GraTAG: Production AI Search via Graph-Based Query Decomposition and Triplet-Aligned Generation with Rich Multimodal Representations

Bo Tang*
AIDS and SIAR, University of Science and Technology of China; MemTensor
China

Junyi Zhu*
ESAT-PSI, KU Leuven
Belgium

Ang Li*
Zhejiang University
China

Yiquan Wu, KunKuang
Zhejiang University
China

Beihong Jin
Institute of Software, Chinese Academy of Sciences
China

Jiahao Wu
The Hong Kong Polytechnic University
China

Hao Wang, Chenyang Xi
Yuchen Feng, Wenqiang Wei
Chunyu Li, Zehao Lin
Zhiyu Li, Feiyu Xiong
MemTensor
China

Beibei Li, Kaiwen Wei
Chongqing University
China

Jingrun Chen†
jingrunchen@ustc.edu.cn
AIDS and SIAR, University of Science and Technology of China
China

Figure 1: Online evaluation (on January 2nd) of `GraTAG` system for the complex multi-faceted query. `GraTAG` features built-in citation, timeline visualization (right column), and a textual-visual choreography mechanism.

Abstract

Generative AI search engines offer a key advantage over traditional search engines through their ability to synthesize fragmented information for queries, yet they still require improvements in relevance, comprehensiveness, and presentation. To these ends, we introduce GraTAG, an AI Search framework that addresses these challenges through three core innovations. First, we introduce **graph-based query decomposition (GQD)** to dynamically break down complex or ambiguous queries into structured sub-queries with explicit dependencies, enabling more precise, stepwise retrieval. Second, we propose **triplet-aligned generation (TAG)**, which dynamically constructs relation triplets from retrieved documents to explicitly model entity relationships, factual dependencies, and logical connections, enabling the model to generate more coherent and comprehensive answers. Third, GraTAG innovates in **rich multimodal presentations** that integrates timeline visualization and textual-visual choreography to reduce cognitive load and enhance information verification. Evaluated on recent real-world queries, GraTAG outperforms eight existing systems in human expert assessments, excelling in relevance, comprehensiveness, and insightfulness. Our work publicly releases a comprehensive, industry-grade full-stack AI search engine, providing a solid reference for the community and demonstrating the effectiveness of the proposed system through rigorous evaluations and ablation studies.

*Co-first author.
†Corresponding author.

2026.

Keywords

Generative AI Search, Query Decomposition, Triplet Aligned Generation

1 Introduction

In the era of information, a significant volume of events, knowledge, and resources has been digitized and made accessible through the Internet. As users continually strive to quickly and accurately locate relevant information within this rapidly growing digital ecosystem, the development of search engines has emerged as a critical solution to meet their fundamental need for efficient and reliable information retrieval [8, 30].

The evolution of information retrieval has been significantly advanced by developments in language modeling. Autoregressive models based on Transformer architectures [42] enabled efficient parallel processing and large-scale unsupervised pretraining [23], leading to the emergence of intelligent behaviors in language models. Reinforcement learning from human feedback [14] further refined these models by aligning their outputs with human preferences. Modern large language models (LLMs) have demonstrated human-level performance in reading comprehension and reasoning [38], with their vast parameters enabling the encoding of extensive knowledge [36]. Building upon these advances, retrieval-augmented generation (RAG) has emerged as a framework that integrates information retrieval with LLMs [25]. Studies have shown that RAG can substantially reduce hallucinations while providing up-to-date information through in-context learning [22]. Meanwhile, LLMs enhance search quality through query rewriting, query expansion and decomposition [9]. Based on RAG, generative AI search engines such as Perplexity AI [4], Tiangong AI [5], and Metaso [33] have emerged to provide synthesized answers rather than merely returning links. While conversational LLM products like Gemini, DeepSeek and ChatGPT also employ RAG, *generative AI search engines distinguish themselves by emphasizing their dual role as both search engines and reading tools*, prioritizing answer quality and reading experiences. A survey conducted in the United States found that over a quarter of adults considered switching to AI search engines in 2023 [37].

Despite recent progress, as shown in Fig. 1, applying RAG to web-scale search still faces key challenges. **(1) Handling complex multi-faceted queries**: Web queries often encompass multiple sub-topics, temporal constraints. Treating such queries holistically results in excessive noise and irrelevant retrievals. **(2) Synthesizing knowledge across fragmented chunks**: Chunking in web RAG breaks semantic coherence and logical links, leading to missing information and hallucination in answer [25, 50]. **(3) Presenting rich and readable answers**: Web users need to quickly grasp information while also gaining comprehensive insights, which calls for clear and multi-dimensional presentations that facilitate efficient understanding. To demonstrate the flaw in existing AI search systems, we collected 300 queries across 8 domains and evaluated the answers of Perplexity AI with the help of human experts. Fig. 2 illustrates that several problems are common in the answers.

To address these challenges, we propose GraTAG, a comprehensive AI search engine framework. To tackle challenge (1), we introduce a **graph-based query decomposition (GQD)** approach that

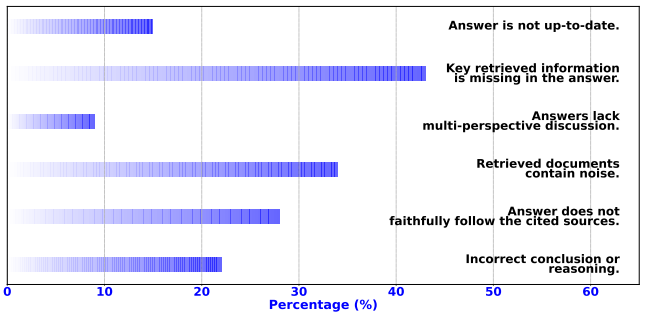

Figure 2: Common issues in generative AI search answers.

systematically breaks down ambiguous or multi-faceted queries into structured sub-queries with explicit dependencies, enabling more fine-grained retrieval with reduced noise and irrelevance. For challenge (2), we design a **triplet-aligned generation (TAG)** mechanism that dynamically constructs relation triplets from retrieved chunks to explicitly model entity relationships, factual dependencies, and logical connections, enabling the model to reason over fragmented information and generate more coherent and comprehensive answers. For challenge (3), we design a **rich multimodal presentation** that integrates timeline visualizations and textual-visual choreography in answer generation to reduce cognitive load, enhance information assimilation, and enable users to verify synthesized content.

We conduct large-scale, multi-faceted human and LLM evaluations, involving over 243,000 expert ratings across 9 criteria, to rigorously assess system performance. The results demonstrate that GraTAG not only achieves the highest overall quality but also excels in answer representation. Additionally, GraTAG is a commercialized (B2B) product and developed with a focus on stability and safety. An online test showcase is presented in Fig. 1. Our contributions can be summarized as follows:

- We propose GraTAG, together with a comprehensive public disclosure of the full-stack design for a commercialized generative-AI search engine. We detail an industrial-grade architecture covering the end-to-end retrieval-and-generation workflow and demonstrate how it addresses core challenges in existing search systems, including irrelevance, incompleteness, and monolithic presentation.
- We present a modular framework designed to address key challenges in AI search: (1) Graph-based Query Decomposition (GQD), which maps complex queries into structured sub-queries with explicit dependencies; (2) Triplet-Aligned Generation (TAG), which constructs semantic relation triplets to improve reasoning depth; and (3) Rich Multimodal Presentation, which integrates timelines and textual-visual choreography to boost verifiability and user experience.
- Comprehensive evaluations conducted on 1000 real-world queries and over 243,000 expert assessments validate the effectiveness of GraTAG. Compared to the strongest baseline, GraTAG improves comprehensiveness by 10.8%, insightfulness by 7.9%, and the overall average score by 4.8%. On the public benchmark BrowseComp, we outperform the best baseline by 17.3%. The relevant system code is available at https://github.com/tangbotony/GraTAG.

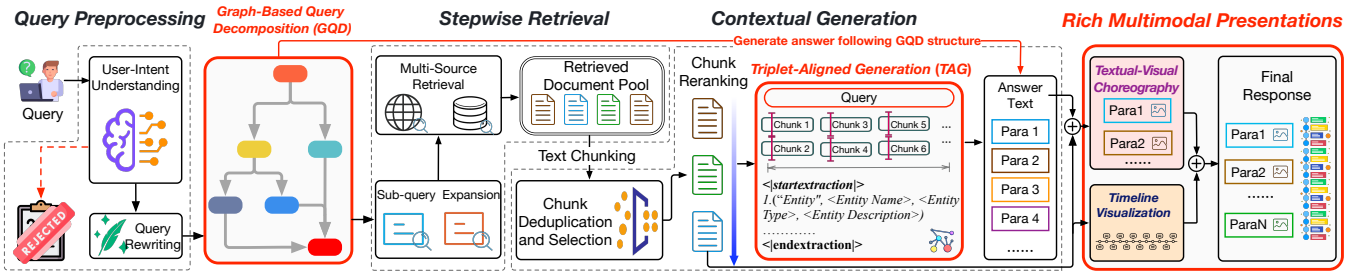

Figure 3: The workflow of GraTAG, which is an AI search system with dedicated components for query preprocessing, retrieval, and generation, featuring graph-based query decomposition, triplet-aligned generation, and rich multimodal presentations.

2 Related Work

2.1 Query Optimization for Retrieval

Effective retrieval is critical to system performance, as the quality of retrieved information significantly shapes the final output. Query rewriting techniques aim to transform user queries into more precise and retrieval-friendly formats, addressing ambiguities and enhancing alignment with indexed data [18, 28, 29, 41]. Similarly, query expansion enriches the input by generating alternative or supplementary queries, ensuring the retrieval of a broader and more contextually relevant set of documents [21]. While these techniques improve query quality, web queries remain complex, often spanning multiple sub-topics or temporal constraints. Direct retrieval thus yields noisy and incomplete documents. GraTAG mitigates this by applying graph-based query decomposition (GQD), which decomposes complex queries into parallel or sequential sub-queries to enrich retrieval while maintaining logical consistency.

2.2 Contextual Generation

After retrieving documents, the generation process synthesizes retrieved content into coherent responses. A main challenge in web-based RAG is that chunking breaks semantic coherence and logical connections, while irrelevant information distracts the model and causes hallucinations [48]. To mitigate these issues, various techniques have been proposed, including reference filtering [15], context selection [47], and reranking [52] to reduce noise and highlight salient information. Recent work has explored graph-based approaches such as GraphRAG [17], GFMRAG [27], and PathRAG [12], which construct knowledge graphs from text to model relationships among retrieved knowledge. However, these methods rely on static corpora or predefined schemas, struggling to adapt to dynamically retrieved web content. GraTAG introduces triplet-aligned generation, which dynamically constructs relation triplets from retrieved web documents to restore logical dependencies disrupted by chunking, thereby mitigating hallucinations and enhancing coherence.

2.3 Generative AI Search Engines

Recent systems such as Perplexity AI [4], Tiangong AI [5], Metaso [33], You.com [49], and Microsoft Copilot [32] integrate LLMs with web retrieval to deliver synthesized answers. However, these remain closed-source with limited architectural disclosure. We present GraTAG as a comprehensive case study, detailing our full-stack system design and targeted solutions to query complexity, knowledge fragmentation, and response structuring.

3 GraTAG AI Search

In this section, we first give an overview of our AI search system GraTAG, and then elaborate on core components: graph-based query decomposition (GQD) and triplet-aligned generation (TAG).

3.1 Overview

GraTAG is an end-to-end production-ready RAG system comprising seven key stages. As shown in Fig. 3, the pipeline begins with Query Preprocessing, which uses a fine-tuned LLM to filter unsafe content and canonicalize spatiotemporal expressions. Next, Graph-Based Query Decomposition (GQD) (Sec. 3.2) decomposes complex queries into structured sub-queries with explicit dependencies to enable hierarchical reasoning. To improve coverage, Sub-query Expansion generates semantic variations, followed by Stepwise Retrieval, which performs multi-source retrieval guided by the GQD structure. The retrieved documents are then processed with Deduplication and Reranking to remove redundancy and suppress noise. Subsequently, Triplet-Aligned Generation (TAG) (Sec. 3.3) dynamically constructs relation triplets from evidence chunks to restore logical coherence. Finally, Rich Multimodal Presentations (Sec. 3.4) delivers structured outputs with timelines and citations to enhance user experience. Detailed implementation of those stages is provided in Appendix A.

3.2 Graph-Based Query Decomposition (GQD)

Web search queries often exhibit multiple dimensions of complexity, such as multi-intent (e.g., "compare iPhone 15 Pro battery life vs. Samsung S24 and their prices in Asia"), temporal constraints (e.g., "Tesla stock performance after 2023 Q3 earnings vs. industry trends"), and multi-hop reasoning requiring cross-document inference (e.g., "how did Fed rate changes in 2023 impact tech startup valuations"). To address these challenges, we decompose complex queries into atomic sub-queries to enable more precise retrieval and information aggregation. The decomposition is represented as a directed acyclic graph (DAG) that explicitly encodes information flow among sub-queries. In contrast to linear or tree-based approaches [51], our Graph-based Query Decomposition (GQD) captures parallel and joint dependencies, enabling finer-grained and more flexible reasoning.

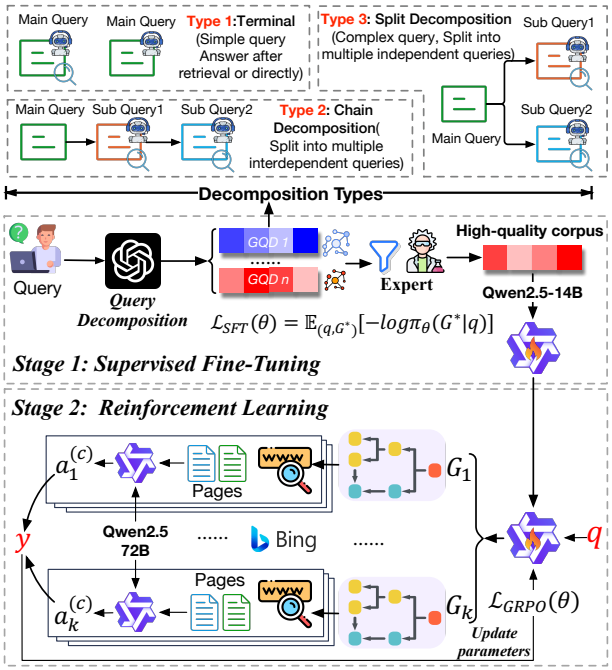

Figure 4: Graph-based query decomposition module.

Specifically, we employ an LLM $\pi_\theta(G \mid q)$ (based on Qwen2.5-72B) to conduct GQD by defining nodes and their pairwise relationships. This GQD model π_θ is post-trained via two stages: supervised fine-tuning on curated query-GQD pairs, followed by reinforcement learning alignment with RAG task performance.

3.2.1 Decomposition Types (Fig. 4 top row). Our GQD model is instructed to follow three decomposition strategies (a combination of them is allowed to fully break down an input query): 1) *Terminal:* The input query is atomic, and decomposition is unnecessary. 2) *Chain decomposition:* A query is sequentially decomposed into a series of sub-queries, where each parent node provides preliminary information for its child node and the descendant nodes, e.g., least-to-most prompting [51]. 3) *Split decomposition:* The query is divided into independent sub-queries that can be processed in parallel.

3.2.2 Stage 1: Supervised Fine-Tuning (SFT) of the GQD Model (Fig. 4 middle row). To improve the performance of GQD model, we first conduct SFT using a set of query-GQD pairs (q, G^*) collected through a systematic curation process. Multiple foundation models are instructed with detailed prompts (see Sec. D.1) to generate candidate GQDs. The generated GQDs are then programmatically validated to ensure that the parent-child relationships meet the requirements and remove duplicate GQDs. Finally, human experts examine the data and select correctly generated high-quality samples. SFT of π_θ is performed via minimizing the loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = \mathbb{E}_{(q,G^*)}\left[-\log \pi_\theta(G^* \mid q)\right]. \tag{1}$$

This stage produces a reference model π_{ref} and initializes the policy model for subsequent reinforcement learning.

3.2.3 Stage 2: Reinforcement Learning (RL) of the GQD Model (Fig. 4 bottom row). To further align GQD generation with downstream RAG performance while stabilizing the reward signal, we adopt a reinforcement learning strategy based on GRPO [16] with environment semi-determinism, variance control, and minimal shaping. For each query q, we sample K candidate GQDs $\{g_k\}_{k=1}^K$ from the current policy π_θ. We then evaluate the quality of each g_k based on the likelihood of the pipeline producing the ground-truth response y. Specifically, for each g_k, we perform C independent retrievals to obtain evidence sets $\{a_k^{(c)}\}_{c=1}^C$. Top-retrieved URLs/chunks for each sub-query are pre-cached to ensure consistency and reduce web fluctuation noise. For high-similarity queries (similarity > 0.95, measured via BGE-M3 embeddings), cached evidence is automatically reused. Variance is further controlled by averaging over multiple retrievals and clipping extreme rewards. Formally, the reward for candidate g_k is computed as:

$$r_k = \frac{1}{C}\sum_{c=1}^{C} \log \mathcal{M}\big(y \mid a_k^{(c)}\big), \tag{2}$$

where $\mathcal{M}(\cdot \mid \cdot)$ denotes the response-generation model (Qwen2.5-72B). Next, we compute group-normalized rewards and use them as advantages:

$$\tilde{r}_k = \frac{r_k - \mu_r}{\sigma_r}, \mu_r = \frac{1}{K}\sum_{j=1}^{K} r_j, \sigma_r = \sqrt{\frac{1}{K}\sum_{j=1}^{K}(r_j - \mu_r)^2}, \hat{A}_{k,t} = \tilde{r}_k. \tag{3}$$

Finally, we optimize the GDQ model π_θ via the GRPO objective:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{\substack{(q,y)\sim\mathcal{D},\\ \{g_k\}\sim\pi_{\theta_{\text{old}}}(\cdot|q)}}\Bigg[\frac{1}{K}\sum_{k=1}^{K}\frac{1}{|g_k|}\sum_{t=1}^{|g_k|}\min\Big(\rho_{k,t}(\theta)\,\hat{A}_{k,t},\ \text{clip}\big(\rho_{k,t}(\theta),\,1-\varepsilon,\,1+\varepsilon\big)\,\hat{A}_{k,t}\Big) - \beta\,\mathbb{D}_{\text{KL}}\big(\pi_\theta \| \pi_{\text{ref}}\big)\Bigg], \tag{4}$$

where

$$\rho_{k,t}(\theta) = \frac{\pi_\theta(g_{k,t} \mid q, g_{k,<t})}{\pi_{\theta_{\text{old}}}(g_{k,t} \mid q, g_{k,<t})}. \tag{5}$$

3.3 Triplet-Aligned Generation (TAG)

Since retrieved documents are split into chunks for information selection and ranking, supporting evidence may span multiple chunks. This fragmentation can break long-range dependencies, leading to information gaps and logical discontinuities that amplify hallucinations. To address this, we propose Triplet-Aligned Generation (TAG), which extracts triplets from the retrieved documents and aligns them with the answer generation process to explicitly bridge missing logic and relations across chunks. By injecting these structural cues, TAG enhances cross-chunk reasoning coherence and mitigates hallucination.

3.3.1 Triplet Extraction Cold Start (Fig. 5 top row). In this paper, triplets encapsulate entity details, inter-entity relations, and implicit factual and logical dependencies, enabling structured cross-chunk reasoning rather than isolated snippet reading. To train triplet extraction stably, we adopt a cold-start stage: For each sub-query, a strong teacher model (e.g., GPT-4o) extracts p parallel triplets $t = \{t^1, \ldots, t^p\}$ for each (sub-query, chunks) pair and then we verify their quality manually. We then SFT the target LLM using sub-queries and their corresponding chunks as input, training the model to produce concise and effective triplets. Skipping this stage typically yields lengthy, noisy triplets that fail to guide response

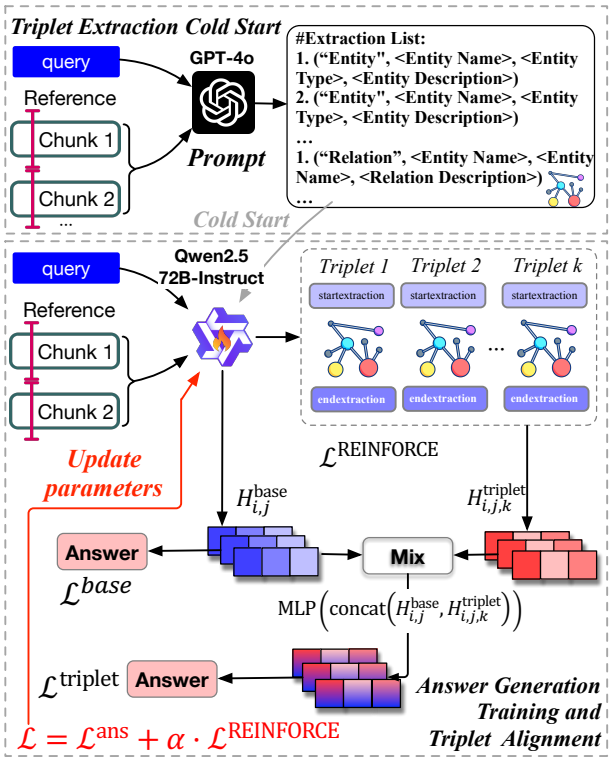

Figure 5: Triplet-aligned generation module.

generation and harm training stability. During fine-tuning, dedicated extraction tokens ⟨!`startextraction`!⟩ mark spans, helping the model internalize graph-like structure.

3.3.2 Answer Generation Training and Triplet Alignment (Fig. 5 bottom row). First, we enhance the model's ability to generate answers both with and without triplet augmentation. Specifically, given an instruction-answer pair (x_i, y_i) and its k-th triplet t_i^k. We first obtain the hidden state $H^{\text{base}}_{i,j} = \text{hs}[x_i; y_i^{(j-1)}]$ of the j-th token in the answer under the original RAG method. In our proposed method, when incorporates the triplet t_i^k, the corresponding hidden state is $H^{\text{triplet}}_{i,j,k} = \text{hs}[x_i; y_i^{(j-1)}; t_i^k]$. We concatenate the two obtained hidden states and input them into a three-layer MLP with ReLU activation to compute the weight for each token in the answer:

$$\omega_{i,j,k} = \text{MLP}(\text{concat}(H^{\text{base}}_{i,j}, H^{\text{triplet}}_{i,j,k})). \tag{6}$$

Meanwhile, we compute the log-likelihoods under the conditions of without and with the triplet:

$$\mathcal{L}^{\text{base}} = \log p_\theta(y_i^j \mid x_i, y_i^{(j-1)}) = f(H^{\text{base}}_{i,j}), \tag{7}$$

$$\mathcal{L}^{\text{triplet}} = \log p_\theta(y_i^j \mid x_i, y_i^{(j-1)}, t_i^k) = f(H^{\text{triplet}}_{i,j,k}), \tag{8}$$

where $f(\cdot) = \text{softmax}(\text{lmhead}(\cdot))$. Finally, we compute the weighted sum of the j-th token to obtain the log-likelihood incorporating the triplet for training answer generation:

$$\mathcal{L}^{\text{ans}} = \omega_{i,j,k} \cdot \mathcal{L}^{\text{base}} + (1 - \omega_{i,j,k}) \cdot \mathcal{L}^{\text{triplet}}. \tag{9}$$

Given that the extracted triplets vary in length and content, we aim to determine the most beneficial triplet for optimizing the answer. Specifically, for i-th instruction-answer pair (x_i, y_i), our objective is to find an optimal triplet $t^{(i)}$ that maximizes $\pi_\theta(y_i \mid x_i, t^{(i)})$. To achieve this, we design a reward function R based on the REINFORCE algorithm to quantify the impact of introducing k-th triplet on answer generation as follows:

$$R_{i,k} = \max(0, \mathcal{L}_i^{\text{base}} - \mathcal{L}_{i,k}^{\text{triplet}} - \bar{R}_i). \tag{10}$$

This reward function increases the likelihood of selecting triplets that perform better than the average reward $\bar{R}_i$. Finally, the REINFORCE loss term is defined as:

$$\mathcal{L}^{\text{REINFORCE}} = -R_{i,k} \cdot \log \pi_\theta(g_k^i \mid x_i). \tag{11}$$

To further encourage succinct triplets, we introduce a length-aware bonus. Let $\ell_{i,k}$ be the tokenized length of t_k, and define the candidate-wise mean $\bar{\ell}_i = 1/K_i \sum_{k=1}^{K_i} \ell_{i,k}$, for the pair (x_i, y_i). We then set $r^{\text{len}}_{i,k} = \gamma\left(\bar{\ell}_i - \ell_{i,k}\right)$, which increases the reward for shorter-than-average triplets and decreases it otherwise ($\gamma > 0$ controls the strength). The augmented reward and the resulting REINFORCE loss are $\tilde{R}_{i,k} = R_{i,k} + r^{\text{len}}_{i,k}$. Finally, the REINFORCE loss is:

$$\mathcal{L}^{\text{REINFORCE}} = -\tilde{R}_{i,k} \cdot \log \pi_\theta(t_k \mid x_i), \tag{12}$$

which encourages the model to select triplets that improve answer generation, while ignoring unhelpful ones. This formulation directly aligns with our GQD-style preference modeling, where log-probability differences capture the relative advantage of candidate knowledge structures. Thus, the total loss function for model training is defined as:

$$\mathcal{L} = \mathcal{L}^{\text{ans}} + \alpha \cdot \mathcal{L}^{\text{REINFORCE}}, \tag{13}$$

where α is a hyperparameter controlling the trade-off between different training objectives.

3.3.3 Answer Generation Inference. For answer generation, the trained model processes sub-queries either sequentially or in parallel based on the dependency structure in the GQD. The generator takes as input both the extracted triplets and the reranked chunks. For nodes with ancestors, the cumulative Q&A pairs of the ancestor chain are prepended to provide contextual information, ensuring that the generation process respects the hierarchical reasoning flow defined by the GQD. When all leaf nodes are processed, their Q&A pairs are concatenated with the main query to form the final comprehensive answer. This triplet-aligned generation approach significantly improves answer coherence and reasoning quality.

3.4 Rich Multimodal Presentations

Traditional chatbots often rely on linear text stacking, which can impose a high cognitive load on users. Given that AI-powered search engines facilitate extensive knowledge transmission, integrating cognitive scaffolding is essential to support user comprehension. Cognitive science research has shown that structured information and multimodal presentations enhance the efficiency of information assimilation [31, 35, 39]. To address these challenges, we incorporate timeline visualizations and textual-visual choreography to optimize the reading experience.

3.4.1 Timeline Visualization (Fig. 6 top row). In online search scenarios, integrating timeline visualizations enables users to better understand the evolution and context of events. We propose a novel timeline visualization scheme. First, we collect all retrieved chunks

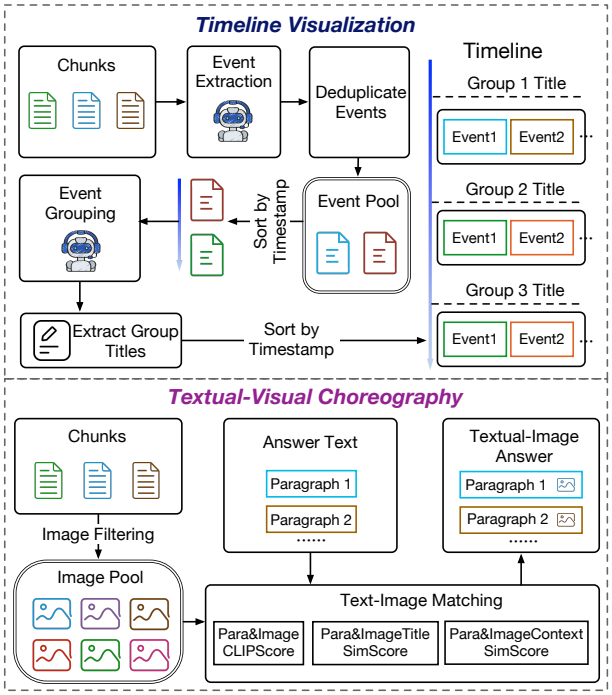

Figure 6: Rich multimodal presentations module.

following the chunk selection (see Fig. 3). Then, we instruct an LLM (Qwen2.5-14B-Instruct) to extract any event time mentioned in each chunk and to generate a corresponding title and summary. If a chunk does not explicitly mention a time, we resort to using the document's report time as extracted by the same LLM. Chunks lacking temporal information in both the chunk and the document are discarded. Because the retained chunks may describe the same content, we then employ `bge-large` to compute text embedding of the concatenated title and summary and calculate pairwise cosine similarities across chunks. Chunks with a similarity score exceeding 0.9 are merged by discarding the one with the later timestamp, resulting in a list of distinct events with timestamps. To make timeline visualization more structured, we further instruct the LLM to group these events and derive relevant keywords based on their summaries. Finally, we present the event titles for each group, sorted according to their timestamps. Our timeline visualization scheme is integrated within the framework, where upstream pipeline components (GQD, retrieval, and chunk ranking) enhance the relevance and precision of the displayed events. This integration distinguishes our solution from other methods [20, 44], as GQD decomposes complex queries into simpler sub-queries, enabling targeted retrieval, while chunk ranking removes noise, together yielding more precise and context-aware results.

3.4.2 Textual-Visual Choreography (Fig. 6 bottom row). A picture is worth a thousand words. `GraTAG` integrates relevant images into textual answers to enhance information assimilation. These images are extracted from retrieved documents. To ensure quality and relevance, we first filter out noisy images, retaining only those

Table 1: Dataset statistics.

Category	Sources	Number
Event Timeline	TopHub, Google Trends	350
Multi-Constraint Factual	Zhihu, Wikipedia	300
Decision-Making	Stack Exchange, Hacker News	200
Technical Troubleshooting	StackOverflow, GitHub	150

Table 2: Pearson correlation coefficients between human and LLM scores under different evaluation criteria.

Metric	Value	Metric	Value
Conciseness	0.804	Numerical Precision	0.713
Relevance	0.822	Factuality	0.729
Timeliness	0.810	Comprehensiveness	0.642
Clarity	0.649	Coherence	0.720
Insightfulness	0.659	**Average**	**0.890**

of high quality. Specifically, a rule-based filtering algorithm eliminates logos, icons, and low-resolution images. Subsequently, we compute the similarity between the textual description associated with the image and the main query using bge-reranker-v2-m3 and remove those smaller than 0.3. To determine the optimal placement of images, we compute the pairwise similarity between generated answer paragraphs and candidate images. This computation involves a weighted average of three measures: (1) the embedding cosine similarity between the generated text paragraph and the image, computed using the clip-vit-huge-patch14; (2) the estimated similarity between the synthesized paragraph and the retrieved document's title, obtained via bge-reranker-v2-m3; and (3) the embedding cosine similarity between the synthesized paragraph and the retrieved document's text using bge-large. Pairwise similarities are assembled into a matrix. Then we determine the optimal image-to-text alignment using the Hungarian algorithm [24].

4 Experiments

4.1 Experimental Setup

Multi-faceted Evaluation Criteria. Due to the open-ended nature of answers in real scenarios, we adopted rating criteria for evaluating generated answers rather than computing their match to a fixed reference answer. We invited experts with master's degrees to develop a multi-faceted rating criteria, including: (1) Conciseness, (2) Numerical Precision, (3) Relevance, (4) Factuality, (5) Timeliness, (6) Comprehensiveness, (7) Clarity, (8) Coherence, and (9) Insightfulness. More detailed definitions for each dimension are provided in Sec. C. Our nine evaluation metrics were chosen based on two principles: (1) each metric is concise, easily interpretable, and independent; and (2) collectively, they capture the majority of observed issues.

Test Set. We construct a test set of 1,000 real-world complex queries, which is organized into four categories: (1) Event Timeline, which requires reconstructing evolving events or comparing entities across time. (2) Multi-Constraint Factual, which involves multiple explicit constraints such as time, location, scope, or causality. (3) Decision-Making, which specifies a target objective together with

Table 3: Multi-faceted comparison of different systems via human expert evaluation (Higher is better, 10-point scale).

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
Perplexity AI [4]	**9.851**	9.630	9.436	8.524	8.553	7.284	9.612	9.853	6.543	8.810
Tiangong AI [5]	9.840	**9.722**	7.812	8.924	8.103	8.020	9.604	9.802	6.535	8.707
Ernie Bot [2]	9.770	9.320	8.883	8.028	8.406	7.798	9.524	**9.900**	5.963	8.621
KIMI [3]	9.840	9.515	8.529	8.224	8.966	8.155	9.223	9.709	6.796	8.773
Metaso [33]	9.760	8.941	8.515	7.408	8.403	5.689	9.383	9.689	4.759	8.061
ChatGLM [11]	9.810	9.420	8.949	**9.124**	8.346	6.168	9.533	9.726	5.047	8.458
Baichuan [1]	9.660	9.596	6.486	7.612	8.220	8.252	9.223	9.612	6.117	8.309
Tongyi [6]	9.803	9.009	7.586	7.212	8.194	7.677	9.293	9.899	5.859	8.281
GraTAG (Ours)	9.813	9.714	**9.533**	8.932	**9.205**	**9.143**	**9.633**	9.810	**7.333**	**9.235**

Table 4: Performance comparison on the BrowseComp.

Model	Correct Number	Acc (%)
Perplexity AI	298	23.54
Tiangong AI	29	2.29
Ernie Bot	255	20.14
KIMI	330	26.07
Metaso	77	6.08
ChatGLM	152	12.01
Baichuan	80	6.32
Tongyi	26	2.05
GraTAG (Ours)	**387**	**30.57**

Table 5: Comparison of timeline visualization.

Approach	Event Count	Precision	Wall Time (s) ↓
CHRONOS [44]	5.12	79%	67.44
GraTAG (Ours)	**10.14**	**84**%	**33.27**

Table 6: Comparison of textual-visual choreography.

Approach	Inclusion (%) ↑	Precision (%) ↑
Metaso [33]	3.0	72.2
GraTAG (Ours)	**80.0**	**90.0**

feasibility or preference constraints. (4) Technical Troubleshooting, which combines procedural intents with concrete environment constraints (e.g., version or platform). The dataset statistics are summarized in Tab. 1. In addition, we evaluate our method on the public benchmark to examine generalization: BrowseComp [43], an English benchmark where answering requires sustained, multi-step retrieval and yields short, verifiable outputs. We required three ratings for each combination of system, query, and rating criterion. Just Tab. 3 alone contains over 243,000 human ratings, underscoring the sheer scale of our evaluation.

LLM Evaluation. Evaluating the generated answers for all experiments using multi-faceted criteria by human experts is prohibitively expensive. Therefore, we considered employing an LLM to evaluate the generated text for a subset of experiments. To validate the effectiveness of the LLM as an evaluator, we also conducted a comparison experiment involving both human and LLM (GPT-4o-2024-08-06) evaluators and calculated the Pearson correlation between their final scores. As shown in Tab. 2, human and LLM scores exhibit a high correlation coefficient, indicating that it is feasible to use LLM as an evaluator. Hence, experiments reported in Tabs. 7, 11 and 12, were evaluated with GPT-4o.

Baselines. We compare GraTAG with eight existing systems, including: **Generative AI search engines**: Perplexity AI [4], Metaso [33], Tiangong AI [5], ChatGLM [11], and Tongyi [6]; and **Conversational LLMs with RAG**: KIMI [3], Ernie Bot [2], and Baichuan [1]. For Perplexity AI, we selected GPT-4o [34] as the backend model. We further conduct ablations on the two core components, GQD and TAG. Additional extended experiments, such as latency analysis and hyperparameter exploration, are provided in the appendices.

4.2 Main Results

4.2.1 Overall Performance. We first compare GraTAG with existing systems under multi-faceted human evaluation in a model-agnostic setting. As shown in Tab. 3, GraTAG achieves the highest average score (9.235 vs. 8.810), outperforming strong baselines particularly in comprehensiveness (9.143 vs. 8.252, $p < 0.001$) and insightfulness (7.333 vs. 6.796, $p < 0.001$). Each response was evaluated by at least three experts. Pairwise Pearson correlations yield an average agreement of 0.890, indicating high inter-rater consistency. We also conduct the GPT-4 evaluation, please refer to Table 10. We further evaluate on the BrowseComp benchmark, which contains 1,266 test samples. As shown in Tab. 4, GraTAG achieves the best accuracy (30.57%), outperforming the strongest baseline by 17.3% and demonstrating strong generalization under standardized evaluation.

4.2.2 Performance on Answer Representation. We evaluate the effectiveness of our method in improving user experience and information accessibility. For **Timeline**, we compare our method with CHRONOS [44] using event count (the number of event presented in the timeline), precision (the percentage of relevant events), and wall time (latency for generating the timeline) for online deployment. As shown in Tab. 5, GraTAG outperforms CHRONOS across all dimensions. For **Textual-Visual**, Metaso [33] also implements textual-visual choreography. We compare it with GraTAG by: inclusion (the rate at which images are incorporated into the generated answers) and precision (the percentage of included images that are contextually relevant). As shown in Tab. 6, GraTAG outperforms Metaso significantly on both metrics.

4.3 Ablation Study

4.3.1 System Components. We conduct a comprehensive ablation study to evaluate the contribution of each component in our pipeline. As shown in Tab. 7, removing any module leads to a measurable drop in overall performance, with the full system achieving the highest average score. Notably, omitting GQD results in the most

Table 7: Ablation study of sub-modules in our approach, "−" indicates skipping the sub-module.

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
Full Approach	**9.797**	**9.747**	**9.559**	**9.222**	9.185	**9.129**	**9.599**	**9.784**	**7.790**	**9.312**
– Query Preprocessing	9.704	9.287	9.352	9.105	9.051	8.810	9.477	9.493	6.890	9.019
– Query Expansion	9.661	9.169	9.491	9.085	**9.200**	8.372	9.363	9.670	6.722	8.970
– GQD	9.615	9.129	7.54	8.695	8.956	7.445	9.032	9.607	6.183	8.467
– Chunk Selection	9.730	9.334	9.392	9.083	9.100	8.582	9.546	9.647	6.924	9.038
– Chunk Rerank	9.711	9.483	9.462	9.031	8.989	8.773	9.465	9.656	6.849	9.047
– TAG	9.564	9.163	9.283	8.397	8.669	8.955	9.103	9.391	7.119	8.849

Table 8: Ablation study on Graph-based Query Decomposition (GQD).

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
– GQD	9.615	9.129	7.540	8.695	8.956	7.445	9.032	9.607	6.183	8.467
Prompt+GPT-4o	**9.798**	9.561	9.419	9.096	9.165	8.825	9.477	**9.807**	7.688	9.203
SFT+Qwen-2.5-14B	9.734	9.088	9.025	8.826	8.959	8.604	9.347	9.636	7.438	8.962
GQD w/ RL (GRPO)	9.797	**9.747**	**9.559**	**9.222**	**9.185**	**9.129**	**9.599**	9.784	**7.790**	**9.312**
GQD w/ RL (Reinforce++)	9.784	9.577	9.436	9.101	8.956	8.987	9.597	9.777	7.415	9.180

Table 9: Ablation study on Triplet Aligned Generation (TAG).

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
Oracle TAG	9.774	9.786	9.554	9.415	9.175	9.053	9.731	9.811	7.864	9.351
– TAG	9.535	9.168	9.302	8.386	8.664	8.998	9.096	9.373	7.091	8.846
Prompt+GPT-4o	9.762	9.682	**9.652**	9.148	8.994	8.638	9.496	**9.869**	7.371	9.179
SFT+Qwen-2.5-14B	9.593	9.292	9.367	8.702	8.832	8.455	9.355	9.488	7.127	8.912
entity-only TAG	9.545	9.175	9.293	8.413	8.664	8.862	9.200	9.363	7.082	8.844
Graph-RAG	9.442	9.207	9.383	8.789	9.053	8.954	9.226	9.510	7.588	9.017
TAG w/ RL (GRPO)	**9.797**	**9.747**	9.559	**9.222**	**9.185**	**9.129**	**9.599**	9.784	**7.790**	**9.312**

significant degradation, particularly in Relevance and Comprehensiveness, underscoring its critical role in generating focused and thorough answers. Removing TAG also causes a notable decline, especially in Coherence, highlighting its importance for coherent answers. Overall, the results demonstrate that the integration of all modules is essential for achieving balanced, high-quality responses across all evaluation criteria.

4.3.2 Graph-based Query Decomposition (GQD). We conduct detailed ablation experiments to evaluate different implementations of GQD. As shown in Tab. 8, the choice of implementation significantly impacts the final answer quality. Using prompt engineering with GPT-4o achieves strong average performance (9.203), demonstrating the effectiveness of the GQD approach itself. Fine-tuning a smaller model (Qwen-2.5-14B-Instruct) with SFT achieves competitive results (8.962), showing that the capability can be distilled into more efficient models. Incorporating reinforcement learning further improves performance, with GRPO achieving the best overall score (9.312) and Reinforce++ [19] also showing strong results (9.180). Notably, GQD with RL shows consistent improvements across all nine evaluation metrics, with particularly substantial gains in Relevance and Comprehensiveness.

4.3.3 Triplet-Aligned Generation (TAG). We evaluate the effectiveness of integrating triplet information into the generation process. As shown in Tab. 9, we compare several TAG variants: entity-only TAG that extracts entities without constructing triplets, Oracle TAG with manually verified relation–entity consistency, prompt engineering with GPT-4o, fine-tuning with Qwen-2.5-14B-Instruct, and the Graph-RAG baseline. Our lightweight TAG design with RL nears Oracle TAG while surpassing both entity-only TAG and Graph-RAG, demonstrating the effectiveness of automatic triplet construction. TAG consistently improves answer quality across implementations, with improvements particularly pronounced in Factuality and Coherence, showing that structured knowledge from triplets enhances both the accuracy and organization of the generated content while maintaining temporal awareness.

4.3.4 Test-Time Latency Analysis. In `GraTAG`, we maximize parallelism to reduce test-time latency, e.g., executing lateral sub-queries in GQD concurrently. We further fine-tune and quantize candidate models, selecting the fastest one that satisfies our performance requirements. In deployment, `GraTAG` achieves a response latency of 14.2 seconds. For comparison, the latencies of Perplexity, Tiangong, ErnieBot, KIMI, Metaso, ChatGLM, Baichuan, and Tongyi are 13.9, 4.0, 6.0, 2.8, 10.4, 7.9, 6.1, and 10.4 seconds, respectively. The latency of `GraTAG` is measured on a cluster of 16 Muxi MXC500 GPUs (each providing approximately 70% of the computing power of an NVIDIA A800), while baselines are evaluated via their publicly available interfaces.

5 Conclusion

In this work, we present `GraTAG`, a generative AI search engine designed to tackle multi-faceted challenges in answer generation and user experience through a fully integrated pipeline. Our approaches not only build on state-of-the-art research but also introduce novel solutions to specific challenges. Extensive experiments demonstrate the superiority of `GraTAG` over existing technologies. The system has been successfully deployed in real-world applications, with finalized industry contracts now exceeding USD 2 million and expansion to English-language scenarios (e.g., Suanova), validating its practical value and scalability.

References

[1] Baichuan AI. 2024. Baichuan. https://ying.baichuan-ai.com/ Accessed: 2025-02-05.

[2] Baidu AI. 2024. Yiyan. https://yiyan.baidu.com/ Accessed: 2025-02-05.

[3] Moonshot AI. 2024. KIMI. https://kimi.moonshot.cn/ Accessed: 2025-02-05.

[4] Perplexity AI. 2024. Perplexity AI. https://www.perplexity.ai/ Accessed: 2025-02-05.

[5] Tiangong AI. 2024. Tiangong AI. http://tiangong.cn/ Accessed: 2025-02-05.

[6] Tongyi AI. 2024. Tongyi. https://tongyi.ai/ Accessed: 2025-02-05.

[7] Alec Berntson. 2023. Azure AI Search: Outperforming Vector Search with Hybrid Retrieval and Reranking. https://techcommunity.microsoft.com/blog/azure-ai-services-blog/azure-ai-search-outperforming-vector-search-with-hybrid-retrieval-and-reranking/3929167 Accessed: 2025-02-01.

[8] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks* 30 (1998), 107–117.

[9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[10] Harrison Chase. 2022. *LangChain.* https://github.com/langchain-ai/langchain

[11] ChatGLM. 2024. ChatGLM. https://chatglm.cn/ Accessed: 2025-02-05.

[12] Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2025. PathRAG: Pruning Graph-based Retrieval Augmented Generation with Relational Paths. *arXiv preprint arXiv:2502.14902* (2025).

[13] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216 [cs.CL]

[14] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30 (2017).

[15] Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. 2023. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *CoRR* (2023).

[16] DeepSeek-AI. 2024. DeepSeek-V3 Technical Report. arXiv:2412.19437 [cs.CL] https://arxiv.org/abs/2412.19437

[17] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024).

[18] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise Zero-Shot Dense Retrieval without Relevance Labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 1762–1777.

[19] Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. 2025. Reinforce++: An efficient rlhf algorithm with robustness to both prompt and reward models. *arXiv preprint arXiv:2501.03262* (2025).

[20] Qisheng Hu, Geonsik Moon, and Hwee Tou Ng. 2024. From Moments to Milestones: Incremental Timeline Summarization Leveraging Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 7232–7246. doi:10.18653/v1/2024.acl-long.390

[21] Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. *arXiv preprint arXiv:2305.03653* (2023).

[22] Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active Retrieval Augmented Generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 7969–7992. doi:10.18653/v1/2023.emnlp-main.495

[23] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361* (2020).

[24] H. W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2, 1-2 (1955), 83–97. doi:10.1002/nav.3800020109 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.3800020109

[25] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[26] Chaofan Li, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023. Making Large Language Models A Better Foundation For Dense Retrieval. arXiv:2312.15503 [cs.CL]

[27] Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. 2025. GFM-RAG: Graph Foundation Model for Retrieval Augmented Generation. *arXiv preprint arXiv:2502.01113* (2025).

[28] Xinbei Ma, Yeyun Gong, Pengcheng He, hai zhao, and Nan Duan. 2023. Query Rewriting in Retrieval-Augmented Large Language Models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*. https://openreview.net/forum?id=gXq1cwkUZc

[29] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query Rewriting in Retrieval-Augmented Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 5303–5315.

[30] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval.* Cambridge University Press.

[31] Richard E. Mayer. 2014. Cognitive Theory of Multimedia Learning. In *The Cambridge Handbook of Multimedia Learning*, Richard E. Mayer (Ed.). Cambridge University Press, Cambridge, 43–71.

[32] Yusuf Mehdi. 2023. Reinventing Search with a New AI-powered Microsoft Bing and Edge, Your Copilot for the Web. Microsoft Official Blog. https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web/

[33] Metaso. 2024. Metaso. https://metaso.cn/ Accessed: 2025-02-05.

[34] OpenAI. 2024. ChatGPT. https://chatgpt.com/ Accessed: 2025-02-05.

[35] Luis P. Prieto, Kshitij Sharma, Łukasz Kidzinski, María Jesús Rodríguez-Triana, and Pierre Dillenbourg. 2018. Multimodal Teaching Analytics: Automated Extraction of Orchestration Graphs from Wearable Sensor Data. *Journal of Computer Assisted Learning* 34, 2 (April 2018), 193–203. doi:10.1111/jcal.12232

[36] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[37] Statista. 2023. https://www.statista.com/statistics/1377993/us-adults-ai-powered-search-engines-usage-choice/ Accessed: 2025-01-21.

[38] Winnie Street, John Oliver Siy, Geoff Keeling, Adrien Baranes, Benjamin Barnett, Michael McKibben, Tatenda Kanyere, Alison Lentz, Robin IM Dunbar, et al. 2024. LLMs achieve adult human performance on higher-order theory of mind tasks. *arXiv preprint arXiv:2405.18870* (2024).

[39] John Sweller, Paul Ayres, and Slava Kalyuga. 2020. Cognitive load theory and educational technology. *Educational Technology Research and Development* 68, 1 (2020), 1–16. doi:10.1007/s11423-019-09701-3

[40] Robert Endre Tarjan and Anthony E Trojanowski. 1977. Finding a maximum independent set. *SIAM J. Comput.* 6, 3 (1977), 537–546.

[41] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*. 10014–10037.

[42] A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* (2017).

[43] Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. BrowseComp: A Simple Yet Challenging Benchmark for Browsing Agents. arXiv:2504.12516 [cs.CL] https://arxiv.org/abs/2504.12516

[44] Weiqi Wu, Shen Huang, Yong Jiang, Pengjun Xie, Fei Huang, and Hai Zhao. 2025. Unfolding the Headline: Iterative Self-Questioning for News Retrieval and Timeline Summarization. *arXiv preprint arXiv:2501.00888* (2025).

[45] Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-Pack: Packaged Resources To Advance General Chinese Embedding. arXiv:2309.07597 [cs.CL]

[46] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115* (2024).

[47] Haoyan Yang, Zhitao Li, Yong Zhang, Jianzong Wang, Ning Cheng, Ming Li, and Jing Xiao. 2023. PRCA: Fitting Black-Box Large Language Models for Retrieval Question Answering via Pluggable Reward-Driven Contextual Adapter. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 5364–5375. doi:10.18653/v1/2023.emnlp-main.326

[48] Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. In *The Twelfth International Conference on Learning Representations*.

[49] You.com. 2024. You.com: The AI Search Engine You Control. https://you.com/. Accessed: 2024-10-04.

[50] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023. Siren's Song in the AI Ocean: A Survey on Hallucination in Large Language Models. *arXiv preprint arXiv:2309.01219* (2023).

[51] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. In *The Eleventh International Conference on Learning Representations*. https:

//openreview.net/forum?id=WZH7099tgfM

[52] Shengyao Zhuang, Bing Liu, Bevan Koopman, and Guido Zuccon. 2023. Open-source Large Language Models are Strong Zero-shot Query Likelihood Models for Document Ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 8807–8817.

A Other Implementations in the Workflow

A.1 Query Preprocessing

To ensure robustness and practical deployment in real-world scenarios, `GraTAG` incorporates a comprehensive query preprocessing pipeline that handles edge cases and user intent clarification before GQD generation. Upon receiving a query, a fine-tuned LLM (currently based on Qwen-2.5-14B) is employed to filter out unsafe or harmful queries and clarifies ambiguous ones by prompting the user with options, such as suggesting a specific region for a general query like "the current state of the economy." If the query is ambiguous or lacks specificity, the LLM prompts the user with clarifying questions or options to refine his/her intent. The query might be rejected if it is regarded as one that warrants refusal.

Following this, any open-source LLM (currently Qwen-2.5-14B) is used to rewrite the query by translating relative spatiotemporal terms (e.g., “last week”) into precise timestamp ranges, resolving vague locations (“nearby”) to specific places, and canonicalizing incomplete entity names.

A.2 Sub-query Expansion

Given a sub-query in the graph after GQD, we ask an LLM to generate multiple relevant queries. These queries are called retrieval queries since they are subsequently entered into a search engine to retrieve more information. Specifically, the LLM is instructed to act as a subject matter expert in a university, expanding the given query to create related questions that assess students’ comprehensive understanding of the topic across multiple dimensions: 1) content mastery, 2) understanding of key elements, 3) contextual analysis, and 4) extended thinking.

A.3 Retrieval

To ensure the recency and relevance of retrieved information, we submit each retrieval query to multiple search engines simultaneously, leveraging the ranking algorithms in search engines to obtain more comprehensive content and mitigate biases. Additionally, we incorporate `Milvus` and `Elasticsearch` for our B2B clients to build local retrieval systems.

Directly feeding raw retrieved documents (e.g., a web page) into an LLM can lead to high perplexity and degraded response quality. To make the input content LLM-friendly and safe, we implement a robust content filtering pipeline. This process involves removing disruptive elements, filtering sensitive or extraneous information, and standardizing formatting.

After filtering, we segment documents into chunks using the `RecursiveCharacterTextSplitter` method [10]. We adopt a small chunk size (i.e., 350) with a relatively large overlap (i.e., 25%) to optimize the performance of the subsequent text embedding model, following the research findings by Azure AI Search [7].

A.4 Chunk Deduplication and Reranking

Notably, retrieved passages vary in relevance and often contain duplicate information, which can distract the model. To mitigate this problem, we use a fine-tuned text embedding model (bge-large [45]) to compute embeddings for the chunks and then calculate pairwise cosine similarities. We aim to identify the largest subset of chunks where the similarity score between any two chunks is no greater than 0.8. Finding the optimal solution to this problem corresponds to solving the maximum independent set problem [40], which is NP-hard. To improve computational efficiency, we adopt a greedy strategy that processes each chunk sequentially, retaining it only if its similarity to all previously retained chunks remains below 0.8. For re-ranking, we finetune a reranker model (bge-reranker-v2-m3 [13, 26] to sort the chunks based on their similarity to the sub-query.

B Further Analysis

B.1 Impact of Fine-Tuning on Answer Generation

In `GraTAG`, we fine-tune LLMs for entity extraction, GQD generation, and answer generation. We validate their effectiveness through ablation studies. As shown in Tab. 11, replacing these modules with GPT-4o and Qwen-2.5-72B confirms that improvements in query rewriting, entity extraction, and GQD generation positively impact final answer quality. Tab. 12 demonstrates that fine-tuning the answer generation model yields consistent performance gains across eight evaluation metrics, including Conciseness and Numerical Precision.

B.2 Effect of Reranking Model Selection

We conduct an ablation study on the reranking component by comparing our fine-tuned reranker model against its base model (bge-reranker-v2-m3) and two LLMs instructed to generate a ranking order based on relevance. As shown in Tab. 13, our fine-tuned method preserves the efficiency of the base model while significantly outperforming it and achieving performance comparable to GPT-4o. The wall time for GPT-4o reflects API response time, whereas the wall time for other models is measured on a local cluster. This demonstrates that fine-tuning allows us to achieve strong performance with more efficient models, maintaining low latency while approaching the quality of much larger language models.

B.3 Impact of Number of Sampled GQDs

We further investigate the optimal number of sampled GQDs. As shown in Tab. 14, performance improves as the number of GQDs increases from 1 to 8. Beyond this point, the performance remains relatively stable with diminishing returns, suggesting that 8 GQDs provide an effective balance between query coverage and computational efficiency.

B.4 Threshold Analysis for Text-Image Choreography

We analyze the matching threshold for Text-Image Choreography using bge-reranker-v2-m3. As shown in Table 15, the threshold controls the trade-off between inclusion (recall) and precision: lower

Table 10: Multi-faceted comparison of different systems via GPT-4 automatic evaluation. Higher value indicates better performance, 10 is the maximum.

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
Perplexity AI	**9.844**	9.528	**9.670**	9.659	8.057	8.194	9.811	9.800	6.547	9.012
Tiangong AI	9.528	8.934	9.315	9.479	7.472	7.239	9.543	9.540	5.895	8.549
Ernie Bot	9.624	8.925	9.334	9.415	7.836	7.692	9.518	9.601	6.362	8.701
KIMI	9.574	9.259	9.449	9.580	7.941	8.176	9.578	9.614	6.312	8.831
Metaso	9.215	8.345	8.924	9.013	7.005	6.249	9.142	8.994	5.203	8.010
ChatGLM	9.723	9.275	9.562	**9.715**	7.947	7.919	**9.826**	**9.809**	6.599	8.931
Baichuan	8.962	8.575	8.812	8.939	7.349	7.357	8.889	8.752	6.171	8.201
Tongyi	9.321	8.467	8.904	9.101	7.289	7.534	9.394	9.740	6.063	8.424
GraTAG (Ours)	9.797	**9.747**	9.559	9.222	**9.185**	**9.129**	9.599	9.784	**7.790**	**9.312**

Table 11: Ablation study of replacing our fine-tuned LLMs with proprietary and open-source models.

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
GPT-4o	9.745	9.622	**9.633**	8.940	8.823	9.065	9.456	9.754	7.147	9.132
Qwen 2.5-72B	9.698	9.485	9.475	8.517	8.912	8.709	9.592	9.670	7.330	9.043
GraTAG (Ours)	**9.797**	**9.747**	9.559	**9.222**	**9.185**	**9.129**	**9.599**	**9.784**	**7.790**	**9.312**

Table 12: Ablation study of replacing our fine-tuned answer generation model with proprietary and open-source models.

Model	Conciseness	Numerical Precision	Relevance	Factuality	Timeliness	Comprehensiveness	Clarity	Coherence	Insightfulness	Average
GPT-4o	9.771	9.681	**9.600**	9.095	8.971	9.110	9.550	9.771	7.661	9.245
Qwen 2.5-72B	9.741	9.577	9.563	8.849	8.794	9.005	9.549	9.72	7.546	9.149
GraTAG (ours)	**9.797**	**9.747**	9.559	**9.222**	**9.185**	**9.129**	**9.599**	**9.784**	**7.790**	**9.312**

Table 13: Ablation study of fine-tuning the rerank model.

Model	Precision	Recall	F1 Score	Wall Time (s)
GPT-4o [34]	**0.717**	0.719	**0.718**	3.6
Qwen 2.5-72B [46]	0.541	**0.894**	0.674	2.4
bge-reranker-v2-m3 [13]	0.568	0.671	0.615	0.1
GraTAG (ours)	0.627	0.735	0.677	0.1

Table 14: Impact of the number of sampled GQDs.

Number	Average Score
1	9.119
4	9.290
8	9.312
16	9.304
32	**9.314**

Table 15: Text-Image Matching Performance with bge-reranker-v2-m3 at Different Thresholds

Threshold	Inclusion (%)	Precision (%)
0.1	93	45
0.2	85	80
0.3	80	90
0.4	68	93
0.5	43	95

thresholds (0.1) achieve high inclusion (93%) but low precision (45%), while higher thresholds (0.5) improve precision (95%) at the cost of reduced inclusion (43%). We select threshold 0.3 as it provides the optimal balance with 80% inclusion and 90% precision, ensuring comprehensive coverage while maintaining answer quality.

C Multi-Faceted Evaluation Criteria

The detailed definitions of the multi-faceted evaluation criteria are provided in Tab. 16. The original text is in Chinese, and we translate it into English to align with the language of this paper.

D Instruction Prompt

D.1 Graph-based Query Decomposition

Prompt for the graph-based query decomposition is provided in Tab. 18.

D.2 Answer Generation

Prompt for answer generation is provided in Tab. 19.

D.3 LLM Evaluation

Tab. 17 presents the prompt used to instruct the LLM to evaluate the generated text based on multi-faceted criteria (see Tab. 16). To reduce task complexity and enhance evaluation quality, we assess one facet per evaluation and fill the metric title and definition accordingly.

Table 16: Multi-faceted evaluation criteria.

```
(1) Conciseness:
    - The response should directly address the user's question.
    - Avoid irrelevant content, unnecessary information, or roundabout explanations.
    - Deduct 1 point for each irrelevant statement.

(2) Numerical Precision:
    - If a question requires a specific number, avoid vague terms like "several" or "many times."
    - Responses should be precise and specific.
    - Deduct 1 point for each ambiguous statement.

(3) Relevance:
    - If the question specifies constraints (e.g., time, location, person, event),
      the answer must adhere to them.
    - Deduct 1 point for each instance of misalignment with the question's constraints.

(4) Factuality:
    - The information must be factually correct, especially for numerical or factual questions.
    - Deduct 1 point for each incorrect numerical or factual statement.

(5) Timeliness:
    - For ongoing news or urgent reports, ensure information reflects the latest updates.
    - The current date is {to be filled}.
    - If the question is not time-sensitive, no points are deducted.
    - For time-sensitive questions, deduct points proportionally based on outdatedness.

(6) Comprehensiveness:
    - The response should comprehensively cover all aspects of the user's inquiry.
    - The user should not need further search to grasp the full context.
    - Deduct 1 point for each missing essential element.

(7) Clarity:
    - The response should be easy to understand, well-structured, and formatted logically.
    - Example: Chronological events should be presented in chronological order.
    - Deduct 1 point for unclear or disorganized presentation.

(8) Coherence:
    - The response should be logically consistent, with smooth transitions between sentences.
    - Deduct 1 point for each instance of incoherent or disjointed phrasing.

(9) Insightfulness:
    - The response should provide insightful or unique perspectives.
    - Base score: 6 points.
    - Award 0.5-1 additional points for each innovative idea or expression.
```

Table 17: Prompt for multi-faceted evaluation.

```
Assume you are an article quality inspector. Please evaluate the response based on {Metric Title}.
I will provide the user's question and the final response.
The maximum score is 10 points, and the scoring rules are as follows:

{Metric Definition}

Please strictly follow the scoring rules. Example output format:

'{
    "Issues Identified": "X",
    "Calculation Process": "10-1.0-1.0-1.0 = 7.0",
    "Score": 7
}'

{Few-Shot Examples}

Your final score: \n"
```

Table 18: Prompt for graph-based query decomposition.

```
Please analyze the following query and return the
explanation in dictionary format.

Response format:
{'is_complex': True/False, 'sub_queries': [], 'parent_child': []}

Analysis Steps and Principles:

1. **Classify the nature of the query**
   - The query can be classified into one of two types:
     (a) A "complex query" that consists of multiple sub-queries.
     (b) A "simple query" that can be directly answered.
   - If the query is classified as "complex," set 'is_complex' to **True**.
   - If the query is "simple," set 'is_complex' to **False**, and leave 'sub_queries' and 'parent_child' as empty lists.

2. **Decomposing a Complex Query**
   - If the query is classified as "complex," break it down into **sub-queries** and store them in the 'sub_queries'
       list.
   - Decomposition principles:
     1) If a query contains multiple **target entities**, split it into multiple sub-queries.
        - Example: *"What are the latest social news and weather news in Shanghai?"*
          - Target entities: *"social news"*, *"weather news"*.
          - Split into: *"What are the latest social news in Shanghai?"* and *"What are the latest weather news in
              Shanghai?"*.
     2) Each sub-query should be **indivisible** and should not require further decomposition.
     3) No duplicate sub-queries.
     4) When referring to **names of people, places, or organizations**, ensure full and precise descriptions.
        - Example: *"What is the area and population of New Jersey, USA?"*
          - Correct split: *"What is the area of New Jersey, USA?"* and *"What is the population of New Jersey, USA?"*.
          - Incorrect split: *"What is the area of New Jersey?"* and *"What is the population of New Jersey?"*.
     5) The total number of sub-queries **should not exceed 6**.

3. **Analyzing Dependencies Between Sub-Queries**
   - If the query is complex, analyze the **dependency relationships** between sub-queries and store them in '
       parent_child'.
   - Example:
     - *"What natural disasters occurred in Indonesia in April?"*
     - *"How long did this natural disaster last?"*
     - The second question **depends** on the first; thus, the first is the *parent*, and the second is the *child*:
       ```json
       {"parent": "What natural disasters occurred in Indonesia in April?",
        "child": "How long did this natural disaster last?"}
       ```
   - Dependency principles:
     1) If sub-queries are **independent**, 'parent_child' remains an empty list.
     2) If the **child question cannot be answered without the parent**, it is a dependent relationship.
        - Example: "What is the latest iPhone model" is the parent node of "What are the specifications of the latest
            iPhone?"
        - The first question must be answered before the second.
     3) Every possible pair of sub-queries should be evaluated for dependency.
        - A query can be both a *parent* and a *child* in different relationships.

Example:
{Few-Shot Examples}

Query: {Query}
Response: \n
```

Table 19: Prompt for answer generation.

```
You are an AI assistant named Xinyu, developed by the
Shanghai Algorithm Innovation Research Institute. You are
performing an encyclopedia Q\&A task. Please generate an
answer based on the provided reference materials and
related Q\&A content.

Question: {Sub-Query}

Related Q\&A:
{Ancestor Node 1: Sub-Query}
{Ancestor Node 1: Answer}
{Ancestor Node 2: Sub-Query}
{Ancestor Node 2: Answer}
...

Reference materials:
{Retrieved Passage 1}
{Retrieved Passage 2}
...

When generating your answer, follow these guidelines:

[Structural Requirements]:
To ensure clarity and organization, you may use one or
more of the following structured formats:
 - **Introduction-Body-Summary**: Introduce the topic, elaborate, and summarize key points.
 - **Paragraphs by Subquestion**: Address each subquestion in a separate paragraph.
 - **Cause and Effect**: Explain the causes and consequences of an event.
 - **Comparison and Contrast**: Describe and compare two or more concepts.
 - **Chronological Order**: Describe events or steps in order of occurrence.
 - **Problem-Solution**: Introduce a problem and explain solutions or strategies.
 - **Pros and Cons**: List the positive and negative aspects of a decision or choice.
 - **Definition and Examples**: Provide a definition and illustrate it with examples.
 - **Logical Reasoning**: Derive conclusions based on assumptions or premises.
 - **List Structure**: Enumerate facts or features for easy readability.
 - **Categorization**: Introduce a concept, group it by categories, and explain in detail.
 - **Theme and Variations**: Explore a core theme and its variations.
 - **Case Study**: Explain a theory or concept through specific cases.
 - **Hierarchical Structure**: Arrange information by importance or sequence.
 - **Issue and Counterarguments**: Present an issue with supporting and opposing views.

[Language Requirements]:
(1) Use concise and clear language.
(2) Ensure that the answer's structure enhances clarity and readability.
(3) The response must directly and accurately address the question, avoiding irrelevant content.
(4) When citing reference materials, ignore template formatting or improper phrasing.
(5) If detailed elaboration is required, output the answer in a structured **Markdown** format.

Your Answer: \n
```