Enhancing Reliability in Community Question Answering with an Expert-Oriented RAG System

Seyyede Zahra Aftabi[1], **Saeed Farzi**[1],
[1] K. N. Toosi University of Technology, Tehran, Iran.
Correspondence: Seyyede Zahra Aftabi

Abstract

In recent years, pre-trained large language models (LLMs) have become a cornerstone for automatically generating answers in question-and-answer (Q&A) communities, significantly reducing user wait times and improving response quality. However, these models require substantial computational resources and are prone to generating hallucinated or unreliable content. To overcome these limitations, we propose an advanced expert-oriented Retrieval-Augmented Generation (RAG) framework as a cost-effective and reliable alternative. Central to our approach is a user-aware question entailment recognition module, which leverages user modeling to identify archived questions with answers that fully or partially address the user's new query. This user modeling significantly improves retrieval relevance, resulting in reduced hallucination and enhanced answer quality. The framework synthesizes expert-written answers from similar questions to generate unified responses. Experimental results on the CQADupStack and SE-PQA datasets show the superiority of our user-aware approach over its user-agnostic counterpart, with ROUGE-1 gains of 3.6% and 0.9%. Both human and AI evaluations confirm the effectiveness of incorporating user modeling in minimizing hallucination and delivering contextually appropriate answers, demonstrating its potential for real-world Q&A systems. The code and data are available on GitHub repository at https://anonymous.4open.science/r/User-Oriented-RAG-CQA.

1 Introduction

Nowadays, question-and-answer communities (Q&A), such as Stack Overflow and Quora, have sparked a revolution in information retrieval and online knowledge sharing. Members within these communities can post questions, provide answers, and engage in discussions through commenting and voting (Roy et al., 2023). When an information need arises, a community member, referred to as a user, submits a question and awaits answers from knowledgeable members, referred to as experts. Experts must navigate through large volumes of unresolved questions and answer those lying within their expertise (Costa and Ortale, 2023). Thus, users highly value these communities for providing the opportunity to obtain authentic human-written answers. However, despite the diligent effort and substantial time devoted by experts in responding to questions, a significant portion remains unanswered, imposing a formidable challenge to Q&A communities (Roy et al., 2023; Roy, 2020; Asaduzzaman et al., 2013).

In recent years, the advent of large language models (LLMs) has brought about a paradigm shift in a wide range of natural language processing (NLP) tasks, including question answering (QA) (Annepaka and Pakray, 2025). Consequently, users are abandoning Q&A communities in favour of AI-driven conversational interfaces that promise immediate responses (Burtch et al., 2024; del Rio-Chanona et al., 2024). However, LLMs often grapple with several deficiencies: (1) training them to acquire broad, general-purpose knowledge is computationally expensive, (2) they often fail to provide valid, expert-like answers (Huang et al., 2024), and (3) they are susceptible to hallucinate, which means they may produce plausible yet fabricated content that is hard to detect by non-experts (Kabir et al., 2024). Ergo, instead of using standalone LLMs, this study suggests employing them in a retrieval-augmented generation (RAG)-based system to develop a context-aware AI assistant in Q&A communities. By delivering fast, accurate answers grounded in the rich, verified knowledge accumulated in community archives, such a system can mitigate hallucinations, alleviate long waiting times, and ultimately help revitalise and sustain Q&A communities.

Our proposed RAG-based system extends the

Findings of the Association for Computational Linguistics: EACL 2026, pages 2551–2569
March 24-29, 2026

standard architecture by introducing two additional components alongside the conventional indexer, retriever, and generator: a user profiler and a post-retriever. The post-retriever component leverages the concept of question entailment (Abacha et al., 2016), wherein a question Q_2 is considered an entailed question for Q_1 if every valid answer to Q_2 also qualifies as a partial or complete answer to Q_1. This concept has practical implications for Q&A communities, as for most unanswered questions, several entailed questions can be found within community archives that have already been satisfactorily resolved (Sun and Song, 2023; Costa and Ortale, 2023). Thereon, after retrieving similar archived questions using the base retriever, the post-retriever filters those questions that are most likely to be an entailed one. The more accurately the irrelevant questions are excluded, the less likely the system is to produce hallucinated responses.

Recognising question entailment (RQE) transcends mere syntactic comparison and requires deep semantic understanding (Sarrouti et al., 2021; Xu and Yuan, 2020). In particular, two questions may elicit identical responses despite differing in wording or may share lexical similarities despite being unrelated (Xu and Yuan, 2020). This challenge is further amplified when questions lack sufficient context. To overcome this issue, one idea is to incorporate the user background knowledge as additional context into the RQE task to help clarify the thematic category of questions. Since users' background knowledge is not explicitly stated in their profiles, the user profiler is introduced to infer this knowledge from users' prior activities and represent it as sequences of thematic tags. To this end, a language model is fine-tuned on a question–tag dataset, where tag diversity is regulated using a hierarchical clustering strategy.

In a nutshell, the principal contributions of the current research are as follows:

- Introducing an advanced RAG-based system for answering community questions with reduced hallucination and lower computational cost.
- Profiling users' background knowledge as sequences of thematic tags based on their history of questions.
- Developing a novel question entailment recognition module as a post-retriever, which reframes RQE as a text generation task and fuses user knowledge into the recognition process to help illuminate question topics.
- Diminishing tag diversity to refine the training data for the user profiler by organising tags into a three-level hierarchy of clusters.

2 Related Work

This section describes related work in two groups: recognizing question entailment and user profiling.

2.1 Recognizing question entailment

Reviewing past research on recognizing question entailment (RQE) reveals various angles from which researchers have attempted to boost performance. Model architecture and knowledge incorporation appear to be the central pillars of this research trajectory. Early efforts relied on conventional machine learning models (Abacha and Demner-Fushman, 2019; Tawfik and Spruit, 2019; Agrawal et al., 2019), which often performed poorly on brief questions due to limited contextual understanding. The field subsequently shifted toward deep learning approaches (Bandyopadhyay et al., 2019) and transformer-based models (Nguyen et al., 2021; Kanakarajan et al., 2022; Alshammari and AlHumoud, 2022).

Further along this trajectory, several studies pursued knowledge acquisition through multi-task learning (Zhou et al., 2019; Aftabi et al., 2024), although such approaches often require high-quality, consistent datasets and are challenging to maintain. Other researchers have advocated knowledge injection via data augmentation (Mrini et al., 2021a; Sarrouti et al., 2021; Monea and Marginean, 2021). While these strategies improve semantic similarity identification, disambiguating short questions with limited context remains challenging. A body of work has incorporated question metadata (Sun and Song, 2023; Ghasemi and Shakery, 2024) or infused global or domain knowledge from knowledge graphs (Yadav et al., 2020; Goodwin and Demner-Fushman, 2019). However, reliance on knowledge graphs may hinder generalisability and scalability. Aligned with this trend, the present study proposes profiling user knowledge and incorporating it as supplementary context into the RQE task.

2.2 User profiling

A review of the literature reveals a divergence in researchers' perspectives on the concept of knowledge: some have examined it from the standpoint

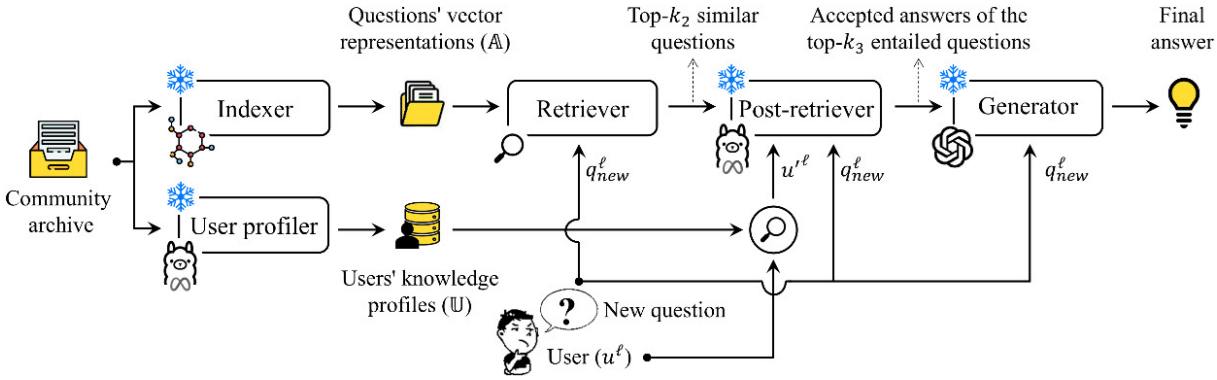

Figure 1: Overview of the advanced knowledge- and retrieval- augmented generation-based community question answering system.

of user expertise (Mumtaz et al., 2019; Krishna and Antulov-Fantulin, 2023; Menaha and Jayanthi, 2023), while others have framed it around user interests (Adishesha et al., 2023; Anandhan et al., 2022). The literature also includes studies that have considered both dimensions (He et al., 2021; Costa and Ortale, 2023; Zahedi et al., 2024). Typically, interests are inferred from users' past questions, whereas expertise is derived from their answers. As the present study aims to understand the topics users most frequently inquire about, the proposed user profiler focuses on analysing users' previously asked questions.

Prominent approaches in this area include transformer-based models (Zahedi et al., 2024; Mumtaz et al., 2019; Adishesha et al., 2023), matrix factorisation methods (Menaha and Jayanthi, 2023; Anandhan et al., 2022), and network analysis techniques (Costa and Ortale, 2023). Matrix factorisation models that exploit the user–tag co-occurrence matrix may fail due to tag diversity and sparsity. Network-based approaches that profile users based on their asking–answering relationships may also suffer from cold-start and dynamicity issues. Consequently, this study adopts transformer-based methods, specifically large language models (LLMs), to model user knowledge more effectively.

3 Methodology

This paper introduces an advanced user-aware retrieval-augmented generation (RAG)-based system for answering user queries in Q&A communities. As depicted in Figure 1, the system consists of five major components: (1) an offline user profiler with a fine-tuned LLaMA-2 model as its backbone, which characterises users' background knowledge as sequences of thematic tags; (2) an offline indexer that employs a pre-trained sentence transformer to encode archived questions; (3) a retriever that identifies similar questions to a given query from the community archive; (4) a post-retriever that filters retrieved candidates using another fine-tuned LLaMA-2 model to retain only those questions that are not only entailed by the input query but also aligned with the user's inferred knowledge state; and (5) a generator, using a pre-trained GPT-4o model, that synthesises a cohesive and accurate answer based on the accepted, expert-authored answers associated with the top-k_3 entailed questions. Each component is described in detail below.

3.1 User profiler

As shown in Figure 2, user profiling is performed in three stages: history retrieval, tag generation, and knowledge modelling. In the first stage, the set $\mathcal{Q}$ containing the k_1 most recent questions submitted by user u in the community is retrieved. In the second stage, each question $q_i \in \mathcal{Q}$ is reformulated using a predefined prompt template to produce a prompt p_i, which is then passed to a fine-tuned LLaMA-2 model. The model generates a sequence of thematic tags, denoted by t'_i, representing the primary themes of the question. In the third stage, all tag sequences generated for user u are decomposed into atomic tags. The distinct tags are sorted by frequency, with ties resolved by their order of first appearance. The resulting list is concatenated into a unified, comma-separated sequence of tags to form the final user knowledge profile, hereafter referred to as u'. These textual profiles effectively capture users' most recent and prominent topics of interest and can be updated in an event-driven fashion.

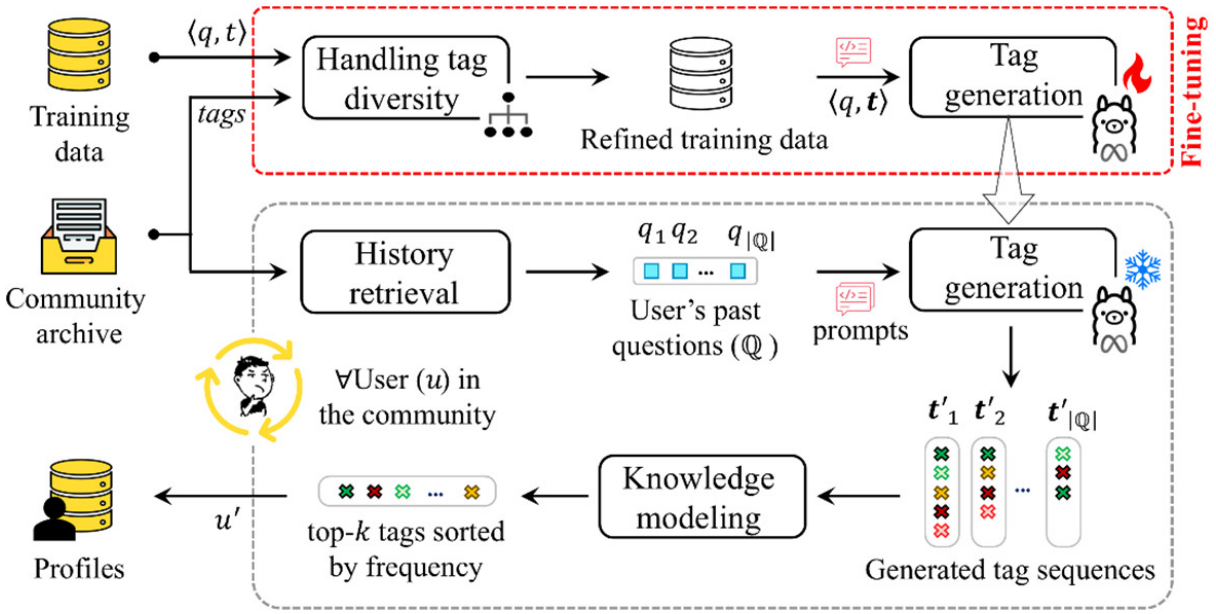

Figure 2: The workflow of the proposed user profiler

While large language models (LLMs) have demonstrated strong performance in text tagging, the extensive diversity of tags in their training data may result in inconsistent tagging for semantically similar questions. To address this issue, a four-step data refinement process is applied to regulate tag diversity prior to fine-tuning LLaMA-2. First, a weighted tag–tag co-occurrence graph is constructed, where nodes represent tags and edges connect tags that co-occur in the same question. Second, the `node2vec` algorithm (Grover and Leskovec, 2016) is employed to generate low-dimensional embeddings of tags, effectively encoding neighbourhood relationships. Third, agglomerative clustering is used to organise tags into a three-level hierarchy, enabling conceptually related tags to be consolidated under representative cluster labels. Finally, the original tags of each question are replaced with a textual sequence of representatives from their corresponding clusters. The updated tags are then deduplicated and ordered by frequency to produce a standardised tag sequence. After data refinement, the LLaMA-2 model is fine-tuned for the tag generation task using a prompt-based learning strategy. The prompt template is illustrated in Appendix, Figure 6. The procedure for maintaining and updating user profiles is detailed in Appendix.

3.2 Indexer

In the offline phase, each question q_j from the community archive is encoded into a dense vector representation e_j using the pre-trained sentence transformer `all-mpnet-base-v2`. The resulting embeddings are stored in a database, denoted as $\mathbb{A}$.

3.3 Retriever

When user u^ℓ submits a new question q^ℓ_{new} to the community, the retriever uses the same encoding model as in the indexing phase to transform the question into a vector representation, denoted by e^ℓ_{new}. It then computes the similarity between e^ℓ_{new} and each archived question embedding $e_j \in \mathbb{A}$ in the shared vector space. Cosine similarity is used as the similarity metric. After computing similarity scores, the top-k_2 most similar archived questions are returned.

3.4 Post-retrieval

Directly feeding the accepted answers of all retrieved questions into the generator may lead to information overload or dilute the relevance of the response by introducing unrelated or marginally related content. Therefore, an intermediate component is required to filter the retrieved questions and retain only those that are entailed. Detecting entailment between a pair of questions requires a binary classification model that predicts a label of 1 if entailment holds and 0 otherwise. Although contextual features are useful, the brevity, vagueness, verbosity, and lexical mismatches of questions can mislead the model and yield incorrect predictions (Hoogeveen et al., 2018). Incorporating user knowledge can help mitigate these issues by revealing the general themes around which a user's queries

revolve. Accordingly, this study introduces a user-aware generative model as the post-retriever component.

The post-retriever model takes as input a triplet $\langle q^{\ell}_{\text{new}}, q_j, u'^{\ell} \rangle$ and reformulates it into a structured prompt (illustrated in Figure 8 in Appendix A, which is then passed to a fine-tuned LLaMA-2 model. The model generates a *positive* label if q_j corresponds to the user's knowledge domain and its accepted answers can adequately respond to q^{ℓ}_{new}; otherwise, it outputs a *negative* label. We adopt *positive* and *negative* instead of *Entailed* and *Not-entailed* to ensure fairness, as the former are single-token labels, whereas the latter comprise multiple tokens. Finally, the top-k_3 entailed candidates are returned. In cold-start scenarios, where a user submits a question for the first time, the current question is recorded as the sole entry in the user's history and passed to the tag generation model in real time. The resulting tag sequence is used as the user knowledge profile.

3.5 Generator

Given the input query and the accepted answers of the top-k_3 entailed questions, synthesised within the prompt shown in Figure 11 in Appendix, the generator produces the final answer. It integrates all relevant information into a coherent, concise, and reliable response. The model is explicitly instructed to disregard its parametric knowledge and rely solely on the provided context. The user may accept the generated answer if it satisfactorily resolves the issue or may wait for expert responses.

4 Experiments

This section provides detailed explanations of the research data and experimental results. The setting parameters are reported in Appendix.

4.1 Data

CQADupStack (Hoogeveen et al., 2015) is a widely used benchmark in community question answering research, compiled from 12 Stack Exchange (SE) communities. SE-PQA (Kasela et al., 2024) is a more recent dataset collected from 50 SE forums. We use curated subsets of each dataset, consisting of 8,000 question pairs for training, 500 for validation, and 2,000 for testing. Each pair (Q_1, Q_2) carries a binary class label indicating entailment (1) or non-entailment (0). Questions are accompanied by metadata, including title, tags, posting date, user ID, answers, and the accepted answer.

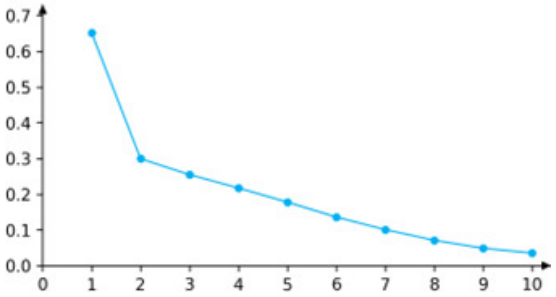

Figure 3: Loss of the LLaMA-2 model during fine-tuning on the CQADupStack-based dataset, plotted against the training epochs.

Metric	BS	R-1	R-2	R-L
recall-based	0.965	0.751	0.683	0.748
precision-based	0.964	0.753	0.684	0.750
F1-based	0.964	0.740	0.669	0.736

Table 1: Performance evaluation of LLaMA-2 for tag generation on the CQADupStack-based test data.

From the 2,000 test questions, 148 questions with accepted answers are selected to form the QA dataset. While the LLaMA-2 model used for the RQE task is fine-tuned on question pairs and their associated labels, the LLaMA-2 model employed in the user profiler is trained solely on Q_1 questions and their corresponding tags. Due to space constraints, we report only the answer quality evaluation results on SE-PQA in the main paper; full experimental details are available in the accompanying GitHub repository.

4.2 Results and analysis

This section presents results and findings corresponding to the research questions (RQs).

RQ1: How can clustering techniques assist in handling tag diversity? We begin by refining the question–tag dataset using hierarchical clustering. Based on the dendrogram illustrated in Figure 7, three thresholds are empirically selected as cut-off points for merging clusters. Visualisation of the resulting three-level hierarchy (Figure 9 in Appendix) reveals well-separated clusters at the primary level, with this separation largely preserved across subsequent levels.

RQ2: How efficient is the tag generation component at discerning the main topics? The second experimental scenario involves fine-tuning the LLaMA-2 model for tag generation and evaluating

Setting	Model	Accuracy	F1-Score	Precision	Recall
User-agnostic	MQU (Mrini et al., 2021b)	94.55	94.60	94.36	94.83
	ReQuEST (Aftabi et al., 2024)	95.00	95.10	94.27	95.95
	Pre-trained GPT-4o	82.05	78.57	99.25	65.02
	Fine-tuned LLaMA-2	95.00	94.89	98.31	91.70
User-aware	Pre-trained GPT-4o	88.15	86.87	98.87	77.47
	Fine-tuned LLaMA-2 (No clustering)	96.15	96.17	96.89	95.45
	Fine-tuned LLaMA-2	**96.60**	**96.58**	98.36	94.86

Table 2: Performance comparison between the proposed user-aware RQE and rivals on test data from CQADupStack.

its performance on a held-out test set. As shown in Figure 3, the cross-entropy loss decreases steadily throughout fine-tuning, indicating effective optimisation. Table 1 reports the evaluation results on the test set. BERTScore (Zhang et al., 2019) values exceeding 90% demonstrate strong semantic alignment between the generated and target tags. High ROUGE-L and ROUGE-1 scores (Lin, 2004) further confirm the model's ability to recover a substantial portion of target tokens in the correct sequential order.

RQ3: How do different indexers compare in terms of retrieval success rate? To address this question, seven indexing methods are evaluated using Hit Ratio at k_2, which measures the proportion of queries with at least one entailed question among the top-k_2 retrieved candidates. The comparative results are reported in Appendix, where the `all-mpnet-base-v2` model emerges as the best-performing indexer.

RQ4: How much does user knowledge profiling contribute to accurately recognising entailed questions? In the fourth experimental scenario, we examine the effectiveness of the proposed post-retriever and assess the individual contributions of user knowledge incorporation and tag diversity regulation through ablation studies. The variant fine-tuned without user knowledge in the prompts is referred to as *user-agnostic RQE*, while the variant fine-tuned without data refinement is denoted as *user-aware RQE (no clustering)*. We further benchmark our model against two recent RQE approaches: MQU (Mrini et al., 2021b), a multi-task model built on $\texttt{BART}_{\texttt{large}}$ that jointly learns RQE and question summarisation using a gradually soft parameter-sharing strategy; and ReQuEST (Aftabi et al., 2024), a compact model comprising a $\texttt{BART}_{\texttt{base}}$ encoder and two decoders, jointly trained for RQE, tag generation, and tag-focused question summarisation. Figure 4 illustrates the loss dynamics during fine-tuning. In Figure 4(a), training loss is plotted on a logarithmic scale over training steps, revealing a consistent downward trend across all models. The two user-aware variants converge more rapidly than the user-agnostic model, indicating that incorporating user knowledge accelerates learning. Figure 4(b) presents validation loss across epochs, highlighting overfitting in all models after several epochs and motivating the use of early stopping. The proposed model achieves the lowest validation loss, suggesting superior generalisation and stability. In contrast, the user-agnostic model overfits earlier and generalises poorly, while the variant without clustering exhibits moderate performance, underscoring the benefit of tag diversity regulation.

Table 2 reports the final performance comparison on the test dataset. Among user-agnostic models, MQU establishes a strong baseline with 94.55% accuracy, while ReQuEST slightly outperforms it with 95%. The fine-tuned LLaMA-2 model achieves comparable accuracy, narrowly trailing ReQuEST and substantially outperforming GPT-4o, which suffers from low recall despite high precision. When shifting to user-aware models, GPT-4o exhibits notable improvements, with a 6.1% increase in accuracy and a 12.45% gain in recall; nevertheless, it remains inferior to the LLaMA-2-based models. The proposed approach outperforms all competitors, achieving a dominant F1 score of 96.58%, corresponding to a 1.69% improvement over its user-agnostic counterpart. This gain reflects a more balanced trade-off between precision and recall, demonstrating the effectiveness of the proposed knowledge-augmented LLM-based approach for entailment recognition.

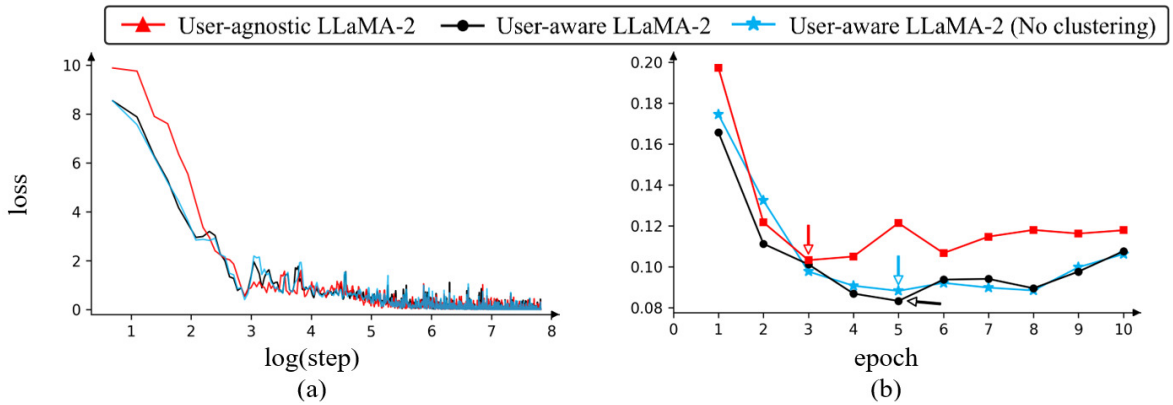

Figure 4: Loss dynamics during fine-tuning. (a) Training loss variation on a logarithmic scale, and (b) validation loss across epochs, with arrows indicating early stopping points before overfitting occurs.

Model	R-1		R-2		R-L		BS		B1	B2	M	P
	F1	Re	F1	Re	F1	Re	F1	Re				[0,∞)
QA data from CQADupStack												
User-agnostic RAG	22.7	32.5	5.3	8.3	12.9	20.2	82.8	82.3	13.5	5.8	14.9	27.6
User-aware RAG (ours)	26.3	**41.5**	6.3	10.8	14.7	**25.5**	83.0	**83.4**	**16.5**	7.1	**18.5**	26.0
Pre-trained GPT-4o	**26.8**	35.7	**7.9**	**11.4**	**16.5**	23.9	83.4	**83.8**	15.4	**7.4**	17.0	**26.47**
QA data from SE-PQA												
User-agnostic RAG	28.7	39.4	6.3	8.6	14.6	21.5	83.2	83.4	17.9	7.4	18.5	**31.0**
User-aware RAG (ours)	**29.6**	**42.2**	**6.6**	**9.6**	**14.8**	**22.7**	**83.2**	**83.7**	**18.5**	**7.7**	**19.1**	32.3
Pre-trained GPT-4o	27.9	35.0	5.7	7.5	14.5	19.6	83.1	83.2	15.8	6.2	15.9	39.27

Table 3: Quantitative analysis of the user-aware RAG system vs. a user-agnostic RAG baseline and pre-trained GPT-4o. (R-1/R-2/R-L: ROUGE-1/2/L, BS: BERTScore, B1/B2: BLEU-1/2, M: METEOR, P: perplexity. Re: recall-based.)

RQ5: How effective is the proposed system in improving answer quality from multiple perspectives? In the final experimental scenario, the parameter k_3 is tuned using values of 3, 5, and 7, with $k_3 = 3$ selected based on F1 performance. Subsequently, three candidate generators are evaluated, and GPT-4o is identified as the most effective model; detailed results are provided in Appendix. Using this configuration, we compare the proposed knowledge- and retrieval-augmented generation system with two baselines: a user-agnostic RAG system and a pre-trained GPT-4o model.

Quantitative evaluation is conducted using BERTScore, ROUGE, BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and perplexity, with results summarised in Table 3. The proposed system consistently outperforms the user-agnostic RAG baseline across all metrics, indicating improved semantic alignment and lexical overlap due to user knowledge integration. Although pre-trained GPT-4o attains marginally higher scores on CQADupStack, the proposed system surpasses it on SE-PQA across all evaluation metrics, demonstrating its competitiveness despite not relying on pre-trained parametric knowledge.

Qualitative evaluation is performed by six AI evaluators and two human experts, who assess generated answers using a structured prompt containing the input query, the gold answer, the three system outputs, and a set of evaluation criteria. Each answer is scored on a scale from 0 to 5 per criterion, and the systems are ranked by overall quality. Figure 5 presents the average AI-evaluator scores using boxplots. While pre-trained GPT-4o achieves the highest overall scores, the proposed system consistently ranks second and substantially outperforms the user-agnostic baseline. The performance gap between the proposed system and GPT-4o is notably smaller for hallucination, completeness, and depth and detail. Table 4 reports

the scores assigned by two human evaluators to 74 randomly selected questions, further confirming the superiority of the proposed system. Additional details are provided in Appendix.

RQ6: Is the response time of the proposed system acceptable for practical deployment? Although real-time responses are not a strict requirement for Q&A platforms, fast turnaround remains desirable. Table 12 reports the average processing time per user for each system component, identifying the post-retriever as the primary bottleneck. However, since retrieved question pairs can be evaluated independently, parallelisation is feasible and could reduce the post-retrieval delay to approximately 0.36 seconds. Overall, the proposed system is capable of delivering accurate and reliable responses in under 30 seconds, making it suitable for practical deployment in community Q&A environments.

5 Conclusions and Future Work

This paper introduced a user-oriented retrieval-augmented generation (RAG)-based community question answering system that harnesses a chain of large language models (LLMs) to generate human-like, contextually grounded answers with minimal hallucination. The proposed system reframes question entailment recognition (RQE) as a generative task and integrates it as a post-retrieval filtering step within the conventional RAG pipeline. Crucially, user awareness is incorporated by profiling user knowledge as a sequence of thematic tags and injecting this information as supplementary context into the RQE process.

Experimental results indicate the superiority of our system over its user-agnostic counterpart, with marked gains in factual accuracy (hallucination), correctness, relevance, and completeness. The evidence highlighted the effectiveness of infusing user context in reducing the performance gap between RAG-based models and pre-trained LLMs, which are costly to maintain as they must be pre-trained on domain-specific data. In future work, we plan to explore soft-prompt tuning techniques as substitutes for hard prompts, refine user profiling by analyzing behavioral patterns across temporal windows, extend the system to support cross-lingual entailment recognition, and adapt it for deployment in specialized domains, such as healthcare and financial industries

Limitations

While our proposed system is cost-effective and requires only a limited number of fine-tuning iterations, its overall performance is influenced by several hyperparameters, such as the number of recent questions from users' activity history analyzed for knowledge inference, the number of similar questions retrieved by the base retriever, and the number of accepted answers aggregated as the final response. Exhaustive exploration of these parameters required substantial computational resources, particularly GPU-intensive trials, which were not feasible under our current conditions. For example, restricting the retrieval size to the top 50 questions may eliminate the chance of truly entailed questions being included in the pool, potentially amplifying the risk of hallucination. Another notable constraint pertains to the use of GPT-4o, which, despite its capabilities, incurs a non-negligible cost. While 20 dollers per month may appear affordable, it remains a substantial barrier in many countries. We relied on its free web interface for answer generation, which restricted our ability to perform extensive prompt engineering and made us limit the size of our QA dataset. Given the known sensitivity of LLMs to the prompt structure, this limitation may have affected our system's performance. Moreover, as different LLMs may yield different results, we recommend that future work empirically assess other LLMs as substitutes or complements to GPT-4o. Our evaluation strategy also imposes several constraints. All AI evaluators were accessed via their publicly available web interfaces in reasoning-enabled mode, requiring substantial manual effort to prompt them and gather their scores.

References

Asma B. Abacha and Dina Demner-Fushman. 2019. A question-entailment approach to question answering. *BMC Bioinformatics*, 20(1).

Asma B. Abacha, Dina Demner-Fushman, and U S National Library. 2016. Recognizing question entailment for medical question answering. In *AMIA Annual Symposium Proceedings*, pages 310–318.

Amogh Subbakrishna Adishesha, Lily Jakielaszek, Fariha Azhar, Peixuan Zhang, Vasant Honavar, Fenglong Ma, Chandra Belani, Prasenjit Mitra, and Sharon Xiaolei Huang. 2023. Forecasting user interests through topic tag predictions in online health communities. *IEEE Journal of Biomedical and Health Informatics*, 27(7):3645–3656.

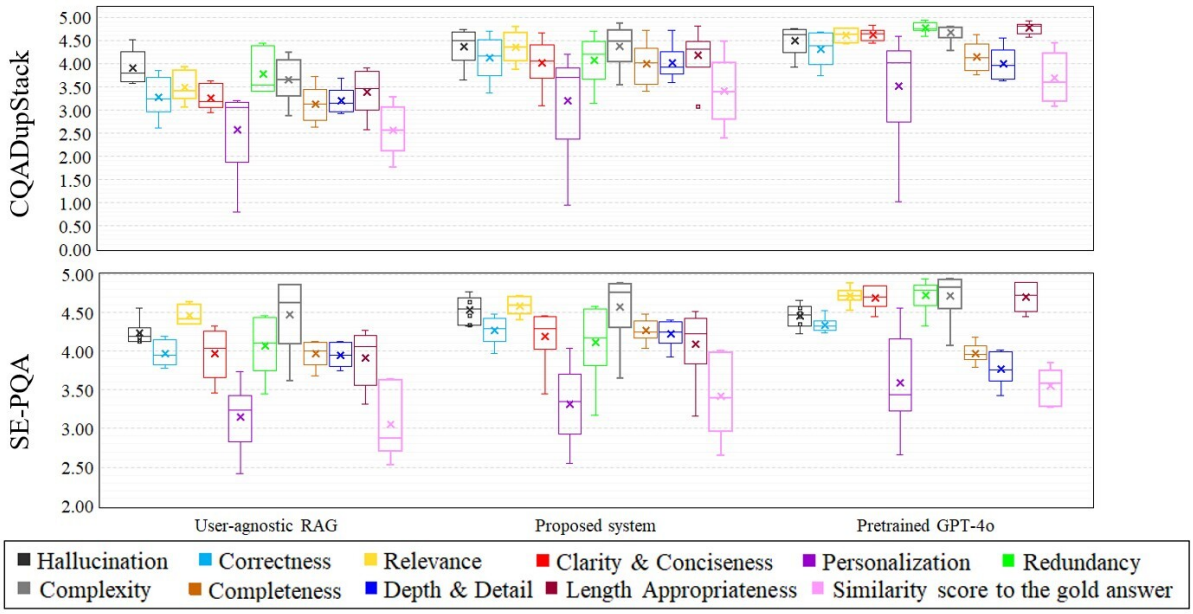

Figure 5: Boxplots of average qualitative scores assigned by six AI evaluators based on 11 criteria.

Model	1	2	3	4	5	6	7	8	9	10	11	Rate(%)
User-agnostic RAG	4.51	3.74	4.01	3.71	3.85	4.44	4.21	3.64	3.51	4.43	3.11	8.8
User-aware RAG (ours)	**4.75**	**4.66**	**4.84**	**4.73**	**4.64**	4.71	**4.77**	**4.61**	**4.55**	4.60	**4.07**	**52.0**
Pre-trained GPT-4o	4.67	4.46	4.68	4.61	4.59	**4.80**	4.76	4.37	4.11	**4.71**	4.00	39.9

Table 4: Average qualitative scores assigned by two human experts to 74 test questions from CQADupStack. Criteria: 1 Hallucination, 2 Correctness, 3 Relevance, 4 Clarity & Conciseness, 5 Personalization, 6 Redundancy, 7 Complexity, 8 Completeness, 9 Depth & Detail, 10 Length appropriateness, 11 Similarity to gold. Rate: selection rate as best answer.

Seyyede Zahra Aftabi, Seyyede Maryam Seyyedi, Mohammad Maleki, and Saeed Farzi. 2024. Request: A small-scale multi-task model for community question-answering systems. *IEEE Access*, 12:17137–17151.

Anumeha Agrawal, Rosa Anil George, Selvan Suntiha Ravi, Sowmya Kamath S, and Anand Kumar. 2019. Ars_nitk at mediqa 2019: Analysing various methods for natural language inference, recognising question entailment and medical question answering system. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 533–540, Stroudsburg, PA, USA. Association for Computational Linguistics.

Waad Thuwaini Alshammari and Sarah AlHumoud. 2022. Taqs: An arabic question similarity system using transfer learning of bert with bilstm. *IEEE Access*, 10:91509–91523.

Anitha Anandhan, Maizatul Akmar Ismail, and Liyana Shuib. 2022. Expert recommendation through tag relationship in community question answering. *Malaysian Journal of Computer Science*, 35(3):201–221.

Yadagiri Annepaka and Partha Pakray. 2025. Large language models: a survey of their development, capabilities, and applications. *Knowledge and Information Systems*, 67(3):2967–3022.

Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K. Roy, and Kevin A. Schneider. 2013. Answering questions about unanswered questions of stack overflow. In *2013 10th Working Conference on Mining Software Repositories (MSR)*, pages 97–100. IEEE.

Dibyanayan Bandyopadhyay, Baban Gain, Tanik Saikh, and Asif Ekbal. 2019. Iitp at mediqa 2019: Systems report for natural language inference, question entailment and question answering. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 517–522, Stroudsburg, PA, USA. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.

Gordon Burtch, Dokyun Lee, and Zhichen Chen. 2024. The consequences of generative ai for online knowledge communities. *Scientific Reports*, 14(1):10413.

Gianni Costa and Riccardo Ortale. 2023. Ask and ye shall be answered: Bayesian tag-based collaborative recommendation of trustworthy experts over time in community question answering. *Information Fusion*, 99:101856.

R Maria del Rio-Chanona, Nadzeya Laurentsyeva, and Johannes Wachs. 2024. Large language models reduce public knowledge sharing on online q&a platforms. *PNAS Nexus*, 3(9):1–13.

Shima Ghasemi and Azadeh Shakery. 2024. Harnessing the power of metadata for enhanced question retrieval in community question answering. *IEEE Access*, 12:65768–65779.

Travis R. Goodwin and Dina Demner-Fushman. 2019. Bridging the knowledge gap: Enhancing question answering with world and domain knowledge. *Computing Research Repository (CoRR)*, abs/1910.07429.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864.

Tongze He, Caili Guo, and Yunfei Chu. 2021. Enhanced user interest and expertise modeling for expert recommendation. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 556–562. IEEE.

Doris Hoogeveen, Andrew Bennett, Yitong Li, Karin Verspoor, and Timothy Baldwin. 2018. Detecting misflagged duplicate questions in community question-answering archives. *Proceedings of the International AAAI Conference on Web and Social Media*, 12(1):112–120.

Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. 2015. Cqadupstack: A benchmark data set for community question-answering research. In *Proceedings of the 20th Australasian Document Computing Symposium*, pages 1–8, New York, NY, USA. ACM.

Xiaowei Huang, Wenjie Ruan, Wei Huang, Gaojie Jin, Yi Dong, Changshun Wu, Saddek Bensalem, Ronghui Mu, Yi Qi, Xingyu Zhao, Kaiwen Cai, Yanghao Zhang, Sihao Wu, Peipei Xu, Dengyu Wu, Andre Freitas, and Mustafa A. Mustafa. 2024. A survey of safety and trustworthiness of large language models through the lens of verification and validation. *Artificial Intelligence Review*, 57(7):175.

Samia Kabir, David N. Udo-Imeh, Bonan Kou, and Tianyi Zhang. 2024. Is stack overflow obsolete? an empirical study of the characteristics of chatgpt answers to stack overflow questions. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–17, New York, NY, USA. ACM.

Kamal Raj Kanakarajan, Bhuvana Kundumani, Abhijith Abraham, and Malaikannan Sankarasubbu. 2022. Biosimcse: Biomedical sentence embeddings using contrastive learning. In *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis*, pages 81–86, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pranav Kasela, Marco Braga, Gabriella Pasi, and Raffaele Perego. 2024. Se-pqa: Personalized community question answering. In *Companion Proceedings of the ACM Web Conference 2024*, pages 1095–1098, New York, NY, USA. ACM.

Vaibhav Krishna and Nino Antulov-Fantulin. 2023. Temporal-weighted bipartite graph model for sparse expert recommendation in community question answering. In *Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization*, pages 156–163, New York, NY, USA. ACM.

C. Y. Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, volume 1, pages 25–26.

R. Menaha and V. E. Jayanthi. 2023. Finding experts in community question answering system using trie string matching algorithm with domain knowledge. *IETE Journal of Research*, pages 1–13.

Andreea Maria Monea and Anca Nicoleta Marginean. 2021. Medical question entailment based on textual inference and fine-tuned biomed-roberta. In *2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 319–326. IEEE.

Khalil Mrini, Franck Dernoncourt, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021a. Joint summarization-entailment optimization for consumer health question understanding. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 58–65, Stroudsburg, PA, USA. Association for Computational Linguistics.

Khalil Mrini, Franck Dernoncourt, Seunghyun Yoon, Trung Bui, Walter Chang, Emilia Farcas, and Ndapa Nakashole. 2021b. A gradually soft multi-task and data-augmented approach to medical question understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1505–1515, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sara Mumtaz, Carlos Rodriguez, and Boualem Benatallah. 2019. Expert2vec: Experts representation in community question answering for question routing. In *Advanced Information Systems Engineering. CAiSE 2019*, volume 11483 of *LNCS*, pages 213–229, Cham. Springer.

Vincent Nguyen, Sarvnaz Karimi, and Zhenchang Xing. 2021. Combining shallow and deep representations for text-pair classification. In *Proceedings of the 19th Workshop of the Australasian Language Technology Association (ALTA 2021)*, pages 68–78.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318.

Pradeep Kumar Roy. 2020. Multilayer convolutional neural network to filter low quality content from quora. *Neural Processing Letters*, 52(1):805–821.

Pradeep Kumar Roy, Sunil Saumya, Jyoti Prakash Singh, Snehasish Banerjee, and Adnan Gutub. 2023. Analysis of community question-answering issues via machine learning and deep learning: State-of-the-art review. *CAAI Transactions on Intelligence Technology*, 8(1):95–117.

Mourad Sarrouti, Asma B. Abacha, and Dina Demner-Fushman. 2021. Multi-task transfer learning with data augmentation for recognizing question entailment in the medical domain. In *2021 IEEE 9th International Conference on Healthcare Informatics (ICHI)*, pages 339–346. IEEE.

Yong Sun and Junfang Song. 2023. Research on question retrieval method for community question answering. *Multimedia Tools and Applications*, 82:24309–24325.

Noha Tawfik and Marco Spruit. 2019. Uu_tails at mediqa 2019: Learning textual entailment in the medical domain. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 493–499, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zhuojia Xu and Hua Yuan. 2020. Forum duplicate question detection by domain adaptive semantic matching. *IEEE Access*, 8:56029–56038.

Shweta Yadav, Vishal Pallagani, and Amit Sheth. 2020. Medical knowledge-enriched textual entailment framework. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1795–1801, Stroudsburg, PA, USA. International Committee on Computational Linguistics.

Mohammad Sadegh Zahedi, Maseud Rahgozar, and Reza Aghaeizadeh Zoroofi. 2024. Mater: Bi-level matching-aggregation model for time-aware expert recommendation. *Expert Systems with Applications*, 237:121576.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *Computing Research Repository (CoRR)*, abs/1904.09675:1–43.

Huiwei Zhou, Xuefei Li, Weihong Yao, Chengkun Lang, and Shixian Ning. 2019. Dut-nlp at mediqa 2019: An adversarial multi-task network to jointly model recognizing question entailment and question answering. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 437–445.

Appendices

<s> [INST] <<SYS>>
You are a Tag Generator. Respond only with a list of tags; do not include any additional text or explanations.
<</SYS>>\n
Please generate at least 5 tags for the provided question. Tags can include multi-word phrases if appropriate and should help hierarchically categorize the question's topics.
Question:\n**{Question}**
Tags:
[/INST]\n
{Tags}</s>

Figure 6: The prompt template for fine-tuning the LLaMA-2 model for the tag generation task

A Fine-tuning LLaMA-2 Models

To fine-tune the LLaMA-2 model for tag generation, each data instance is transformed into a structured prompt following the template illustrated in Figure 6. The template conforms to the standard prompt format of the `LLaMA-2-chat` model. During fine-tuning, the placeholders `{Question}` and `{Tags}` are populated with the corresponding data; however, during the testing phase, the `{Tags}` field is left blank. It is worth noting that, apart from the `{Tags}` field, the remaining parts of the prompt are excluded from the cross-entropy loss calculation.

For the RQE task, two additional prompt templates are required. Figure 8 presents the templates designed for user-agnostic and user-aware RQE. The former includes only the bodies of the question pairs, whereas the latter additionally incorporates the user knowledge profile (Type II in Figure 8). During testing, the `{Label}` subsequence is omitted from the prompt. Both models are optimised using cross-entropy loss, computed solely over the generated tokens (i.e., loss over completion only).

B Updating User Knowledge Profiles

To facilitate future updates, user knowledge profiles can be stored in separate JSON files per user, structured as follows:

```
{
  "User_id": {
    "Tags": [[str, ...], ...],
    "Tag_frequencies": {str: int, ...}
  }
}
```

Updates may be performed either periodically or in an event-driven manner. We recommend the

Algorithm 1: Event-driven update of user profile

Inputs: new question (q^{ℓ}_{new}), user id (u^{ℓ}), maximum number of recent questions (k_1)

Output: Updated knowledge profiles (u-ℓ.json)

if *exist*(u-ℓ.json) **then**
  $j \leftarrow$ open u-ℓ.json and parse it
  $T, F \leftarrow j$["Tags"], j["Tag_frequncies"]
else
  $j \leftarrow$ create u-ℓ.json and open it
  $T, F \leftarrow [\,], \{\}$
end
$t' \leftarrow \text{LLaMA}(q^{\ell}_{new})$
if $|T| \geq k_1$ **then**
  $T_{old} \leftarrow T$.pop_front( )
  foreach tags t in T_{old} do
    $F[t] \leftarrow F[t] - 1$
    if $F[t] \leq 0$ then remove t from F
  end
end
append t' to T
foreach tags t in t' do
  if $t \in F$ **then**
    $F[t] \leftarrow F[t] + 1$
  else
    $F[t] \leftarrow 1$
  end
end
store T and F back into u-ℓ.json
return u-ℓ.json

latter, whereby a user's knowledge profile is updated immediately upon question submission, as described in the Algorithm 1.

The proposed update procedure is time-efficient, exhibiting constant time complexity $O(1)$. It is also memory-efficient and scalable, with storage requirements growing linearly with the number of users, i.e., $O(N)$. Assuming one byte per character under UTF-8 encoding, each user requires, on average, approximately $k_1 \times 2L$ bytes to store their profile, where L denotes the average number of characters in a tag sequence.

C Configuration and Setting Parameters

The proposed system is implemented in Python and executed in the Google Colab environment using an NVIDIA A100-SXM4 GPU, with 83.5 GB of system RAM and 40 GB of dedicated GPU memory. Table 6 lists all hyperparameters and their initial values. The chat version of LLaMA-2 with 7 billion parameters is employed, incorporating 33.6 million trainable parameters through Low-Rank Adaptation (LoRA).

D Hierarchical Clustering of Tags

The dendrogram for the CQADupStack dataset, shown in Figure 7, is used to establish three clustering thresholds at distances of 32, 16, and 8, indicated by horizontal dotted lines. Each threshold implies that clusters with distances equal to or greater than the specified value remain unmerged. Similarly, thresholds of 60, 35, and 17 are selected for the SE-PQA dataset. Clustering performance is evaluated using the Silhouette coefficient, the Calinski–Harabasz index, and the Davies–Bouldin index, with results reported in Table 8.

For the CQADupStack dataset, Figure 9 visualises tag clusters at the three hierarchical levels. The t-SNE (t-distributed stochastic neighbour embedding) algorithm is used to project tag embeddings of size 1×128 into a two-dimensional space. As illustrated in Figure 10, dataset refinement not only modifies the distribution of tag counts but also affects the token counts within tag sequences.

E Indexer Selection

Table 5 reports the average time required to generate vector representations for archived questions. In addition, for $k_2 \in \{10, 20, 50\}$, we report the number of retrieved questions containing at least one entailed candidate in the top-k_2 results.

F Generator Selection

Figure 11 illustrates the prompt template used to generate answers from the accepted answers of the top-k_3 entailed questions. The performance of three LLMs—LLaMA-2, GPT-4o-mini, and GPT-4o—used as the generator component is compared in Table 9, where GPT-4o achieves the highest performance across all metrics except perplexity. Furthermore, Table 10 reports the answer quality evaluation of GPT-4o with varying values of k_3 ($k_3 \in \{3, 5, 7\}$), among which $k_3 = 3$ yields the best F1-based performance.

G Quantitative and Qualitative Analysis

Figure 12 presents the prompt template used to request answer generation from the pre-trained GPT-4o model. Using an additional prompt template shown in Figure 13, the quality of these responses is evaluated and compared with those generated by the proposed system. This evaluation template defines 11 criteria to support a comprehensive assessment of answer quality from multiple perspectives.

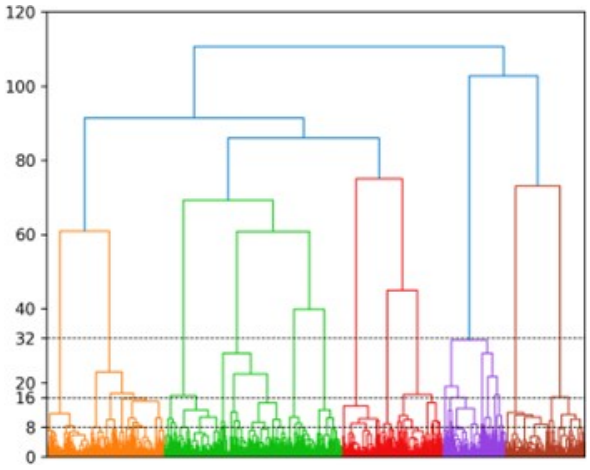

Figure 7: The dendrogram with selected thresholds for hierarchical clustering of tags in CQADupStack.

method	**HR@k_2**			**avg time***
	10	20	50	
bert-base-cased	3	5	5	2.89e-2
unsup-simcse-roberta-base	12	17	21	1.71e-2
tf-idf	26	30	50	**1.03e-4**
paraphrase-minilm-l6-v2	32	33	46	9.63e-4
paraphrase-multilingual-minilm-l12-v2	38	42	47	1.71e-3
jina-colbert-v1-en	40	47	60	3.86e-3
all-mpnet-base-v2	**72**	**80**	**95**	3.97e-2

Table 5: Comparative analysis of indexing methods. *Average indexing time per question on a T4 GPU (seconds).

A criterion-wise statistical comparison of the three systems is reported in Table 11, including 95% confidence intervals and pairwise p-values computed based on scores from six AI evaluators. The mean average qualitative scores are further summarised in Table 12.

Model/System	**Hyperparameter**	**Value**
Node2vec	p	1
	q	0.5
	d	128
	`walk_length`	10
	`num_walks`	60
	`window`	10
Agglomerative clustering	metric	euclidean
	linkage	ward
	`n_clusters`	none
	thresholds	32, 16, 8
LLaMA-2	`max_epochs`	10
	learning_rate	$1e{-}4^*$, $3e{-}5^\diamond$
	`batch_size`	32
	`max_seq_length`	512^*, $750^\diamond$
	`max_new_tokens`	30^*, $1^\diamond$
	`lora_r`	64^*, $64^\diamond$
	`lora_alpha`	64^*, $16^\diamond$
	`lora_dropout`	0.1^*, $0.3^\diamond$
	`torch_dtype`	bfloat16
System	k_1, k_2, k_3	10, 50, 3

Table 6: Values of setting parameters. $*$ used for tag generation; $\diamond$ used for the RQE task.

Component	**Time (s)**	**Description**	**Online**	**GPU**
User profiler	1.680	Analyzing the 10 most recent questions of each user	×	A100
Indexer	0.014	Representing the user's new question	✓	T4
Retriever	0.602	Retrieving the 50 most similar questions from 248,426 questions	✓	T4
Post-retrieval	18.00	Analyzing 50 question pairs sequentially	✓	A100
Generator	5.660	Prompting GPT-4o to generate final answer	✓	–

Table 7: Average processing time of system components for each user.

<s> [INST] <<SYS>>
Help recognize question entailment
<</SYS>>\n
Entailment means:
1. every answer to Q2 must be a partial or complete answer to Q1
2. Q2 must be related to the topics of interest of Q1's asker, denoted by Background Knowledge.
Respond with "positive" for entailment and "negative" for not-entailment. No other words.
Example1:
Q1: How can I read a PDF?
Background Knowledge: python, programming, pandas
Q2: Help me how to open different files such as pdf, docx, etc. in Linux.
Answer: negative

Example2:
Q1: How can I read a PDF?
Background Knowledge: linux, debian, filesystems
Q2: Help me how to open different files such as pdf, docx, etc. in Linux.
Answer: positive
Now, evaluate the following:
Q1: **{Question1}**
Background Knowledge: **{User Knowledge}**
Q2: **{Question2}**
Answer:
[/INST]\n
{Label}

Figure 8: The prompt template for fine-tuning the LLaMA-2 model for the RQE task. The plus signs indicate the additional sequences included in the user-aware RQE prompts (Type II prompts).

	CQADupStack			SE-PQA		
Metric	**Level 1**	**Level 2**	**Level 3**	**Level 1**	**Level 2**	**Level 3**
Silhouette Score	0.48	0.22	0.19	0.36	0.47	0.38
Calinski–Harabasz Index	10069.97	254.78	98.44	12586.96	1617.98	662.31
Davies–Bouldin Index	0.68	1.88	3.09	2.52	0.99	1.28
Average within-cluster mean distances	107.08	81.55	70.05	308.78	192.41	122.35
Average cluster size (Min, Max)	963 (342,1580)	462 (58,974)	141 (4,491)	2060 (994,4800)	956 (307,2866)	418 (97,1790)

Table 8: Clustering evaluation metrics across three hierarchical levels.

Model	**R-1**		**R-2**		**R-L**		**BS**		**P**	
	F1	**Re**	**F1**	**Re**	**F1**	**Re**	**F1**	**Re**	**[0,∞)**	**F1**
LLaMA-2	23.6	39.8	4.8	9.0	13.5	25.2	82.3	**82.7**	14.4	15.25
GPT-4o-mini	24.6	**43.1**	5.5	10.4	13.2	**25.6**	82.5	83.1	14.9	29.18
GPT-4o	**26.3**	41.5	**6.3**	**10.8**	**14.7**	25.5	**83.0**	83.4	**16.5**	**25.99**

Table 9: Comparative analysis of various LLMs as the generator component using the CQADupStack dataset. R-1: ROUGE-1, R-2: ROUGE-2, R-L: ROUGE-L, BS: BERTScore, B1: BLEU-1, B2: BLEU-2, M: METEOR, P: Perplexity, F1: F1-based, Re: Recall-based.

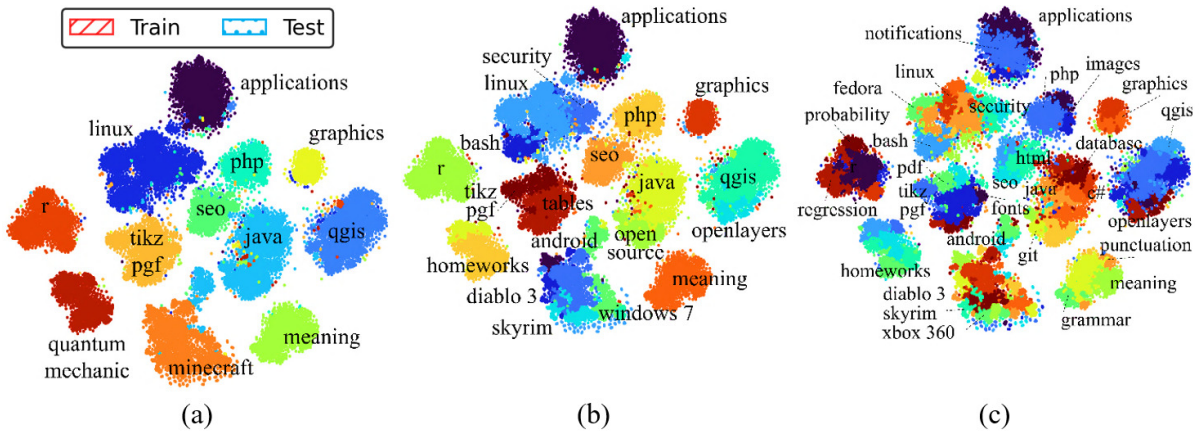

Figure 9: Three-level hierarchical clusters of tags in CQADupStack: (a) level 1 comprising 12 clusters (threshold = 32), (b) level 2 having 25 clusters (threshold = 16), and (c) level 3 containing 82 clusters (threshold = 8).

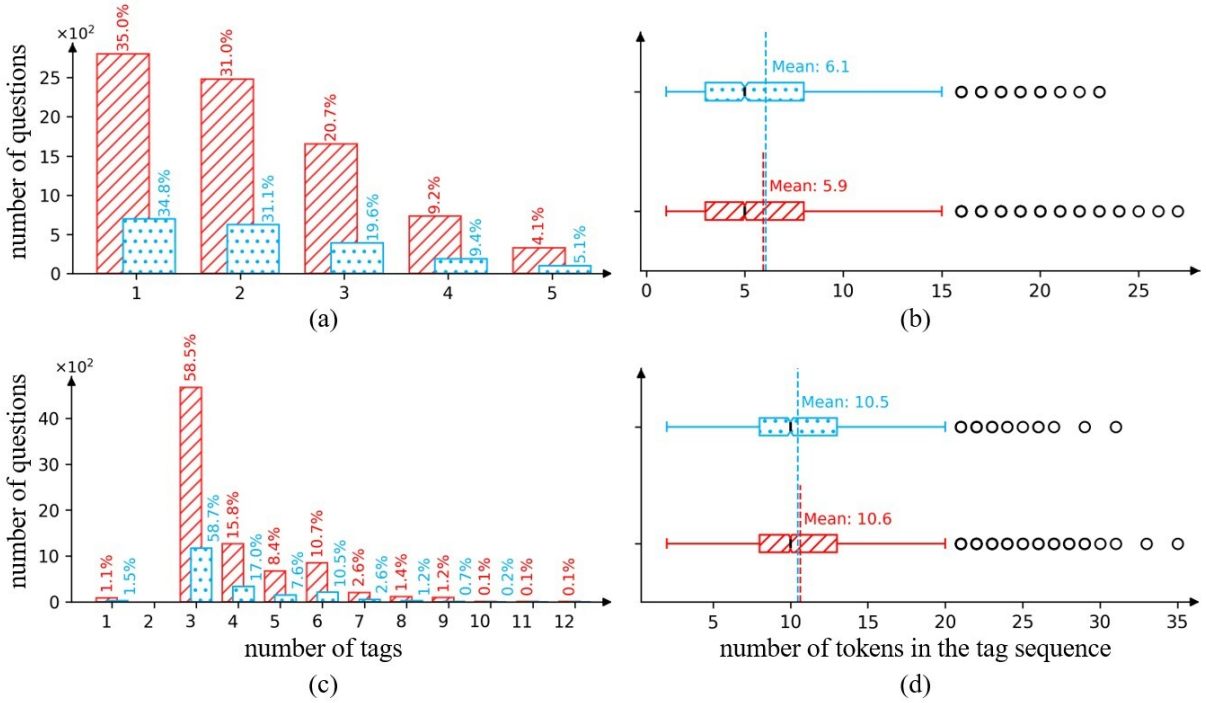

Figure 10: Distribution of questions in the training and test sets from the CQADupStack based on the number of tags, and the number of tokens in their tag sequences, (a and b) before and (c and d) after data refinement.

Model	R-1		R-2		R-L		BS		P	
	F1	Re	F1	Re	F1	Re	F1	Re	[0,∞)	F1
$k_3 = 3$	**26.3**	41.5	6.3	10.8	14.7	25.5	83.0	83.4	16.5	25.99
$k_3 = 5$	25.6	44.1	6.2	11.7	14.2	27.2	82.9	83.5	15.5	18.9
$k_3 = 7$	25.7	**44.8**	**6.1**	**11.6**	**14.3**	**27.8**	**82.8**	**83.5**	**15.7**	**19.0**

Table 10: Comparing the effect of the number of candidate answers on the quality of generated responses using the CQADupStack dataset. R-1: ROUGE-1, R-2: ROUGE-2, R-L: ROUGE-L, BS: BERTScore, B1: BLEU-1, B2: BLEU-2, M: METEOR, P: Perplexity, F1: F1-based, Re: Recall-based.

You are an AI language model tasked with synthesizing an accurate, complete, and well-structured answer based solely on the provided expert-written answers. Follow these strict guidelines:

- Analyze each provided answer carefully.
- Extract relevant words, phrases, sentences, or subsequences that contribute to answering the given question.
- Synthesize a comprehensive, continuous, and well-structured response without any headings, subheadings, or explicit references to the original answers (e.g., do not say "as stated in Answer 1").
- Rephrase extracted content where necessary to align with the exact requirements of the question. If an answer provides a solution to a slightly different but related problem, adapt the phrasing while preserving factual accuracy.
- Incorporate all relevant information from the answers, ensuring that multiple valid solutions, perspectives, or explanations are included where applicable. No relevant information should be omitted.
- Remove non-relevant parts that do not contribute to answering the question.
- Do not introduce any external information beyond what is contained in the provided answers. If an answer is not covered in the provided content, do not generate additional details from external knowledge.
- If none of the provided answers sufficiently address the question, clearly state: "The question could not be answered based on the available context."
- Prioritize precision over recall, ensuring that responses are accurate and directly relevant to the question. However, the answer should also be as complete as possible while maintaining clarity and conciseness.

Now, based on these instructions, analyze the following question and answers, then generate the best possible response.

Question: **{Question}**
Answer1: **{CandidateAnswer1}**
Answer2: **{CandidateAnswer2}**
Answer3: **{CandidateAnswer3}**
Response:

Figure 11: Distribution of questions in the training and test sets from the CQADupStack based on the number of tags, and the number of tokens in their tag sequences, (a and b) before and (c and d) after data refinement.

Answer the following question with high accuracy, ensuring that all relevant aspects are covered. Try to keep your answer under 300 tokens. Adjust the explanation depth and the terminology based on the question's complexity—use simple, example-driven explanations for simple questions, but provide technical, domain-specific details for complex ones. Maintain conciseness without sacrificing clarity or completeness. Provide short, high-quality examples to reinforce key points.

Now, answer this question:
Question: **{Question}**
Answer:

Figure 12: Distribution of questions in the training and test sets from the CQADupStack based on the number of tags, and the number of tokens in their tag sequences, (a and b) before and (c and d) after data refinement.

- Instruction:
You are given a Question, four AI-generated Answers (Answers 1 to 4), and a Gold Answer written by a human for reference. You are also provided with the User Knowledge, which describes the question asker's main areas of interest or knowledge. As an AI language model, Your task is to evaluate and compare Answers 1–4 based on the following 11 criteria, each scored on a 0–5 scale (0 = very poor, 5 = excellent).

- Inputs:
Question: **{Question}**
User Knowledge: **{User knowledge}**
Answer1: **{Generated answer 1}**
Answer2: **{Generated answer 2}**
Answer3: **{Generated answer 3}**
Gold Answer: **{Accepted answer}**

- Evaluation Criteria:
1. Hallucination (Factual correctness): Does the response contain false or misleading information based on world knowledge? (Lower score if hallucinations are present.)
2. Correctness: Is the information in the response factually and contextually accurate? Does it correctly address the question?
3. Relevance: Does the response directly address the question requirements and align with the user's specific information needs?
4. Clarity & Conciseness: Is the response clearly written, logically structured, and free from unnecessary verbosity? Is it easy to understand?
5. Personalization: Does the response adapt appropriately to the user's stated interests, background, and preferences?
6. Redundancy: Does the response avoid unnecessary repetition of ideas or content? (Lower score if redundancy exist)
7. Complexity Appropriateness: Is the level of complexity appropriate for the user's knowledge level? (Not too technical for beginners, not oversimplified for experts.)
8. Completeness: Does the response fully address all parts of the question, including any sub-questions?
9. Depth & Detail: Does the response provide sufficient explanation, background, and supporting details to meet the user's information needs?
10. Length Appropriateness: Is the response an appropriate length—neither too brief to be helpful nor overly long?
11. Comparative Score vs. Human Answer: How closely does the AI response match the quality, clarity, and meaning of the human-written Gold Answer?

- Output Format:
Provide a table of scores (0-5) for each criterion for all answers.
Provide another table with brief justifications for each score.
Rank the answers from best to worst.
Finally, Indicate which response is the best, and why.

Figure 13: The prompt template designed for qualitative assessment of generated responses.

Metric	Confidence Interval			p-value		
	UA RAG vs Proposed	UA RAG vs GPT-4o	Proposed vs GPT-4o	UA RAG vs Proposed	UA RAG vs GPT-4o	Proposed vs GPT-4o
Hallucination	[0.24, 0.68]	[-0.80, -0.37]	[-0.30, 0.05]	2.9E-08	1.1E-08	7.7E-02
Correctness	[0.54, 1.14]	[-1.33, -0.75]	[-0.39, 0.00]	3.6E-12	1.7E-12	1.2E-02
Relevance	[0.58, 1.15]	[-1.39, -0.83]	[-0.40, -0.10]	2.2E-11	1.1E-13	1.3E-04
Clarity & Conciseness	[0.56, 0.92]	[-1.53, -1.18]	[-0.72, -0.51]	5.5E-17	7.4E-31	4.6E-19
Personalization	[0.40, 0.83]	[-1.15, -0.73]	[-0.46, -0.18]	2.0E-12	4.0E-17	3.2E-09
Redundancy	[0.17, 0.42]	[-1.10, -0.89]	[-0.79, -0.60]	1.7E-08	1.4E-37	7.6E-28
Complexity Appropriateness	[0.48, 0.97]	[-1.27, -0.78]	[-0.40, -0.20]	1.1E-09	3.6E-14	6.7E-09
Completeness	[0.60, 1.14]	[-1.28, -0.75]	[-0.31, 0.01]	8.7E-14	3.1E-12	4.1E-02
Depth & Detail	[0.55, 1.08]	[-1.06, -0.54]	[-0.12, 0.15]	8.4E-12	7.2E-08	8.2E-01
Length Appropriateness	[0.57, 1.00]	[-1.58, -1.17]	[-0.68, -0.50]	5.8E-13	6.7E-27	1.0E-23
Similarity to gold	[0.60, 1.10]	[-1.36, -0.87]	[-0.46, -0.08]	1.0E-16	1.1E-17	1.0E-03
QA data from CQADupStack						
Hallucination	[-0.03, 0.61]	[-0.56, 0.12]	[-0.22, 0.38]	2.8E-02	1.8E-01	5.2E-01
Correctness	[-0.17, 0.76]	[-0.79, 0.05]	[-0.46, 0.32]	5.5E-04	6.7E-02	7.0E-01
Relevance	[-0.27, -0.52]	[-0.58, 0.07]	[-0.43, 0.17]	1.2E-01	1.3E-01	4.1E-01
Clarity & Conciseness	[0.06, 0.39]	[-0.88, -0.57]	[-0.66, -0.34]	8.6E-05	1.1E-11	2.8E-07
Personalization	[-0.12, 0.46]	[-0.69, -0.19]	[-0.52, -0.02]	4.3E-03	5.0E-04	2.4E-02
Redundancy	[-0.11, 0.20]	[-0.80, -0.52]	[-0.75, -0.48]	5.2E-01	1.4E-11	9.0E-12
Complexity Appropriateness	[-0.13, 0.33]	[-0.43, 0.05]	[-0.30, 0.02]	2.2E-02	1.5E-02	8.5E-02
Completeness	[-0.12, 0.72]	[-0.36, 0.34]	[-0.05, 0.62]	2.2E-03	9.7E-01	1.3E-01
Depth & Detail	[-0.12, 0.67]	[0.15, 0.51]	[0.14, 0.77]	9.7E-03	3.6E-01	1.4E-02
Length Appropriateness	[-0.04, 0.40]	[-0.96, -0.61]	[-0.78, -0.43]	5.3E-03	1.1E-11	3.2E-08
Similarity to gold	[-0.05, 0.77]	[-0.87, -0.13]	[-0.52, -0.25]	1.8E-05	1.4E-02	5.1E-01

Table 11: Statistical analysis of answer generation systems.

Metric	CQADupStack			SE-PQA		
	UA RAG	Proposed	GPT-4o	UA RAG	Proposed	GPT-4o
Hallucination	3.91	4.37	4.50	4.24	4.53	4.45
Correctness	3.28	4.12	4.32	3.97	4.27	4.34
Relevance	3.49	4.35	4.60	4.46	4.58	4.71
Clarity & Conciseness	3.27	4.01	4.62	3.97	4.19	4.69
Personalization	2.58	3.20	3.52	3.15	3.32	3.59
Redundancy	3.78	4.07	4.77	4.06	4.11	4.72
Complexity Appropriateness	3.64	4.37	4.67	4.47	4.57	4.71
Completeness	3.13	3.99	4.14	3.97	4.26	3.97
Depth & Detail	3.20	4.01	4.00	3.95	4.22	3.77
Length Appropriateness	3.39	4.18	4.76	3.92	4.10	4.70
Similarity score to gold	2.56	3.40	3.67	3.05	3.41	3.54

Table 12: Mean qualitative scores provided by six AI evaluators.