally, the embedding evaluation based on cosine similarity between question–answer pairs achieved 82%, with ParsBERT outperforming alternative models. These findings indicate that the pipeline not only ensures robust preprocessing

Table 3 Overview of five RAG configurations, from baseline semantic retrieval to the proposed hybrid task-injected layered approach

Scenario	Configuration	Description
S1	**Semantic RAG**	A layered architecture that applies semantic retrieval for document selection, followed by a fallback general response when no relevant document is found.
S2	**Hybrid semantic–BM25 retrieval**	A dual retrieval mechanism that integrates FAISS-based semantic search with BM25 keyword scoring, combining both signals to improve ranking robustness and accuracy across heterogeneous Persian texts by injecting task-specific.
S3	**Layered retrieval RAG**	A three-layer retrieval pipeline (FAQ $\rightarrow$ documents $\rightarrow$ fallback) using semantic search, but without task injection.
S4	**TILR-RAG (task-injected layered retrieval RAG)**	Extends the layered retrieval framework by injecting task-specific vectors into query embeddings to improve domain alignment.
S5	**TILHR-RAG (task-injected layered hybrid retrieval RAG)**	The full proposed version, combining task injection with layered retrieval and a HybridRetriever that integrates weighted BM25 with semantic search to achieve optimal accuracy and efficiency.

but also supports the generation of semantically aligned QA pairs, thereby strengthening the overall reliability of the dataset construction process.

Building on these strong baseline outcomes, the next stage of our study shifted toward assessing how various retrieval-augmented generation (RAG) configurations could capitalize on the reliable preprocessing, question generation, and embedding pipeline to enhance end-to-end performance. In doing so, we not only examined the robustness of each component in isolation but also their integration into complete RAG architectures tailored for Persian text.

This evaluation, however, was carried out under several constraints, most notably the lack of Persian-specific RAG datasets and the dominance of English-centric resources. To mitigate these challenges, we employed localized prompt engineering, custom preprocessing modules, and efficient local deployment strategies. As shown in Table 3, five experimental configurations were developed within this framework, each representing a progressive refinement of RAG. S1 (semantic RAG) serves as the baseline, relying exclusively on semantic embeddings to retrieve the most relevant document chunks while falling back to a general response when needed. S2 (hybrid

semantic–BM25 retrieval) extends this baseline by integrating semantic similarity with BM25 keyword scoring, thus balancing deep contextual alignment with lexical matching – an especially valuable enhancement for heterogeneous Persian corpora where orthographic variations, sparse contexts, and domain-specific terminology often reduce the reliability of purely semantic search.

Beyond these, the pipeline introduces task-awareness and structured layering. S3 (layered retrieval RAG) organizes retrieval into three tiers: first checking FAQs, then searching broader document collections, and finally falling back to general reasoning if needed. S4 (TILR-RAG) advances this structure by injecting task-specific vectors directly into query embeddings, enhancing semantic alignment with the underlying knowledge domains. Finally, S5 (TILHR-RAG) represents the most comprehensive approach, combining task injection, layered retrieval, and hybrid (semantic + lexical) search mechanisms. This configuration maximizes both precision and coverage, offering the strongest balance between efficiency, contextual accuracy, and robustness across diverse question types. Each of the five evaluated methods was systematically tested across a wide range of Persian queries, and performance was assessed using semantic accuracy, retrieval quality, response latency, memory efficiency, and the reliability of generated answers.

Having outlined the progressive design of the five retrieval configurations, it is essential to establish a rigorous evaluation protocol to compare their effectiveness. The following framework defines the metrics and criteria used to systematically assess system performance across accuracy, efficiency, and resource utilization.

4.1 Evaluation Framework

The evaluation framework measures three dimensions: semantic accuracy, latency, and memory consumption. Similarity computations are batched for efficiency and reproducibility; resource usage is profiled with a *psutil*-based monitor that samples resident memory at fixed intervals.

Semantic similarity is computed using cosine similarity:

$$sim(a, b) = \frac{(a \cdot b)}{\|a\|_2 \cdot \|b\|_2} \tag{3}$$

A query–FAQ match is accepted if the FAQ similarity exceeds a configured threshold $\tau_{\{FAQ\}}$:

$$Accept_{FAQ} \Leftrightarrow sim(q'', f) \geq \tau_{\{FAQ\}} \tag{4}$$

Document matches pass if the combined hybrid score meets the document threshold $\tau_{\{Doc\}}$:

$$Accept_{Doc} \Leftrightarrow S_{combined}(i) \geq \tau_{\{Doc\}} \tag{5}$$

Latency is measured as elapsed time from query submission to final answer (averaged over multiple runs). Memory is reported as peak resident set size (RSS) during inference, sampled at fixed intervals; all reported values include mean $\pm$ standard deviation over at least three independent runs to reflect measurement variability. The framework exposes thresholds $(\tau_{\{FAQ\}}, \tau_{\{Doc\}})$, margin parameters, and fusion weights for systematic optimization and ablation studies.

4.2 Reproducibility

Reproducibility is enforced through deterministic initialization (fixed random seeds), fully documented preprocessing and chunking scripts, and regenerable indexing pipelines. Query embeddings are computed and normalized as:

$$e_q = \frac{encode(q)}{\|encode(q)\|_2} \tag{6}$$

Task classification uses the top and second highest similarity scores against task vectors:

$$score_1,\ i_1 = max_j\ sim(e_q, t_j) \tag{7}$$

$$score_2,\ i_2 = second\_max_j\ sim(e_q, t_j) \tag{8}$$

The routing decision is:

$$\hat{\tau}_{task} = \begin{cases} general, & if\ score_1 < \theta_{low} \\ general, & if\ |score_1 - score_2| < \delta \\ task_{i_1}, & otherwise \end{cases} \tag{9}$$

with $\theta_{low} = 0.55$ and $\delta = 0.01$ as default values (both exposed for tuning). Semantic accuracy over a test set of N examples is reported as the mean cosine between predicted and reference answers:

$$SemAcc = \left(\frac{1}{N}\right) \sum_{\{i=1\}}^{\{N\}} sim(e_{\hat{a}_i}, e_{a_i}) \tag{10}$$

where e_* is the embedding function (e.g., multilingual E5) and $\text{sim}(\cdot, \cdot)$ is cosine similarity between two embedding vectors. Here, $\hat{a}_i$ (predicted answer) is the response generated by the system for query q_i, while a_i (reference answer) is the gold-standard response provided in the dataset. All preprocessing scripts, indexing routines, and evaluation code are released as supplementary material so that reported results can be independently reproduced and audited.

In addition to the supplementary material provided with this paper, we have made the full implementation publicly accessible to further promote transparency and reproducibility. All preprocessing scripts, indexing routines, and evaluation pipelines are released as open-source code at [https://github.com/zoughi/TILHR-RAG], enabling full reproducibility of the reported results. With this reproducibility setup in place, the evaluation outcomes can be interpreted with confidence, as observed differences across retrieval strategies reflect genuine architectural effects rather than experimental noise. Building on this foundation, we conducted an empirical assessment of all configurations to examine their practical performance. The comparative analysis emphasized key dimensions – semantic accuracy, retrieval robustness, computational efficiency, and memory usage – thereby highlighting both the strengths and trade-offs of the five methods. This systematic evaluation offers a comprehensive view of how incremental refinements from S1 through S5 shape overall system effectiveness.

Table 4 presents the evaluation of the five retrieval configurations (S1–S5), demonstrating progressive improvements in semantic accuracy, efficiency, and scalability. S1 (semantic RAG) records the lowest semantic

Table 4 Normalized comparison of accuracy, latency, and memory usage across five retrieval configurations (S1–S5) on an A100 GPU

Response Mode	S1 (Semantic RAG)	S2 (Hybrid Semantic–BM25 Retrieval)	S3 (Layered Retrieval RAG)	S4 (TILR-RAG)	S5 (TILHR-RAG)
Average semantic accuracy	86.53	87.79	88.43	89.10	89.67
Elapsed time (second)	923.75	2189.27	590.01	551.77	574.03
Average memory usage	2475.27 MB	2489.33 MB	1509.07 MB	1541.00 MB	1566.62 MB

accuracy (86.53%) with high memory usage (2475.27 MB) and a relatively long response time (923.75 s), underscoring the limitations of relying solely on semantic embeddings with a fallback layer. S2 (hybrid semantic–BM25 retrieval) increases accuracy to 87.79% but also incurs the highest memory usage (2489.33 MB). Moreover, this configuration suffers from the longest response latency (2189.27 s), suggesting that the integration of lexical and semantic scoring introduces additional computational overhead despite improving robustness. S3 (layered retrieval RAG) further enhances performance, achieving higher accuracy (88.43%) and reduced latency (590.01 s). Its memory footprint (1509.07 MB) is substantially lower than S1 and S2, indicating greater efficiency. The reduction in latency is largely due to the fact that many queries are answered at the FAQ layer, which provides much faster responses compared to the semantic retrieval layer. Unlike the FAQ layer, the semantic stage must compare each query against all document chunks rather than against predefined question–answer pairs, making it inherently slower.

The task-injected approaches, S4 (TILR-RAG) and S5 (TILHR-RAG), deliver the best trade-offs across all metrics. By embedding task-specific vectors into query representations, S4 boosts semantic accuracy to 89.10% while maintaining low memory usage (1541.00 MB) and the fastest overall latency (551.77 s). S5 (TILHR-RAG) achieves the highest semantic accuracy (89.67%), with efficient response time (574.03 s) and only a marginal increase in memory usage (1566.62 MB). These results confirm that incorporating both task-awareness and hybrid retrieval mechanisms not only improves alignment with domain-specific queries but also optimizes computational efficiency. Overall, the findings underscore that layered and task-injected hybrid frameworks outperform baseline methods, offering the most effective balance between accuracy, latency, and resource consumption in Persian RAG systems.

Beyond the aggregate comparison of system-level efficiency, it is also important to examine how the S5 scenario performs across different categories of real-world queries. Such an analysis provides a more fine-grained view of the model's strengths and limitations, revealing whether the observed improvements in accuracy and efficiency are consistently maintained across diverse task types. Table 5 presents the answer accuracy across different task categories, demonstrating consistent and robust performance of the proposed system. Among all categories, event support achieved the highest accuracy (96.46%), indicating that the retrieval and generation pipeline is particularly effective when handling structured, event-related content. By contrast, thesis support (87.01%) and postdoctoral support (87.51%) showed

Table 5 Semantic accuracy across task categories

Task Category	Semantic accuracy
Executive guideline	88.74
Postdoctoral support	87.51
Thesis support	87.01
Event support	96.46
Study opportunity	88.65
General	89.64
Overall average	89.66

Table 6 Performance of S5 (TILHR-RAG) with four embedding models on accuracy, latency, and memory (A100 GPU)

S5 (TILHR-RAG)	Bert-base-parsbert-uncased	Tooka-SBERT-V2-large	Multilingual-e5-base	Multilingual-e5-large
Average semantic accuracy	77.90	63.32	89.48	89.67
Elapsed time (second)	508.41	743.15	500.33	574.03
Average memory usage	2158.22 MB	1250.43 MB	1512.68 MB	1566.62 MB

slightly lower performance, likely due to the greater lexical variability and semantic overlap within academic and research-related queries. The executive guideline (88.74%), study opportunity (88.65%), and general (89.64%) categories exhibited balanced performance close to the overall mean. With an overall average accuracy of 89.66%, the results highlight the system's ability to generalize effectively across diverse query types while maintaining high semantic alignment, with only minor fluctuations attributable to domain-specific linguistic complexities.

While task-level evaluation demonstrates the system's ability to generalize effectively across diverse query types, a deeper understanding of performance also requires examining the role of underlying embedding models. Since embeddings directly determine the quality of semantic retrieval and influence both accuracy and efficiency, comparing different embedding backbones provides critical insights into the scalability and robustness of the proposed S5 configuration. Table 6 compares the performance of S5 (TILHR-RAG) across four embedding models, highlighting the trade-offs between semantic accuracy, efficiency, and resource consumption. The multilingual-e5-large variant achieves the highest semantic accuracy (0.8967) with near-best latency ($\sim$574 s) and moderate memory usage ($\sim$1.57 GB), closely followed by multilingual-e5-base (0.8948, $\sim$500 s, $\sim$1.51 GB).

These results confirm the strength of the multilingual-e5 family, whose cross-lingual pretraining enables high-quality embeddings across languages and proves particularly effective for Persian, where orthographic variation and semantic subtlety often challenge monolingual models. In contrast, bert-base-parsbert-uncased provides comparable speed (∼508 s) but markedly lower accuracy (0.7790) and the highest memory footprint (∼2.16 GB), limiting its scalability. Tooka-SBERT-V2-Large is the most memory-efficient model (∼1.25 GB) but underperforms significantly in both accuracy (0.6332) and latency (∼743 s). Overall, these findings demonstrate that multilingual-e5 embeddings are pivotal to TILHR-RAG's success, enabling strong semantic alignment and robust hybrid ranking while maintaining feasible computational costs.

Beyond embeddings, the choice of the large language model (LLM) back-end also plays a decisive role in shaping overall system performance. Since generation quality, latency, and memory footprint are tightly coupled with the underlying LLM, evaluating multiple back-ends provides complementary insights into the practical trade-offs of deploying TILHR-RAG in real-world scenarios. Table 7 compares TILHR-RAG paired with five different LLM back-ends and highlights the trade-offs among accuracy, latency, and resource use, which explain why one configuration may be preferred over another. The smallest generator, TILHR-RAG + LLaMA-3.1 (base), achieves high semantic accuracy (89.67%) while requiring the least runtime (∼574 s) and memory (∼1.57 GB), offering the best balance of efficiency and accuracy for resource-constrained or interactive deployments. In contrast, TILHR-RAG + Qwen-32B reaches the highest accuracy (90.79%) but at substantially higher latency (∼2276 s) and memory (∼2.31 GB), making it most suitable when maximum accuracy is prioritized over operational cost. TILHR-RAG + Gemma2-27B provides a modest accuracy improvement (90.09%) over the

Table 7 Performance of TILHR-RAG with different LLM back-ends, usage under a high-resource GPU setting (A100)

Response Mode	TILHR-RAG-llama3.1:8b	TILHR-RAG-llama3.1:70b	TILHR-RAG-dorna:8b	TILHR-RAG-deepseek:1.5b	TILHR-RAG-gemma2:27b	TILHR-RAG-qwen:32b
Average semantic accuracy	89.67	88.19	Did not respond	DID not respond	90.09	90.79
Elapsed Time (second)	574.03	3912	–	–	996.39	2276
Average memory usage	1566.62 MB	2311.66 MB	–	–	2341 MB	2310 MB

small LLaMA, with intermediate latency ($\sim$996 s) and higher memory usage ($\sim$2.34 GB), representing a middle ground that may suit batch-processing workloads. Scaling LLaMA to the 70B variant leads to significantly longer runtimes ($\sim$3912 s) without accuracy improvements (88.19%), suggesting diminishing returns from sheer model size – likely due to decoding overhead, deployment inefficiencies, or mismatch with retrieval evidence.

In contrast, TILHR-RAG + Dorna-8 occasionally produced incoherent responses, including hallucinations and excessively long, loop-like outputs. These behaviors indicate that while the model has potential, it requires further fine-tuning and adaptation to improve reliability and ensure consistent integration in practical scenarios. Similarly, TILHR-RAG + DeepSeek-1.5 failed to generate usable responses in Persian: when explicitly instructed to respond in Persian, the model consistently returned no valid outputs. These observations point to current limitations in multilingual robustness and highlight areas where additional training or alignment is required. Overall, the optimal generator depends on deployment objectives: the base LLaMA provides the most practical efficiency–accuracy trade-off for production use, while Qwen and Gemma are better aligned with offline, accuracy-critical scenarios. The differences observed reflect the complex interplay among model capacity, inference efficiency, tokenizer/pretraining coverage, and retrieval integration.

To assess practical, deploy ability under constrained hardware, we repeated the five-scenario evaluation on a low-resource NVIDIA GeForce GTX 1080; Figure 8 shows the normalized accuracy, runtime and memory measurements for each configuration. Figure 8 also shows that the relative ranking of the five retrieval configurations remains stable on the GTX 1080 – S5 (TILHR-RAG) and S4 (TILR-RAG) still deliver the highest semantic accuracy (89.69% and 89.22%, respectively) while S1 and S2 remain the weakest (86.54% and 87.69%). However, latency increases dramatically on the low-resource GPU: elapsed times inflate to the order of 2×10^4–9×10^4 seconds (figure values), i.e. roughly 36–40$\times$ slower than on the A100 for the layered and task-injected variants and even larger factors for the purely semantic/hybrid baseline; S2 again exhibits the worst runtime (88,329.58 in the normalized units) consistent with the extra overhead of combining BM25 and semantic scoring. Memory usage is comparatively stable across devices and follows the same pattern as before (S2 $\approx$ 2.49 GB highest; S3–S5 $\approx$ 1.51–1.56 GB), indicating that the primary hardware sensitivity is latency rather than resident memory. Practically, these results show that while accuracy is robust to hardware downscaling, end-to-end response times on a GTX 1080 render interactive deployment impractical without further optimization (e.g.,

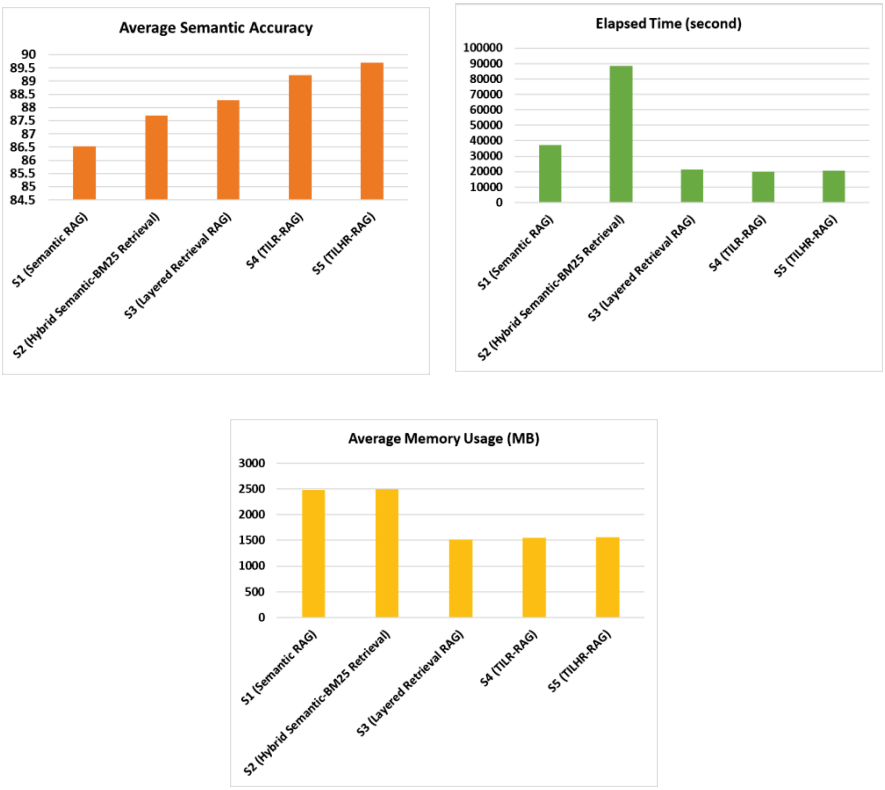


Figure 8 Normalized performance of five retrieval configurations (S1–S5) on a low-resource GPU (GTX 1080).

model quantization, smaller generator back-ends, precomputed embeddings, reduced n_{probe}, top-k, or offloading retrieval). In short, S4/S5 remain the best choices for accuracy–efficiency trade-offs, but on low-resource GPUs the system requires engineering adaptations to achieve acceptable throughput.

5 Conclusion and Discussion

This study introduced a retrieval-augmented generation (RAG) framework specifically tailored to the challenges of Persian text processing, addressing the scarcity of native resources and the limitations of existing English-centric pipelines. By systematically evaluating five retrieval configurations – ranging from a baseline semantic search model to the proposed task-injected

layered hybrid retrieval RAG (TILHR-RAG) – we demonstrated that progressively integrating layered retrieval, hybrid scoring, and task-specific vector injection leads to measurable gains in both semantic accuracy and system efficiency. Across extensive experiments, the proposed TILHR-RAG achieved the strongest overall balance, attaining 89.67% semantic accuracy with moderate latency and memory consumption, thereby outperforming both baseline and intermediate variants.

The results further highlight the critical role of task-specific query augmentation and multilingual embeddings in handling the orthographic variability, compound structures, and domain-specific terminology characteristic of Persian. Injecting task-aware vectors into query embeddings consistently improved semantic alignment, reducing response time without substantial memory overhead. Similarly, the use of multilingual E5 embeddings proved pivotal, as their cross-lingual pretraining offered superior contextual representation for Persian compared to monolingual baselines such as ParsBERT or domain-specific SBERT variants. These findings underscore that carefully designed hybrid RAG architectures can overcome resource constraints in low-resource languages by exploiting transfer learning and structured retrieval pipelines.

From a practical perspective, the trade-offs between accuracy, latency, and memory observed across model back-ends (e.g., LLaMA, Qwen, Gemma, DeepSeek) suggest that deployment choices should be guided by application requirements: smaller models such as LLaMA 3.1 provide the best accuracy–efficiency balance for interactive or resource-constrained environments, while larger back-ends like Qwen-32B deliver peak accuracy at higher operational cost, making them suitable for offline or batch settings. The failure of DeepSeek-1.5B under enforced Persian generation also highlights the importance of robust multilingual pretraining in ensuring reliability.

Overall, this work provides both a methodological blueprint and empirical evidence that layered, task-aware, hybrid retrieval combined with multilingual embedding models enables scalable and precise Persian-language RAG systems. Future research should extend this pipeline to other low-resource languages, explore adaptive weighting of semantic and lexical signals, and investigate integration with larger multimodal corpora. Such directions will not only enhance retrieval robustness but also broaden the applicability of RAG systems to diverse linguistic and domain-specific contexts.

Acknowledgements

The author would like to express sincere gratitude to the ICT Research Institute (Iran Telecommunication Research Center – ITRC) for their valuable support and contributions. With more than 52 years of scientific experience, ITRC is recognized as the most experienced research institution in the field of Information and Communication Technology (ICT) in Iran, and its support has played an important role in the completion of this research.

References

[1] Petroşanu, D. M., Pîrjan, A., and Tăbuşcă, A. (2023). Tracing the influence of large language models across the most impactful scientific works. *Electronics*, *12*(24), 4957.

[2] Kumar, P. (2024). Large language models (LLMs): survey, technical frameworks, and future challenges. *Artificial Intelligence Review*, *57*(10), 260.

[3] Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., ... and Li, Q. (2024, August). A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining* (pp. 6491–6501).

[4] Han, B., Susnjak, T., and Mathrani, A. (2024). Automating systematic literature reviews with retrieval-augmented generation: a comprehensive overview. *Applied Sciences*, *14*(19), 9103.

[5] Wiratunga, N., Abeyratne, R., Jayawardena, L., Martin, K., Massie, S., Nkisi-Orji, I., ... and Fleisch, B. (2024, June). CBR-RAG: case-based reasoning for retrieval augmented generation in LLMs for legal question answering. In *International Conference on Case-Based Reasoning* (pp. 445–460). Cham: Springer Nature Switzerland.

[6] Shiraee Kasmaee, A. (2025). *Domain-Specific Text Embedding Models for Information Retrieval* (Doctoral dissertation).

[7] Barron, R. C., Grantcharov, V., Wanna, S., Eren, M. E., Bhattarai, M., Solovyev, N., ... and Alexandrov, B. S. (2024, December). Domain-specific retrieval-augmented generation using vector stores, knowledge graphs, and tensor factorization. In *2024 International Conference on Machine Learning and Applications (ICMLA)* (pp. 1669–1676). IEEE.

[8] Amur, Z. H., Kwang Hooi, Y., Bhanbhro, H., Dahri, K., and Soomro, G. M. (2023). Short-text semantic similarity (stss): Techniques, challenges and future perspectives. *Applied Sciences*, *13*(6), 3911.

[9] Jolfaei, S. A., and Mohebi, A. (2025). A review on persian question answering systems: from traditional to modern approaches. *Artificial Intelligence Review*, *58*(5), 127.

[10] Nurzhanov, A., and Sharipbay, A. (2024). Modern AI models for text analysis: A comparison of ChatGPT and RAG. *Journal of Data Analytics and Artificial Intelligence Applications*, *1*(1), 1–13.

[11] Tahir, H. A., Panula, T., Feli, M., and Omobolaji, A. R. (2025). Development of a Medical Question-Answering System Using Large Language Models with Retrieval Augmented Generation and Prompt Engineering. *Health Technology*.

[12] Petroşanu, D. M., Pîrjan, A., and Tăbuşcă, A. (2023). Tracing the influence of large language models across the most impactful scientific works. *Electronics*, *12*(24), 4957.

[13] Chakraborty, S., Chowdhury, R., Shuvo, S. R., Chatterjee, R., and Roy, S. (2025). A scalable framework for evaluating multiple language models through cross-domain generation and hallucination detection. *Scientific Reports*, *15*(1), 29981.

[14] Toftdahl, J. C., Shahin, K. I., and Hylle, T. (2024, July). Enhancing Data Retrieval with Custom Embedding Models and ChatGPT. In *2024 IEEE 24th International Conference on Software Quality, Reliability and Security (QRS)* (pp. 455-461). IEEE.

[15] Mahboub, A., Za'ter, M. E., Al-Rfooh, B., Estaitia, Y., Jaljuli, A., and Hakouz, A. (2024). Evaluation of semantic search and its role in retrieved-augmented-generation (rag) for arabic language. *arXiv preprint arXiv:2403.18350*.

[16] Abdelazim, H., Tharwat, M., and Mohamed, A. (2023). Semantic Embeddings for Arabic Retrieval Augmented Generation (ARAG). *International Journal of Advanced Computer Science & Applications*, *14*(11).

[17] Badaro, G., Baly, R., Hajj, H., El-Hajj, W., Shaban, K. B., Habash, N., ... and Hamdi, A. (2019). A survey of opinion mining in Arabic: A comprehensive system perspective covering challenges and advances in tools, resources, models, applications, and visualizations. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, *18*(3), 1–52.

[18] Atlam, H. F. (2025). LLMs in Cyber Security: Bridging Practice and Education. *Big Data and Cognitive Computing*, *9*(7), 184.

[19] Kuratomi, G., Pirozelli, P., Cozman, F. G., and Peres, S. M. (2025). A RAG-Based Institutional Assistant. *arXiv preprint arXiv:2501.13880*.

[20] Ding, W., Liang, P., Tang, A., and Van Vliet, H. (2014). Knowledge-based approaches in software documentation: A systematic literature review. *Information and Software Technology*, *56*(6), 545–567.

[21] Teixeira de Lima, R., Gupta, S., Berrospi, C., Mishra, L., Dolfi, M., Staar, P., and Vagenas, P. (2024). Know Your RAG: Dataset Taxonomy and Generation Strategies for Evaluating RAG Systems. *arXiv e-prints, arXiv-2411*.

[22] Yang, J., Liu, Z., Xiao, S., Li, C., Lian, D., Agrawal, S., ... and Xie, X. (2021). Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems*, *34*, 28798–28810.

[23] Hasan, S., and Rezai, A. (2025). LLM: Retreival vs. ParametricMemory Tradeoff: A Comparison of Retrieval-Augmented Generation and Standalone LargeLanguage Models Using RAGAS Answer Accuracy.

[24] Burger, J., Cardie, C., Chaudhri, V., Gaizauskas, R., Harabagiu, S., Israel, D., ... and Weishedel, R. (2001, July). Issues, tasks and program structures to roadmap research in question & answering (Q&A). In *Document Understanding Conferences Roadmapping Documents* (pp. 1–35).

[25] Shehmir, S., and Kashef, R. (2025). LLM4Rec: A Comprehensive Survey on the Integration of Large Language Models in Recommender Systems – Approaches, Applications and Challenges. *Future Internet*, *17*(6), 252.

[26] Moniri, S., Schlosser, T., and Kowerko, D. (2024). Investigating the Challenges and Opportunities in Persian Language Information Retrieval through Standardized Data Collections and Deep Learning. *Computers*, *13*(8), 212.

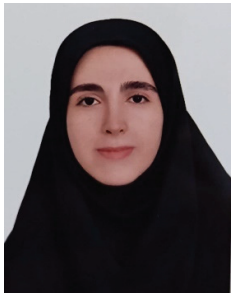
Biographies

Toktam Zoughi received her Ph.D. in computer engineering (artificial intelligence) from Amirkabir University of Technology, Tehran, Iran, in 2019, and her M.Sc. in computer engineering (artificial intelligence) from Shiraz University, Shiraz, Iran, in 2010. Since January 2020 she has been an Assistant Professor in the Department of Electrical and Computer Engineering at Shariaty College, Technical and Vocational University (TVU), Tehran, Iran. Her research interests include natural language processing, deep learning, machine learning, speech processing, and image processing.

Ehsan Arianyan received the M.S. and Ph.D. degrees from Amirkabir University of Technology, Tehran, Iran, in 2010 and 2015, respectively. Now, he is the head of the information technology research faculty and an assistant professor at the ICT Research Institute (ITRC). He is the author of more than 20 peer-reviewed papers as well as 5 books. His areas of interest include cloud computing, big data, parallel processing**,** and data centers.

Maryam Mahmoudi holds a B.S. in Software Engineering and an M.S. in Information Technology Engineering. She has been a researcher at the ICT Research Institute since 2012. Her research focuses on information retrieval, data mining, natural language processing, artificial intelligence, and generative AI. She is actively engaged in the enhancement and evaluation of large language models and intelligent assistants, with a particular focus on benchmark design and model assessment.

Mahtab Aghdamifard received her technical diploma in computer and network systems from Honarafarinan Technical School. She earned both her associate's and bachelor's degrees in computer engineering from Shariaty Technical and Vocational College, Tehran, Iran. Her research interests include natural language processing (NLP), particularly Persian language processing, artificial intelligence, and data-driven approaches. She has been involved in several NLP-related projects and is interested in applying language technologies to real-world problems. She is currently a junior data scientist focusing on text processing and intelligent systems, with a particular interest in developing chatbots and intelligent assistants.



Mohadese Nikoogoftar received her diploma in mathematics and physics from Farzanegan4 High School and a bachelor's degree in computer engineering from the Dr. Shariaty Technical and Vocational College. She is currently pursuing a master's degree in computer engineering with a specialization in artificial intelligence at the University of Science and Culture. Her areas of expertise include Python programming and artificial intelligence, with a focus on chatbot development and predictive model design. She also has a strong interest in data analysis and enjoys working on projects that combine technical precision with practical problem-solving.