Multimedia Systems (2026) 32:510
https://doi.org/10.1007/s00530-026-02559-5

REGULAR PAPER

STaR: a soft-labeling and triplet-aware retriever for efficient retrieval-augmented QA

Jia-Yang Jiang[1,2] · Chih-Yung Chang[1] · Youxi Li[3] · Tzu-Chia Huang[1] · Diptendu Sinha Roy[4]

Received: 18 October 2025 / Accepted: 12 July 2026

Abstract
Retrieval-augmented generation (RAG) has emerged as a powerful paradigm for improving the factuality and grounding of large language models (LLMs). However, existing RAG systems suffer from several persistent limitations, including retrieval inefficiency, high computational costs, and limited semantic understanding due to the reliance on hard labels and recursive retrieval methods. To address these issues, this study proposes STaR, a novel retriever fine-tuning framework that integrates BM25 similarity graph-based soft labeling with a triplet similarity learning strategy based on Sentence-BERT (SBERT). Our approach constructs graded semantic labels through BM25 and similarity graph traversal, overcoming the binary constraints of traditional supervision and enriching the learning signal for more precise vector alignment. Furthermore, this study introduces a triplet-aware SBERT training architecture that explicitly models relative semantic distances between queries and candidate passages, significantly enhancing retrieval ranking precision and semantic robustness. This study also adopts a dual-level evaluation strategy that combines lexical level and semantic level accuracy, leveraging their complementary strengths to better reflect retrieval quality and enhance downstream generation. Experimental results on a proprietary dataset of 98,037 QA pairs demonstrate that our method outperforms strong baselines—achieving HR@5 = 0.93 with consistent gains in MRR and favorable ARP. Compared with traditional and hybrid retrieval methods, STaR not only improves semantic precision but also reduces unnecessary computation and redundancy, making it well-suited for resource-constrained and real-world QA deployments.

Keywords Retrieval-augmented generation · Large language models · Question-answering systems · Soft labeling · Retriever fine-tuning

1 Introduction

Retrieval-augmented generation (RAG) has become a core technique for enhancing the factuality and grounding of large language models (LLMs) in knowledge-intensive tasks [1]. By retrieving relevant external documents to support response generation, RAG mitigates the limitations of generation without external knowledge and has been widely adopted in question answering (QA), chatbots, and domain-specific assistants.

Communicated by Xun Yang.

✉ Jia-Yang Jiang
812414034@o365.tku.edu.tw

✉ Chih-Yung Chang
cychang@mail.tku.edu.tw

Youxi Li
liyouxi2010@gmail.com

Tzu-Chia Huang
142870@o365.tku.edu.tw

Diptendu Sinha Roy
diptendu.sr@nitm.ac.in

[1] The Department of Computer Science and Information Engineering, Tamkang University, New Taipei 25137, Taiwan

[2] The Department of Information Engineering, Minxi Vocational and Technical College, Longyan 364021, China

[3] Hangzhou Ruiyuan Jiling Technology Co. Ltd, HangZhou 310000, China

[4] The Department of Computer Science and Engineering, National Institute of Technology, Shillong 793003, India

Published online: 10 August 2026

Despite the progress made by dense retrievers like Dense Passage Retrieval (DPR) [2] and ColBERT [3], several limitations persist. First, most retrievers are trained using binary hard labels that classify passages as either relevant or irrelevant.

This rigid supervision neglects the nuanced spectrum of semantic relatedness and suppresses the expressiveness needed for generalization [4, 5]. Second, pointwise and pairwise training objectives lack explicit regularization over the embedding space, often causing semantically similar but contextually irrelevant passages to cluster together, thereby increasing misretrieval. Third, these retrievers are typically sensitive to domain shifts and require substantial labeled data for effective adaptation. Hybrid/blended retrieval combines dense and sparse indexes with hybrid query strategies to improve effectiveness but increases system complexity rather than performing iterative refinement [6]. Interleaving or multi-step retrieval alternates reasoning and retrieval (e.g., IRCoT) to iteratively refine evidence, which can improve precision but entails additional retrieval calls and compute overhead [7]. Beyond binary supervision, the noisy-label literature and instance-dependent label-enhancement methods indicate richer supervisory signals; while these advances are not designed for open-domain retrieval, they motivate moving beyond hard labels and highlight adaptation challenges when corpora evolve [8, 9].

To address these challenges holistically, this study proposes STaR (Soft-labeling and Triplet-Aware Retriever), a novel retrieval optimization framework that jointly enhances semantic supervision and embedding space discrimination. Rather than using binary or heuristic supervision, STaR constructs a dynamic similarity graph from the corpus and generates structured soft labels based on multi-hop graph paths between documents. This approach captures fine-grained semantic relevance and facilitates scalable training across domains. To complement this, this study introduces a triplet-based SBERT optimization strategy that explicitly regularizes embedding geometry. By pulling semantically relevant passages closer to the query anchor and pushing distractors away, our method preserves global relational structure and mitigates retrieval errors stemming from embedding collapse or contextual ambiguity. Our main contributions are as follows:

(1) Compared with hard-label supervision and label-distribution enhancement methods from partial/weak-label learning [9, 10], this study introduces a BM25 similarity graph-based soft labeling strategy that assigns fine-grained relevance via structured multi-hop relations, improving expressiveness and scalability for retrieval settings.

(2) Beyond pairwise contrastive objectives commonly used in dense retrievers (e.g., DPR) [2] and label-free autoencoding training (ART) [11], this study instantiates a triplet-based SBERT objective [12] coupled with graph-derived soft labels to shape the embedding space with relational consistency, effectively separating semantically similar yet contextually irrelevant passages.

(3) This study demonstrates that combining soft-label supervision and triplet optimization enables our model to achieve superior performance in both in-domain and cross-domain QA tasks, offering improvements in retrieval quality, robustness, and deployment efficiency.

The remainder of the paper is organized as follows. Section II reviews and compares previous relevant work. Section III explains the notations, assumptions, problem description, and objectives. Section IV introduces the proposed STaR. Section V presents the simulation and performance evaluation. Section VI discusses the summary and future work.

2 Related works

Retrieval serves as a foundational component in Retrieval-Augmented Generation (RAG) systems, enabling large language models (LLMs) to access external knowledge for factual grounding and contextual reasoning. Prior research on retrieval techniques in the RAG framework can be broadly categorized into three major directions: Text-based embedding methods, Hybrid semantic and knowledge-augmented retrieval methods and structured knowledge and Retriever fine-tuning and supervision strategies. This section reviews representative work in each category to highlight the evolution of retrieval strategies and their underlying modeling assumptions.

2.1 Text-based embedding methods

Early retrieval systems such as Term Frequency-Inverse Document Frequency (TF-IDF) [13, 14] and BM25 [15] represented queries and passages using high-dimensional sparse vectors derived from word frequency statistics. These models maintained popularity due to their interpretability and efficiency but lacked the ability to capture paraphrastic variation or contextual meaning. To mitigate these limitations, shallow embedding methods such as Word2Vec [16] and GloVe [17] were proposed. These approaches learned fixed embeddings from co-occurrence patterns to reflect semantic proximity. However, the resulting vectors were context-independent and static, making them insufficient for complex tasks like open-domain QA that require understanding polysemy and compositional semantics. Overall, while these methods performed well in low-resource settings, their reliance on surface-level lexical features limited their expressiveness and adaptability.

More recent advances sought to refine static embeddings rather than abandon them. For example, WordFast improves on traditional embeddings by linearly fusing FastText and Word2Vec, achieving better performance in tasks such as sentiment analysis [18]. In parallel, subword-enhanced static embeddings such as FastText [19] enrich type-level vectors with character n-grams, improving robustness for rare and morphologically rich forms while remaining context-independent. Subsequent semantic-specialization post-processing refines pre-trained static, type-level vectors; for instance, Attract-Repel injects synonymy/antonymy constraints to tighten semantic neighborhoods and separate lexical contrasts, explicitly remaining context-independent [20]. More recently, lexical-subspace approaches encode symmetric (e.g., synonymy, antonymy) and asymmetric (e.g., hypernymy, meronymy) lexical relations as subspaces within the distributional space, further specializing static embeddings while still avoiding contextualized token-level representations [21]. While static embeddings are token-context independent and struggle with temporal shifts, diachronic/geo-conditioned static variants attempt to mitigate this by conditioning on time and location; for example, Gong et al. [22] enrich type-level representations with temporal and spatial signals without introducing token-level contextualization.

Beyond static type-level vectors, Transformer-based encoders such as BERT [23], RoBERTa [24], and DeBERTa [25] produce representations conditioned on surrounding context. These encoders improve polysemy disambiguation and compositional semantics and form the basis of many modern sentence and passage embeddings. Building on these encoder foundations, recent work has released strong general-purpose embedding families and packaged resources to support retrieval-oriented representation learning. In particular, C-Pack [26] provides packed resources and evaluation suites for general Chinese embeddings, and it is often used to benchmark strong embedding families such as BGE in dense retrieval and semantic matching.

In addition, large language models have also been explored as representation enhancers for downstream retrieval and matching tasks. For instance, LE-DLCM [27] leverages large language models to decouple learner and course modeling and to obtain knowledge-enriched course embeddings and improve representation quality in recommendation settings, illustrating how LLMs can augment embedding quality via external knowledge and richer semantic encoding. Similar ideas have also been explored across other information-access settings, where LLMs are used to enrich representations beyond static corpora statistics. Recent studies have begun to outline this emerging design space of LLM-enhanced representation learning across recommender and matching pipelines. While these directions improve expressiveness, they do not eliminate the need for controllable and task-aligned supervision when adapting embeddings to retrieval-style relevance.

In summary, static and shallow embeddings are efficient and interpretable, but their reliance on fixed lexical or distributional statistics limits their ability to model paraphrases and context-dependent semantics. This motivates richer contextual representations and more informative supervision for aligning embeddings with retrieval-oriented relevance, as further discussed in the following sections on knowledge-augmented retrieval and dense retriever fine-tuning.

2.2 Hybrid semantic and knowledge-augmented retrieval methods

To improve retrieval robustness and semantic grounding, hybrid methods combine sparse and dense signals or augment retrievers with structured knowledge. For example, Blended RAG [6] blends dense and sparse indexes with hybrid, query-aware strategies while RAP-Gen [28] employs a hybrid patch retriever that combines lexical and semantic matching for code to supply a CodeT5 generator with high-quality candidates. However, most fusion strategies rely on fixed weights, which fail to adapt across heterogeneous query intents or domains. Other approaches incorporated knowledge graphs to enhance factual consistency. ReAct [29] interleaved retrieval and reasoning steps using external tools, while Think-on-Graph (ToG) [30] performed multi-hop graph traversal for compositional QA. Although these systems improved grounding accuracy, they generally required additional inference-time modules, introducing latency and scalability issues. Lightweight alternatives such as query expansion–based reranking pipelines (e.g., EAR) [31] and knowledge-constrained re-rankers for KGQA [32] reduced computational overhead but depended on manually crafted rules or static ontologies, limiting their adaptability in dynamic environments.

Recent work targets these limitations via adaptivity and tighter integration. DAT [33] dynamically tunes the balance between BM25 and dense retrieval on a per-query basis using LLM-informed feedback, outperforming fixed-weight hybrids. In domain-specific applications, HybridRAG [34] combines vector and graph-based retrieval to extract accurate answers from financial transcripts. Broadening the scope, a high-ranking hybrid RAG system for the KDD Cup 2024 CRAG track [35] improves complex reasoning through attribute predictors, structured-knowledge extractors, and other retrieval–extraction optimizations. From a system design perspective, the KG-Vector RAG framework [36] shows how structured knowledge integration can enhance both factual grounding and runtime efficiency. Moreover, query-aware graph neural architectures [37]

employ graph-based attention to dynamically prioritize relevant paths, yielding strong performance on multi-hop, reasoning-intensive tasks.

In sum, hybrid methods deepen semantic grounding and expand retrieval breadth, yet they often trade simplicity for architectural complexity and still lack fine-grained supervisory signals to explicitly shape embedding geometry. Our approach targets this gap by deriving similarity-graph–based soft labels and applying triplet-aware SBERT optimization, providing continuous, topology-aware supervision that improves discrimination without incurring heavy inference-time toolchains.

2.3 Retriever fine-tuning and supervision strategies

Dense retrieval models such as dense passage retrieval (DPR) [2] and ColBERT [3] encode queries and passages into dense vectors with contextual encoders and are typically fine-tuned under hard-label supervision. These methods are trained on QA datasets with binary relevance signals using contrastive or ranking objectives (e.g., in-batch InfoNCE), thereby discretizing relevance. However, such coarse supervision under-specifies graded similarity and provides limited global geometric constraints in the embedding space, which can hinder generalization to unseen domains and make it difficult to separate topically similar yet contextually irrelevant passages.

Within this hard-label regime, research has mainly pursued stronger negatives under discrete supervision. ANCE [38] mines globally hard negatives from an asynchronously updated ANN index, consistently improving over random in-batch negatives, yet the supervision signal remains binary. STAR/ADORE [39] further stabilizes training and refreshes hard negatives to optimize ranking with explicit positive and negative labels. RocketQA [40] combines cross-batch negatives, denoised hard negatives, and data augmentation, making discrete-label training more effective for open-domain QA.

Beyond mining strategies, journal work has systematized hard-label supervised training via architectural adjustments while keeping supervision discrete. SDA (Sparse–Dense–Attentional) retrieval [41] analyzes dual-encoder capacity and margins, and trains sparse–dense or attentional hybrids with BM25-mined negatives plus discrete positives to strengthen first-stage retrieval. Aggretriever [42] aggregates token-level signals into a single-vector representation and is fine-tuned with explicit positives and negatives under standard DPR setups, improving both in-domain accuracy and zero-shot robustness without costly pre-training.

In parallel, relevance-aware learning signals have also been explored in unsupervised dense retrieval to reduce the noise introduced by pseudo-positive pairs. Contriever [43] learns dense retrievers through contrastive pre-training on unlabeled corpora; however, pseudo-positive pairs constructed from unlabeled data may contain weakly relevant or noisy examples, leading to false positives during contrastive learning. Lei et al. proposed a relevance-aware contrastive pre-training method, also referred to as ReContriever [44], in which an intermediate-trained retriever is used as a proxy relevance estimator to assess the reliability of pseudo-positive pairs and adaptively reweight the contrastive loss. This model-driven relevance estimation strategy improves the robustness of unsupervised dense retrieval by reducing the influence of unreliable positives.

Different from this model-driven pre-training paradigm, this study focuses on task-specific retriever fine-tuning for RAG-QA and derives graded relevance supervision from corpus-level similarity structures. Instead of estimating pair reliability from the intermediate state of a retriever during pre-training, the proposed framework constructs soft labels from the target corpus before retriever optimization, thereby providing an interpretable and reproducible supervision signal for task-specific retrieval. Therefore, STaR is positioned as a corpus-structure-driven soft-labeling strategy that is complementary to adaptive model-driven relevance estimation.

Beyond retrieval-specific relevance weighting, dynamic difficulty-aware supervision has also been explored in sequence-to-sequence learning. Peng et al. [45] proposed Token-Level Self-Evolution Training, which dynamically emphasizes under-explored target tokens and applies token-specific label smoothing to improve generalization. Although this method is designed for token-level generation rather than query-passage retrieval, it highlights the value of adapting supervision according to learning difficulty. In contrast, this study focuses on retrieval-level supervision and uses graph traversal depth and decay as corpus-level controls for propagating graded relevance over the similarity graph. This design emphasizes interpretability and reproducibility in task-specific retriever fine-tuning, while difficulty-aware graph weighting remains a potential extension of the current soft-labeling strategy.

3 Assumptions and problem formulation

This section presents the assumptions, problem formulation and objective of this study.

3.1 Notations and assumptions

Given a document $\mathcal{D}$, represented as a sequence of sentences $\mathcal{D} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{|\mathcal{D}|}\}$, where $|\mathcal{D}|$ denotes the total number of sentences within the document. Each sentence $\mathcal{S}_i$

consists of a sequence of words $\mathcal{S}_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,|s_i|}\}$, where $|s_i|$ denotes the total number of words in $\mathcal{S}_i$. Our goal is to develop a Question Answering (QA) system capable of accurately responding to a set of *n* user-generated questions $\mathcal{Q} = \{\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_n\}$. For each question $\mathcal{Q}_i$, if the answer can be derived from information within the document $\mathcal{D}$, our QA system should be able to provide the corresponding accurate answers $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n\}$.

Assume a correct answer $\mathcal{A}_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,|\mathcal{A}_i|}\}$ is composed of multiple sentences. Each sentence $s_{i,j} = \{w_{i,j,1}, w_{i,j,2}, \ldots, w_{i,j,|s_{i,j}|}\}$ in answer $\mathcal{A}_i$ is composed of multiple words $w_{i,j,k}$, where $|s_{i,j}|$ is the number of words in the sentence $s_{i,j}$. Similarly, let $\widehat{\mathcal{A}}_i = \{\widehat{s}_{i,1}, \widehat{s}_{i,2}, \ldots, \widehat{s}_{i,|\widehat{\mathcal{A}}_i|}\}$ denote a predicted answer, which is also composed of multiple sentences. Each sentence $\widehat{s}_{i,j} = \{\widehat{w}_{i,j,1}, \widehat{w}_{i,j,2}, \ldots, \widehat{w}_{i,j,|\widehat{s}_{i,j}|}\}$ in the predicted answer, is composed of multiple words $\widehat{w}_{i,j,k}$.

3.2 Evaluation metrics

The performance of our developed QA system is primarily evaluated based on the following three performance metrics: answerability, lexical-level accuracy, and semantic accuracy. The "answerability" means that our QA system should have the ability to determine whether the scope of the user's *i*-th question $\mathcal{Q}_i$ falls within the content of the document $\mathcal{D}$. Once $\mathcal{Q}_i$ is identified as being within the scope of $\mathcal{D}$, the system's performance is further assessed using lexical-level accuracy and semantic accuracy. The lexical-level accuracy evaluates the system's ability to assess the correctness of the predicted answer compared to the ground truth answer at the lexical level. The Semantic accuracy measures the system's capability to determine the alignment between the predicted answer and the annotated answer at the semantic level. The following presents the details of the proposed three performance metrics.

3.2.1 Performance metric of answerability

To evaluate the performance of the first metric, the following notations are defined. Let $y_j^{\mathcal{M}}$ be a Boolean variable which denotes the ground truth label of the *j*-th query $\mathcal{Q}_j$, representing whether or not the query exists in the database $\mathcal{D}$. More specifically, $y_j^{\mathcal{M}}$ is defined as:

$$y_j^M = \begin{cases} 1, if\ Q_j\ is\ in\ scope\ of\ D \\ 0, otherwise \end{cases} \tag{1}$$

similarly, let $\hat{y}_j^{\mathcal{M}}$ be a Boolean variable which indicates whether the system predicts that $\mathcal{Q}_j$ exists in $\mathcal{D}$. It is defined as:

$$\hat{y}_j^M = \begin{cases} 1, if\ the\ system\ predicts\ \mathcal{Q}_j\ in\ scope\ of\ D \\ 0, otherwise \end{cases} \tag{2}$$

let $TP_i^{\mathcal{M}}$ denote true positive for the result predicted by mechanism $\mathcal{M}$. That is, the value of $TP_i^{\mathcal{M}}$ is true if the question $\mathcal{Q}_i$ exists in the database $\mathcal{D}$ and the prediction of $\mathcal{M}$ is correct. The value of TP_i^M can be calculated by Exp. (3).

$$TP_i^M = \begin{cases} 1, y_j^M\ \hat{y}_j^M = 1 \\ 0, otherwise \end{cases} \tag{3}$$

Let $FP_i^{\mathcal{M}}$ and $FN_i^{\mathcal{M}}$ denote false positive and false negative for the results predicted by mechanism $\mathcal{M}$, respectively. The values of $FP_i^{\mathcal{M}}$ and $FN_i^{\mathcal{M}}$ can be derived by Exps. (4) and (5), respectively.

$$FP_i^M = \begin{cases} 1, (1-y_j^M)\ \hat{y}_j^M = 1 \\ 0, otherwise \end{cases} \tag{4}$$

$$FN_i^M = \begin{cases} 1, (1-\hat{y}_j^M)\ y_j^M = 1 \\ 0, otherwise \end{cases} \tag{5}$$

Let $TN_i^{\mathcal{M}}$ denote true negative for the result predicted by mechanism. The value of $TN_i^{\mathcal{M}}$ can be calculated by Exp. (6), respectively.

$$TN_i^M = \begin{cases} 1, (1-y_j^M)\ (1-\hat{y}_j^M) = 1 \\ 0, otherwise \end{cases} \tag{6}$$

Let $\mathcal{Q} = \{\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_{|\mathcal{Q}|}\}$ be the set of $|\mathcal{Q}|$ questions. Let $TP^{\mathcal{M}}_{\text{answerability}}$, $FP^{\mathcal{M}}_{\text{answerability}}$, $FN^{\mathcal{M}}_{\text{answerability}}$ and $TN^{\mathcal{M}}_{\text{answerability}}$ denote the outcomes of determining, using mechanism $\mathcal{M}$, whether these questions fall within the scope of document $\mathcal{D}$. The results are categorized as true positive, false positive, false negative and true negative, respectively. That is,

$$TP^M_{answerability} = \sum_{i=1}^{|\mathcal{Q}|} TP_i^M \tag{7}$$

$$FP^M_{answerability} = \sum_{i=1}^{|\mathcal{Q}|} FP_i^M \tag{8}$$

$$FN^M_{answerability} = \sum_{i=1}^{|\mathcal{Q}|} FN_i^M \tag{9}$$

$$TN^{M}_{answerability} = \sum_{i=1}^{|\mathcal{Q}|} TN^{M}_{i} \tag{10}$$

Let $\mathcal{P}^{\mathcal{M}}_{\text{answerability}}$ and $\mathcal{R}^{\mathcal{M}}_{\text{answerability}}$ denote the precision and recall by applying mechanism M, respectively. The values of $\mathcal{P}^{\mathcal{M}}_{\text{answerability}}$ and $\mathcal{R}^{\mathcal{M}}_{\text{answerability}}$ can be calculated by Exps. (11) and (12), respectively.

$$P^{M}_{answerability} = \frac{TP^{M}_{answerability}}{TP^{M}_{answerability} + FP^{M}_{answerability}} \tag{11}$$

$$R^{M}_{answerability} = \frac{TP^{M}_{answerability}}{TP^{M}_{answerability} + FN^{M}_{answerability}} \tag{12}$$

Let $\mathcal{F}1^{\mathcal{M}}_{\text{answerability}}$ denote F1-score by applying mechanism $\mathcal{M}$. The value of $\mathcal{F}1^{\mathcal{M}}_{\text{answerability}}$ can be computed by Exp. (13).

$$F1^{M}_{answerability} = \frac{2 * P^{M}_{answerability} * R^{M}_{answerability}}{P^{M}_{answerability} + R^{M}_{answerability}} \tag{13}$$

3.2.2 Performance metric for lexical-level accuracy

The first performance aims to determine whether or not the question falls in the scope of document $\mathcal{D}$. In case that the question falls in the scope of $\mathcal{D}$, the second performance metric evaluates the accuracy of the answer content at the word level. For model $\mathcal{M}$, let Boolean variable $\delta^{\mathcal{M}}_{i,j,k}$ represent whether or not the k-th word $w_{i,j,k}$ in the j-th sentence is in label $\mathcal{A}_i$. The value of $\delta^{\mathcal{M}}_{i,j,k}$ can be derived from Exp. (14)

$$\delta^{M}_{i,j,k} = \begin{cases} 1, if\ w_{i,j,k} \in A_i \\ 0, if\ w_{i,j,k} \notin A_i \end{cases} \tag{14}$$

Furthermore, let Boolean variable $\rho^{\mathcal{M}}_{i,j,k}$ represent whether or not the k-th word $w_{i,j,k}$ in the j-th sentence is in predicted answer $\widehat{\mathcal{A}}_i$. The value of $\rho^{\mathcal{M}}_{i,j,k}$ can be derived from Exp. (15)

$$\rho^{M}_{i,j,k} = \begin{cases} 1, if\ w_{i,j,k} \in \hat{A}_i \\ 0, if\ w_{i,j,k} \notin \hat{A}_i \end{cases} \tag{15}$$

Let $TP^{\mathcal{M}}_{i,j,k}$, $FP^{\mathcal{M}}_{i,j,k}$, $FN^{\mathcal{M}}_{i,j,k}$ and $TN^{\mathcal{M}}_{i,j,k}$ denote true positive, false positive, false negative and true negative for the result predicted by mechanism $\mathcal{M}$. Specifically, $TP^{\mathcal{M}}_{i,j,k}$ denote that the k-th word in the j-th sentence of the generated answer is present in the labeled answer $\mathcal{A}_i$ and is correctly predicted. $FP^{\mathcal{M}}_{i,j,k}$ denote that the k-th word in the j-th sentence of the generated answer is present in the labeled answer $\mathcal{A}_i$ but is incorrectly predicted. $FN^{\mathcal{M}}_{i,j,k}$ denote that the k-th word in the j-th sentence of the generated answer is not present in the labeled answer $\mathcal{A}_i$ but is incorrectly predicted. $TN^{\mathcal{M}}_{i,j,k}$ denote that the k-th word in the j-th sentence of the generated answer is not present in the labeled answer $\mathcal{A}_i$ and is correctly predicted. The value of $TP^{\mathcal{M}}_{i,j,k}$, $FP^{\mathcal{M}}_{i,j,k}$, $FN^{\mathcal{M}}_{i,j,k}$ and $TN^{\mathcal{M}}_{i,j,k}$ can be calculated by Exps. (16) to Exps. (19), respectively.

$$TP^{M}_{i,j,k} = \begin{cases} 1, \delta^{M}_{i,j,k}\, \rho^{M}_{i,j,k}=1 \\ 0, otherwise \end{cases} \tag{16}$$

$$FP^{M}_{i,j,k} = \begin{cases} 1, (1-\delta^{M}_{i,j,k})\, \rho^{M}_{i,j,k}=1 \\ 0, otherwise \end{cases} \tag{17}$$

$$FN^{M}_{i,j,k} = \begin{cases} 1, (1-\rho^{M}_{i,j,k})\, \delta^{M}_{i,j,k}=1 \\ 0, otherwise \end{cases} \tag{18}$$

$$TN^{M}_{i,j,k} = \begin{cases} 1, (1-\delta^{M}_{i,j,k})\, (1-\rho^{M}_{i,j,k})=1 \\ 0, otherwise \end{cases} \tag{19}$$

Let $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_{|\mathcal{A}|}\}$ be the set of $|\mathcal{A}|$ labeled answers for each question. Let $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_{|s_{i,j}|}\}$ be the set of $|s_{i,j}|$ sentences for each answers. Let $\mathcal{W} = \{\mathcal{W}_1, \ldots, \mathcal{W}_{|\mathcal{W}_{i,j,k}|}\}$ be the set of $|\mathcal{W}_{i,j,k}|$ words for each sentences. Let $TP^{\mathcal{M}}_{\text{accuracy}}$, $FP^{\mathcal{M}}_{\text{accuracy}}$, $FN^{\mathcal{M}}_{\text{accuracy}}$, and $TN^{\mathcal{M}}_{\text{accuracy}}$ denote true positive, false positive, false negative, and true negative at the word level, representing the matching status for each word. The values of $TP^{\mathcal{M}}_{\text{accuracy}}$, $FP^{\mathcal{M}}_{\text{accuracy}}$$FN^{\mathcal{M}}_{\text{accuracy}}$, and $TN^{\mathcal{M}}_{\text{accuracy}}$ can be calculated by Exps. (20) to (23), respectively.

$$TP^{M}_{accuracy} = \sum_{i=1}^{|A|} \sum_{j=1}^{|s_{i,j}|} \sum_{k=1}^{W_{i,j,k}} TP^{M}_{i,j,k} \tag{20}$$

$$FP^{M}_{accuracy} = \sum_{i=1}^{|A|} \sum_{j=1}^{|s_{i,j}|} \sum_{k=1}^{W_{i,j,k}} FP^{M}_{i,j,k} \tag{21}$$

$$FN^{M}_{accuracy} = \sum_{i=1}^{|A|} \sum_{j=1}^{|s_{i,j}|} \sum_{k=1}^{W_{i,j,k}} FN^{M}_{i,j,k} \tag{22}$$

$$TN^{M}_{accuracy} = \sum_{i=1}^{|A|} \sum_{j=1}^{|s_{i,j}|} \sum_{k=1}^{W_{i,j,k}} TN^{M}_{i,j,k} \tag{23}$$

Let $\mathcal{P}^{\mathcal{M}}_{\text{accuracy}}$ and $\mathcal{R}^{\mathcal{M}}_{\text{accuracy}}$ denote the precision and recall by applying mechanism M, respectively. The values of $\mathcal{P}^{\mathcal{M}}$ and $\mathcal{R}^{\mathcal{M}}$ can be calculated by Exps. (24) and (25), respectively.

$$P^{M}_{accuracy} = \frac{TP^{M}_{accuracy}}{TP^{M}_{accuracy} + FP^{M}_{accuracy}} \tag{24}$$

$$R^{M}_{accuracy} = \frac{TP^{M}_{accuracy}}{TP^{M}_{accuracy} + FN^{M}_{accuracy}} \tag{25}$$

Let $\mathcal{F}1^{\mathcal{M}}_{\text{accuracy}}$ denote F1-score of $\mathcal{C}$ predicted by mechanism $\mathcal{M}$. The value of $\mathcal{F}1^{\mathcal{M}}_{\text{accuracy}}$ can be denoted by Exp. (26).

$$F1^{M}_{accuracy} = \begin{cases} \frac{2*P^{\mathcal{M}}_{accuracy}*R^{M}_{accuracy}}{P^{\mathcal{M}}_{accuracy}+R^{M}_{accuracy}}, & if y^{M}_{j} = 1 \; and \; \hat{y}^{M}_{j} = 1 \\ 0, & if \; y^{M}_{j} = 0 \; or \; \hat{y}^{M}_{j} = 0 \end{cases} \tag{26}$$

3.2.3 Performance metric of semantic accuracy

The third performance metric evaluates the semantic accuracy of the QA system's answers. This metric evaluates whether the predicted answer accurately conveys the intended meaning of the correct answer, even if there are lexical differences. For a given question $\mathcal{Q}_{\not}$, let $\mathcal{A}_{\not}$ and $\widehat{\mathcal{A}}_{\not}$ be the ground truth answer and the answer predicted by the system, respectively. To quantify semantic similarity, the answers $\mathcal{A}_{\not}$

and $\widehat{\mathcal{A}}_{\not}$ will be transformed to sentence embeddings $\mathcal{V}_{\not}$ and$\widehat{\mathcal{V}}_{\not}$, respectively. Let $\mathcal{S}_{\not}$ denote the similarity score between $\mathcal{A}_{\not}$ and $\widehat{\mathcal{A}}_{\not}$. That is, using Exp. (27).

$$\rho_j = \begin{cases} \frac{v_j \, \hat{v}_j}{\|v_j\| \, \|\hat{v}_j\|}, if \; y^{M}_{j}{=}1 \; and \; \hat{y}^{M}_{j}{=}1 \\ 0, if \; y^{M}_{j}{=}0 \; or \; \hat{y}^{M}_{j}{=}0 \end{cases} \tag{27}$$

where $\mathcal{V}_{\not} \bullet \widehat{\mathcal{V}}_{\not}$ represents the dot product of the two vectors, $\|\mathcal{V}_{\not}\|$ and $\|\widehat{\mathcal{V}}_{\not}\|$ are the magnitudes of the vectors $\mathcal{V}_{\not}$ and $\widehat{\mathcal{V}}_{\not}$, respectively. When both vectors are positive, the cosine similarity score $\mathcal{S}_{cos} \in [0,1]$, with higher values indicating stronger semantic similarity.

Let $\mathcal{Q} = \{\mathcal{Q}_1, \ldots, \mathcal{Q}_{|\mathcal{Q}|}\}$ be the set of $|\mathcal{Q}|$ questions. Let ρ denote the outcomes of determining, using mechanism $\mathcal{M}$, representing the similarity between the predicted answer and the ground truth answer. The value of ρ can be denoted by Exp. (28).

$$\rho = \sum_{j=1}^{|\mathcal{Q}|} \rho_j \tag{28}$$

3.3 Objectives

Finally, the overall performance of the QA system is quantified by a weighted combination of the three metrics described above: Answerability F1-score, lexical F1-score and Semantic similarity. Let the weights assigned to each metric beα, β and γ respectively, with$\alpha + \beta + \gamma = 1$. The overall goal is to maximize the system's performance score $\mathcal{P}$. The final performance score $\mathcal{P}$ can be calculated using Exp. (29).

$$P = \alpha F1^{M}_{answerability} + \beta F1^{M}_{accuracy} + \gamma\rho \tag{29}$$

Let Ω be the set of all possible mechanisms for solving the QA problem and $\mathcal{P}_{\mathcal{M}}$ be the total score of mechanism $\mathcal{M}$. The goal of this paper is to design the best mechanism $\mathcal{M}^{best}$ that achieves the highest score among all possible mechanisms in Ω. Exp. (30) reflects the objective of this paper.

$$M^{best} = \arg\max_{M \in \Omega} \mathcal{P}_{\mathcal{M}} \tag{30}$$

4 The proposed STaR mechanism

This paper introduces a novel training framework, called STaR, which aims to develop a training model for a Retrieval-Augmented Generation (RAG) retriever. The goal of the training model is to generate high-quality QA training data and a training model for improving the retriever's precision and generalizability. As shown in Fig. 1, the proposed framework comprises three main stages, namely Keyword Extraction and Similarity Computation, Soft Label Generation via Similarity Graphs, and Multi-Dimensional Pairing through Triplet Similarity SBERT Architecture. The first stage employs CKIP tokenization to extract keywords from the text and utilizes the BM25 algorithm to calculate similarity scores between documents, constructing a textual similarity graph. The second stage leverages the similarity graph to generate soft similarity labels that capture the nuanced relationships between a query and its related documents. Finally, the last stage incorporates a triplet similarity-based SBERT architecture to optimize multi-dimensional relationships, enabling the model to handle question-to-multiple-text matching effectively. The following presents the details of the three stages.

4.1 Keyword extraction and similarity computation

This stage begins with the extraction of key textual features and the computation of document similarity. It uses CKIP

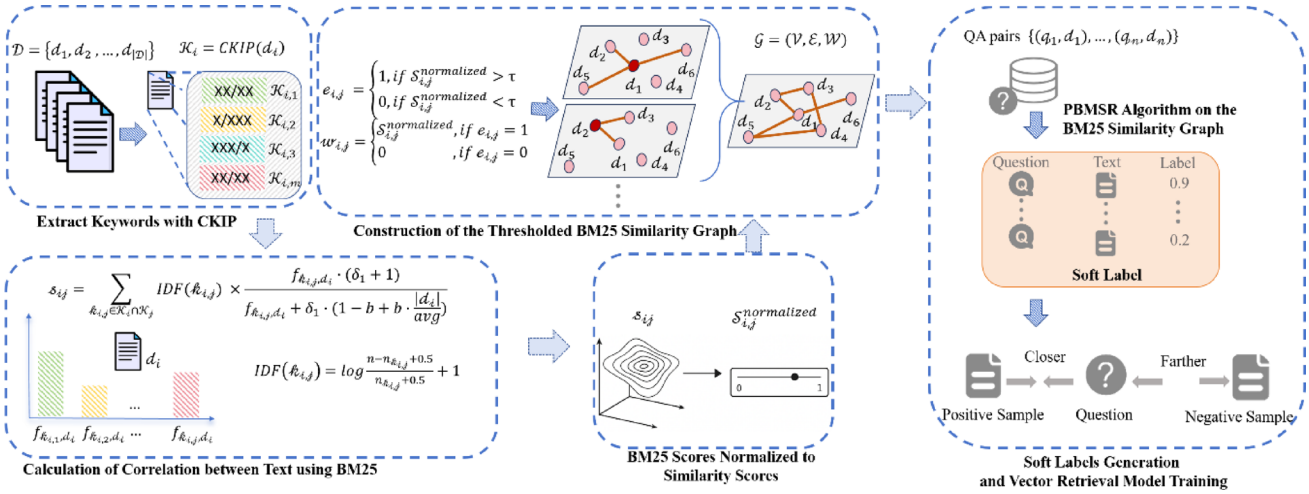

Fig. 1 The architecture of the proposed StaR

tokenization for linguistic processing and BM25 for similarity measurement, both of which lay the groundwork for constructing a textual similarity graph.

As shown in Fig. 1, the CKIP tokenizer processes the input text $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$, where document $\mathcal{D}$ consists of n documents and d_i represents a document. Then, each document d_i is further segmented into a set of tokens. The result of the CKIP tokenizer can be defined as Exp. (31).

$$K_i = CKIP(d_i) \tag{31}$$

where $\mathcal{K}_i = \{\mathcal{K}_{i,1}, \mathcal{K}_{i,2}, \ldots, \mathcal{K}_{i,m}\}$ is the set of extracted keywords from document d_i and m denotes the maximal number of tokens in each document d_i.

Let notation s_{ij} denote the similarity score of documents d_i and d_j. After the document segmentation, the BM25 algorithm is applied to calculate the score similarity s_{ij} for each pair of documents $d_i \in \mathcal{D}$ and $d_j \in \mathcal{D}$. This study utilizes BM25 to compute similarity scores between texts when establishing a training set for a RAG (Retrieval-Augmented Generation) retriever, as BM25 balances the effects of high- and low-frequency terms, making the results more precise and reliable.

The relevance scores derived from BM25 reflect the degree of similarity between texts, which is crucial for constructing a text retrieval system. The BM25 measures textual relevance based on term frequency and document length, ensuring the model captures content most relevant to the query. The score s_{ij} can be calculated based on:

$$s_{ij} = \sum_{k_{i,j} \in k_i \cap k_j} IDF\left(k_{i,j}\right) \frac{f_{k_{i,j},d_i}\ (\delta_1 + 1)}{f_{k_{i,j},d_i} + \delta_1 \left(1 - b + b\,\frac{|d_i|}{avg}\right)} \tag{32}$$

where $f_{k_{i,j},d_i}$ represents the frequency of keyword key in document d_i, δ_1 and b are hyper parameters with typical values $\delta_1 = 1.2$ and $b = 0.75$, $|d_i|$ represents the length of document d_i,avg represents the average document length. The $IDF(k_{i,j})$, representing the inverse document frequency of keyword $k_{i,j}$, is defined as

$$IDF(k_{i,j}) = \log \frac{n - n_{k_{i,j}} + 0.5}{n_{k_{i,j}} + 0.5} + 1 \tag{33}$$

where n represents the total number of documents, and $n_{k_{i,j}}$ is the number of documents containing the keyword $k_{i,j}$. Then the relevance scores $s_{i,j}$ of two documents d_i and d_j are normalized using the Min–Max normalization method to convert them into similarity scores ranging between 0 and 1. This step is crucial for SBERT training, enabling the model to effectively learn textual similarities. The relevance scores for all query terms are aggregated to compute the overall relevance score of the candidate document $\mathcal{Q}$ relative to the primary document $\mathcal{D}$. To facilitate SBERT similarity training, the BM25 relevance scores are transformed into normalized similarity scores using the Min–Max normalization technique, as defined by the equation:

$$S_{i,j}^{normalized} = \frac{s_{i,j} - \min\limits_{i,j \in D}(s_{i,j})}{\max\limits_{i,j \in D}(s_{i,j}) - \min\limits_{i,j \in D}(s_{i,j})} \tag{34}$$

where $\mathcal{S}_{i,j}^{normalized}$ represents the original relevance score, while $\min\limits_{i,j \in \mathcal{D}}(s_{i,j})$ and $\max\limits_{i,j \in \mathcal{D}}(s_{i,j})$ represent the minimum and maximum $BM25Scores$ across all documents, respectively. This transformation maps BM25 scores to the 0–1 range, aligning them with SBERT's output and ensuring consistent similarity evaluations. Such normalization enhances the accuracy and reliability of SBERT during similarity training.

4.2 Construction of BM25 similarity graphs

This stage aims to construct a textual similarity graph $\mathcal{G}$. The primary purpose of constructing $\mathcal{G}$ is to effectively organize and represent the relationships between texts based on their similarity. This structured representation facilitates subsequent multi-hop similarity propagation for soft-label generation, providing graded supervision signals for retriever fine-tuning.

Recall that $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$ represents the input text, where n represents the total number of documents, and each d_i corresponds to a specific textual document. This text collection serves as the foundation for constructing the similarity graph. In this stage, CKIP is used to segment all texts and extract key terms. Subsequently, based on these keywords, the BM25 algorithm is employed to assess the relevance between texts based on specific keywords. At this stage, the normalized similarity scores $\mathcal{S}_{ij}^{normalized}$ are calculated for all document pairs as part of the BM25 process. These scores are subsequently used as weights in constructing a textual similarity graph.

The textual similarity graph is formally defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ where node set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ corresponds to the text collection $\mathcal{D}$, with each node v_i representing a unique document d_i. The edge set $\mathcal{E}$ defines the connections between nodes. An edge e_{ij} is established between nodes v_i and v_j if the similarity score $\mathcal{S}_{ij}^{normalized}$ between their corresponding texts exceeds a threshold τ. Otherwise, no edge is created. That is,

$$e_{i,j} = \begin{cases} 1, if\ \delta_{i,j}^{normalized} > \tau \\ 0, if\ \delta_{i,j}^{normalized} < \tau \end{cases} \tag{35}$$

The weight set $\mathcal{W}$ quantifies the strength of connections between nodes. The weight w_{ij} of an edge reflects the similarity score $\mathcal{S}_{ij}^{normalized}$ between the corresponding documents d_i and d_j. If $e_{ij} = 1$, the weight $w_{ij} = \mathcal{S}_{ij}^{normalized}$. Otherwise, the weight is 0. That is,

$$w_{i,j} = \begin{cases} \delta_{i,j}^{normalized}, if\ e_{i,j}=1 \\ 0, if\ e_{i,j}=0 \end{cases} \tag{36}$$

From a computational perspective, the thresholded BM25 similarity graph is constructed as an offline preprocessing step for soft-label generation rather than as an online inference module. In the naive case, document-level similarity computation may involve pairwise scoring over training passages. After similarity scores are obtained, the thresholding operation stores the graph in a sparse form by retaining only edges whose normalized BM25 similarity exceeds τ. Therefore, although the initial graph construction introduces additional preprocessing cost, the storage and subsequent propagation cost depend on the retained sparse edges rather than on a fully connected document graph. The multi-hop propagation is further bounded by the maximum traversal depth d_{max}, while the decay factor α attenuates the contribution of distant paths and reduces the influence of low-confidence long-range relations. As a result, the graph construction and traversal cost is incurred during training-data preparation, while the trained retriever does not require graph traversal during online retrieval.

4.3 Establishment of similarity-graph-based soft labels

In traditional question-answering (QA) systems, the relationship between a question and candidate texts is typically defined using hard labels, such as binary relevance indicators. However, such labels fail to capture fine-grained semantic relationships, limiting the model's ability to generalize in complex retrieval tasks. To address this limitation, this stage constructs a similarity-graph–based soft labeling framework. Instead of relying solely on hard labels, we build a thresholded BM25 similarity graph in which documents are treated as nodes and edges are weighted by normalized BM25 scores above a threshold. We then perform multi-hop propagation with decay control on this graph to derive soft labels. These soft labels are subsequently used to fine-tune SBERT under our triplet-aware objective, encouraging semantically related items to be closer than near-negatives and imposing stronger relative-distance constraints on the embedding space. This adjustment improves retrieval accuracy, ensures better query-to-document matching, and enables more efficient similarity computation. By integrating SBERT fine-tuning with soft labeling, document representations become more precise, enhancing the overall performance of the QA system.

To enhance semantic representation learning, a soft labeling framework is introduced for constructing training pairs in SBERT fine-tuning. As shown in Fig. 2, given a set of documents $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$, each document d_i generates a corresponding question q_i, forming a set of QA pairs $\{(q_1, d_1), \ldots, (q_n, d_n)\}$. For each training pair (q_i, d_j) , a label y_{ij} is assigned to indicate the relevance between the question q_i and document d_j. In case that $=$ is hold, the soft label of pair (q_i, d_j) will be set as $y_{ij} = 1$. Let w_{ij} is the weight of the edge between nodes d_i and d_j. If condition $i \neq j$ and $e(d_i, d_j) \in KG$ is satisfied, the soft label of pair (q_i, d_j) will be $y_{ij} = w_{ij}$. Otherwise, the remaining case is $e(d_i, d_j) \notin KG$. This implies that the number of hops from node d_i to node d_j is larger than one. Let $\mathcal{P}_{ij}$ denotes a path from d_i to d_j in similarity graph. Let d_{max} represents the

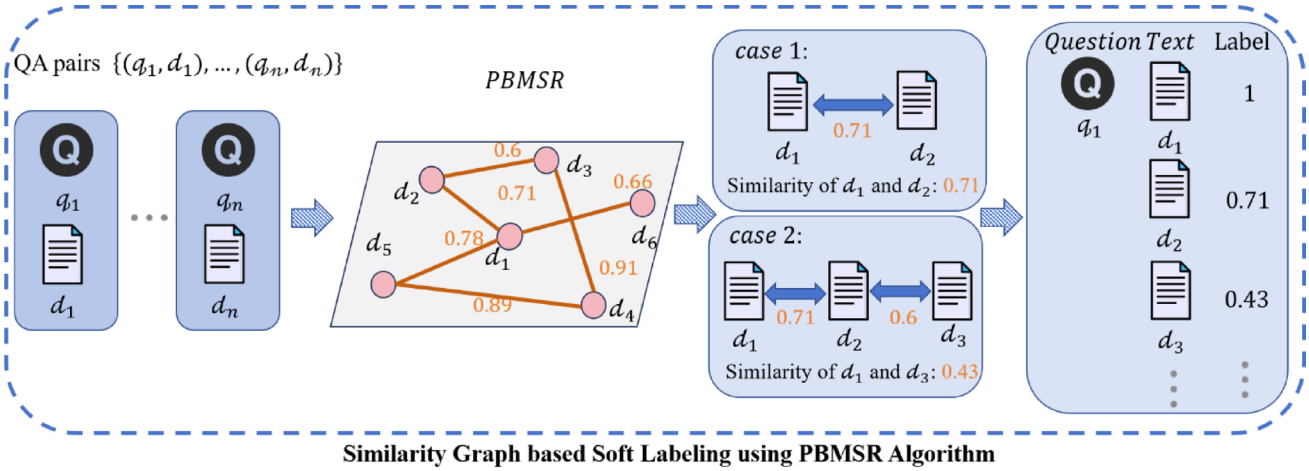

Fig. 2 The architecture of the establishment of similarity-graph-based soft labels

maximum allowable path length. Let $|\mathcal{P}_{i,j}|$ denotes the number of nodes of path $\mathcal{P}_{i,j}$. The longer the path, the smaller $\alpha^{|\mathcal{P}_{i,j}|-1}$, reducing the influence of long paths and mitigating noise in similarity calculations. As a result, the soft label of pair (q_i, d_j) along path $\mathcal{P}_{i,j}$ will be,

$$\tilde{y}_{i,j} = \alpha^{|P_{i,j}|-1} \prod_{e(d_i,d_j)\in P_{i,j}} w_{i,j} \tag{37}$$

This method generates a soft-label dataset that accurately reflects the semantic relationships between questions and texts. It incorporates indirect relationships, enabling the model to capture nuanced connections, while the rich semantic information enhances the model's ability to handle complex contexts.

The following formally defines the soft label $y_{i,j}$ of training pair (q_i, d_j).

Definition: *Soft label* $y_{i,j}$ of (q_i, d_j)

The soft label $y_{i,j}$ of a QA pair (q_i, d_j) is defined as follows:

$$y_{i,j} = \begin{cases} 1, if i = j \\ w_{i,j}, if (d_i, d_j) \in KG \\ \max\limits_{p_{i,j}|p_{i,j}|\le d_{\max}} (\tilde{y}_{i,j}), otherwise \end{cases} \tag{38}$$

4.4 Multi-dimensional pairing through triplet similarity SBERT architecture

Traditional SBERT-based contrastive learning typically relies on a hard-label binary classification framework, where a query q is paired with a single relevant document (d^+) and trained to maximize their similarity, while all other documents are implicitly considered irrelevant. This approach, while effective in certain cases, presents inherent limitations. The reliance on pre-defined hard labels fails to capture the continuous and dynamic nature of semantic similarity, leading to suboptimal generalization in retrieval-based tasks. Moreover, traditional SBERT contrastive learning often selects the most and least similar documents as positive and negative samples, respectively, which can result in overfitting to extreme cases rather than learning nuanced distinctions between documents with varying degrees of relevance.

To address these issues, the proposed Tri-SBERT model introduces a randomized triplet selection strategy that dynamically determines positive (d^+) and negative (d^-) samples for each query. Instead of selecting (d^+) as the most similar document and (d^-) as the least similar document from the entire corpus, Tri-SBERT randomly samples two documents from the dataset and determines their roles based on their cosine similarity with respect to the query. The document exhibiting a higher similarity score with the query is designated as (d^+), while the document with a lower similarity score is assigned as (d^+). This design prevents the model from being biased towards extreme examples and allows for a more diverse and adaptive learning process, ensuring that the training pairs capture a wider spectrum of semantic relationships. Furthermore, this randomized selection process reduces computational overhead by eliminating the need for an exhaustive search over the entire dataset, making Tri-SBERT both more scalable and efficient.

As shown in Fig. 3, recall that $\mathcal{D} = \{d_1, d_2, \ldots, d_n\}$ be a set of textual documents, where each document d_i generates a corresponding query q_i. The goal of Tri-SBERT is to learn an embedding function $f : (q, d) \to \mathbb{R}^d$ that maps queries and documents to a shared latent space, ensuring that relevant documents are closer to their corresponding queries than irrelevant ones. The triplet selection and loss function are defined as follows. Given a query q, two documents d_i and d_j are randomly sampled from the document set $\mathcal{D}$. Their cosine similarity with the query is computed as:

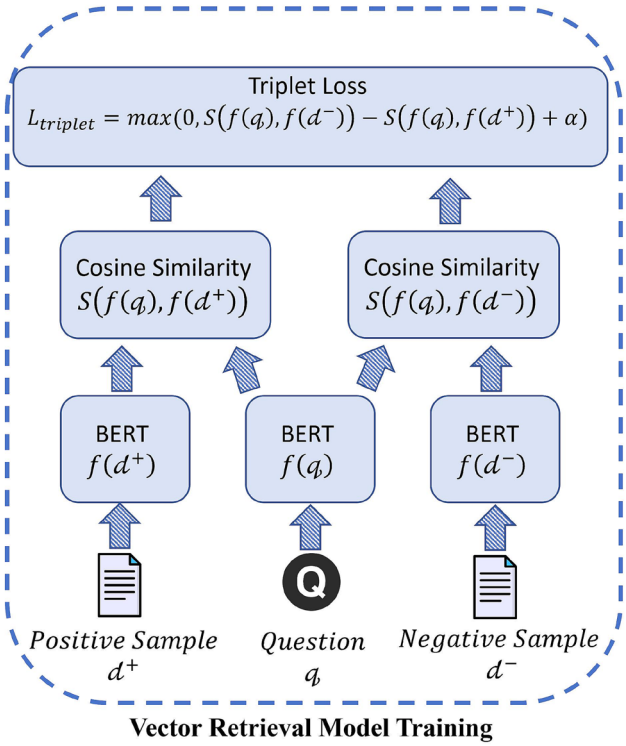

Fig. 3 The architecture of the Multi-Dimensional Pairing Through Triplet Similarity SBERT

$$\rho_i = \cos\left(f(q), f(d_i)\right) \tag{39}$$

$$\rho_j = \cos\left(f(q), f(d_j)\right) \tag{40}$$

the positive and negative documents are then assigned based on their similarity values:

$$(d^+) = \arg\max\left(\rho_i, \rho_j\right) \tag{41}$$

$$(d^-) = \arg\min\left(\rho_i, \rho_j\right) \tag{42}$$

this ensures that within each randomly sampled pair, the document more relevant to q is used as (d^+), while the less relevant one is treated as (d^-).

The model is trained using a triplet loss, which enforces a margin-based separation between the positive and negative documents:

$$L_{triplet} = \max\left(0,\ S(f(q),\ f(d^-)) - S(f(q),\ f(d^+)) + \alpha\right) \tag{43}$$

here, α is a margin hyperparameter that ensures a sufficient separation between positive and negative samples, reinforcing the model's ability to discriminate between relevant and irrelevant documents. The similarity function $S()$ denotes cosine similarity.

Through this formulation, Tri-SBERT directly optimizes relative similarity ranking rather than relying on absolute similarity scores, making the learned embeddings more discriminative and generalizable.The effectiveness of Tri-SBERT can be theoretically demonstrated by analyzing its ability to achieve a more structured semantic space compared to traditional SBERT contrastive learning. Let f_{Tri} and f_{SBERT} be the embedding functions learned by Tri-SBERT and SBERT, respectively. Define the expected similarity margin for Tri-SBERT as

$$\begin{aligned}\Delta_{Tri} =& \mathbb{E}_{d^+,d^-}\left[\cos\left(f_{Tri}\left(q\right), f_{Tri}\left(d^+\right)\right)\right. \\ &\left. - \cos\left(f_{Tri}\left(q\right), f_{Tri}\left(d^-\right)\right)\right]\end{aligned} \tag{44}$$

similarly, the expected similarity margin for SBERT as

$$\begin{aligned}\Delta_{SBERT} =& \mathbb{E}_{d^+,d^-}\left[\cos\left(f_{SBERT}\left(q\right), f_{SBERT}\left(d^+\right)\right)\right. \\ &\left. - \cos\left(f_{SBERT}\left(q\right), f_{SBERT}\left(d^-\right)\right)\right]\end{aligned} \tag{45}$$

The following theorem establishes that the expected separation between positive and negative similarity scores is larger for Tri-SBERT, leading to better semantic discrimination and generalization.

Theorem: *For any given query, the expected separation between the positive and negative document similarities is greater in Tri-SBERT than in SBERT, formally expressed as*

$$\Delta_{Tri} \geq \Delta_{SBERT} \tag{46}$$

It implies that Tri-SBERT consistently enforces a wider margin between relevant and irrelevant documents, thus improving retrieval performance.

Proof: Since SBERT uses binary classification, it assigns fixed labels to positive and negative samples without enforcing a relative ranking constraint. In contrast, Tri-SBERT explicitly optimizes a dynamic ranking loss, which guarantees that the similarity difference between positive and negative pairs is at least α, leading to greater separation in the embedding space. By the triplet loss constraint in Tri-SBERT, that is

$$\left[\cos\left(f_{\text{Tri}}(q), f_{\text{Tri}}(d^+)\right) - \cos\left(f_{\text{Tri}}(q), f_{\text{Tri}}(d^-)\right)\right] > \alpha \tag{47}$$

since (d^+) and (d^-) are randomly sampled, they cover a broader range of semantic similarity than the hard-labeled pairs in SBERT, leading to a higher expected separation margin:

$$\mathbb{E}[\Delta_{Tri}] > \mathbb{E}[\Delta_{SBERT}] \tag{48}$$

thus, Tri-SBERT consistently achieves a greater semantic separation between positive and negative examples, leading to superior retrieval accuracy and generalization.

4.5 Utilization of trained Tri-SBERT in RAG-QA system

Following the completion of Tri-SBERT training, the model is employed to enhance the RAG QA system. This stage involves two primary steps: encoding all answers in the QA system into a vector database and performing query-based retrieval to generate the final response. These processes are elaborated below.

To enable efficient and accurate information retrieval, all candidate answers $\mathcal{D} = \{d_1, \ldots, d_n\}$ in the QA system are preprocessed using the trained Tri-SBERT model. Each textual answer d_i is encoded into a dense vector representation $\vec{d}_i$ as follows:

$$\vec{d}_i = f_{Tri}(d_i) \in \mathbb{R}^d \tag{49}$$

where $f_{Tri} : \mathcal{Q} \to \mathbb{R}^d$ maps textual queries from the input space $\mathcal{Q}$ to a d-dimensional latent space.

The encoded vectors $\vec{d}_i$, along with their corresponding textual answers d_i, are stored as key-value pairs in a vector database $\mathcal{K}$:

$$k = \{(d_1, \vec{d}_1), ..., (d_n, \vec{d}_n)\} \tag{50}$$

this vector database functions as a semantic repository, enabling fast and precise similarity-based retrieval. When a user inputs a query q, it is first encoded into a dense vector representation $\vec{q}$ using the same Tri-SBERT model:

$$\vec{q} = f_{Tri}(q) \in \mathbb{R}^d \tag{51}$$

the query vector $\vec{q}$ is then compared with all the stored answer vectors in the vector database $\mathcal{K}$. The similarity between the query vector $\vec{q}$ and each answer vector $\vec{d}_i$ is computed using the cosine similarity metric, defined as:

$$sim\left(\vec{q}, \vec{d}_i\right) = \frac{\vec{q}\,\vec{d}_i}{\| \vec{q} \| \| \vec{d}_i \|} \tag{52}$$

A subset of relevant documents is retrieved based on a predefined threshold τ:

$$\mathcal{D}_{retrieved} = \left\{ d_i \in \mathcal{D} | sim(\vec{q}, \vec{d}_i) \geq \tau \right\} \tag{53}$$

where $\mathcal{T}$ is an empirically chosen threshold ensuring relevance.

Once the retrieval stage is completed, the retrieved documents $\mathcal{D}_{retrieved}$ serve as contextual evidence for the generative model g_{RAG}, which produces the final response:

$$a^* = g_{RAG}(q, \mathcal{D}_{retrieved}) \tag{54}$$

where a^* denotes the optimal answer generated for the input query q. The model aims to maximize the conditional probability $\mathcal{A}(a|q, \mathcal{D}_{retrieved})$, where $a \in \mathcal{A}$ is a candidate answer from the set of all possible answers $\mathcal{A}$.

The integration of triplet-trained embeddings with our BM25 similarity graph–based soft-labeling retrieval framework helps align retrieved documents with the query at a finer semantic granularity, thereby improving the accuracy and relevance of downstream generation. By leveraging Tri-SBERT's structured embedding space, the RAG-QA system can mitigate common retrieval errors, better separate confusable near-negatives, and capture subtle semantic relationships, leading to higher-quality answers.

5 Performance evaluation

This section presents a comprehensive performance evaluation of the proposed STaR framework, benchmarking it against representative retrieval baselines under diverse configurations. This study assesses retrieval effectiveness using a suite of quantitative metrics and performs rigorous hyperparameter tuning to ensure fair comparisons. The objective is to demonstrate that STaR not only surpasses traditional and hybrid retrieval models in accuracy, but also maintains robust and consistent performance under computational constraints.

5.1 Datasets

To support the training and evaluation of our retrieval-augmented QA system, this study constructed a large-scale, domain-specific dataset in collaboration with Ai3 Artificial Intelligence Co., Ltd. The dataset was collected from an internal enterprise knowledge base covering product documentation, customer-support knowledge, and domain tutorials. To satisfy privacy constraints, all materials were de-identified before processing, and any personal identifiers or sensitive fields were removed or masked, while preserving the informational content needed for retrieval and question answering.

After quality filtering and de-duplication on the raw enterprise corpus, we retained 34,279 high-quality question–passage (QA) pairs, each generated via GPT-4 with prompt engineering to ensure semantic clarity, domain representativeness, and internal consistency. Before QA

Table 1 Dataset overview

Dataset name	Category	Source	Data type	Number of entries
Video subtitles	Enterprise data	Ai3	QA-PAIR	3030
Customer KPI	Domain knowledge	Ai3	QA-PAIR	7986
CRM	Enterprise data	Ai3	QA-PAIR	5335
Ai3 company vision	Enterprise data	Ai3	QA-PAIR	5880
Ai3 product introduction	Enterprise data	Ai3	QA-PAIR	1404
Digital transformation	Domain knowledge	Ai3	QA-PAIR	3906
CTI terminology	Domain knowledge	Ai3	QA-PAIR	6738
Chatbot history	Domain knowledge	Ai3	QA-PAIR	4680

construction, the raw texts were normalized with lightweight cleaning, including the removal of duplicated entries, non-informative symbols, and formatting artifacts, followed by consistent punctuation and whitespace normalization. Each passage was then used to generate a question under prompt constraints that enforced passage-grounded answering, and low-quality generations were filtered by rule-based checks, including supportability by the associated passage and the removal of ambiguous or overly generic questions. For downstream retrieval and graph construction, all texts were tokenized using CKIP, and BM25 scores were computed with fixed settings consistent with Sect. 4 to ensure reproducible similarity estimation. Table 1 summarizes the dataset composition across content domains and categories, including entry counts for each source.

These QA pairs reflect realistic information-seeking behaviors across enterprise scenarios such as customer service, legal advisory, and technical product support. For evaluation, we reserved 200 manually verified question–passage pairs as a held-out test set, where each question corresponds to a unique positive passage. The remaining 34,079 exact-match QA pairs were randomly divided into training and validation splits with a 90/10 ratio using a fixed random seed (42) for reproducibility. The split is finalized before any similarity-graph construction or training-set augmentation. To prevent leakage introduced by augmentation, all augmented instances derived from the same original QA pair were kept within the same split, and the validation and test splits were used exclusively for hyperparameter selection and final reporting, respectively. We further enforced split disjointness by verifying zero overlap of original QA-pair identifiers, as well as exact-match normalized questions and passages, across the splits. The thresholded BM25 similarity graph used for soft-label generation is constructed only from the training split passages, and all multi-hop traversals for generating training soft labels are restricted to nodes within the training graph. Neither validation nor test queries or answers are used in graph construction or soft-label generation.

Table 2 Experimental environment of this research system

Component	Specification
Hardware environment	
CPU	i9-12,900
GPU	4090 24G
RAM	64 GB
CUDA version	v12.1
cuDNN version	v9.2.1
Software environment	
Python	3.8.16
Torch	2.0.0
Transformers	4.28.1

To enrich semantic diversity and emulate realistic retrieval challenges, we applied our proposed similarity graph–based expansion and soft-labeling framework only to the training split, expanding the training set to 98,037 QA pairs by introducing multi-hop similarity-graph expansions, paraphrastic variants, and semantically proximal passages derived from the original training data. Each augmented instance was annotated with a graded relevance score computed based on similarity graph traversal depth and SBERT cosine similarity.

Table 1 indicates that the corpus covers both enterprise data and domain knowledge sources, supporting diverse information-seeking scenarios. Unlike conventional QA datasets that treat relevance as binary, our corpus encodes continuous semantic signals, enabling fine-grained retriever optimization with graded soft labels and triplet-based objectives, rather than relying solely on binary classification supervision. For evaluation purposes, this study curated a dedicated test set comprising 200 manually verified QA pairs, each associated with a unique gold-standard answer. This set was independently reviewed to ensure semantic clarity, contextual specificity, and lexical distinctiveness. All baseline retrievers were retrained on our dataset under consistent experimental conditions to ensure fair comparisons. Retrieval performance was assessed using standard metrics, including Hit Rate (HR), Mean Reciprocal Rank (MRR), and Average Rank Position (ARP), supporting comprehensive analysis across semantic relevance and positional ranking dimensions.

5.2 Experimental environment

All experiments were conducted on a high-performance computing system, with detailed specifications summarized in Table 2.

The hardware setup included an Intel Core i9-12900 CPU, 64 GB of RAM, and an NVIDIA GeForce RTX 4090 GPU

equipped with 24 GB of memory. The system was configured with CUDA 12.1 and cuDNN 9.2.1 for GPU acceleration.The software environment consisted of Python 3.8.16 and PyTorch 2.0.0 as the core machine learning framework. Additionally, this study employed the HuggingFace Transformers library to support model initialization, tokenizer management, and embedding operations. This configuration ensured compatibility and efficiency during training, fine-tuning, and evaluation phases of all retriever models.

5.3 Model parameter optimization experiments

To rigorously evaluate the retrieval performance and robustness of the proposed STaR retriever, this study conducted a series of parameter optimization experiments targeting three key hyperparameters: the similarity threshold τ, the maximum graph traversal depth d_{max}, and the decay factor α. These parameters govern the construction of the similarity graph and the propagation of soft labels, which are central to the model's ability to balance semantic expansion and retrieval precision. The results of these experiments are presented in Table 3, where Hit Rate metrics (HR@1, HR@3, HR@5) are reported across all configuration combinations. The HR metric measures the proportion of queries for which the correct answer appears within the $Top-N$ retrieved results. Higher HR values indicate better retrieval effectiveness. These metrics offer a practical benchmark for evaluating the retriever's ability to return relevant answers under different semantic constraints.

Table 3 Parameter optimization experiment—HR metrics

τ	d_{max}	α	HR-1 ↑	HR-3 ↑	HR-5 ↑
0.4	3	0.5	0.72	0.77	0.83
		0.7	0.75	0.81	0.87
		1	0.74	0.79	0.86
	5	0.5	0.74	0.80	0.85
		0.7	0.76	0.82	0.89
		1	0.75	0.80	0.87
	7	0.5	0.75	0.81	0.86
		0.7	0.77	0.83	0.90
		1	0.77	0.81	0.89
0.6	3	0.5	0.76	0.79	0.87
		0.7	0.79	0.84	0.92
		1	0.77	0.83	0.91
	5	0.5	0.77	0.83	0.87
		0.7	0.82	0.86	0.92
		1	0.79	0.85	0.91
	7	0.5	0.78	0.84	0.88
		0.7	0.84	0.87	0.93
		1	0.80	0.85	0.92
0.8	3	0.5	0.74	0.78	0.85
		0.7	0.78	0.82	0.90
		1	0.76	0.81	0.87
	5	0.5	0.76	0.81	0.86
		0.7	0.81	0.82	0.89
		1	0.78	0.83	0.88
	7	0.5	0.77	0.82	0.88
		0.7	0.83	0.85	0.90
		1	0.79	0.83	0.88

The similarity threshold τ was varied across {0.4, 0.6, 0.8} to represent increasing strictness in semantic similarity. As shown in Table 3, this range was chosen to examine the trade-off between semantic coverage and retrieval precision: lower τ values promote broader graph connectivity but introduce noise, whereas higher τ values enforce stricter matching at the cost of graph fragmentation. Specifically, τ = 0.4 allows looser semantic connections, promoting broad graph connectivity and higher recall potential; however, it also introduces increased noise from weakly related nodes. In contrast, τ = 0.8 enforces strict semantic matching, reducing spurious links but fragmenting the graph structure and limiting multi-hop propagation. The intermediate value τ = 0.6 was chosen to balance these opposing effects by preserving meaningful associations while mitigating semantic drift. Thresholds below 0.4 were empirically tested and found to degrade retrieval quality due to over-inclusion of irrelevant passages, while values above 0.8 overly pruned the graph, leading to sparse coverage and reduced hit rates.

The graph traversal depth d_{max} was set to {3, 5, 7}, corresponding to shallow, intermediate, and deep graph exploration. As shown in Table 3, these values were selected based on practical considerations in multi-hop similarity propagation: a depth of 3 allows local neighborhood propagation, 5 captures mid-range semantic paths, and 7 enables full exploration within a typical semantic neighborhood. Depths smaller than 3 were too limited to reach semantically relevant nodes beyond direct edges, while values above 7 significantly increased computational cost and introduced semantically weaker nodes, leading to retrieval diffusion.

The decay factor $\alpha \in \{0.5, 0.7, 1.0\}$ controls the attenuation of soft label confidence over longer graph paths. As shown in Table 3, this range was selected to systematically examine different levels of trust allocation across semantic distances. Specifically, α = 0.5 emphasizes short, high-confidence paths; α = 1.0 applies uniform confidence to all paths, regardless of length; and α = 0.7 offers a moderated decay that balances semantic breadth with reliability. Lower values such as $\alpha < 0.5$ were excluded due to their tendency to overly suppress multi-hop associations, thereby diminishing recall. Conversely, values exceeding 1.0 are theoretically inappropriate, as they would amplify rather than attenuate the influence of longer, weaker connections—contradicting the intended regularization effect of the decay mechanism.

As summarized in Table 3, the configuration τ = 0.6, d_{max} = 7, and α = 0.7 delivers the highest overall retrieval performance, achieving top values across all three metrics:

HR@1, HR@3, and HR@5. This combination exemplifies a well-balanced interaction among the three hyperparameters. The intermediate similarity threshold $\tau = 0.6$ preserves sufficient semantic coverage while filtering out weakly related nodes, ensuring meaningful connectivity without excessive noise. Simultaneously, the graph traversal depth $d_{max} = 7$ enables more comprehensive multi-hop propagation over the similarity graph, reaching semantically rich nodes beyond local neighborhoods without incurring significant diffusion or overreach. The decay factor $\alpha = 0.7$ introduces a moderated attenuation that prioritizes shorter, high-confidence paths while still allowing contribution from informative mid-range paths.

To further illustrate and validate these findings, Fig. 4 provides a heatmap-based visualization of the retrieval performance results detailed in Table 3 Each subfigure corresponds to a fixed similarity threshold $\tau \in \{0.4, 0.6, 0.8\}$, displaying the Hit Rate (HR) values under varying combinations of decay factor α (horizontal axis) and graph depth d_{max} (vertical axis). Within each heatmap, color intensity indicates performance magnitude—darker shades represent higher HR values, thereby highlighting optimal regions in the hyperparameter space.

Across all τ values, $\alpha = 0.7$ consistently outperforms $\alpha = 0.5$ and $\alpha = 1.0$, as shown by the darker central column in each heatmap. This moderate decay promotes reliable soft-label propagation by emphasizing informative mid-range paths while avoiding over- or under-attenuation.

Among the three τ configurations, the middle row of subfigures corresponding to $\tau = 0.6$ consistently exhibits higher HR values across all α and d_{max} settings, with numerical results that exceed those under $\tau = 0.4$ and $\tau = 0.8$. Unlike $\tau = 0.4$, which permits excessive semantic expansion, and $\tau = 0.8$, which leads to graph sparsity, $\tau = 0.6$ provides an effective trade-off that maintains semantic relevance while preserving connectivity.

In summary, Fig. 4 highlights that the optimal retrieval performance emerges from a synergistic configuration: a moderate similarity threshold ($\tau = 0.6$), sufficient graph depth ($d_{max} = 7$), and a balanced decay rate ($\alpha = 0.7$). This setting consistently occupies the darkest region across all subfigures, reaffirming its effectiveness in navigating the trade-off between semantic expansion and noise suppression.

In addition to the Hit Rate (HR) metrics discussed earlier, this study further assesses the quality of document ranking and retrieval efficiency using Mean Reciprocal Rank (MRR) and Average Rank Position (ARP). These complementary metrics offer finer-grained insights into not only whether the correct answer appears in the $Top - N$ results but also how

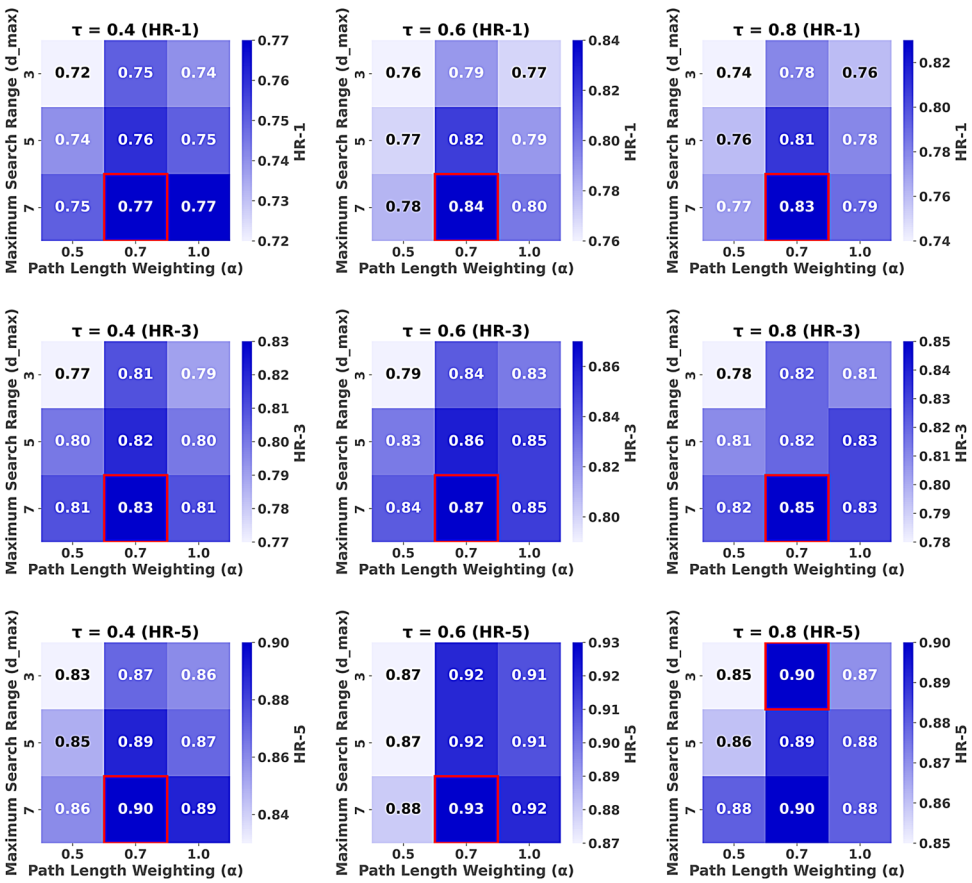

Fig. 4 Parameter interaction effects on HR metrics

early it is ranked within the list. Table 4 reports the results of all parameter configurations under these two metrics.

In alignment with the findings from Table 3, the hyperparameter configuration $\tau = 0.6$, $d_{max} = 7$, and $\alpha = 0.7$ again demonstrates the most favorable retrieval performance under ranking-sensitive metrics. As shown in Table 4, this setting achieves the highest MRR@5 score of 0.86, and outperforms all other configurations in MRR@1 and MRR@3 as well. These results indicate that this combination not only facilitates the inclusion of relevant documents but also ensures they are ranked in top positions with high precision. Although the corresponding ARP (Average Rank Position) value of 1.29 is not the absolute minimum, it remains highly competitive and reflects effective retrieval efficiency.

These outcomes collectively reinforce the design intuition behind our approach: expanding the semantic search radius through deeper graph traversal ($d_{max} = 7$) captures richer contextual associations; adopting a moderate similarity threshold ($\tau = 0.6$) balances recall and noise suppression; and applying a controlled decay factor ($\alpha = 0.7$) helps preserve confidence in relevant but non-local paths without over-amplifying noisy signals. This configuration not only significantly enhances retrieval coverage, as evidenced by superior Hit Rate scores, but also achieves strong performance in ranking precision and average rank positioning, thereby validating the practical effectiveness and robustness of our proposed method in real-world knowledge-intensive retrieval tasks.

To provide an intuitive visualization of the ranking performance across different hyperparameter settings, Fig. 5 presents a series of heatmaps based on the MRR metrics reported in Table 4. Each row corresponds to a specific MRR level—MRR@1, MRR@3, and MRR@5—while each column represents a different similarity threshold $\tau \in \{0.4, 0.6, 0.8\}$. Within each heatmap, the horizontal axis denotes the path length weighting factor α, and the vertical axis indicates the maximum search depth d_{max}. The color intensity of each cell reflects the MRR value at the corresponding configuration, with darker shades representing higher scores.

Across all three MRR levels, the configuration $\tau = 0.6$, $d_{max} = 7$, and $\alpha = 0.7$ consistently yields top performance, as reflected by the darkest cells in the center heatmaps. Specifically, this setting achieves the best scores in all three metrics—MRR@1 (0.84), MRR@3 (0.85), and MRR@5 (0.86)—validating the robustness of this configuration for ranking-sensitive retrieval.

Table 4 Parameter optimization experiment—MRR and ARP metrics

τ	d_{max}	α	MRR-1 ↑	MRR -3 ↑	MRR -5 ↑	ARP ↑
0.4	3	0.5	0.72	0.74	0.75	1.37
		0.7	0.75	0.78	0.79	1.34
		1	0.74	0.76	0.77	1.38
	5	0.5	0.74	0.76	0.77	1.32
		0.7	0.76	0.79	0.80	1.37
		1	0.75	0.77	0.78	1.37
	7	0.5	0.75	0.77	0.78	1.30
		0.7	0.77	0.80	0.81	1.36
		1	0.77	0.79	0.80	1.38
0.6	3	0.5	0.76	0.77	0.79	1.36
		0.7	0.79	0.81	0.83	1.36
		1	0.77	0.79	0.81	1.40
	5	0.5	0.77	0.79	0.80	1.25
		0.7	0.82	0.84	0.85	1.27
		1	0.79	0.81	0.82	1.35
	7	0.5	0.78	0.80	0.81	1.28
		0.7	0.84	0.85	0.86	1.29
		1	0.80	0.82	0.83	1.34
0.8	3	0.5	0.74	0.76	0.77	1.35
		0.7	0.78	0.80	0.82	1.34
		1	0.76	0.78	0.80	1.30
	5	0.5	0.76	0.78	0.79	1.28
		0.7	0.81	0.82	0.83	1.28
		1	0.78	0.80	0.81	1.29
	7	0.5	0.77	0.79	0.80	1.30
		0.7	0.83	0.84	0.85	1.20
		1	0.79	0.81	0.82	1.24

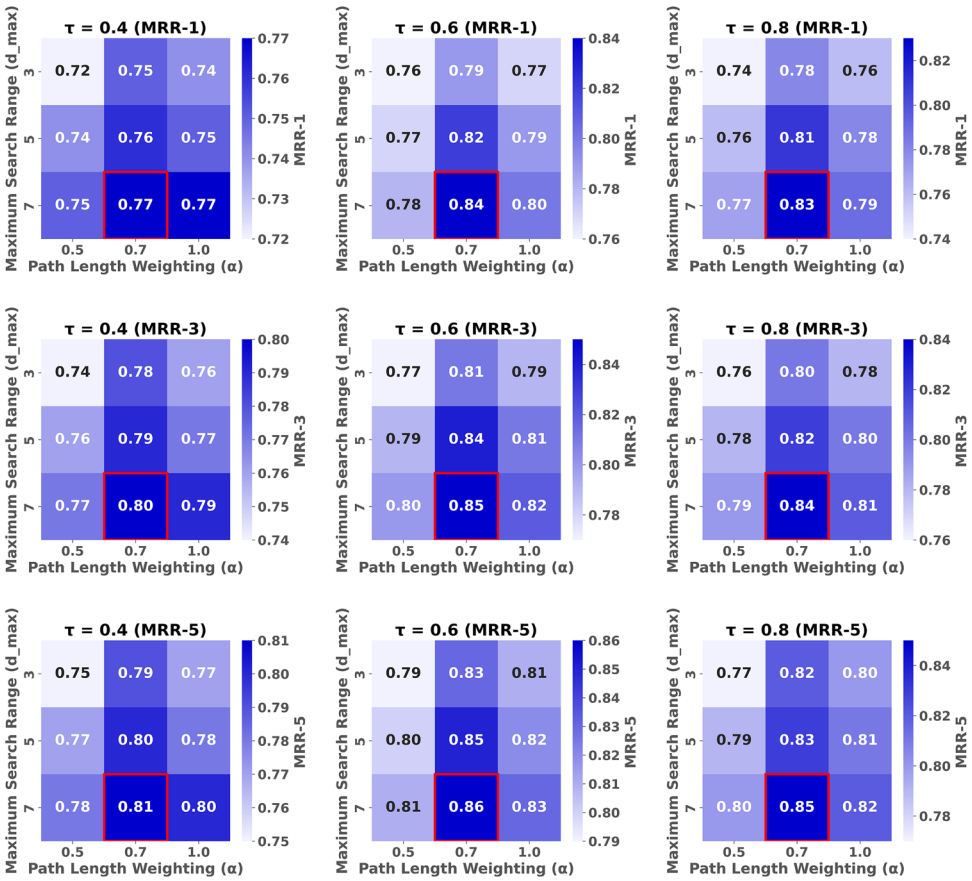

Fig. 5 Parameter interaction effects on MRR metrics

In addition, increasing d_{max} from 3 to 7 consistently improves MRR values across all α levels, reaffirming the importance of deeper graph traversal for capturing multi-hop semantic relations. Likewise, $\alpha = 0.7$ remains the most favorable decay factor, outperforming both the overly conservative $\alpha = 0.5$ and the uniform weighting $\alpha = 1.0$, by effectively balancing confidence propagation along informative but longer semantic paths.

Taken together, Fig. 5 visually substantiates the conclusions drawn from Table 4, highlighting that the optimal balance among semantic coverage, ranking quality, and graph traversal depth is achieved under the configuration $\tau = 0.6$, $d_{max} = 7$, and $\alpha = 0.7$.

It should be noted that the maximum traversal depth and decay factor are fixed after validation-based hyperparameter selection rather than assigned as arbitrary static priors. The parameter optimization results show that the configuration with a moderate similarity threshold, sufficient graph traversal depth, and balanced decay rate achieves the best overall trade-off between semantic coverage and noise suppression. Keeping these parameters fixed during training ensures that all soft-label training instances are generated under the same graph propagation rule, which improves reproducibility and makes the effect of each hyperparameter interpretable.

Nevertheless, fixed graph-level parameters may not fully capture the varying difficulty of different queries, passages, or graph paths. For example, semantically ambiguous query-passage pairs may require stronger attenuation or more conservative propagation, whereas easier pairs may benefit from broader graph expansion. Inspired by dynamic difficulty-aware training strategies such as token-level self-evolution [45], a possible extension is to estimate node-level, edge-level, or query-passage-level difficulty during retriever fine-tuning and dynamically adjust graph edge weights, decay coefficients, or soft-label confidence. Unlike token-level methods designed for sequence generation, such an extension would operate at the retrieval-pair or graph-relation level, making it more suitable for RAG-QA retriever optimization.

To complement the MRR-based analysis, Fig. 6 visualizes the parameter interaction effects on retrieval efficiency using the Average Rank Position (ARP) metric. Similar to the layout in Fig. 5, each subfigure corresponds to a fixed similarity threshold $\tau \in \{0.4, 0.6, 0.8\}$, with the x-axis representing the decay factor α and the y-axis representing the maximum graph traversal depth d_{max}. Darker shades indicate lower ARP values, denoting higher ranking efficiency.

Notably, the best ARP value of 1.20 is achieved under the configuration $\tau = 0.8$, $d_{max} = 7$, and $\alpha = 0.7$. This suggests

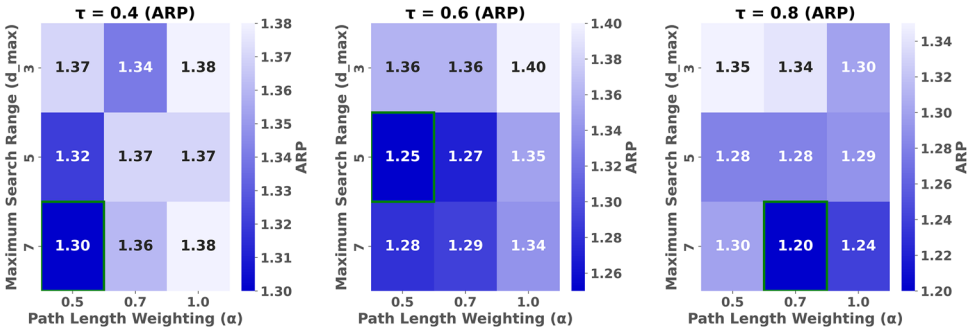

Fig. 6 Parameter interaction effects on ARP metrics

Table 5 Ablation study on SBERT and soft label

method	HR- 1 ↑	HR-3 ↑	HR-5 ↑	MRR-1 ↑	MRR-3 ↑	MRR-5 ↑	ARP ↓
w/ Binary SBERT	0.82	0.84	0.87	0.82	0.83	0.84	1.25
w/ Triplet SBERT	0.79	0.80	0.85	0.79	0.79	0.80	1.22
w/o soft label	0.69	0.73	0.79	0.69	0.71	0.72	1.35
STaR	0.84	0.87	0.93	0.84	0.85	0.86	1.29

that stricter semantic filtering helps suppress spurious associations, allowing highly relevant nodes to surface earlier in the retrieval results. Compared to looser thresholds such as $\tau = 0.4$, which admit broader but noisier graph connectivity, the higher threshold $\tau = 0.8$ enhances ranking efficiency by preserving only the most semantically pertinent connections.

In summary, Fig. 6 confirms that while configurations optimized for HR and MRR also achieve strong ARP performance, absolute ranking efficiency may favor slightly stricter semantic constraints. Nonetheless, the overall effectiveness and robustness of the $\tau = 0.6$, $d_{max} = 7$, and $\alpha = 0.7$ setting remain evident across all evaluation metrics.

5.4 Ablation and robustness studies

To assess validate the effectiveness of the proposed tri-similar SBERT model, this study conducted an ablation study comparing the retrieval performance of binary SBERT, ternary SBERT, and the proposed tri-similar SBERT as well as investigating the impact of using hard and soft labels during training.

To investigate the effectiveness of the proposed STaR framework, this study conducts an ablation study focusing on two critical design components: the use of soft-label propagation and the similarity-based triplet loss within the SBERT retriever. Table 5 summarizes the results across all evaluation metrics (HR, MRR, and ARP) under four configurations.

The "w/o soft label" configuration eliminates the soft-label training mechanism entirely and instead employs traditional hard labels for model supervision. As expected, this results in the most substantial performance degradation across all metrics, with HR@5 dropping from 0.93 to 0.79, MRR@5 declining from 0.86 to 0.72, and ARP increasing to 1.35. These results underscore the central role of soft labels in guiding semantic expansion and contextual matching.

This study further compares two variants of SBERT training: "w/ Triplet SBERT" employs a traditional distance-based triplet loss, where anchor-positive–negative relationships are optimized using margin-based distance objectives. In contrast, our full STaR model adopts a similarity-based triplet formulation that directly maximizes the semantic alignment in the embedding space via cosine similarity. As shown in Table 5, although the distance-based variant slightly improves ARP (1.22 vs. 1.29), it underperforms across all HR and MRR metrics compared to STaR, suggesting that similarity-based optimization better aligns with the goal of semantic retrieval.

Interestingly, the "w/ Binary SBERT" configuration—where SBERT is trained using a binary classification loss instead of triplet learning—achieves competitive results (e.g., HR@1 = 0.82), but still falls short of the STaR setup. This highlights the importance of fine-grained pairwise relational modeling, which is better captured by triplet learning under soft-label supervision.

Figure 7 presents a delta heatmap that quantifies the performance gains of the proposed STaR model over three baseline configurations: Binary SBERT, Triplet SBERT, and the version without soft-label training. Each cell in the heatmap represents the difference in performance for a specific evaluation metric, with darker green indicating larger gains and orange indicating a performance drop.

The results highlight several key observations. First, the most significant improvements are observed when comparing STaR to the "w/o soft label" variant. Across all Hit Rate (HR@1, HR@3, HR@5) and Mean Reciprocal Rank (MRR@1, MRR@3, MRR@5) metrics, STaR achieves substantial gains, with deltas ranging from +0.12 to +0.15. This confirms the crucial role of the soft-label propagation

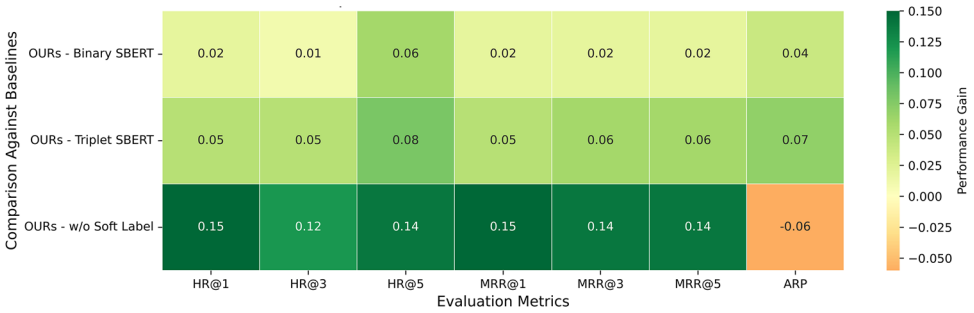

Fig. 7 Delta heatmap: performance gains of STaR over baseline variants

Table 6 Backbone robustness of STaR on the Ai3 dataset

Method	HR- 1 ↑	HR-3 ↑	HR-5 ↑	MRR-1 ↑	MRR-3 ↑	MRR-5 ↑	ARP ↓
BM25 [15]	0.72	0.74	0.79	0.72	0.73	0.74	1.25
ToG [30]	0.82	0.83	0.87	0.82	0.82	0.83	1.16
STaR(RoBERTa-base) [24]	0.83	0.86	0.92	0.83	0.84	0.85	1.28
STaR(DeBERTa-base) [25]	0.85	0.88	0.93	0.85	0.86	0.87	1.26
STaR(BERT-base) [23]	0.84	0.87	0.93	0.84	0.85	0.86	1.29

mechanism in enabling deeper semantic understanding and more precise ranking. Notably, the ARP metric shows a slight decline (– 0.06) under STaR compared to the "w/o soft label" baseline, which may be attributed to the broader semantic expansion introduced by soft labels, potentially including relevant but non-top-ranked items.

In the comparison with "Triplet SBERT", which shares the same triplet-based loss function but uses distance rather than cosine similarity, STaR exhibits consistent improvements across all metrics, with deltas between + 0.05 and + 0.08. This suggests that computing similarity-based contrastive training—rather than distance-based—enhances representation learning for retrieval tasks, leading to better $Top - N$ ranking and positioning.

Lastly, while the difference between STaR and "Binary SBERT" is smaller, it remains positive across the board. STaR achieves modest improvements ranging from + 0.01 to + 0.06 in HR and MRR metrics and a + 0.04 gain in ARP. This indicates that while binary classification remains a competitive baseline, the combination of soft-label training and triplet-aware similarity optimization provides a more nuanced learning signal, resulting in overall better retrieval performance.

Overall, the heatmap in Fig. 7 provides a clear visual confirmation that both the soft-label mechanism and the similarity-aware triplet SBERT contribute synergistically to the performance advantages observed in STaR. The consistent and wide-ranging gains across evaluation dimensions validate the robustness and efficacy of the proposed approach.

In addition to the component ablations above, we further examine whether the effectiveness of STaR is sensitive to the choice of pretrained encoder backbone.

As shown in Table 6, we conducted a backbone robustness study on the Ai3 dataset by replacing the BERT-base encoder with alternative pre-trained encoders while keeping the STaR training protocol unchanged. All variants share the same Ai3 data split, the same similarity-graph soft labels computed from the Ai3 training passages, and the same triplet optimization and hyperparameter settings. Evaluation is performed on the held-out Ai3 test set of 200 manually verified QA pairs, and retrieval effectiveness is measured by HR, MRR, and ARP as defined in Sect. 5.3. For reference, we also report BM25 and ToG as non-STaR baselines to contextualize absolute performance; the backbone robustness comparison is among the STaR variants.

Overall, the performance remains stable across different backbones, indicating that the observed gains are robust to the choice of pre-trained encoder backbone. In particular, HR@5 varies within a narrow range from 0.92 to 0.93, and MRR@5 ranges from 0.85 to 0.87, while ARP remains within a narrow range from 1.26 to 1.29 across all variants. Here, ARP measures the average rank of the first relevant passage, where lower is better; although ToG has a slightly lower ARP, the ARP range across STaR variants remains narrow, supporting the robustness conclusion. Among the compared backbones, DeBERTa achieves the strongest ranking quality with the highest MRR@5 of 0.87, which is consistent with its stronger contextual modeling capacity, while RoBERTa provides competitive performance with HR@5 of 0.92 and MRR@5 of 0.85 under the same protocol. These results confirm that the gains reported for STaR primarily arise from the similarity-graph soft supervision and triplet-aware embedding optimization, rather than from choosing a specific pre-trained backbone.

Despite the overall stability, small differences can be observed across backbones. DeBERTa-base yields the best ranking quality in our setting (MRR@5 = 0.87, ARP = 1.26), suggesting that its enhanced contextual modeling and

Table 7 Comparative experiments-HR,MRR and ARP ndicators

method	HR- 1 ↑	HR-3 ↑	HR-5 ↑	MRR-1 ↑	MRR-3 ↑	MRR-5 ↑	ARP ↓
BM25[15]	0.72	0.74	0.79	0.72	0.73	0.74	1.25
ToG [30]	0.82	0.83	0.87	0.82	0.82	0.83	1.16
STaR w/o Soft Label	0.69	0.73	0.79	0.69	0.71	0.72	1.35
STaR-WeakSim (τ=0.4)	0.77	0.83	0.90	0.77	0.80	0.81	1.36
STaR-StrongDecay (α=1)	0.80	0.85	0.92	0.80	0.82	0.83	1.34
STaR- ShortestPath (d_{max}=3)	0.79	0.84	0.92	0.79	0.81	0.83	1.36
STaR-Binary SBERT	0.82	0.84	0.87	0.82	0.83	0.84	1.25
STaR	0.84	0.87	0.93	0.84	0.85	0.86	1.29

attention design can slightly improve fine-grained ranking among near-negative candidates. BERT-base performs competitively (HR@5=0.93, MRR@5=0.86), indicating that the proposed supervision and triplet optimization already dominate performance under a fixed training protocol. RoBERTa-base shows comparable but slightly lower results (HR@5=0.92, MRR@5=0.85), which may be due to the fact that we intentionally keep identical hyperparameter settings across backbones for a controlled robustness study rather than re-tuning each backbone. Overall, the performance gaps are small ($\leq$0.02 on MRR@5), reinforcing that STaR's improvements are primarily driven by the similarity-graph soft supervision and triplet-aware optimization instead of a specific pretrained backbone.

5.5 Top-N retrieval rate and benchmark comparisons

To validate retrieval performance, we compare STaR with traditional retrieval baselines and representative retriever baselines on a held-out test set. The test set is the 200 manually verified QA pairs described in Sect. 5.1 and is never used in similarity-graph construction, soft-label generation, training, or hyperparameter selection. STaR achieves the highest Hit Rate across all $Top-N$ settings, with an HR@5 of 0.93, outperforming BM25 (0.79) and ToG (0.87). Similarly, STaR obtains an MRR@5 of 0.86, indicating that relevant results are more likely to appear near the top of the ranked list. While its ARP (1.29) is marginally higher than that of ToG (1.16), STaR still achieves strong ranking efficiency, indicating that relevant answers are consistently retrieved near the top positions.

To further evaluate model robustness, several variants of STaR were examined under different architectural and hyperparameter configurations. STaR w/o Soft Label and STaR-Binary SBERT both show a clear performance drop, confirming the importance of soft semantic supervision and triplet-based training. Moreover, three additional variants—STaR-WeakSim (τ = 0.4), STaR-StrongDecay (α = 1), and STaR-ShortestPath (d_{max} = 3)—were tested to explore the impact of different similarity graph traversal strategies. All three demonstrate inferior performance compared to the full STaR model. STaR-WeakSim adopts a lower similarity threshold, leading to overly cautious matching. STaR-StrongDecay applies aggressive decay to long-path semantics, reducing the influence of deeper associations. STaR-ShortestPath limits the maximum semantic depth explored in the graph, resulting in insufficient semantic propagation. These outcomes, as summarized in Table 7, highlight that the STaR configuration delivers an optimal trade-off among retrieval depth, semantic coverage, and ranking accuracy, validating its design choices under realistic conditions.

Table 8 Retrieval Recall@k on Natural Questions and TriviaQA under standard open-domain retrieval evaluation

Method	Natural Questions		Trivia QA	
	R@20 ↑	R@100 ↑	R@20 ↑	R@100 ↑
BM25 [15]	59.1	73.7	66.9	76.7
Contriever [43]	67.8	80.6	73.9	82.9
ReContriever [44]	69.4	82.6	75.9	84.1
DPR [2]	78.4	85.4	79.4	85.0
ANCE [38]	81.9	87.5	80.3	85.3
BGE-base [26]	82.3	88.5	80.0	85.9
STaR	84.8	89.9	83.5	87.8

To improve comparability with prior dense-retrieval literature, we additionally evaluate STaR on two widely used open-domain retrieval benchmarks, Natural Questions [46] and TriviaQA [47]. We report Recall@k at k=20 and 100 using the standard benchmark splits and metric definition. We include one sparse baseline (BM25) and several strong dense retrievers, namely Contriever, ReContriever, DPR, ANCE, and BGE. These baselines represent major training paradigms, including unsupervised dense pretraining, supervised hard-label fine-tuning, and strong general-purpose embedding families. For baseline numbers, we follow the results reported in the corresponding original papers under their standard evaluation settings. STaR is evaluated on the same benchmark splits using the same Recall@k definition, providing a directly comparable reference.

Table 8 summarizes Recall@k results on Natural Questions and TriviaQA. To improve comparability with prior dense-retrieval literature, we compare STaR with BM25 and

representative dense retrievers, including Contriever [43] and ReContriever [44], following standard open-domain retrieval evaluation settings. ReContriever is particularly relevant because it represents an adaptive, model-driven relevance-aware contrastive pre-training paradigm. For baseline methods, we follow the results reported in the corresponding original papers under their standard evaluation settings when available, and STaR is evaluated using the same benchmark splits and Recall@k definitions.

The results in Table 8 show that the reported dense retrievers consistently outperform BM25 on both benchmarks, suggesting that contextual encoders can capture semantic relevance beyond lexical overlap. Since some baseline results are taken from the corresponding original papers, Table 8 should be interpreted as a benchmark-level reference comparison under standard Recall@k definitions, rather than as a fully controlled reimplementation of all systems under an identical training and indexing pipeline. Under this setting, STaR achieves the highest reported Recall@k scores among the compared methods. On Natural Questions, STaR improves R@20 from 82.3 to 84.8 and R@100 from 88.5 to 89.9, corresponding to absolute gains of 2.5 and 1.4 percentage points over the strongest reported baseline for each metric. On TriviaQA, STaR improves R@20 from 80.3 to 83.5 and R@100 from 85.9 to 87.8, corresponding to absolute gains of 3.2 and 1.9 percentage points.

The comparison with ReContriever further clarifies the competitive position of STaR against adaptive model-driven relevance estimation. ReContriever reduces false positives by dynamically estimating pseudo-positive reliability during unsupervised pre-training, whereas STaR improves task-specific retriever fine-tuning by constructing corpus-level graded relevance signals before triplet-aware optimization. These results indicate that target-corpus similarity-graph supervision provides effective and complementary relevance signals for RAG-QA retrieval and that the proposed training strategy remains effective beyond the proprietary Ai3 corpus on established public QA retrieval benchmarks.

The performance gains can be explained by the way STaR constructs supervision and shapes embedding geometry. Conventional hard-label training assigns uniform relevance to all positives and mainly optimizes local pairwise objectives. In contrast, STaR derives graded targets from a thresholded BM25 similarity graph via controllable multi-hop propagation with decay, which provides richer supervision among near-neighbor passages and avoids overly penalizing semantically close but non-identical candidates. In addition, the triplet-based objective enforces relative-distance constraints among positives, weak negatives, and hard negatives, strengthening separation in the embedding space and improving ranking robustness. Consequently, STaR increases the likelihood that a correct evidence passage appears within the top retrieved set, which is reflected by consistent gains at both R@20 and R@100.

To further evaluate the competitive standing and transferability of STaR beyond the component-level ablation analysis, we additionally conduct experiments on selected datasets from the BEIR benchmark [48]. BEIR provides a heterogeneous retrieval evaluation suite covering diverse retrieval scenarios, and nDCG@10 is widely used as a primary ranking metric. In this experiment, STaR is first fine-tuned on Natural Questions and then directly evaluated on the selected BEIR datasets without target-dataset fine-tuning. No BEIR training split or relevance judgments are used to update STaR or tune its hyperparameters; the BEIR relevance judgments are used only for evaluation. Therefore, this setting should be interpreted as an NQ-trained cross-dataset transfer evaluation on selected BEIR datasets rather than a full BEIR leaderboard comparison. We report results on eleven representative BEIR datasets, including MS MARCO, NFCorpus, HotpotQA, FiQA-2018, CQADupStack, Quora, DBPedia, SCIDOCS, FEVER, Climate-FEVER, and SciFact.

Table 9 reports nDCG@10 scores on the selected BEIR datasets. We compare STaR with BM25 as a sparse lexical baseline and representative dense retrieval methods, including BERT [23], SimCSE [49], RetroMAE [50], coCondenser [51], Contriever [43], and ReContriever [44]. ReContriever is particularly relevant to this comparison because it represents an adaptive, model-driven relevance-aware contrastive pre-training paradigm. Baseline results are taken from the corresponding original papers when available, and the reproduced Contriever results are included as an additional implementation reference. The Avg row reports the mean nDCG@10 over the selected datasets, and Avg Rank reports the average rank across datasets, where a lower rank indicates better overall performance.

As shown in Table 9, STaR achieves the best average nDCG@10 among the compared methods on the selected BEIR datasets. Compared with ReContriever, STaR improves the average nDCG@10 from 39.9 to 49.5 and reduces the average rank from 3.0 to 1.0. The improvement is especially clear on datasets such as MS MARCO, HotpotQA, FiQA-2018, DBPedia, FEVER, and Climate-FEVER, where STaR obtains large absolute gains over both sparse and dense baselines. These results suggest that the NQ-trained STaR retriever retains competitive transferability on heterogeneous retrieval tasks represented by the selected BEIR datasets.

Overall, the selected BEIR results complement the NQ and TriviaQA evaluations by assessing STaR under more heterogeneous retrieval scenarios. Although this experiment does not cover the full BEIR suite, it provides additional

Table 9 Retrieval performance on selected BEIR datasets

DATASET	BM25	BERT	SimCSE	RetroMAE	coCondenser	Contriever	Contriever (reproduced)	ReContriever	STaR
MS MARCO	22.8	0.6	8.8	4.5	7.7	20.6	21.1	21.8	41.9
NFCorpus	32.5	2.5	14.0	15.3	14.4	31.7	30.0	31.9	34.7
HotpotQA	60.3	4.9	23.3	25.0	24.4	48.1	44.1	50.1	68.8
FiQA-2018	23.6	1.4	14.8	9.3	5.2	24.5	26.2	26.2	40.1
CQADupStack	29.9	2.5	20.2	17.0	9.8	28.4	28.4	28.7	37.7
Quora	78.9	3.9	81.5	69.0	66.7	83.5	83.6	84.3	85.9
DBPedia	31.3	3.9	13.7	4.6	15.1	29.2	27.6	29.3	40.4
SCIDOCS	15.8	2.7	7.4	7.4	1.9	14.9	15.0	15.6	18.6
FEVER	75.3	4.9	20.1	7.1	25.3	68.2	66.9	68.9	80.9
Climate-FEVER	21.3	4.1	17.6	4.4	9.8	15.5	15.6	15.6	24.2
SciFact	66.5	9.8	38.5	53.1	48.1	64.9	65	66.4	71.0
Avg	41.7	3.7	23.6	19.7	20.8	39.0	38.5	39.9	49.5
Avg Rank	2.6	8.9	6.4	7.0	7.3	4.5	4.2	3.0	1.0

Table 10 Direct graph use across STaR stages

Stage	Offline stage	Direct graph operation	Per-query graph traversal
Graph construction	Yes	Yes	No
Soft-label propagation	Yes	Yes	No
Retriever fine-tuning	Yes	No	No
Online retrieval	No	No	No

evidence of STaR's competitiveness against strong sparse and dense retrieval baselines, including ReContriever.

These results confirm that the proposed STaR method not only improves retrieval accuracy but also shows strong robustness across different configurations, making it effective for real-world, large-scale QA applications.

5.6 Computational trade-off analysis

The graph-based expansion in STaR introduces non-trivial offline preprocessing cost because the thresholded BM25 similarity graph must be constructed and traversed to generate graded soft labels. In our implementation, the graph is constructed over the training passages, and graph propagation is then used to produce the expanded 98,037 soft-labeled QA training instances. This cost mainly arises from BM25-based similarity computation, sparse graph construction, and multi-hop propagation over the retained graph edges. However, this procedure is performed only during training-data construction and retriever fine-tuning. Once the Tri-SBERT retriever has been trained, the similarity graph is no longer used in the online retrieval stage. Table 10 summarizes whether the thresholded BM25 similarity graph is directly constructed or traversed in each stage of STaR. The graph is used only during offline graph construction and soft-label generation. After retriever fine-tuning, online retrieval requires only query encoding and vector similarity search, without per-query graph traversal.

During inference, all candidate passages are encoded into dense vectors and stored in the vector database in advance. For each user query, the deployed system only needs to encode the query and perform vector similarity search over the pre-encoded passage representations. Thus, STaR does not introduce additional graph traversal latency at inference time. Compared with using the same encoder and vector index without graph-based soft-label training, the raw cost of query encoding and vector search remains comparable. The main practical benefit of STaR is that the improved retriever ranking quality increases the probability that relevant evidence appears near the top of the retrieved list, which may reduce redundant candidate inspection, repeated retrieval attempts, or the need for heavier multi-step retrieval pipelines in RAG-QA deployment.

This trade-off is particularly suitable for enterprise or domain-specific knowledge bases that are updated periodically rather than continuously, because the offline graph-construction cost can be amortized over many subsequent user queries. For rapidly changing corpora, however, rebuilding the entire similarity graph may introduce additional preprocessing overhead, especially when the target corpus is frequently updated.

6 Conclusions

This paper presented STaR, a retrieval fine-tuning framework for retrieval-augmented generation that combines similarity-graph–based soft labeling with triplet-aware SBERT optimization. By constructing graded relevance targets through multi-hop similarity propagation and enforcing relative-distance constraints with a triplet objective, STaR captures fine-grained semantic proximity between queries and candidate passages and improves retriever discrimination under the same inference setting. Experiments

on the proprietary Ai3 QA dataset show that STaR consistently outperforms strong baselines across Hit Rate, Mean Reciprocal Rank, and Average Rank Position, achieving an HR@5 of 0.93. Additional evaluations on public benchmarks further indicate that the proposed training signal generalizes beyond the proprietary corpus. Ablation and robustness studies verify that both the soft-label supervision and the triplet-based optimization contribute to the gains and that performance remains stable across different encoder backbones. Overall, STaR provides a practical and scalable approach to improving retrieval quality for QA systems under resource constraints. Future work will investigate incremental similarity-graph maintenance, efficient edge pruning, and adaptive graph updating strategies to reduce preprocessing overhead when the target corpus evolves. In addition, although the current STaR framework uses validation-selected deterministic graph-derived labels, future studies will explore adaptive, difficulty-aware, and uncertainty-guided soft-label generation, in which graph edge weights, decay coefficients, or soft-label confidence can be dynamically adjusted according to model-estimated uncertainty and node-level, edge-level, or query-passage-level difficulty.

Author contributions Jia-Yang Jiang: Writing – original draft, Visualization, Validation, Software, Formal analysis, Data curation, Methodology, Conceptualization. Chih-Yung Chang: Supervision, Resources, Project administration. Youxi Li: Software, Resources. Tzu-Chia Huang: Validation, Data curation. Diptendu Sinha Roy: Supervision, Writing – review & editing. All authors read and approved the final manuscript.

Funding The work was supported in part by the National Science and Technology Council of the Republic of China, Taiwan, under Grant NSTC 113-2221-E-032-033-MY2 (Corresponding author: Chih-Yung Chang.)

Data availability The datasets generated during the current study are not publicly available but are available from the corresponding author.

Declarations

Conflict of interest The authors declare no competing interests.

Ethical approval and consent to participate This study does not involve either human subjects or animals.

References

1. Lewis, P., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. Adv. Neural. Inf. Process. Syst. **33**, 9459–9474 (2020)
2. V. Karpukhin, B. Oğuz, S. Min, et al., "Dense Passage Retrieval for Open-Domain Question Answering," in *Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, Online, 2020, pp. 6769–6781.
3. O. Khattab and M. Zaharia, "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT," in *Proc. 43rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR)*, Virtual Event, Jul. 2020, pp. 39–48.
4. Algan, G., Ulusoy, I.: MetaLabelNet: Learning to Generate Soft-Labels from Noisy-Labels. IEEE Trans. Image Process. **31**, 4352–4362 (2022)
5. Xia, X., Lu, P., Gong, C., et al.: Regularly truncated M-estimators for learning with noisy labels. IEEE Trans. Pattern Anal. Mach. Intell. **46**(5), 3522–3536 (2024). https://doi.org/10.1109/TPAMI.2023.3347850
6. K. Sawarkar, A. Mangal, and S. R. Solanki, "Blended RAG: Improving RAG Accuracy with Semantic Search And Hybrid Query-Based Retrievers," arXiv:2404.07220*, 2024*.
7. H. Trivedi, N. Balasubramanian, T. Khot, et al., "Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions," in *Proc. 61st Annu. Meet. Assoc. Comput. Linguist. (ACL)*, 2023, pp. 10014–10037.
8. Song, H., Kim, M., Park, D., et al.: Learning from noisy labels with deep neural networks: a survey. IEEE Trans. Neural Netw. Learn. Syst. **34**(11), 8135–8153 (2023). https://doi.org/10.1109/TNNLS.2022.3152527
9. Xu, N., Qiao, C., Zhao, Y., et al.: Variational label enhancement for instance-dependent partial label learning. IEEE Trans. Pattern Anal. Mach. Intell. **46**(12), 11298–11313 (2024). https://doi.org/10.1109/TPAMI.2024.3455260
10. Xu, N., Liu, Y.-P., Zhang, Y., et al.: Progressive enhancement of label distributions for partial multilabel learning. IEEE Trans. Neural Netw. Learn. Syst. **34**(8), 4856–4867 (2023). https://doi.org/10.1109/TNNLS.2021.3125366
11. Sachan, D.S., Lewis, M., Yogatama, D., et al.: Questions are all you need to train a dense passage retriever. Trans. Assoc. Comput. Linguist. **11**, 651–669 (2023). https://doi.org/10.1162/tacl_a_00564
12. N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks," in *Proc. EMNLP-IJCNLP*, Hong Kong, China, Nov. 2019, pp. 3982–3992.
13. Spärck Jones, K.: A statistical interpretation of term specificity and its application in retrieval. J. Doc. **28**(1), 11–21 (1972)
14. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manag. **24**(5), 513–523 (1988)
15. S. Robertson and H. Zaragoza., " The Probabilistic Relevance Framework: BM25 and Beyond, " Found. Trends Inf. Retr., vol. 3, No. 4, pp. 333–389. 2009.
16. T. Mikolov, K. Chen, G. Corrado, et al., "Efficient Estimation of Word Representations in Vector Space." arXiv:1301.3781, 2013.
17. J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP), 2014, pp. 1532–1543.
18. C. Ouni, E. Benmohamed, and H. Ltifi, "Sentiment Analysis Deep Learning Model Based on a Novel Hybrid Embedding Method," Soc. Netw. Anal. Min., vol. 14, art. no. 150, 2024.
19. Bojanowski, P., Grave, E., Joulin, A., et al.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017). https://doi.org/10.1162/tacl_a_00051
20. Mrkšić, N., Vulić, I., Ó Séaghdha, D., et al.: Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints. Trans. Assoc. Comput. Linguist. **5**, 309–324 (2017). https://doi.org/10.1162/tacl_a_00063
21. Arora, K., Chakraborty, A., Cheung, J.C.K.: Learning lexical subspaces in a distributional vector space. Trans. Assoc. Comput. Linguist. **8**, 311–329 (2020). https://doi.org/10.1162/tacl_a_00316

22. H. Gong, S. Bhat, and P. Viswanath, "Enriching Word Embeddings with Temporal And Spatial Information," in *Proc. Conf. Nat. Lang. Learn. (CoNLL)*, 2020, pp. 1–11.
23. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. North Amer. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. Conf. (NAACL-HLT)*, Minneapolis, MN, USA, 2019, pp. 4171–4186, https://doi.org/10.18653/v1/N19-1423.
24. Y. Liu, M. Ott, N. Goyal, et al., "RoBERTa: A Robustly Optimized BERT Pretraining Approach," arXiv:1907.11692, 2019.
25. P. He, X. Liu, J. Gao, and W. Chen, "DeBERTa: Decoding-Enhanced BERT with Disentangled Attention," arXiv:2006.03654, 2020.
26. S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, D. Lian, and J.-Y. Nie, "C-Pack: Packed Resources For General Chinese Embeddings," in *Proc. 47th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR)*, Washington, DC, USA, Jul. 2024, pp. 641–649, https://doi.org/10.1145/3626772.3657878.
27. J. Ma, Z. Zhao, Z. Xie, Y. Zhang, and G. Zhou, "LE-DLCM: Decoupled Learner and Course Modeling with Large Language Models for Enhanced Course Recommendation," *Knowl.-Based Syst.*, vol. 334, Art. no. 115135, Feb. 2026, https://doi.org/10.1016/j.knosys.2025.115135.
28. W. Wang, Y. Wang, S. Joty, et al., "RAP-Gen: Retrieval-Augmented Patch Generation with CodeT5 for Automatic Program Repair," in *Proc. 31st ACM Joint Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng. (ESEC/FSE)*, San Francisco, CA, USA, 2023, pp. 146–158, https://doi.org/10.1145/3611643.3616256.
29. S. Yao, J. Zhao, D. Yang, et al., "ReAct: Synergizing Reasoning And Acting in Language Models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Kigali, Rwanda, 2023.
30. J. Sun, Z. Wang, X. Xu, et al., "Think-on-Graph: Deep And Responsible Reasoning of Large Language Models with Knowledge Graph," arXiv:2307.07697*, 2023*.
31. Y.-S. Chuang, W. Fang, S.-W. Li, et al., "Expand, Rerank, And Retrieve: Query Reranking for Open-Domain Question Answering," in *Proc. Findings of the Association for Computational Linguistics (ACL Findings)*, Toronto, Canada, Jul. 2023, pp. 12131–12147.
32. K. Sun, N. Jedema, K. Sharma, et al., "Efficient And Accurate Contextual Re-Ranking for Knowledge Graph Question Answering," in *Proc. Int. Conf. on Language Resources and Evaluation and the Conf. on Computational Linguistics (LREC-COLING)*, Turin, Italy, May 2024, pp. 5585–5595.
33. H.-L. Hsu and J. Tzeng, "DAT: Dynamic alpha tuning for hybrid retrieval in retrieval-augmented generation," arXiv:2503.23013, Mar. 2025.
34. B. Sarmah, R. Dandapat, M. Shamsi, et al., "HybridRAG: Integrating Knowledge Graphs And Vector Retrieval Augmented Generation for Efficient Information Extraction," in *Proc. 5th ACM Int. Conf. on AI in Finance (ICAIF)*, Brooklyn, NY, USA, Nov. 2024.
35. Y. Yuan, L. Zhang, H. Xu, et al., "A Hybrid RAG System with Comprehensive Enhancement on Complex Reasoning," arXiv:2408.05141, 2024.
36. Wan, Y., Li, J., Xu, K., et al.: Empowering LLMs by Hybrid Retrieval-Augmented Generation for Domain-Centric Q&A in Smart Manufacturing. Adv. Eng. Inform. **65**, 103212 (2025)
37. V. Agrawal, F. Wang, and R. Puri, "Query-Aware Graph Neural Networks for Enhanced Retrieval-Augmented Generation," arXiv:2508.05647, Jul. 2025.
38. L. Xiong, C. Xiong, Y. Li, et al., "Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.
39. J. Zhan, J. Mao, Y. Liu, et al., "Optimizing Dense Retrieval Model Training with Hard Negatives," in *Proc. 44th Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. (SIGIR)*, 2021, pp. 1503–1512.
40. Y. Qu, Y. Ding, J. Liu, et al., "RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering," in *Proc. NAACL-HLT*, 2021, pp. 5835–5847.
41. Luan, Y., Eisenstein, J., Toutanova, K., et al.: Sparse, dense, and attentional representations for text retrieval. Trans. Assoc. Comput. Linguist. **9**, 329–345 (2021)
42. Lin, S.-C., Li, M., Lin, J.: Aggretriever: a simple approach to aggregate textual representations for robust dense passage retrieval. Trans. Assoc. Comput. Linguist. **11**, 436–452 (2023)
43. G. Izacard, M. Caron, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave, "Unsupervised Dense Information Retrieval with Contrastive Learning," Trans. Mach. Learn. Res., Aug. 2022.
44. Y. Lei, L. Ding, Y. Cao, C. Zan, A. Yates, and D. Tao, "Unsupervised Dense Retrieval with Relevance-Aware Contrastive Pre-Training," in *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada, Jul. 2023, pp. 10932–10940, https://doi.org/10.18653/v1/2023.findings-acl.695.
45. K. Peng, L. Ding, Q. Zhong, Y. Ouyang, W. Rong, Z. Xiong, and D. Tao, "Token-Level Self-Evolution Training for Sequence-to-Sequence Learning," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics* (Volume 2: Short Papers), Toronto, Canada, Jul. 2023, pp. 841–850, https://doi.org/10.18653/v1/2023.acl-short.73.
46. Kwiatkowski, T., Palomaki, J., Redfield, O., et al.: Natural Questions: A Benchmark for Question Answering Research. Trans. Assoc. Comput. Linguist. **7**, 452–466 (2019). https://doi.org/10.1162/tacl_a_00276
47. M. Joshi, E. Choi, D. Weld, and L. Zettlemoyer, "TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension," in *Proc. 55th Annu. Meet. Assoc. Comput. Linguist. (ACL)* (Vol. 1: Long Papers), Vancouver, Canada, 2017, pp. 1601–1611, https://doi.org/10.18653/v1/P17-1147.
48. N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models," in *Proc. Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.
49. T. Gao, X. Yao, and D. Chen, "SimCSE: Simple Contrastive Learning of Sentence Embeddings," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, Nov. 2021, pp. 6894–6910, https://doi.org/10.18653/v1/2021.emnlp-main.552.
50. S. Xiao, Z. Liu, Y. Shao, and Z. Cao, "RetroMAE: Pre-Training Retrieval-oriented Language Models Via Masked Auto-Encoder," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, Dec. 2022, pp. 538–548, https://doi.org/10.18653/v1/2022.emnlp-main.35.
51. L. Gao and J. Callan, "Unsupervised Corpus Aware Language Model Pre-training for Dense Passage Retrieval," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, May 2022, pp. 2843–2853, https://doi.org/10.18653/v1/2022.acl-long.203.