RAGTKGC: Undertaking Temporal Knowledge Graph Completion with Retrieval Augmented Generation

No Author Given

No Institute Given

Abstract. Knowledge Graphs (KGs) have recently gained attention as a symbolic formalism for capturing factual and semantic knowledge. KGs are constantly evolving due to emergence of new information, thus shifting from static to Temporal Knowledge Graphs (TKGs). Due to this evolving nature, TKGs tend to be incomplete. Therefore, Temporal Knowledge Graph Completion (TKGC) aims to predict the missing elements of partial facts, based on existing knowledge and its temporal validity. To solve this task, numerous solutions employ language models either pre-trained or Large Language Models (LLMs), taking advantage of their inherent ability to process and understand natural language which is extremely valuable when dealing with such semantic networks. In this paper, we propose a new Retrieval Augmented Generation (RAG) based framework, RAGTKGC, which focuses on combining logical rule mining with LLMs for TKGC. We present a novel method for mining temporal logical rules and constructing a relevant history for the target link. Additionally, we test whether RAG with LLMs can act as either a knowledge extractor or a rule miner, with the final purpose of boosting the usefulness of the provided history. Finally, the obtained results confirm the applicability of RAGTKGC, with metrics on par with State-of-The-Art (SoTA) frameworks, and initial insights into the suitability of LLMs for the proposed framework.

Keywords: Knowledge Graph Completion · Large Language Models · Temporality.

1 Introduction

Knowledge Graphs (KG) have recently been used in many downstream tasks such as text classification, recommender systems, etc. KGs represent knowledge in the form of entities and potentially different relations between these entities [16]. Therefore, KGs provide an excellent endeavor to store and analyse vast amounts of heterogeneous data. They keep factual knowledge in the form of triples represented as <subject, relationship, object>, e.g., <Lionel_Messi, playedFor, Barcelona_FC>. However, the real world is dynamic and rapidly changing, and new information is continuously added in KGs. Hence, their static form may store outdated knowledge, leading to erroneous inferences. To mitigate this issue, Temporal KGs (TKG) make use of the temporal information,

validating each stored fact with a time-related attribute [37]. This leads to more accurately derived statements. The above presented triple format is extended to a quadruple, where a timestamp is added, e.g., <Lionel_Messi, playedFor, Barcelona_FC, 2015>.

Due to their automated generation, KGs suffer from incompleteness significantly reducing their potential usage as external or background knowledge source in downstream tasks. Thus, the task of KG Completion (KGC) emerged, with the purpose of predicting missing information based on existing facts [4]. Many solutions were developed over the years, classified by Ji et al. [18] as embedding-based, such as TransE [3], relation path reasoning like the DIVA [5] framework, reinforcement-learning path finding algorithms akin to MINEVRA [7], rule-based reasoning exemplified by KALE [14] and models such as ANALOGY [26] used in the triple classification paradigm.

Despite their success, most systems are designed for static KGC and cannot be applied to TKGs. Whether they predict future events or not, Temporal KGC methods can be classified as interpolation or extrapolation-based. The former refers to identifying trends in KGs and using them to infer missing items, while the latter focuses on extending the known information by predicting future events from historical snapshots. Wang et al. [37] present a variety of models developed throughout the years. Interpolation-based models build upon static solutions such as TTransE [21], BoxTE [31] and ChronoR [35], while extrapolation-based ones may mine logical rules like TLogic [28], or take advantage of neural networks similar to TANGO [15].

However, these methods are prone to insufficient temporal information extraction from facts or insufficient information mining of associations in relations [39]. An alternative is to use Language Models (LMs) which are designed for understanding natural language texts, thus making them suitable for KGC. Lately, LMs' competence as Temporal Knowledge Bases was acknowledged [9,41]. Some recent studies [39,32] design quadruples as an input to a PLM to leverage their ability of encoding and understanding human-written texts, while others explore various techniques of retrieving relevant past information related to the target fact and constructing prompts designed for querying and/or fine-tuning Large Language Models (LLMs) [22,25,29,38]. All of them focus on extrapolation of known information for future timestamps.

Our work investigates LMs suitability for the TKGC task using Retrieval-Augmented-Generation (RAG) [24]. We present RAGTKGC, a framework composed of two methods which can also be used as standalone. Inspired by the work of Liao et al. [25], the first component shows how a simple, yet powerful addition to their rule mining algorithm can increase the performance of LLMs, while helping PLMs to reach SoTA results. Additionally, we study the capacity of GPT 4.1[1] [33] to extend the history of past related events for each queried fact, by comprehending and utilizing mined logical rules to extract internal knowledge relevant to the given quadruple, or act as a sole information extractor without any additional guidelines.

[1] `https://openai.com/index/gpt-4-1/`

To summarize, our approach aims to answer the following research questions:
RQ1: How does history modeling affect a model's performance?
We test three different methods for extracting relevant events that may provide the necessary information for a model to correctly predict the missing object of the target quadruple. RAGTKGC aims to increase the performance of mining logical rules for underrepresented paths in a TKG.
RQ2: Does RAGTKGC retrieve supplementary relevant facts?
Thanks to its intense training schedule, the GPT 4.1 model stores an incredible amount of information, making it suitable for the role of information extractor. We test its capacity to enrich the history for a given fact.
RQ3: Can PLMs compete with LLMs for the TKGC task?
LLMs possess emergent abilities that manifest only after a certain threshold regarding the number of internal parameters. However, this also leads to greater memory usage and low-latency responses, which are not suitable in scenarios when resources are limited. Therefore, using prompts constructed with different history modeling approaches, including RAGTKGC, we compare fine tuned instances of Flan-T5-Small [6] with LLaMA2-7B [36], highlighting the ability of PLMs to challenge their larger counterparts.

The rest of the paper is structured as follows. In Sec. 2 we position our contribution in relation with related work. Sec. 3 gives the preliminary definitions providing the basis of our work while Sec. 4 describes the RAGTKGC framework. Sec. 5 presents the experimentation setup and discusses the obtained results. Finally, Sec. 6 concludes the paper and discusses future contributions.

2 Related Work

TKGC requires models to understand historical events. Existing methods formalize it into history modeling and future prediction [13]. The addition of temporality creates correlation between events, making the prediction of future tedious.

2.1 Classical Methods

TKGC methods can be classified into interpolation or extrapolation-based methods depending whether they forecast future events or not. Interpolation refers to the capacity of processing known information and inferring missing values that have not been explicitly calculated. Extrapolation focuses on deriving new information that is not already stated in existing facts.

Many interpolation-based methods were developed throughout the years [37]. Leblay et al. [21] build upon the classic TransE algorithm by substituting its scoring function with three alternatives that jointly encode entities, relationships and time in the same embedding space. Messner et al. [31] present a temporally-adapted version of BoxE, extending it with dedicated time embeddings, allowing to flexibly represent temporal information, and show that their model is fully expressive and captures a rich class of temporal inference patterns. Sadeghian

et al. [35] develop a model that associates the timestamp with the relation and treats the combination of the two as a rotation from the head to the tail entity.

Extrapolation-based techniques focus on predicting the future. Liu et al. [28] present an explainable framework that is based on temporal logical rules extracted via temporal random walks, which are directly learnt from TKGs. Han et al. [15] extend the idea of neural ordinary differential equations, by presenting a model which preserves the continuous nature of dynamic multi-relational graph data and encodes both temporal and structural information into continuous-time dynamic embeddings. Further details on these methods are described in [1].

2.2 Language Model-based Methods

Following the success of PLMs for a wide range of Natural Language Processing (NLP) tasks, subsequent studies adapted them for TKGC. Xu et al. [39] transform a series of sampled facts into PLM inputs and convert intervals between timestamps into different prompts to make coherent sentences with implicit semantic information and fine-tune a BERT [8] model with a masking strategy. Ong et al. [32] propose a novel methodology of combining language models and graph structure to enable the encoding of unseen entities and relations for TKGC, successfully applying it on RoBERTa [27].

With the advent of LLMs, Lee et al. [22] solve the TKGC task by leveraging the In-Context Learning (ICL), where task examples are added into the prompt in natural language, with GPT-NeoX-20B [2] by constructing a history of past information related to the queried fact. Specifically, they model the history by experimenting with different aspects of historical facts, such as including past quadruples that only contain either the target subject or both subject and relationship, and whether the position of the subject and/or relationship should mirror the target quadruple or may appear in any valid combinations. Liao et al. [25] uses an improved version of the TLogic [28] algorithm to mine logical rules for each relation in a given KG. Intuitively, pairs of entities may have different interactions over time, and some relations tend to be temporally and logically sequential. Additionally, they finetune an instance of LLaMA2 [36] using the LORA algorithm [17], leading to SoTA results on a variety of datasets. Sharing similarities in constructing the history with [22], Luo et al. [29] introduce the reverse logic method which aims to alleviate the shortcomings of LLMs on performing reverse inference. They design three prompts: the ordinary construction, treating the structure of backward inferences as forward ones, the text-aware one that adds the word *reverse* at the beginning of each relationship name, and the position-aware one that switches subjects with objects. Finally, Xia et al. [38] make use of the promising Chain of Thought [42] technique which requires LLMs to reason step-by-step, by incrementally querying the model with k-hop quadruples related to the target subject, ranging from one to a fixed k. They fuse the predicted results of the LLM, namely Mixtral 8x7B [19], with graph-based models to obtain more comprehensive results for predicting over TKGs.

However, the above mentioned history modeling methods efficiently operate on facts pertaining to a densely populated subgraph structure, but struggle

with the sparse counterparts. In contrast, we focus on increasing the rule mining extraction capacity of underrepresented paths in a given TKG, by starting the mining algorithm from quadruple examples that take into consideration all possible path combinations. Additionally, we propose a RAG framework that explores the suitability of a powerful LLM, namely GPT 4.1, to expand the history size by appending extra information that is not available in a specific KG. Finally, we test the capacity of PLMs to solve the TKGC task under the same settings as LLMs, which can lead to a great reduction of needed resources.

3 Preliminaries

In the following subsection, we define important concepts that will be used throughout the paper. Definitions 3 and 4 are based on definitions from [25].

Definition 1 (Temporal Knowledge Graph). *Temporal KGs store structured time-specific knowledge [37]. Let TKG = {E, R, T, Q} be a directed graph, where E, R, T represent the set of entities, relationships and timestamps. Q stands for the set of facts such as* $q = \langle s, r, o, t \rangle$*, where* $s, o \in E$ *are the subject and object,* $r \in R$ *is the relationship, and* $t \in T$ *is the timestamp which is usually represented as a single point or time interval.*

Definition 2 (Temporal Knowledge Graph Completion). *The task of TKGC focuses on predicting a missing component* $\langle s, r, o, t \rangle$*, then it can be classified as head* $\langle ?, r, o, t \rangle$*, tail* $\langle s, r, ?, t \rangle$ *or relationship* $\langle s, ?, o, t \rangle$ *prediction.*

Definition 3 (Temporal Random Walk). *A non-increasing temporal random walk W starting from s to o, where* $s, o \in E$*, is defined as a cycle of edges* $(\langle s, r_1, o, t_2 \rangle \in Q,\ \langle s, r_2, o, t_1 \rangle \in Q)$ *with* $t_2 > t_1$*. The time constraints ensure that the edges are traversed only backward in time.*

Definition 4 (Temporal Logical Rule). *Let* $\langle s_1, r_h, o_2, t_2 \rangle \leftarrow \langle s_1, r_b, o_2, t_1 \rangle$ *with* $t_2 > t_1$ *be defined as a cyclic temporal logical rule R. The left part is called rule head and* r_h *is the head relation, while the right one represents the rule body, with* r_b *being the body relation. Multiple different rule bodies can support a rule head, which can be denoted as* TR*, implying that if the rule body holds then the rule head is true for a future timestamp* t_2*. The confidence of a rule* $conf(TR)$ *is defined as dividing the rule support by the body support, where the support is the number of quadruples satisfying rule bodies or rule heads with time constraints within* TR*.*

4 Methodology

In this section, we present the history modeling algorithm and how input prompts are designed, the RAG with GPT 4.1 approach, and the process of fine tuning LMs.

4.1 History Modeling

Let $q = \langle s, r, ?, t \rangle$ be the target quadruple, where the object o is missing and needs to be predicted. Thus, to ensure that the models have enough information when solving the TKGC task, a history of past-related events, denoted as H_q, is appended to the target quadruple.

Modified Temporal Logic Rule-based Retrieval. Building on top of the algorithm provided by Liao et al. [25], we modify their approach by starting the temporal random walk from quadruples that include entities which form unique walkable paths instead of beginning from random quadruples specific for a certain relationship. The original algorithm begins the rule learning mechanism by fixing a certain relationship r_h. Then, a random quadruple $\langle s, r_h, o, t \rangle$ is sampled, followed by a temporal random walker which iteratively crawls paths (i.e., combination of edges) of different lengths from s to o. The procedure is repeated for a fixed number of walks, outputting rules with a certain level of confidence.

However, this method may not be able to mine rules, specific for a relationship, that contain underrepresented paths. For a fixed relationship and number of walks, the temporal random walker results are influenced by the available paths (relationships) of s and o. If the amount of quadruples is greater than the number of walks, the algorithm will not start from all of them, leaving out quadruples with potentially different paths than the vast majority of examples. Also, due to a random selection of the starting quadruple, there is a high chance of sampling entities sharing similarities in their structure, leading to comparable mined rules. Therefore, the obtained rules may be applicable to the vast majority of quadruples, but the problem still stands for the underrepresented paths. For example, in Figure 1, seven quadruples contain the *Consult* relationship. Five of them link entities that do not share any other interaction before the target relationship, therefore no rules can be mined. If the number of walks is set below the number of available quadruples, some of them will be left out, and due to the random selection of the starting quadruple, it is not guaranteed that any of those examples sharing past connections will be picked, leaving out a potential rule for an underrepresented path, such as before *Consult* happens, two entities might be engaged in a *RequestAid* relationship.

To mitigate this issue, we propose an improved sampling technique that takes into account all possible paths, leading to an increased number of mined rules. An example is depicted in Figure 1. We start by counting any distinct incoming and outgoing relationships for each known entity in the TKG. Next, a label is assigned to every entity, according to the relationships it's part of. Entities part of the same relationships will have matching labels. Finally, we replace entities in every quadruple with their label, and keep only one representative fact for all available combinations. Thus, for each relationship, we obtain a set of quadruples that encapsulate entities with all available path combinations, enabling the temporal rule walker to take into consideration any possible route from s to o, even the underrepresented ones.

Prompt Design. To perform any given task, the LLM input needs to be formatted in a way that is both understandable and semantically-rich. Therefore,

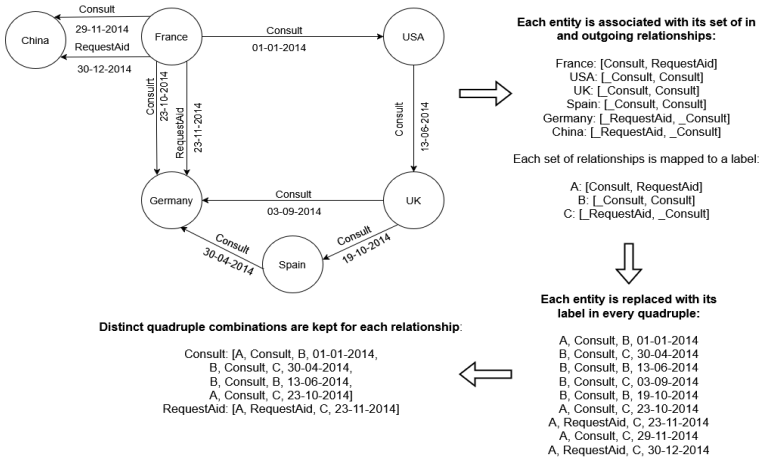

Fig. 1. An improved strategy for sampling quadruples corresponding to each relationship. Inverse relationships are preceded by "_".

Prompt	2.0: [Sudan, Consult, Ethiopia] 5.0: [Sudan, Reject, South_Sudan] 5.0: [Sudan, Praise_or_endorse, South_Sudan] 5.0: [Sudan, Meet_at_a_'third'_location, Umar_al_Bashir] 5.0: [Sudan, Express_intent_to_cooperate, South_Sudan] 6.0: [Sudan, Make_statement, South_Sudan] 6.0: [Sudan, Express_intent_to_cooperate, South_Sudan] 6.0: [Sudan, Consult, South_Sudan] 7.0: [Sudan, Express_intent_to_cooperate,
Target Output	**South_Sudan**

Table 1. Example of an input prompt and its associated target output.

a prompt is a sequence of natural language inputs that are specific for the designated assignment [34]. We build the history of events in a similar fashion to [25]. Given a target quadruple $\langle s, r, o, t \rangle$, we firstly select any other fact satisfying $\langle s, r, o, t - w \rangle$, where $w \in N+$ represents the time window length backward starting from the target timestamp. Next, the top k rules of the fixed r are sampled in descending order based on their confidence score, and append any fact conforming to them to the input prompt. The size of the retrieved history is set to n facts, chronologically ordered. In our case, history size is set to 50 for fair comparison with [25], while k equals the total number of rules for each fixed r.

The next step is to transform the obtained history into TKGC-specific format. Therefore, we follow a similar approach to [22,29], where each history fact added to the prompt is written as $t : [s, r, o]$, while the target quadruple is formulated as $t : [s, r$, and concatenated at the end of it.

To ensure a fair comparison between PLMs and LLMs, we do not add any system prompt, because the input text size is very limited for PLMs, as opposed to their larger alternatives. Table 1 depicts an example of an input prompt.

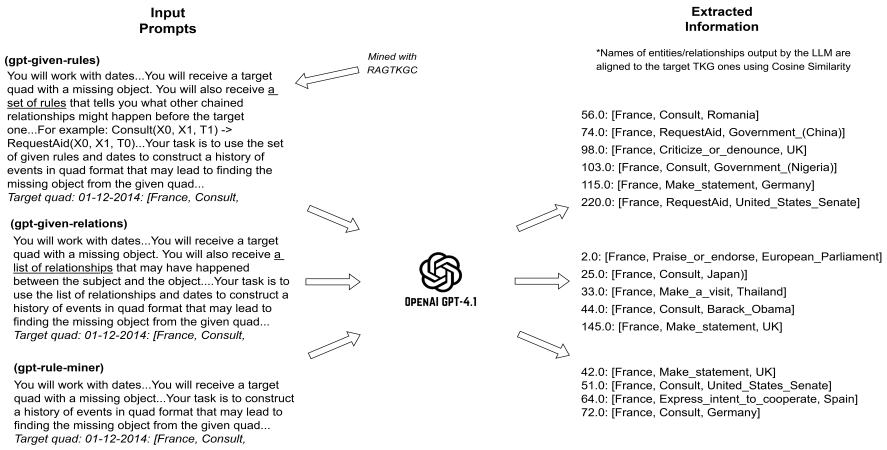

Fig. 2. Pipeline of RAGTKGC with GPT 4.1 for a target quadruple.

4.2 Retrieval Augmented Generation with GPT 4.1.

We explore the usage of GPT 4.1 as a RAG model to extend the history of known events beside what is explicitly stored in a specific TKG, with information extracted from its own knowledge. We design three distinct system prompts, the model is asked to generate additional information regarding a target quadruple by: (1) leveraging relation-specific rules mined under RAGTKGC that have a confidence score greater than a set p value, (2) using only the names of all known relationships or (3) using no additional information beside the target fact.

The first method retrieves facts missing from the KG, ensuring GPT-4.1 outputs follow a regular format. The second and third treat the LLM as a temporal rule miner without storing explicit patterns. Although outputs may include grammatical errors or unfamiliar entity/relation names, we map them to known ones using cosine similarity. Figure 2 illustrates RAGTKGC with GPT-4.1, where retrieved facts are added to the history. Datasets are labeled as *gpt-given-rules* (method 1), *gpt-given-relations* (method 2), and *gpt-rule-miner* (method 3). Example of prompts for each method, as well as the final prompts that are input to the tested models are available at our repository[2].

4.3 Fine tuning LMs

TKGC is a difficult task due to the interconnections between stored quadruples. Thus, it is of paramount importance to fine tune any tested model to ensure it is aligned with the target TKG.

Models. To facilitate our experiments, two different models are used: LLaMA2-7B [36] and Flan-T5-Small [6]. Both models are open-source, the first one is an

[2] `https://github.com/AnonOctopus/RAGTKGC`

Datasets	# Train	# Test	# Validation	# Entities	# Relations	Time Interval
ICEWS14	74845	7371	8514	7128	230	1 day
ICEWS18	373018	49545*	45995	23033	256	1 day

Table 2. Dataset statistics. * only the first 10000 samples are used in our experiments.

LLM, while the last one is an instance of a PLM. They differ in size, ranging from 77 millions (Flan-T5-Small) to 7 billions (LLaMA2-7B). LLaMA2-7B-gtkg checkpoints are directly taken from [25] github repository[3].

Procedures. Due to its size, LLaMA2-7B is trained using the LORA [17] algorithm, which greatly reduces the number of trained parameters without affecting the final performance of the model. Flan-T5-Small does not require plenty of resources, therefore we can run a full supervised fine tuning approach.

Baselines. To highlight the benefits of the history modeling algorithms, we firstly fine tune LLaMA2-7B and Flan-T5-Small with the raw instances of each dataset, which means nothing but the target quadruple is input into the model. These variants are suffixed with *raw*. For a fair comparison between our results and those of [22] and [25], we construct training and testing prompts for each dataset using both their history modeling approaches, as well as ours. Lee et al. [22] retrieve facts in a naive manner, thus it is further referred to as the *standard* approach and set the history size to 50. Liao et al. [25] propose a novel framework called GenTKG, therefore we associated it with the *gtkg* notation. Lastly, our approach is entitled *ragtkgc*.

5 Experimentation

5.1 Datasets & Metrics

Datasets. Well-known benchmarks for the TKGC task include WIKI [21], YAGO [30], GDELT [23], ICEWS14 [11] and ICEWS18 [20]. Experiments were run on the ICEWS14 and ICEWS18 datasets. Both comprise information about political events, with train/evaluation/test splits, having the timestamp level of a day. Table 2 shows statistics of the two datasets. Due to limited resources, we have only tested on the first 10000 instances from the test split of ICEWS18.

Metrics. The prediction accuracy of the tested models is measured using Hits@k, with k being either 1 or 3, meaning the target entity must be ranked among the first k generated entities. Results are reported in a time-aware filtered manner [12], indicating that if multiple entities are valid responses for a given quadruple, only the target one is kept while the rest are removed, to ensure correct ranking of it. Due to their non-deterministic nature, LMs may generate responses that resemble the target entity like synonyms or semantically-related words. Therefore, we report two additional metrics for similarity: Cosine Similarity@1 (CS@1), which measures how similar are the target entity and the first-ranked generated

[3] `https://github.com/mayhugotong/GenTKG`

prediction, based on the cosine of the angle between their vector representations, and BERTScore@1 [40], which leverages the pre-trained contextual embeddings from BERT for both the target and first-ranked generated entity, and measure their likeness by cosine similarity.

5.2 Implementation Details

Experiments were operated on two Google Colab virtual machines. The first one was used for the LLM due to its large size, and was equipped with an A100 GPU with 80GB RAM, while the other one was fitted with a T4 GPU with 50GB RAM, serving Flan-T5-Small. Both models were loaded locally from their corresponding HuggingFace endpoints.

On ICEWS14, fine tuning LLaMA2-7B models took, on average, 45 minutes to train and 185 minutes to test. Flan-T5-Small training lasted around 90 minutes and 23 minutes for testing. On ICEWS18, the LLM's training time had an approximate duration of 35 minutes, while testing took nearly 300 minutes. Fine tuning the PLM necessitated, on average, 80 minutes, and testing lasted 32 minutes. In order to obtain the GPT 4.1-generated histories, queries were sent directly to OpenAI's[4] official checkpoint. Each newly created dataset produced a cost of approximately 6.5USD.

Hyperparameters for fine tuning LLaMA2-7B were set as follows: a LORA rank of 8, LORA alpha of 16, LORA dropout rate set to 0.05, a per device batch size of 1, learning rate with a value of 3e-4, number of training epochs set to 1, and gradient accumulation steps set to 8. For training Flan-T5-Small models, per device batch size was set to [2,8], learning rate to 3e-4 with e cosine scheduler, number of training epochs to [1,3], a weight decay of 0.1, and the training precision set to bf16. Combinations differ based on the used dataset.

Each model has a pre-defined input context size, therefore, we filtered out any tokenized input prompt that exceeded one model's limit. For LLaMA2-7B, no sample was eliminated, however, for Flan-T5-Small, several ones were removed depending on the history modeling algorithm and dataset. LLaMA2-7B instances were fine tuned on 1024 equally-distributed samples from the train set. Flan-T5-Small variations used the filtered train splits, as mentioned above. LLMs predictions were decoded using a beam search strategy [10] with three distinct beams, returning top-3 possible answers. For Flan-T5-Small, we employ a permutation search, where we combine first n predicted tokens on each position and keep top-3 most probable answers.

5.3 Results

In this section, we present our main results and aim to answer the research questions as discussed in Section 1. Code and datasets, as well as guidelines and prompt examples, are provided at `https://github.com/AnonOctopus/RAGTKGC`.

[4] https://platform.openai.com/docs/models/gpt-4.1

Mining Algorithm	ICEWS14		
	Time (s)	Nr of relations with rules	Total number of rules
gtkg	16.593	413	7996
ragtkgc	1013.062	421	33972
	ICEWS18		
	Time (s)	Nr of relations with rules	Total number of rules
gtkg	30.343	465	12553
ragtkgc	6327.188	475	68709

Table 3. Statistics of the mining algorithms

How does history modeling affect a model's performance? (RQ1). The TKGC task requires the model to output a certain missing component of the input quadruple. However, it is difficult to generate the correct response when no additional context information is provided, because such facts may be the result of multiple former interactions between a variety of entities that are encapsulated in a TKG.

Firstly, Table 3 presents the metrics of our approach (ragtkgc) compared to the gtkg algorithm in terms of rule extraction efficiency. We set the number of walks to 200, for both. Although it takes significantly more time, we are able to mine approximately five times more rules, which underlines the ability of finding not only frequently occurring paths, but rare and diverse ones too. Figure 3 and 4 show further statistics on the rule mining performance differences between the two approaches. To ensure a fair comparison, we only kept metrics for those relationships that have rules mined by both algorithms, noting that our method was the one able to generate rules for more relations. Both methods follow an uptrend in terms of the average number of mined rules when an increased number of quadruples per each relationship is available, on each dataset. However, the obtained number of rules is significantly higher in favor of our approach, regardless of the available quadruple interval. On average, ragtkgc mined four times more rules than gtkg when the number of available quadruples per relation was maximum eight on ICEWS14, and 30 on ICEWS18 respectively. This proves that our method is able to discover hidden patterns even for underrepresented paths, while maintaining the same level of performance for the popular ones.

Secondly, we tested several algorithms of modeling the history of past-related facts for a given quadruple and compared it with the raw approach, where contextual data is missing. As shown in Table 4, all versions of both models achieved considerably better results than the baseline, regardless of the dataset. Compared to the standard method, both rule-mining based algorithms recorded increased results, highlighting the benefit of selecting temporally-connected facts based on specific logical rules. Notably, our approach had an average increase of 3.23% H@1 and 5.45% H@3, while also managing to overcome the gtkg method by 0.83% H@1 and 4.6% H@3. Thus, RAGTKGC allows for a more precise selection of facts that may lead to correct responses.

A further analysis of the semantically-related metrics reveals that even though all models encountered difficulties in predicting the correct target entities, their responses were semantically close to them, as CS@1 values range between 61.1%

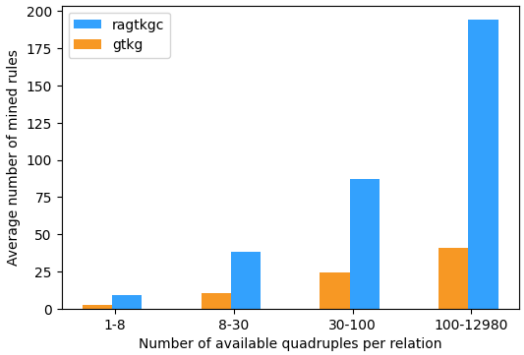

Fig. 3. Rules Mining Performance on ICEWS14

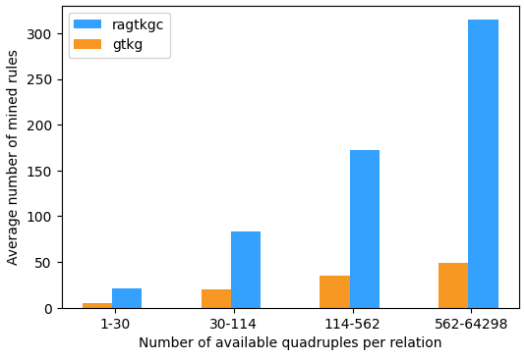

Fig. 4. Rules Mining Performance on ICEWS18.

and 69.9% on ICEWS14, and 63.1% to 65% on ICEWS18 respectively. Interesting insights are further given by the BERTScore@1 score, which has values between 90.5% to 92.3% on ICEWS14 and 88.5% to 91% on ICEWS18. Such high values reveal that both the PLM and LLM were able to leverage the given context and their internal knowledge to provide an answer that is relevant to the input quadruple, even though it is not the target one. Our approach topped both metrics, achieving an average score of 67.4% CS@1 and 91.2% BERTScore among both datasets, which is 1.8% and 0.3% more than the second best one.

Does RAG with GPT 4.1 retrieve more relevant facts? (RQ2). LLMs can serve as knowledge bases, storing more information as the number of internal parameters increases. Therefore, we selected a prominent model that was proven capable of a wide variety of tasks, namely GPT 4.1. Table 5 and 6 show the results of our proposed methodology. For each tested dataset, we have selected 40% of the most challenging target quadruples, where no version of any model was able to predict the correct output. For example, out of 7371 test instances

from the ICEWS14 dataset, a total number of 3000 samples were selected for GPT 4.1 to generate additional histories.

All tested models seem to benefit from any of the three approaches, registering increased performances compared to the non-RAG results from Table 4, apart from standard model variations on the ICEWS14 dataset. Their output is greatly affected by any of the three approaches, culminating with a top decrease of approximately 11.2% in H@1 and 12.7% in H@3, respectively. However, on ICEWS18, the same model benefits the most out of all when we apply our RAG with GPT 4.1 method. It is indeed a strange behaviour, as there were no differences in modeling the input prompt, highlighting the unpredictable behaviour of LLMs. Following conclusions are noted without these results, as they are considered outliers. Also, LLaMA2-7B-gtkg registered lower metric values on ICEWS18, however not of the same magnitude as the ones mentioned above, with only a 0.5% reduction of H@1 and 0.3% of H@3.

When asked to output facts following already mined rules, models' H@1 metrics raised with an average of 0.47%, topped by LLaMA2-7B-standard on the ICEWS18 dataset (1.2%), while the H@3 seem to have different behaviour depending on the tested dataset. On ICEWS14, most metrics were affected by this approach, with an average decrease of 0.13%, as LLaMA2-7B-ragtkgc reduced its metric by 1.2%. An in-depth analysis shows that although newly retrieved facts may add beneficial information, they might also append biased and misleading facts, thus guiding the model into choosing wrong entities. However, on the ICEWS18 dataset, this approach is rather beneficial with an average increase of 0.7%, culminating with a 2.9% raise for LLaMA2-7B-standard. Therefore, the structure of the analysed TKG is of paramount importance, as the latter set has more entities and relationships, compared to the former one, leading to more accurate mined rules. Similar results were obtained when GPT 4.1 acted without any restrictions, fulfilling the role of a logical rule miner. This may show that the LLM does not actually follow any given rule, which highlights the need for better formulated prompts. The LLM registered its best results when it was provided with the relationship names specific to the tested TKG. The H@1 met-

Datasets	ICEWS14				ICEWS18			
MODEL	H@1	H@3	CS	BertScore	H@1	H@3	CS	BertScore
Llama2-7B-raw	0.171	0.277	0.611	0.905	0.083	0.122	0.631	0.885
Llama2-7B-standard	0.313	0.436	0.692	0.913	0.121	0.196	0.641	0.89
Llama2-7B-gtkg	0.317	0.383	0.685	0.909	0.172	0.242	0.625	0.895
Llama2-7B-ragtkgc	0.338	*0.468*	0.698	0.919	0.181	*0.338*	0.649	0.896
Flan-T5-Small-raw	0.287	0.287	0.655	0.915	0.197	0.197	0.630	0.907
Flan-T5-Small-standard	0.335	0.335	0.692	0.922	0.201	0.201	0.634	0.906
Flan-T5-Small-gtkg	0.350	0.350	0.699	0.923	***0.227***	**0.227**	0.649	***0.91***
Flan-T5-Small-ragtkgc	***0.354***	**0.354**	***0.699***	***0.923***	0.226	0.226	***0.65***	0.91

Table 4. TKGC prediction results, after testing every model on the dataset version specified by their suffix. The best result for Flan-T5-Small is in **bold**, the best result for LLaMA2-7B is underlined, while the best result among all is written in *italic*.

ric increased on average with 0.71%, while H@3 followed the same behaviour as above, averaging no change in values on the ICEWS14 dataset and increased ones on ICEWS18 (0.8%). It may seem as GPT 4.1 better searches for internal knowledge when a partial ontology is provided, rather than logical rules.

Among all algorithms, the values for CS@1 and BERTScore@1 seem to not be affected by the addition of the extra information, as only the first one increased with an average of 0.12%. This is in line with the H@1 metric, as the first-ranked predictions did not change significantly. Thus, it is crucial to craft prompts in such a way that the LLM can better target pertinent information. However, regardless of the method, the average gains are low. This may show that even though GPT 4.1 can add additional context, retrieving relevant temporal information is a difficult task, as all facts must follow a given format and map their components' names to the ones already present in the TKG. A deeper analysis has shown that the LLM did retrieve facts with entities not present in the TKG, however, those cannot be used as the target is an already known one. As mentioned above, we map unrecognized nodes to existing ones based on their cosine similarity. This approach is rather limited, resulting in invalid facts. For example, in the ICEWS14 dataset, there is no entity named "USA", but similar ones like "Organization_of_American_States" or "Central_America", which may be selected as replacements, but still might not entirely refer to the same entity as GPT 4.1.

Can PLMs compete with LLMs for the TKGC task? (RQ3). Although LLMs are an excellent tool for NLP tasks, they require a vast amount of resources when loading in memory, and have a slow inference time which makes them unfit for real-time execution. Thus, we compare the performance of a PLM, namely Flan-T5-Small, which has significantly less parameters and generates faster responses, with LLaMA2-7B. Based on the results from Table 4, the smaller model has an average H@1 and H@3 scores of 27.2%, while the LLMs metrics are 21.2% and 30.8%. We can note that Flan-T5-Small correctly responded more times than the LLaMA2-7B variant, but its generated entities were rather limited at higher

Version	**gpt-given-rules**				**gpt-given-relations**				**gpt-rule-miner**			
Model	**H@1**	**H@3**	**CS@1**	**BertScore**	**H@1**	**H@3**	**CS@1**	**BertScore**	**H@1**	**H@3**	**CS@1**	**BertScore**
Llama2-7B-standard	0.209	0.302	0.639	0.893	0.208	0.3	0.636	0.891	0.208	0.303	0.642	0.892
Delta	-0.104	-0.134	-0.053	-0.01	-0.105	-0.136	-0.056	-0.012	-0.105	-0.133	-0.050	-0.011
Llama2-7B-gtkg	0.321	0.384	0.685	0.91	0.322	0.386	0.688	0.908	0.32	0.385	0.687	0.91
Delta	+0.004	+0.001	0	+0.001	+0.005	+0.003	+0.003	-0.001	+0.003	+0.002	+0.002	+0.001
Llama2-7B-ragtkgc	0.341	*0.456*	0.695	0.917	0.341	*0.455*	0.694	0.915	0.341	*0.458*	0.698	0.917
Delta	+0.003	-0.012	-0.003	-0.002	+0.003	-0.013	-0.004	-0.004	+0.003	-0.01	0	-0.002
Flan-T5-Small-standard	0.218	0.218	0.638	0.91	0.219	0.220	0.639	0.91	0.217	0.217	0.638	0.91
Delta	-0.117	-0.117	-0.054	-0.011	-0.116	-0.115	-0.053	-0.011	-0.118	-0.118	-0.054	-0.011
Flan-T5-Small-gtkg	***0.355***	**0.355**	***0.701***	***0.922***	0.358	0.358	***0.703***	0.922	***0.355***	**0.355**	***0.701***	***0.923***
Delta	+0.005	+0.005	+0.002	-0.001	+0.008	+0.008	+0.004	-0.001	+0.005	+0.005	+0.002	0
Flan-T5-Small-ragtkgc	0.355	0.355	0.7	0.922	***0.359***	**0.359**	0.702	***0.922***	0.355	0.355	0.700	0.922
Delta	+0.001	+0.001	+0.001	-0.001	+0.004	+0.004	+0.003	-0.001	+0.001	+0.001	+0.001	-0.001

Table 5. Results of TKGC on ICEWS14 dataset version that uses a specific GPT 4.1 RAG method. Delta shows the difference between this and the standard approach in Table 4 The best result for Flan-T5-Small is in **bold** and the best result for LLaMA2-7B is underlined. The best result among all is written in *italic*.

Version	gpt-given-rules				gpt-given-relations				gpt-rule-miner			
Model	**H@1**	**H@3**	**CS@1**	**BertScore**	**H@1**	**H@3**	**CS@1**	**BertScore**	**H@1**	**H@3**	**CS@1**	**BertScore**
Llama2-7B-standard	0.133	0.225	0.638	0.894	0.135	0.221	0.637	0.892	0.134	0.224	0.641	0.893
Delta	+0.012	+0.029	-0.003	+0.004	+0.014	+0.025	-0.004	+0.002	+0.013	+0.028	0	+0.003
Llama2-7B-gtkg	0.167	0.239	0.62	0.895	0.17	0.239	0.624	0.895	0.168	0.239	0.622	0.894
Delta	-0.005	-0.003	-0.005	0	-0.002	-0.003	-0.001	0	-0.004	-0.003	-0.003	-0.001
Llama2-7B-ragtkgc	0.190	*0.336*	0.649	0.898	0.194	*0.338*	0.651	0.897	0.19	*0.338*	0.651	0.897
Delta	+0.009	-0.002	0	+0.002	+0.013	0	+0.002	+0.001	+0.009	0	+0.002	+0.001
Flan-T5-Small-standard	0.211	0.211	0.638	0.908	0.213	0.213	0.64	0.908	0.212	0.212	0.641	0.908
Delta	+0.01	+0.01	+0.004	+0.002	+0.012	+0.012	+0.006	+0.002	+0.011	+0.011	+0.007	+0.002
Flan-T5-Small-gtkg	***0.234***	**0.234**	***0.652***	***0.911***	***0.238***	**0.238**	***0.655***	***0.911***	***0.236***	**0.236**	***0.656***	***0.911***
Delta	+0.007	+0.007	+0.003	+0.001	+0.011	+0.011	+0.006	+0.001	+0.009	+0.009	+0.007	+0.001
Flan-T5-Small-ragtkgc	0.227	0.227	0.651	0.91	0.229	0.229	0.653	0.91	0.228	0.228	0.653	0.91
Delta	+0.001	+0.001	+0.001	0	+0.003	+0.003	+0.003	0	+0.002	+0.002	+0.003	0

Table 6. Results of TKGC on the ICEWS18 dataset version that uses a specific GPT 4.1 RAG method. Delta shows the difference between this and the standard approach in Table 4. The best result for Flan-T5-Small is in **bold**, the best result for LLaMA2-7B is underlined, while the best result among all is written in *italic*.

ranks. However, it is arguably more efficient than its larger counterpart, and proves that PLMs, with enough data during training, can outscore bigger and more powerful models. Flan-T5-Small has also managed to pass LLaMA2-7B on CS@1 and BERTScore@1, with average values of 66.4% and 91.5%, compared to 65.4% and 90.2%. The PLM's responses are semantically closer to the target entity than those of the LLM, which proves that even if it possesses less internal knowledge it can generate answers with at least the same level of pertinence.

6 Conclusion & Future Directions

In this paper, we present RAGTKGC, a two folded framework that mines logical rules and extends the history of known facts with extra knowledge from LLMs. The first component shows how a slight change in the mining algorithm can lead to considerably more mined logical rules. This leads to improved versions of models' input prompts, that ultimately obtain higher metrics values. However, it takes significantly more time to mine and the increase in performance is not proportional with the number of obtained rules. Additionally, it is limited to rules of length one. Future work will focus on longer chains too, as well as a more efficient mining method. The second component tests the ability of using a powerful LLM, namely GPT 4.1, as an additional tool under a RAG framework that adds extra facts to a target quadruple history. Designing three different prompts, the model successfully retrieved facts that were able to increase the performance of the tested models. Despite this benefit, the performance increase is still low, while generating the appended history takes time and financial efforts. Prompts need to be refined, considering methods that raise the contextual awareness of any tested model. Finally, we show that smaller PLMs can outperform larger models on TKGC tasks, making them better suited for real-time, resource-constrained scenarios. Future directions include testing varied architectures and datasets.

References

1. Mehwish Alam, Genet Asefa Gesese, and Pierre-Henri Paris. Neurosymbolic methods for dynamic knowledge graphs. In *Handbook on Neurosymbolic AI and Knowledge Graphs*, volume 400 of *Frontiers in Artificial Intelligence and Applications*, pages 577–601. IOS Press, 2025.
2. Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. Gpt-neox-20b: An open-source autoregressive language model. *CoRR*, abs/2204.06745, 2022.
3. Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013.
4. Borui Cai, Yong Xiang, Longxiang Gao, He Zhang, Yunfeng Li, and Jianxin Li. Temporal knowledge graph completion: A survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 6545–6553. ijcai.org, 2023.
5. Wenhu Chen, Wenhan Xiong, Xifeng Yan, and William Yang Wang. Variational knowledge graph reasoning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1823–1832. Association for Computational Linguistics, 2018.
6. Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *J. Mach. Learn. Res.*, 25:70:1–70:53, 2024.
7. Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.
8. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.
9. Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. Time-aware language models as temporal knowledge bases. *Trans. Assoc. Comput. Linguistics*, 10:257–273, 2022.

10. Markus Freitag and Yaser Al-Onaizan. Beam search strategies for neural machine translation. In Thang Luong, Alexandra Birch, Graham Neubig, and Andrew M. Finch, editors, *Proceedings of the First Workshop on Neural Machine Translation, NMT@ACL 2017, Vancouver, Canada, August 4, 2017*, pages 56–60. Association for Computational Linguistics, 2017.
11. Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4816–4821. Association for Computational Linguistics, 2018.
12. Julia Gastinger, Timo Sztyler, Lokesh Sharma, Anett Schuelke, and Heiner Stuckenschmidt. Comparing apples and oranges? on the evaluation of methods for temporal knowledge graph forecasting. In Danai Koutra, Claudia Plant, Manuel Gomez Rodriguez, Elena Baralis, and Francesco Bonchi, editors, *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part III*, volume 14171 of *Lecture Notes in Computer Science*, pages 533–549. Springer, 2023.
13. Saiping Guan, Xueqi Cheng, Long Bai, Fujun Zhang, Zixuan Li, Yutao Zeng, Xiaolong Jin, and Jiafeng Guo. What is event knowledge graph: A survey. *IEEE Trans. Knowl. Data Eng.*, 35(7):7569–7589, 2023.
14. Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Jointly embedding knowledge graphs and logical rules. In Jian Su, Xavier Carreras, and Kevin Duh, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 192–202. The Association for Computational Linguistics, 2016.
15. Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8352–8364. Association for Computational Linguistics, 2021.
16. Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2022.
17. Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.
18. Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514, 2022.
19. Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet,

Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. *CoRR*, abs/2401.04088, 2024.
20. Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6669–6683, Online, November 2020. Association for Computational Linguistics.
21. Julien Leblay and Melisachew Wudage Chekol. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1771–1776. ACM, 2018.
22. Dong-Ho Lee, Kian Ahrabian, Woojeong Jin, Fred Morstatter, and Jay Pujara. Temporal knowledge graph forecasting without knowledge using in-context learning. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 544–557. Association for Computational Linguistics, 2023.
23. Kalev Leetaru and Philip A. Schrodt. Gdelt: Global data on events, location, and tone. *ISA Annual Convention*, 2013.
24. Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.
25. Ruotong Liao, Xu Jia, Yangzhe Li, Yunpu Ma, and Volker Tresp. Gentkg: Generative forecasting on temporal knowledge graph with large language models. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 4303–4317. Association for Computational Linguistics, 2024.
26. Hanxiao Liu, Yuexin Wu, and Yiming Yang. Analogical inference for multi-relational embeddings. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2168–2178. PMLR, 2017.
27. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.
28. Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 4120–4127. AAAI Press, 2022.
29. Ruilin Luo, Tianle Gu, Haoling Li, Junzhe Li, Zicheng Lin, Jiayi Li, and Yujiu Yang. Chain of history: Learning and forecasting with llms for temporal knowledge graph completion. *CoRR*, abs/2401.06072, 2024.

30. Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. YAGO3: A knowledge base from multilingual wikipedias. In *Seventh Biennial Conference on Innovative Data Systems Research, CIDR 2015, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org, 2015.
31. Johannes Messner, Ralph Abboud, and İsmail İlkan Ceylan. Temporal knowledge graph completion using box embeddings. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 7779–7787. AAAI Press, 2022.
32. Ryan Ong, Jiahao Sun, Yi-Ke Guo, and Ovidiu Serban. Dynamic link prediction: Using language models and graph structures for temporal knowledge graph completion with emerging entities and relations. *Expert Syst. Appl.*, 272:126648, 2025.
33. OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.
34. Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Trans. Knowl. Data Eng.*, 36(7):3580–3599, 2024.
35. Ali Sadeghian, Mohammadreza Armandpour, Anthony M. Colas, and Daisy Zhe Wang. Chronor: Rotation based temporal knowledge graph embedding. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6471–6479. AAAI Press, 2021.
36. Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.
37. Jiapu Wang, Boyue Wang, Meikang Qiu, Shirui Pan, Bo Xiong, Heng Liu, Linhao Luo, Tengfei Liu, Yongli Hu, Baocai Yin, and Wen Gao. A survey on temporal knowledge graph completion: Taxonomy, progress, and prospects. *CoRR*, abs/2308.02457, 2023.
38. Yuwei Xia, Ding Wang, Qiang Liu, Liang Wang, Shu Wu, and Xiaoyu Zhang. Chain-of-history reasoning for temporal knowledge graph forecasting. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 16144–16159. Association for Computational Linguistics, 2024.
39. Wenjie Xu, Ben Liu, Miao Peng, Xu Jia, and Min Peng. Pre-trained language model with prompts for temporal knowledge graph completion. In Anna Rogers,

Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 7790–7803. Association for Computational Linguistics, 2023.

40. Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
41. Ruilin Zhao, Feng Zhao, Guandong Xu, Sixiao Zhang, and Hai Jin. Can language models serve as temporal knowledge bases? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 2024–2037. Association for Computational Linguistics, 2022.
42. Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models, 2025.