EventRAG: Supportive Event Retrieval on Hypergraph for Future Forecasting

Anonymous ACL submission

Abstract

Recent advancements in Retrieval-Augmented Generation (RAG) have significantly enhanced the ability of large language models to ground their responses in external knowledge. However, solving complex future forecasting problems remains a challenge due to the need for retrieving supportive events. Current methods focusing on textual-similarity or entity-relevance are not able to capture supportive events due to incompleteness of the knowledge base and the inherent nuanced nature of events. This paper introduces EventRAG, an event-oriented RAG framework specifically designed for future forecasting tasks. Specifically, we first propose the supportive event retrieval where we construct the event hypergraph index on the knowledge base. Based on that, we denote the event supportiveness as random variables and maximize the expectation. We establish the maximum expected event cover program to solve this maximization. Finally, EventRAG integrates the retrieval and reasoning into the event-oriented agentic reasoning process. It enables the framework to retrieve the needed information to perform complicated forecasting. We conducted experiments and in-depth analysis to evaluate the effectiveness of EventRAG. The results demonstrate that EventRAG significantly outperforms competitive RAG baselines in future forecasting. The code and dataset are available on the ARR system.

1 Introduction

Retrieval-Augmented Generation (RAG) has emerged as a transformative paradigm in natural language processing, addressing critical limitations of large language models (LLMs), such as knowledge cutoff issues (Gupta et al., 2024). By integrating retrieval mechanisms with generative models, RAG systems dynamically access external knowledge to enhance response accuracy and relevance in complex intelligent reasoning (Zhao et al., 2024). Given a question, recent advanced

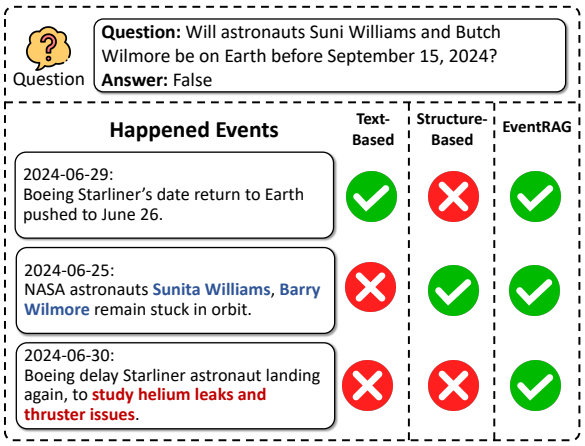


Figure 1: An example of future forecasting tasks such as event prediction. Retrieval abilities of each RAG methodology. EventRAG is able to retrieve supportive events even without explicit entity connections.

RAG systems aim to retrieve more relevant information via: 1) improved textual semantic similarity between query and knowledge such as query enriching (Gao et al., 2023) and retrieval requirement understanding (Oh et al., 2024). 2) nuanced entity-centric structured indexing as tree (Sarthi et al.) and graph (Edge et al., 2024). These developments achieve great performance in numerous applications in scenarios such as finance (Li et al., 2023; Islam et al., 2023), medical (Dou et al., 2024), law (Fei et al., 2023).

While current RAG performs well in problems depending on external knowledge, answering the complex future forecasting tasks remains a significant challenge. These tasks require RAG systems to retrieve happened events that are not only relevant but supportive of answering the questions. The supportiveness of an event for a question implies that the event contains information that can be used to answer the question, offering evidence or reasoning to back up the response. An event that is highly relevant may not necessarily provide strong support and vice versa. However, effectively retrieving events with high supportiveness poses

challenges. Firstly, real-world events and relations are often incompletely observable. Many events may have hidden details, unrecorded intermediate steps, or ambiguous relationships. Moreover, the complexity of event relationships themselves poses a challenge. Events can be mutually affected in complicated ways that are often difficult to represent accurately in a knowledge base.

These factors create restrictions for retrieval methods to fully identify highly supportive events. Currently, text-based RAG methods focus on query-document similarity (Gao et al., 2023) while structure-based RAG focuses on entity relations (Edge et al., 2024). As a result, traditional retrieval methods may either retrieve events that are only textual-similar or entity-related but lack the necessary support for answering the question or miss important events that could provide strong support. As shown in Figure 1, to solve this event prediction task, text-based retrieval methods can retrieve the first event through textual similarity. Entity-centric structural methods find the second event due to the core entities "Sunita Williams" and "Barry Wilmore". However, retrieving the third event requires deeper understanding that this event can support answering the question.

To bridge this gap, we propose `EventRAG`, a new event-centric RAG framework designed to retrieve support and related events for future forecasting. Our approach introduces two key innovations:

Supportive Event Retrieval `EventRAG` first constructs event hypergraph index on the knowledge base. It extracts events from documents and represents them as a hypergraph. The nodes represent the participants or actions of the event, whereas the hyperedges model the events themselves. This hypergraph structure offers a more natural and holistic way to capture the complex relations between events. To retrieve supportive events and knowledge on the hypergraph, we denote the event supportiveness as random variables and maximize the expectation. We introduce the Maximum Expected Event Cover to solve that, which establishes an optimization program that jointly maximizes event probabilistic supportiveness and structural connectivity. Moreover, we prove that this program bounds optimal supportive event retrieval. This counteracts potential biases in the partial observability of event supportiveness.

Event-Oriented Agentic Reasoning We employ a multi-step, agentic generation process to iteratively retrieve and reason about the answer. This process seamlessly integrates our supportive event retrieval and reasoning, allowing the system to adaptively adjust its approach and search for the knowledge it requires at each step.

We conduct experiments to testify the effectiveness of `EventRAG`. Compared to state-of-the-art RAG baselines among textual-based RAG and graph-based RAG, EventRAG achieves significant improvement. The results indicate the necessity of developing event-oriented RAG systems and demonstrate the validity of our method. We list our contributions:

- We introduce `EventRAG`, an event-oriented RAG system. It retrieves supportive events via the hypergraph index and the solving of the maximum expected event cover program.
- `EventRAG` integrates and retrieval and reasoning in an agentic RAG process. This multi-step process iteratively retrieves needed knowledge and answers the question.
- We conduct experiments to evaluate the effectiveness of `EventRAG`. The results demonstrate the validity of our method.

2 Preliminaries

2.1 Event Representation

In natural language processing (NLP) and knowledge representation, an event e is typically defined as a structured occurrence (Dölling et al., 2013). Events comprise participants (or arguments) $p \in \mathbb{P}$, representing entities involved in the event, and one action a, often identified by a lexical trigger word (Doddington et al., 2004).

In our implementation, events may include multiple actions $a \in \mathbb{A}$ to accommodate coarser-grained event definitions, where a single action word inadequately captures the scope (e.g. "protest" and "arrest" in a civil unrest event). Moreover, while structural representations encode core components, they risk information loss by omitting contextual nuances. To mitigate this, we augment the structure with a textual description $\mathcal{X}$. Thus, our full event representation is formalized as $e = (\mathbb{P}, \mathbb{A}, \mathcal{X})$.

2.2 Future Forecasting

Future Forecasting is a critical research area within the field of artificial intelligence and natural language processing, aimed at anticipating future events based on historical data and contextual information. This task involves analyzing large volumes of news articles, social media posts, or other temporal data sources, to identify patterns and trends

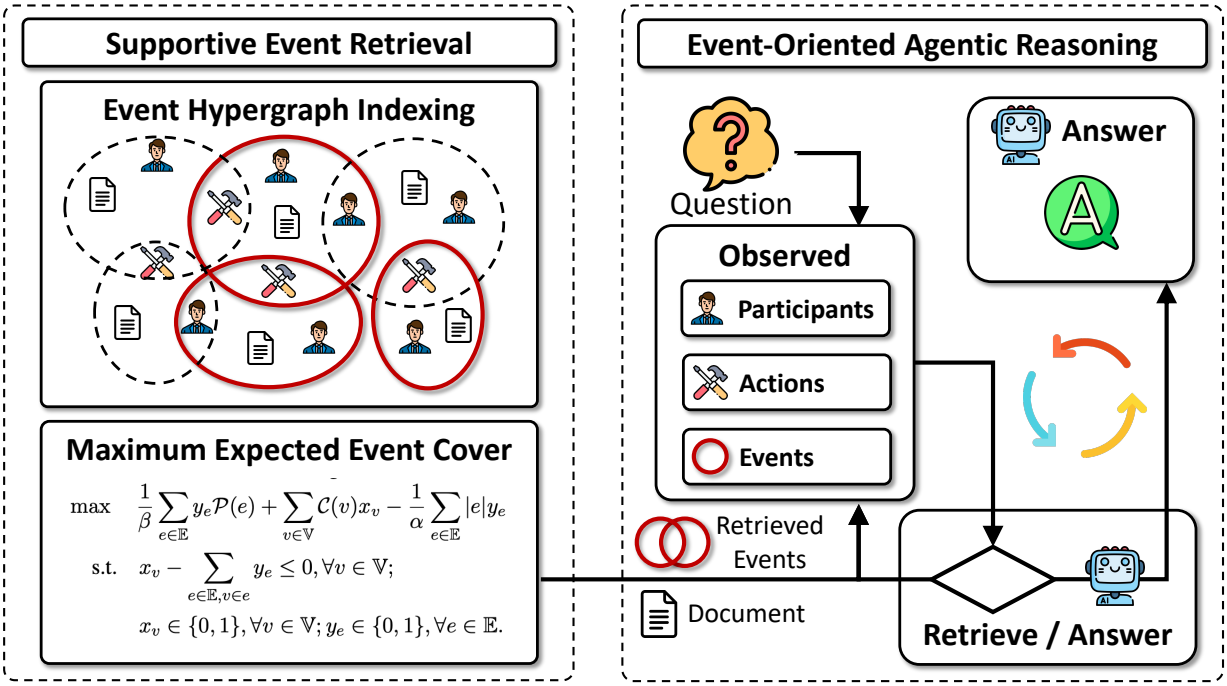


Figure 2: Overview of EventRAG. Given a question, our event-oriented agentic reasoning answers it in the multi-step process. In each step, it first retrieves supportive events and documents via our supportive event retrieval, then decides to answer or update states for the next step.

that can help forecast the occurrence of specific events. Specifically, given a binary classification prediction question $\mathcal{Q}$, a large set of documents $\mathbb{D}$ records all related information. The RAG system should retrieve supportive information such as events and documents from $\mathbb{D}$ and reason the whether the event in $\mathcal{Q}$ will happen or not:

$$\mathcal{Y} = \text{Reason}(\mathcal{Q}, \text{Retrieve}(\mathcal{Q}, \mathbb{D})), \quad (1)$$

where $\mathcal{Y} \in [0, 1]$ is the predicted probability of how likely the event asked by $\mathcal{Q}$ would happen.

3 Method

Our EventRAG framework addresses future forecasting tasks as shown in Figure 2. The system retrieves supportive events by indexing the knowledge base as an event hypergraph and models the event retrieval as the maximum expected event cover program (Section §3.1). Then it integratedly conducts iterative reasoning through an agentic reasoning process (Section §3.2).

3.1 Supportive Event Retrieval

Future forecasting tasks demand that RAG systems retrieve not just relevant but also supportive past events to answer questions. An event is considered supportive if it contains information to answer the question, providing evidence or rationales. Relevance does not always imply strong support, and vice versa. Effectively retrieving highly supportive events is challenging. First, real-world events and their relations are often not completely observable. There may be hidden details and unrecorded intermediate connections. Second, the complexity of event relations themselves is a problem. Events can interact intricately, and accurately representing them is difficult. These issues limit retrieval methods from fully identifying highly supportive events. Currently, text-based RAG methods focus on query-document similarity (Ma et al., 2024), while structure-based RAG emphasizes entity relations (Edge et al., 2024). As a result, these retrieval methods may retrieve only superficially relevant events lacking the required support.

To address this, EventRAG constructs an event hypergraph index in the knowledge base. Events are extracted from documents represented as hypergraphs. Nodes represent participants or actions of the event, and hyperedges represent events. The hypergraph structure captures the complex event semantics more naturally and comprehensively. We then introduce the maximum expected event cover program for retrieval. This program formulates an optimization problem to maximize both event structural connectivity and probabilistic supportiveness. We also prove that this program bounds optimal supportive event retrieval.

3.1.1 Event Graph Indexing

Specifically, to construct the hypergraph index, the first step is to extract events from the knowledge

base. Given the knowledge base $\mathbb{D}$, EventRAG begins with event extraction in each document $\mathcal{D} \in \mathbb{D}$ and constructs the event hypergraph. However, directly extracting events would encounter entity coreference problems where the same entity would be in different wording between events. To mitigate this problem, we first extract named entities then mining events based on them.

Entity Extraction The foundation of event extraction lies in accurate entity recognition. Precise entity identification reduces event extraction errors through better argument binding. Given a document $\mathcal{D} \in \mathbb{D}$, we extract named entities with LLMs, which are the event participants in the later stage. We show the prompt in Appendix A.7 (a). We provide several extraction examples in the prompt.

Event Extraction After harvesting the named entities which are the event participants, we perform event extraction based on them. We also implement this process using LLMs. The prompt is in the Appendix A.7 (b). We describe each event in sentences resulting in $\mathcal{X}$ and also extract the date and location. We provide demonstrations for few-shot typing. We also resolve the event co-reference by merging events that have large overlapped participants and actions. We concatenate their descriptions into the final description $\mathcal{X}$. After this, we acquire the structured event e.

Hypergraph Indexing We next construct the event hypergraph index based on the extracted events. An event hypergraph is a generalization of a graph in which an edge (called a hyperedge) is an event. It can connect any number of vertices, rather than being restricted to pairwise connections. Formally, a hypergraph $\mathcal{H}$ is defined as a pair $(\mathbb{V}, \mathbb{E})$. The node set $\mathbb{V} = \{v_1^{\mathcal{P}}, v_2^{\mathcal{P}}, \ldots, v_n^{\mathcal{P}}, v_1^{\mathcal{A}}, v_2^{\mathcal{A}}, \ldots, v_m^{\mathcal{A}}\}$ represents all participants and actions which is a finite set of vertices of $\mathcal{H}$[1]. $\mathbb{E} = \{e_1, e_2, \ldots, e_l\}$ is a family of non-empty subsets of $\mathbb{V}$, each representing a hyperedge which is also the event[2]. In particular, each hyperedge e in $\mathbb{E}$ corresponds to an event $e = (\mathbb{P}, \mathbb{A}, \mathcal{X}, \mathcal{D})$. where $\mathbb{P}$ and $\mathbb{A}$ are participants and actions of e. $\mathcal{X}$ is the event description generated before. $\mathcal{D}$ is the document where e sources. Each edge e plays the role of indexing associated with its source document $\mathcal{D}$. Unlike structure-based RAG indexing such as GraphRAG, hyperedges in $\mathcal{H}$ can encompass two or more vertices, enabling the retrieval of higher-order semantics of events demonstrated in Figure 2.

[1]We use v without superscripts to indicate node on $\mathcal{H}$ when we don't distinguish participants or actions.

[2]We use e represents both hyperedge and event since they are the same in our work. $\mathbb{E}$ is the set of them.

3.1.2 Maximum Expected Event Cover

In this section, we elaborate on how we do retrieval on our constructed event hypergraph index. Starting with the queried entities $\tilde{\mathbb{P}}$ and actions $\tilde{\mathbb{A}}$, we aim to retrieve query-relevant and question-supportive events and the documents:

$$(\tilde{\mathbb{E}}, \tilde{\mathbb{D}}) = \text{Retrieve}(\mathcal{H}, \tilde{\mathbb{P}}, \tilde{\mathbb{A}}). \quad (2)$$

$\tilde{\mathbb{E}}$ is the retrieved events while $\tilde{\mathbb{D}}$ is the retrieved documents indexed by $\tilde{\mathbb{E}}$. It first links $\tilde{\mathbb{P}}$ and $\tilde{\mathbb{A}}$ to nodes $\tilde{\mathbb{V}} \subseteq \mathbb{V}$ on the hypergraph. We model the retrieval as selecting a set of supportive and relevant events $\tilde{\mathbb{E}} \subseteq \mathbb{E}$.

To select supportive $\tilde{\mathbb{E}}$, the first thing is to measure the supportiveness of events in $\mathcal{H}$. However, the precise supportiveness is hard to obtain since the events are partially observed in the real world. Therefore, we use Bernoulli random variable $X(e) \sim \text{Bernoulli}(\mathcal{P}(e))$ to represent whether event e supports answering the question. $\mathcal{P}(e)$ is the probability of $X(e) = 1$. We then use LLMs to estimate a probability $\mathcal{P}(e)$. LLMs, pretrained on broad semantic and reasoning patterns, can assess the event against the query to measure its potential usefulness (Bynum and Cho, 2024). Inspired by Halawi et al. (2024), we calculate the supportive probability of document $\mathcal{D}$ as that of e. The prompt is in Appendix A.7 (c). Therefore, our retrieval is equivalent to we can select $\tilde{\mathbb{E}}$ to have the maximum total supportive probability.

Besides, we also consider the topology relevance of events on $\mathcal{H}$. This program should ensure that the retrieved events $\tilde{\mathbb{E}}$ are structurally related to $\tilde{\mathbb{V}}$. To do that, we assign a relevance score $\mathcal{C}_{\mathcal{V}}$ for all nodes v on $\mathcal{H}$ via the Personal Page Rank (Yang and Wang, 2024). The implementation details of the PPR algorithm are in the Appendix A.4. Based on $\mathcal{C}(v)$, we require that the selected events $\tilde{\mathbb{E}}$ can cover the nodes with the highest total relevance scores $\mathcal{C}(v)$. Last, to prevent it from selecting an excessive number of events and preferring events covering a large number of nodes, we add a regularization term. Formally, the supportiveness measurement of $\tilde{\mathbb{E}}$ is defined as:

$$f^{\tilde{\mathbb{E}}} = \frac{1}{\beta} \sum_{e \in \tilde{\mathbb{E}}} X(e) + \sum_{v \in \bigcup_{e \in \tilde{\mathbb{E}}} e} \mathcal{C}(v) - \frac{1}{\alpha} \sum_{e \in \tilde{\mathbb{E}}} |e|. \quad (3)$$

α and β are hyper-parameters. $|e|$ is the number of

nodes in e. Note that $f^{\tilde{\mathbb{E}}}$ is a random variable for any $\tilde{\mathbb{E}}$. Its first term are random variables while the rest two are not. The expectation of $f^{\tilde{\mathbb{E}}}$ is:

$$Ef^{\tilde{\mathbb{E}}} = \frac{1}{\beta}\sum_{e\in\tilde{\mathbb{E}}}\mathcal{P}(e) + \sum_{v\in\bigcup_{e\in\tilde{\mathbb{E}}}e}\mathcal{C}(v) - \frac{1}{\alpha}\sum_{e\in\tilde{\mathbb{E}}}|e|. \tag{4}$$

Our purpose is the select $\tilde{\mathbb{E}}$ that can maximize $Ef^{\tilde{\mathbb{E}}}$. To do that, we construct an integer program (IP) to get such desired $\tilde{E}$. For $e \in \mathbb{E}$, the binary variable y_e indicates whether the event e is selected ($y_e = 1$) or not ($y_e = 0$). Similarly, for $v \in \mathbb{V}$, the binary variable x_v indicates whether the participant or action v is covered ($x_v = 1$) or not ($x_v = 0$). Thus, for any $\tilde{\mathbb{E}}$, The corresponding binary variables are assigned as: for $v \in \mathbb{V}$, $x_v^{\tilde{\mathbb{E}}} = 1$ if v is covered by $\tilde{\mathbb{E}}$, otherwise $x^{\tilde{\mathbb{E}}} = 0$; for $e \in \mathbb{E}$, $y_e^{\tilde{\mathbb{E}}} = 1$ if $e \in \tilde{\mathbb{E}}$, otherwise $y_e^{\tilde{\mathbb{E}}} = 0$. Then, $Ef^{\tilde{\mathbb{E}}}$ in Eq.(4) can be re-writed as:

$$Ef^{\tilde{\mathbb{E}}} = \frac{1}{\beta}\sum_{e\in\mathbb{E}}y_e^{\tilde{\mathbb{E}}}\mathcal{P}(e) + \sum_{v\in\mathbb{V}}\mathcal{C}(v)x_v^{\tilde{\mathbb{E}}} - \frac{1}{\alpha}\sum_{e\in\mathbb{E}}|e|y_e^{\tilde{\mathbb{E}}} \tag{5}$$

Thus, we solve the following IP:

$$\begin{aligned}\max \quad & \frac{1}{\beta}\sum_{e\in\mathbb{E}}y_e\mathcal{P}(e) + \sum_{v\in\mathbb{V}}\mathcal{C}(v)x_v - \frac{1}{\alpha}\sum_{e\in\mathbb{E}}|e|y_e \\ \text{s.t.} \quad & x_v - \sum_{e\in\mathbb{E},v\in e}y_e \le 0, \forall v\in\mathbb{V}; \\ & x_v \in \{0,1\}, \forall v\in\mathbb{V}; y_e\in\{0,1\}, \forall e\in\mathbb{E}.\end{aligned} \tag{6}$$

For the optimal solution sol^* of IP (6), where sol^* is given by $\{x_v^* \mid v \in \mathbb{V}\} \cup \{y_e^* \mid e \in \mathbb{E}\}$, we choose $\tilde{\mathbb{E}}$ as $\{e \mid y_e^* = 1\}$. In the following theorem, we give an upper bound for $E(OPT - f(\tilde{\mathbb{E}}))$, where $\tilde{\mathbb{E}}$ is chosen by the IP (6).

Theorem 1. *For a single sampling of all $X(e), e \in \mathbb{E}$, there is a best $f^{\tilde{\mathbb{E}}}$. We denote OPT as the random variable representing this best $f^{\tilde{\mathbb{E}}}$. It is proved that $E(OPT - f^{\tilde{\mathbb{E}}}) \le \frac{2}{\beta}\sum_{e\in\mathbb{E}}(\mathcal{P}(e) - \mathcal{P}^2(e))$, where $\tilde{\mathbb{E}}$ is chosen by IP (6).*

We leave the proof in the Appendix. Ensured by this upper bound, our method can retrieve events that are close to the optimal supportive events.

3.2 Event-Oriented Agentic Reasoning

In this section, we introduce our integrated event-oriented agentic reasoning. In the previous section, we describe the retrieval for events that require queried entities and actions. However, answering future forecasting tasks requires deep reasoning. In addition, LLMs should retrieve the information needed to answer the question. To achieve that, we establish a multi-step agentic RAG process. We show the whole process in Algorithm 1.

Algorithm 1 Event-Oriented Agentic Reasoning

Require: Question Q, max steps l
Ensure: Final answer

Initialize $\dot{\mathbb{R}} \leftarrow \emptyset$, $\tilde{\mathbb{E}} \leftarrow \emptyset$, $t \leftarrow 0$,
$\tilde{\mathbb{P}} \leftarrow \text{ParticipantsOf}(Q)$
$\tilde{\mathbb{A}} \leftarrow \text{ActionsOf}(Q)$
while $t < l$ **do**
  $\mathcal{K} \leftarrow \text{M}(\dot{\mathbb{R}}, \dot{\mathbb{E}}, \dot{\mathbb{P}}, \dot{\mathbb{A}})$
  if $\mathcal{K}$ indicates **Retrieval then**
    $\dot{\mathbb{P}}, \dot{\mathbb{A}}, \mathcal{R}_{\text{new}} \leftarrow Parse(\mathcal{K})$
    $\mathbb{R} \leftarrow \mathbb{R} \cup \{\mathcal{R}_{\text{new}}\}$
    $(\tilde{\mathbb{E}}, \tilde{\mathbb{D}}) = \text{Retrieve}(\mathcal{H}, \dot{\mathbb{P}}, \dot{\mathbb{A}})$ ▷ Eq.2
    $\dot{\mathbb{E}} \leftarrow \dot{\mathbb{E}} \cup \tilde{\mathbb{E}}$, $\dot{\mathbb{D}} \leftarrow \dot{\mathbb{D}} \cup \tilde{\mathbb{D}}$
    $\dot{\mathbb{P}} \leftarrow \dot{\mathbb{P}} \cup \bigcup_{e\in\tilde{\mathbb{E}}} \text{ParticipantsOf}(e)$
    $\dot{\mathbb{A}} \leftarrow \dot{\mathbb{A}} \cup \bigcup_{e\in\tilde{\mathbb{E}}} \text{ActionsOf}(e)$
    $t \leftarrow t + 1$
  else
    $\text{Answer}(\dot{\mathbb{R}}, \dot{\mathbb{E}}, \dot{\mathbb{D}}, \dot{\mathbb{P}}, \tilde{\mathbb{A}})$
    exit loop
if $t \ge l$ **then** ▷ Force final answer
  $\text{Answer}(\dot{\mathbb{R}}, \dot{\mathbb{E}}, \dot{\mathbb{D}}, \dot{\mathbb{P}}, \dot{\mathbb{A}})$

There are some key modules in this process.

- Reasoning History $\dot{\mathbb{R}}$. Previous reasoning thoughts $\mathcal{R}$ initialized with an empty set.
- Oberserved Events $\dot{\mathbb{E}}$. It stores the retrieved events initialized with an empty set. The documents $\dot{\mathbb{D}}$ that are the sources of these events.
- Observed Participants $\dot{\mathbb{P}}$ and Actions $\dot{\mathbb{A}}$. It contains observed participants and actions via retrieving events. They are initialized by participants and actions from the question.

After initialization, in each step, we provide current $\dot{\mathbb{R}}$, $\dot{\mathbb{E}}$, and $\dot{\mathbb{P}}$ to the LLM and harvest the response $\mathcal{K} = \text{M}(\mathbb{R}, \dot{\mathbb{E}}, \dot{\mathbb{P}})$ where M represents the LLM. $\mathcal{K}$ would tell that the LLM chooses to answer the question or keep retrieving events. We show this plan prompt in Appendix A.7 (d). If the LLM chooses to answer, we call another prompt to answer the question based on $\dot{\mathbb{R}}$, $\dot{\mathbb{E}}$,$\dot{\mathbb{P}}$ and $\dot{\mathbb{A}}$[3]. Prompt is in Appendix A.7 (e).

If the LLM chooses to keep retrieving, we parse new $\tilde{\mathbb{P}}$ and $\tilde{\mathbb{S}}$ from $\mathcal{K}$. We ask M consider states $\dot{\mathbb{P}}$ and $\dot{\mathbb{A}}$ when generating the queries. The LLM also generates thoughts on why it should conduct

[3]Either $\dot{\mathbb{R}}$, $\dot{\mathbb{E}}$,$\dot{\mathbb{P}}$ and $\dot{\mathbb{A}}$ can be abcent according to tasks.

Method	# $\mathcal{D}$	Brier Score ↓	
		GPT-4O	*Gemini-Pro*
ScratchPAD (w.o. RAG) (Halawi et al., 2024)	-	25.42 ± 0.09	25.39 ± 0.41
Text-Based RAG			
Naive RAG	10.00	21.22 ± 0.30 (+4.20)	22.18 ± 0.39 (+3.21)
	20.00	20.31 ± 0.48 (+5.11)	20.98 ± 0.06 (+4.41)
APP (Halawi et al., 2024)	10.00	20.02 ± 0.26 (+5.40)	21.76 ± 0.24 (+3.60)
	20.00	19.79 ± 0.52 (+5.63)	21.06 ± 0.03 (+4.33)
HyDE (Gao et al., 2023)	20.00	20.59 ± 0.26 (+4.83)	20.87 ± 0.21 (+4.51)
RankLlama (Ma et al., 2024)	20.00	20.20 ± 1.44 (+5.22)	22.73 ± 0.08 (+2.66)
Structure-Based RAG			
HippoRAG (Gutiérrez et al., 2024)	20.00	20.94 ± 0.28 (+4.48)	22.82 ± 0.13 (+2.56)
EventRAG (Ours)	22.36	**18.48 ± 0.57 (+6.94)**	**20.10 ± 0.39 (+5.29)**

Table 1: Main results. # $\mathcal{D}$ is the average number of retrieval documents. Brier score is the lower the better. Bold stands for the best performance. Increments are compared w.o. RAG. We report mean and std values on twice runs.

this retrieval in the current step based on all the information. Then it retrieves events as Eq.2. After that, we update $\dot{\mathbb{E}}$ by the retrieved events. The participants and actions of the newly retrieved events are used to update $\dot{\mathbb{P}}$ and $\dot{\mathbb{A}}$. To avoid infinite steps, we set up a maximum step l. If the process step reaches l, we force the LLM to respond with the same answering prompt.

4 Experiments

This section comprehensively evaluates the proposed EventRAG through systematic experiments and analyses. We begin by introducing the dataset in Section §4.1and baselines in Section §4.2. The core findings are presented in Section §4.3, where we benchmark our approach against state-of-the-art baselines across multiple metrics. To quantify the contribution of individual components, Section §4.4 conducts ablation studies on key architectural choices. Finally, Section §4.5 provides multifaceted discussions: (1) a Regularization Study analyzing optimization stability, (2) a Maximum Step Study exploring effects on maximum agentic reasoning steps. We show then **Details Implementation specifics**, including hyperparameter configurations and training protocols, ensuring reproducibility in Appendix A.2. We examine the structural **statistics of event hypergraphs** index in Appendix §A.3 to establish foundational insights into hypergraph properties. We also analysis the **cost consumption** in Appendix A.6.

4.1 Dataset

We use PROPHET as the test dataset[4]. This is a real-world future forecasting dataset. It consists of 99 binary classification forecasting questions mainly in 2024 such as "Will Tim Walz win the VP debate against J.D. Vance?". Each question is paired with 100 news articles as knowledge base. This dataset is validated by a supportiveness estimation to ensure its inferablity. The evaluation metric of PROPHET is the Brier Score (Brier, 1950):

$$\texttt{Brier Score} = \frac{1}{N}\sum_{n}^{N}(\mathcal{Y}_n - \hat{\mathcal{Y}}_n)^2, \quad (7)$$

where $\hat{\mathcal{Y}} \in \{0, 1\}$ is the ground true indicating the queried event happened or not. $\mathcal{Y}$ is the predicted probability of the model.

4.2 Baselines

We compare our method to ScrathPAD which is the zero-shot ScrathPAD prompting method without RAG. The RAG baselines are Naive RAG, APP (Halawi et al., 2024), Rankllama (Ma et al., 2024), HyDE (Gao et al., 2023). To ensure comparability, **we uniformly use this prompt for reasoning**. We show this prompt in the Appendix A.7 (f). We leave the details of the baselines in the Appendix A.5.

4.3 Main Results

We compare EventRAG to the competitive baselines and show the results in Table 1. EventRAG presents the comparative performance against these baselines on the future forecasting task, measured by the Brier Score (lower is better). Our method performs best among all methods, demonstrating its superior accuracy in predicting future events. Specifically, EventRAG outperforms the strongest text-based RAG baseline (APP) by 1.31 points under GPT-4O and 2.68 points under Gemini-Pro, and surpasses the best structure-based base-

[4]https://github.com/TZWwww/PROPHET

Method	Brier Score ↓	# $\mathcal{D}$
EventRAG	18.48 ± 0.41	22.36
w.o. EC	19.64 ± 1.32 (-1.16)	26.82
w.o. SS	20.60 ± 0.28 (-2.12)	16.48
w.o. AR	19.64 ± 0.21 (-1.16)	17.26

Table 2: Ablation study. # $\mathcal{D}$ stands for the average number of retrieval documents. EC stands for Maximum Expected Event Cover. AR is event-oriented agentic reasoning. SS is supportive score of the event.

line (HippoRAG) by 2.46 and 2.72 points, respectively. These results validate the effectiveness of EventRAG and its motivation of event-centric RAG.

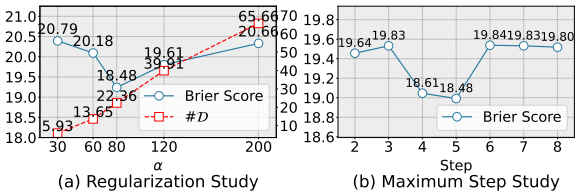
Compared with text-based RAG, methods like Naive RAG and HyDE show moderate improvements over direct prompting but plateau due to their reliance on surface-level semantics. RankLlama, despite its instruction-aware retrieval, underperforms EventRAG, emphasizing the need for event-aware structuring. In the structure-based RAG, HippoRAG's knowledge graph approach improves over text-based methods but remains inferior to EventRAG, as entity-centric graphs fail to encode multi-participant events or latent dependencies.

EventRAG's consistent superiority across both GPT-4O and Gemini-Pro underscores its model-agnostic design. The framework's reliance on event hypergraph index and probabilistic supportive event retrieval reduces dependency on the generative capabilities of LLMs, making it versatile for different backbone models.

4.4 Ablation Experiments

To further investigate the contributions of different components in EventRAG, we conducted ablation studies by selectively removing key modules from the full model. The results are summarized in Table 2. w.o. EC stands for ablating Maximum Expected Event Cover where we directly retrieve events covering the queried entities and actions from the event hypergraph. w.o. SS doesn't leverage the supportive score of the event where we uniformly treat all events to the same score. w.o. AR is without event-oriented agentic reasoning in which we solve the task in a one-step Maximum Expected Event Cover program retrieval.

The removal of the Maximum Expected Event Cover (w.o. EC) led to a significant degradation in performance, with the Brier Score increasing by 1.16 points(from 18.48 to 19.64). This indicates that the event cover program is crucial for selecting the most relevant events for reasoning. Disabling the Supportive Score (w.o. SS) of events led to



(a) Regularization Study (b) Maximum Step Study

Figure 3: Regularization study and maximum step study.

a more substantial drop in performance, with the Brier Score increasing by 2.12 points (from 18.48 to 20.60). This suggests that the probabilistic scoring of events is critical for evaluating their supportive and utility in answering the question. Similarly, removing the Event-Oriented Agentic Reasoning (w.o. AR) process also resulted in a 1.16-point increase in Brier Score (from 18.48 to 19.64). This highlights the importance of the iterative reasoning process in refining the answer through multiple steps of retrieval and reasoning.

In all, the combined effect of all these components is evident in the superior performance of the full EventRAG model compared to its ablations. The results demonstrate that the integration of all modules is essential for achieving robust and accurate performances in future forecasting.

4.5 Discussions

Regularization Study The regularization parameter α governs the balance between event coverage and retrieval specificity in the Maximum Expected Event Cover program. As shown in Figure 3 (a), increasing α initially increase the number of retrieved events by prioritizing structurally coherent pathways, leading to improved Brier Scores (optimal at $\alpha = 120$). Beyond this point, overly relax regularization causes performance degradation. Large α would incurs noisy information for retrieval. This demonstrates that moderate regularization ensures a synergistic balance between event supportiveness and structural relevance, countering noise while preserving forecasting utilities.

Maximum Step Study The maximum reasoning step l in the agentic process directly impacts reasoning depth. Figure 3 (b) reveals that increasing l from 1 to 5 progressively improves performances, as iterative retrieval enables the system to resolve ambiguities and incorporate supportive events. However, performance plateaus beyond $l = 5$, suggesting diminishing returns from additional steps. Notably, excessive steps ($l > 5$) marginally degrade results due to noise accumulation or redundant retrievals. These findings highlight the effectiveness of multi-step agentic reasoning. It

Method	Brier Score ↓
IP (6)	18.48 ± 0.41
IP (8)	19.63 ± 0.99

Table 3: Diversity inspectation of program Eq.(6).

also shows the necessity of setting l to balance thoroughness and efficiency, with $l = 5$ achieving optimal trade-offs in our experiments.

Event Diversity The retrieved events are desired to be diversified leading to more supportive information. In this part, we further inspect retrieved event diversity by IP (6). We add additional the following constraints to adapt it into a new IP:

$$\sum_{e\in\mathbb{E}, v\in e} y_e \leq 3, \forall v \in \mathbb{V}. \quad (8)$$

These constraints restrict the number of nodes from the same event should not exceed 3. And rerun the experiments. The results are in Table 3. We find the performances drops indicating the sufficient event diversity of our method.

5 Related Work

5.1 Retrieval-Augmented Generation (RAG)

Text-based methods focus on retrieving knowledge through textual signals, prioritizing similarity. Recent advancements include query refinement (Gao et al., 2023), self-reflection mechanisms in SELF-RAG (Asai et al., 2024), and iterative error correction in Corrective RAG (Liu et al., 2024). However, these methods often overlook structural or interactive aspects of knowledge organization.

Structure-based approaches address this by leveraging hierarchical or graph-based structures. RAPTOR (Sarthi et al., 2024) introduces recursive abstraction for tree-structured retrieval, enhancing multi-scale document representation. Graph-based methods, such as query-focused summarization (Edge et al., 2024) and HippoRAG (Teyler et al., 2024) integrates graph traversals to capture global context and long-term dependencies (Procko and Ochoa, 2024).

Agentic RAG integrates autonomous agents to orchestrate retrieval and generation. AMOR (Guan et al., 2024a) employs modular knowledge agents trained via process feedback, while GEAR (Shen et al., 2024) combines graph reasoning with agentic decision-making. Agent-G (Lee et al.) proposes a unified framework with an agent, retriever bank, and critic module, solving questions using hybrid knowledge sources.

Among these advanced RAG methods, we are the first to propose event-oriented RAG systems for complex knowledge-grounded event reasoning such as future forecasting.

5.2 Event Graph

As a event-centric knowledge base, event graph is constructed for storing abstract-level (Zhang et al., 2020) or instance-level (Gottschalk and Demidova, 2018) events and their relations. It can be used for future forecasting (Li et al.) or causality tracking (Tao et al., 2023). Although similar as previous event graph methods that are knowledge bases, our method firstly construct event hypergraph running as index for retrieval.

5.3 Future Forecasting

Forecasting on Event Graphs A foundational approach involves modeling event relations to predict downstream consequences. Zhan et al. (2024) leverage event causality graphs to simulate chains of cause-effect relationships, enabling structured reasoning about future scenarios. Similarly, Tao et al. (2024) enhance prediction by event schema graph guidance. These methods construct the event graph as knowledge to improve prediction.

Forecasting by LLMs Methods forecast future events by retrieving external information such as news, then reasoning the answer (Halawi et al., 2024; Guan et al., 2024b). Ye et al. (2024) establish a multi-agent framework where each agent has its role in utility of forecasting. Wang et al. (2024) introduce the future time series prediction driven by LLMs assisted by external information.

Compared with these methods, we propose an event-oriented RAG framework which can be plugged in other form of future forecasting system.

6 Conclusion

We introduced `EventRAG`, an event-oriented RAG framework tailored for future forecasting. By constructing an event hypergraph index and employing the Maximum Expected Event Cover Program, `EventRAG` effectively retrieves supportive events that are crucial for answering. Additionally, `EventRAG` integrates retrieval and reasoning through a multi-step, event-oriented agentic reasoning process, enabling adaptive knowledge acquisition and robust reasoning. Our experiments demonstrate significant improvements over state-of-the-art RAG baselines, highlighting the effectiveness of our method and the necessity of developing event-oriented RAG systems.

Limitations

In this work, we use close-source LLMs for reasoning and agentic operations. However, training LLMs on this would further benefit this process. We leave it to future work.

Ethics Statement

Research-Only Purpose Our algorithm is strictly designed for academic and research purposes. Any commercial exploitation, malicious intent (including but not limited to defamation, harassment, or discriminatory practices), or misuse beyond its intended scope is explicitly prohibited. **Data Accountability Disclaimer** The creators of this algorithm bear no responsibility for any financial, unauthorized, unethical, or harmful application of the algorithm. **Prohibition of Harmful Outcomes** Any deployment of this algorithm that leads to financial, physical, psychological, or socio-economic harm is categorically condemned. Users are urged to implement safeguards against such risks.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*.

Glenn W Brier. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Lucius EJ Bynum and Kyunghyun Cho. 2024. Language models as causal effect generators. *arXiv preprint arXiv:2411.08019*.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Johannes Dölling, Tatjana Heyde-Zybatow, and Martin Schäfer. 2013. *Event structures in linguistic form and interpretation*, volume 5. Walter de Gruyter.

Chengfeng Dou, Ying Zhang, Yanyuan Chen, Zhi Jin, Wenpin Jiao, Haiyan Zhao, and Yu Huang. 2024. Detection, diagnosis, and explanation: A benchmark for Chinese medial hallucination evaluation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4784–4794, Torino, Italia. ELRA and ICCL.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Zhiwei Fei, Xiaoyu Shen, Dawei Zhu, Fengzhe Zhou, Zhuo Han, Songyang Zhang, Kai Chen, Zongwen Shen, and Jidong Ge. 2023. Lawbench: Benchmarking legal knowledge of large language models. *arXiv preprint arXiv:2309.16289*.

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)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Simon Gottschalk and Elena Demidova. 2018. Eventkg: A multilingual event-centric temporal knowledge graph. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 272–287. Springer.

Jian Guan, Wei Wu, Zujie Wen, Peng Xu, Hongning Wang, and Minlie Huang. 2024a. Amor: A recipe for building adaptable modular knowledge agents through process feedback. *arXiv preprint arXiv:2402.01469*.

Yong Guan, Hao Peng, Xiaozhi Wang, Lei Hou, and Juanzi Li. 2024b. Openep: Open-ended future event prediction. *arXiv preprint arXiv:2408.06578*.

Shailja Gupta, Rajesh Ranjan, and Surya Narayan Singh. 2024. A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape and future directions. *arXiv preprint arXiv:2410.12837*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *arXiv preprint arXiv:2405.14831*.

Danny Halawi, Fred Zhang, Chen Yueh-Han, and Jacob Steinhardt. 2024. Approaching human-level forecasting with language models. *arXiv preprint arXiv:2402.18563*.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. Financebench: A new benchmark for financial question answering. *arXiv preprint arXiv:2311.11944*.

Meng-Chieh Lee, Qi Zhu, Costas Mavromatis, Zhen Han, Soji Adeshina, Vassilis N Ioannidis, Huzefa Rangwala, and Christos Faloutsos. Agent-g: An agentic framework for graph retrieval augmented generation.

Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. 2023. Large language models in finance: A survey. In *Proceedings of the fourth ACM international conference on AI in finance*, pages 374–382.

Zhongyang Li, Xiao Ding, and Ting Liu. Constructing narrative event evolutionary graph for script event prediction.

Yunzhi Liu, Yizhong Wang, Zeqiu Wu, Akari Asai, and Hannaneh Hajishirzi. 2024. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.

Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. 2024. Instructir: A benchmark for instruction following of information retrieval models. *arXiv preprint arXiv:2402.14334*.

Trevor T. Procko and Oscar Ochoa. 2024. Graph retrieval-augmented generation: A survey. *arXiv preprint arXiv:2408.08921*.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Pritish Sarthi, Sanaullah Abdullah, Anirudh Tuli, Shreya Khanna, Amanpreet Singh Goldie, and Christopher D. Manning. 2024. RAPTOR: Recursive abstractive processing for tree-organized retrieval. *arXiv preprint arXiv:2401.18059*.

Zhili Shen, Chenxin Diao, Pavlos Vougiouklis, Pascual Merita, Shriram Piramanayagam, Damien Graux, Dandan Tu, Zeren Jiang, Ruofei Lai, Yang Ren, et al. 2024. Gear: Graph-enhanced agent for retrieval-augmented generation. *arXiv preprint arXiv:2412.18431*.

Zhengwei Tao, Zhi Jin, Xiaoying Bai, Haiyan Zhao, Chengfeng Dou, Yongqiang Zhao, Fang Wang, and Chongyang Tao. 2023. Seag: Structure-aware event causality generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4631–4644.

Zhengwei Tao, Zhi Jin, Yifan Zhang, Xiancai Chen, Haiyan Zhao, Jia Li, Bing Liang, Chongyang Tao, Qun Liu, and Kam-Fai Wong. 2024. A comprehensive evaluation on event reasoning of large language models. *arXiv preprint arXiv:2404.17513*.

T. J. Teyler, P. Discenna, and J. W. Rudy. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *arXiv preprint arXiv:2405.14831*.

Xinlei Wang, Maike Feng, Jing Qiu, Jinjin Gu, and Junhua Zhao. 2024. From news to forecast: Integrating event analysis in llm-based time series forecasting with reflection. *arXiv preprint arXiv:2409.17515*.

Mingji Yang and Xiaokui Wang. 2024. Efficient algorithms for personalized pagerank computation: A survey. *arXiv preprint arXiv:2403.05198*.

Chenchen Ye, Ziniu Hu, Yihe Deng, Zijie Huang, Mingyu Derek Ma, Yanqiao Zhu, and Wei Wang. 2024. Mirai: Evaluating llm agents for event forecasting. *arXiv preprint arXiv:2407.01231*.

Chuanhong Zhan, Wei Xiang, Chao Liang, and Bang Wang. 2024. What would happen next? predicting consequences from an event causality graph. *arXiv preprint arXiv:2409.17480*.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. Aser: A large-scale eventuality knowledge graph. In *Proceedings of the web conference 2020*, pages 201–211.

Siyun Zhao, Yuqing Yang, Zilong Wang, Zhiyuan He, Luna K Qiu, and Lili Qiu. 2024. Retrieval augmented generation (rag) and beyond: A comprehensive survey on how to make your llms use external data more wisely. *arXiv preprint arXiv:2409.14924*.

A Appendix

A.1 Proof of Theorem 1

We show the proof of **Theorem 1** as the following:

Proof. Consider a single sample $s = \{s_e \mid e \in \mathbb{E}\}$ of all $X(e)$, where $e \in \mathbb{E}$. Let $\tilde{\mathbb{E}}_s$ be the set that maximizes $f^{\tilde{\mathbb{E}}_s}$ under this sample. Recall that $sol^* = \{x_v^* \mid v \in \mathbb{V}\} \cup \{y_e^* \mid e \in \mathbb{E}\}$ is an optimal solution of IP (6), and $\tilde{\mathbb{E}} = \{e \mid y_e^* = 1\}$. For a feasible solution $sol = \{x'_v \mid v \in \mathbb{V}\} \cup \{y'_e \mid e \in \mathbb{E}\}$, denote $g(sol)$ as the objective value of sol in IP (6).

For $v \in \mathbb{V}$, let $x_v^s = 1$ if there exists an event in $\tilde{\mathbb{E}}_s$ covering x_v, otherwise, let $x_v^s = 0$. Similarly, let $y_e^s = 1$ if $e \in \tilde{\mathbb{E}}_s$, otherwise, let $y_e^s = 0$. Notably, $sol_s = \{x_v^s \mid v \in \mathbb{V}\} \cup \{y_e^s \mid e \in \mathbb{E}\}$ is a feasible solution of IP (6). Since sol_s and sol^* are a feasible and an optimal solution of IP (6), respectively, it follows that $g(sol_s) - g(sol^*) \leq 0$. Furthermore, we have

$$\begin{aligned} & f^{\tilde{\mathbb{E}}_s} - f^{\tilde{\mathbb{E}}} \\ =& f^{\tilde{\mathbb{E}}_s} - g(sol_s) - (f^{\tilde{\mathbb{E}}} - g(sol^*)) \\ & + g(sol_s) - g(sol^*) \\ \leq& f^{\tilde{\mathbb{E}}_s} - g(sol_s) - (f^{\tilde{\mathbb{E}}} - g(sol^*)) \end{aligned} \tag{9}$$

In this sample, by the definition of $f^{\tilde{\mathbb{E}}}$, we have

$$f^{\tilde{\mathbb{E}}_s} = \frac{1}{\beta} \sum_{e \in \mathbb{E}} y_e^s s_e + \sum_{v \in \mathbb{V}} \mathcal{C}(v) x_v^s - \frac{1}{\alpha} \sum_{e \in \mathbb{E}} |e| y_e^s$$

Thus, it deduces that

$$f^{\tilde{\mathbb{E}}_s} - g(sol_s) = \frac{1}{\beta} \sum_{e \in \mathbb{E}} y_e^s (s_e - \mathcal{P}(e)) \tag{10}$$

Similarly, we have

$$f^{\tilde{\mathbb{E}}} - g(sol^*) = \frac{1}{\beta} \sum_{e \in \mathbb{E}} y_e^* (s_e - \mathcal{P}(e)) \tag{11}$$

Then by Equations (9)-(11), we have

$$\begin{aligned} & f^{\tilde{\mathbb{E}}_s} - f^{\tilde{\mathbb{E}}} \\ \leq& f^{\tilde{\mathbb{E}}_s} - g(sol_s) - (f^{\tilde{\mathbb{E}}} - g(sol^*)) \\ =& \frac{1}{\beta} \sum_{e \in \mathbb{E}} (y_e^s - y_e^*)(s_e - \mathcal{P}(e)) \end{aligned} \tag{12}$$

Since $0 \leq \mathcal{P}(e) \leq 1$ and $y_e^s, y_e^*, s_e \in \{0, 1\}$, it follows that $(y_e^s - y_e^*)(s_e - \mathcal{P}(e)) \leq s_e - (2s_e - 1)\mathcal{P}(e)$. Thus by Eqaution (12), we have

$$f^{\tilde{\mathbb{E}}_s} - f^{\tilde{\mathbb{E}}} \leq \frac{1}{\beta} \sum_{e \in \mathbb{E}} (s_e - (2s_e - 1)\mathcal{P}(e)) \tag{13}$$

By Equation (13), and by enumerating all samples on random variables $X(e)$ where $e \in \mathbb{E}$, it follows that

$$E(OPT - f^{\tilde{\mathbb{E}}}) \leq \frac{2}{\beta} \sum_{e \in \mathbb{E}} \mathcal{P}(e) - \mathcal{P}^2(e).$$

□

A.2 Implementation Details

Event Hypergraph In Event Hypergraph Indexing, we use `gpt-4o-mini-2024-07-18` for Entity Extraction, Event Extraction, and Event Typing. We use networkx[5] toolkit for hypergraph implementation. Since networkx doesn't have hypergraph operation, we use DiGraph to implement hypergraph. In our implementation, the event hyperedge is a node in the DiGraph, and its participant nodes and action nodes connect to it.

Event-Oriented Agentic Reasoning We use LangGraph[6] framework to implement the event-oriented agentic reasoning. We wrap the supportive events retrieval as a tool in LangGraph, and use the function call module of LLMs to call this tool. We show the tool implementation in Figure 5. When do entity linking on the hypergraph, we find two most similar nodes of each query participant or action. We use `all-MiniLM-L6-v2` model in similarity computing.

In maximum expected event cover retrieval, we set regularization parameter $\alpha = 80$, and supportive encouraging parameter $\beta = 200$ in Eq. 6. In event-oriented agentic reasoning, we set the maximum step $l = 5$. All experiments in this work are under twice runs. We report the mean and std values. The version for GPT-4O is `gpt-4o-2024-08-06` while Gemini-Pro is `gemini-1.5-pro-latest`.

A.3 Event Hypergraph Statistics

The statistical profile of event hypergraphs, in Table 4, reveals three key structural properties. First, the average of 1.98 participants a per event e indicates frequent multi-nodes interactions. Second, the moderate action $\mathcal{A}$ multiplicity (1.33 a/e).

Notably, the scale statistics demonstrate significant complexity: each question $\mathcal{Q}$ contains 461.30 events ($e/\mathcal{Q}$) implying *high-frequency* of event patterns. The participant-event ratio (324.42 $p/\mathcal{Q}$ vs. 2.82 e/p) further reveals *asymmetric role distributions*: while most participants engage in 2-3 events,

[5] https://networkx.org/

[6] https://www.langchain.com/langgraph

Event e			Participant p		Action a	
p / e	a / e	e / $\mathcal{Q}$	e / p	p / $\mathcal{Q}$	e / a	a / $\mathcal{Q}$
1.98	1.33	461.30	2.82	324.42	1.29	476.52

Table 4: Statistics of event hypergraph. e, p, a, $\mathcal{Q}$ stand for event, event participant, event action, and the question. We report the average numbers.

a minority serve as hubs across dozens of interactions. These characteristics collectively justify our EventRAG's capacity for indexing events.

A.4 Personal Page Rank

The Personal PageRank (PPR) algorithm is implemented as follows to acquire the relevance scores $\mathcal{C}(v)$ of each node v on the hypergraph $\mathcal{H}$.

Let $\mathbf{M}$ be the transition probability matrix of the hypergraph $\mathcal{H}$. For a hypergraph with n nodes, $\mathbf{M}$ is an $n \times n$ matrix, where the element M_{ij} represents the probability of transitioning from node i to node j.

We start by initializing the relevance scores of all nodes. Let $\mathcal{C}^{(0)}(v)$ be the initial relevance score of node v. We set $\mathcal{C}^{(0)}(v)$ based on the initial assignment of relevance score $\mathcal{C}_{\mathcal{V}}$ for all nodes v on $\mathcal{H}$. That is, $\mathcal{C}^{(0)}(v) = \mathcal{C}_{\mathcal{V}}(v)$.

The PPR algorithm then iteratively updates the relevance scores. In the t^{th} iteration, the relevance score of node v, $\mathcal{C}^{(t)}(v)$, is calculated as follows:

$$\begin{aligned}\mathcal{C}^{(t)}(v) =&(1-\alpha)\sum_{u\in \text{In}(v)} M_{uv}\mathcal{C}^{(t-1)}(u) \\ &+\alpha\mathcal{C}^{(0)}(v)\end{aligned} \tag{14}$$

where $\text{In}(v)$ is the set of in-neighbors of node v, α is a teleportation probability (a scalar value between 0 and 1, we set $\alpha = 0.15$). The first term $(1-\alpha)\sum_{u\in \text{In}(v)} M_{uv}\mathcal{C}^{(t-1)}(u)$ represents the contribution from the in-neighbors of node v, and the second term $\alpha\mathcal{C}^{(0)}(v)$ is the teleportation term, which helps to prevent the algorithm from getting trapped in cycles.

We repeat this iterative process until the change in the relevance scores between two consecutive iterations is below a certain threshold $\epsilon = 1^{-6}$. That is, we stop when

$$\sum_{v\in V} |\mathcal{C}^{(t)}(v) - \mathcal{C}^{(t-1)}(v)| < \epsilon \tag{15}$$

where V is the set of all nodes in the hypergraph $\mathcal{H}$. At the end of the iteration process, the final relevance score of each node v is $\mathcal{C}(v) = \mathcal{C}^{(t)}(v)$, which will be used in the maximum expected event cover program for event retrieval.

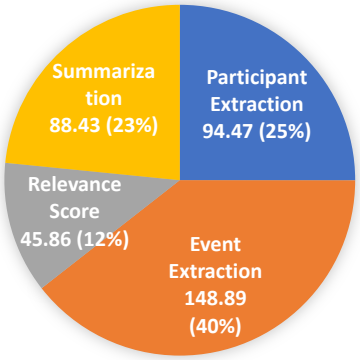


Figure 4: Cost analysis. The amount of token consumed per unit of data, measured in thousands.

A.5 Baselines

ScrathPAD: This is the zero-shot ScrathPAD prompting method without RAG. We adopt the scratchpad prompt introduced by Halawi et al. (2024). For other RAG methods, to ensure comparability, **we uniformly use this prompt for reasoning**. We show this prompt in the Appendix A.7 (f).

Naive RAG: Given the length of news articles, we perform summarization of these articles in advance. The RAG method then retrieves relevant news articles by calculating the embedding similarity between the question and the news summaries. For this purpose, we employ all-MiniLM-L6-v2 models in SentenceTransformer[7]. After retrieving the news, we utilize the scratchpad prompt for reasoning.

APP: This method, introduced by Halawi et al. (2024), is specifically text-based RAG designed for future forecasting. It also begins with summarizing the news articles. Subsequently, it uses an LLM to compute the relevance score. Following this, it too makes use of the scratchpad prompt for the reasoning process.

Rankllama: Rankllama is a retrieval method capable of understanding complex retrieval instructions (Ma et al., 2024). It encodes both the question and the news articles (using summaries of the news) with the model. After the retrieval step, it provides answers in the format of the scratchpad prompt.

HyDE: For a given query, this method leverages an instruction - following language model (such as InstructGPT) to generate a "hypothetical document" that captures relevance patterns (Gao et al., 2023). This is a query refinement method. In the event prediction scenario, we generate potential future events that could influence the answer. Then, rele-

[7]https://sbert.net

```
class FindEventsByEntitiesAndActionsInput(BaseModel):
      entities: list[str] = Field(description="Names of the
            input named entities rather than date or time. It
            must be directly a list of string of entities.
            They can not be dates or time. Eg. ['A', 'B',
            'C'].",
            required = True)
      actions: list[str] = Field(description="Names of the
            input actions. It must be directly a list of
            string of actions. Eg. ['A', 'B', 'C'].",
            required = True)


tool = StructuredTool.from_function(
      Func = _find_events_by_paricipants_and_actions,
      Name = "FindEventsByEntitiesAndActions",
      Description = "Find events of a certain event type on
            the graph which are simultaneously related to the
            input named entities and actions.",
      args_schema = FindEventsByEntitiesAndActionsInput,)
```

Figure 5: Core code for retrieval tool and its inputs.

vant news articles are retrieved.

HippoRAG: This is a representative graphrag framework inspired by the human brain's long-term memory system. It enhances large language models (LLMs) by enabling efficient multi-hop reasoning and knowledge integration from external documents, using a knowledge graph and personalized PageRank algorithm (Gutiérrez et al., 2024).

A.6 Cost Analysis

We analyze the cost consumption of our method. In the process of hypergraph construction, the token consumption are mainly in four steps: participant extraction, event extraction, relevance score, and summarization. We show the consumption in Figure 4. Th total consumption of a data is 337,760 tokens. We use `gpt-4o-mini-2024-07-18` model, that only cost 0.2266 $.

A.7 Prompts

We show all prompts (a-f) as the following.

(a) Named Entity Extraction

Your task is to extract named entities from the given paragraph. Don't extract date, time. Respond with a list of entities.
Document: $\mathcal{D}$
Output:

(b) Event Extraction

Your task is to extract all events about the given named entities from the given passages. Each event contains at least one participant and one action. Also, summarize the event.
Pay attention to the following requirements:
- Each event is in the JSON format as ["participants": ["", "", "", ...], "action": "", "time": "", "location": "", "text": "", ...].
Document: $\mathcal{D}$
Named Entities: $\mathbb{P}$
Output:

(c) Supportive Probability

Please consider the following forecasting question and its background information. After that, I will give you a news article and ask you to rate its relevance with respect to the forecasting question.
Question: $\mathcal{Q}$
Question Background: *background*
Resolution Criteria: *resolution criteria*
Article: $\mathcal{D}$
Please rate the relevance of the article to the question, at the scale of 1-6
1 – irrelevant
2 – slightly relevant
3 – somewhat relevant
4 – relevant
5 – highly relevant
6 – most relevant
Guidelines:
- If the article has events of similar types which may happened on different subjects, it also consider relevant to the question.
- You don't need to access any external sources. Just consider the information provided.
- If the text content is an error message about JavaScript, paywall, cookies or other technical issues, output a score of 1.
Rating:

(d) Agentic Plan

Your task is to answer the question about reasoning and predicting a future event in several turns via operation on an event graph. You need to answer the question step by step. In each step, you can answer the question or retrieve more events from a pre-built event graph via calling tools. Previous Memory records summarization of all previous retrieved information and thoughts. Observed Entities and Observed actions store named entities and actions you already retrieved. You must take next step by considering Previous Memory, Observed Entities, and Observed Actions. Answer the question know when you think the information is enough for reasoning the answer. You must generate the thought of calling the tools. You can only use observed entities and actions as arguments of the tool call. Previous Tool Calls records previous tool calling you have made. Don't make duplicated tool call with the same arguments that have been made before. Don't mention in the output words.
Question: $\mathcal{Q}$
Question Background: *background*
Previous Memory (could be empty): $\mathbb{R}$
ALL Observed Entities (could be empty): $\mathbb{P}$
ALL Observed Actions (could be empty): $\mathbb{A}$
Previous Tool Calls: *tool calls*
Your Thought:

(e) Answer

Question: $\mathcal{Q}$
Question Background: *background*
Resolution Criteria: *resolution criteria*
I have observed events related to the question.
Observed Events: $\mathbb{E}$
Instructions:
Your goal is to aggregate the information and make a final prediction.
1. Provide at least 3 reasons why the answer might be no: {Insert your thoughts}
2. Provide at least 3 reasons why the answer might be yes: {Insert your thoughts}
3. Rate the strength of each of the reasons given in the last two responses: { Insert your rating of the strength of each reason }
4. Aggregate your considerations: {Insert your aggregated considerations}
5. Output your answer (a number between 0 and 1) with an asterisk at the beginning and end of the decimal: {Insert your answer}

(f) ScratchPAD

Question: $\mathcal{Q}$
Question Background: *background*
Resolution Criteria: *resolution criteria*
We have retrieved the following information for this question: *retrieved articles*
Instructions:
1. Provide at least 3 reasons why the answer might be no: { Insert your thoughts }
2. Provide at least 3 reasons why the answer might be yes: { Insert your thoughts }
3. Rate the strength of each of the reasons given in the last two responses: { Insert your rating of the strength of each reason }
4. Aggregate your considerations: { Insert your aggregated considerations }
5. Output your answer (a number between 0 and 1) with an asterisk at the beginning and end of the decimal: { Insert your answer }