Self-Exploring Language Models for Explainable Link Forecasting on Temporal Graphs via Reinforcement Learning

Zifeng Ding[1]* **Shenyang Huang**[2,3,4]*
Zeyu Cao[1] **Emma Kondrup**[2,3] **Zachary Yang**[2,3] **Xingyue Huang**[4] **Yuan Sui**[5]
Zhangdie Yuan[1] **Yuqicheng Zhu**[6] **Xianglong Hu**[7] **Yuan He**[7]
Farimah Poursafaei[2,3] **Michael Bronstein**[4,8] **Andreas Vlachos**[1]†
[1]University of Cambridge, [2]Mila - Quebec AI Institute,
[3]McGill University, [4]University of Oxford,
[5]National University of Singapore, [6]University of Stuttgart, [7]Amazon, [8]AITHYRA
`{zd320, av308}@cam.ac.uk`

Abstract

Forecasting future links is a central task in temporal graph (TG) reasoning, requiring models to leverage historical interactions to predict upcoming ones. Traditional neural approaches, such as temporal graph neural networks, achieve strong performance but lack explainability and cannot be applied to unseen graphs without retraining. Recent studies have begun to explore using large language models (LLMs) for graph reasoning, but most of them are constrained to static graphs or small synthetic TGs and lack the evaluation of the quality of reasoning traces generated by LLMs. In this work, we present **Rea**soning-Enhanced **L**earning for **T**emporal **G**raphs (ReaL-TG), a reinforcement learning framework that fine-tunes LLMs to perform explainable link forecasting on real-world TGs. ReaL-TG uses outcome-based reward to encourage models to self-explore reasoning strategies from graph structure and to produce explanations that directly justify their predictions. To enable evaluation on LLM-generated reasoning traces, we propose a new evaluation protocol combining ranking metrics with an LLM-as-a-Judge system that assesses both the quality of reasoning and the impact of hallucinations. Experiments with ReaL-TG-4B, obtained by fine-tuning Qwen3-4B under our framework, show that it outperforms much larger frontier LLMs, including GPT-5 mini, on ranking metrics, while producing high-quality explanations confirmed by both the LLM judge and human evaluation.

1 Introduction

Temporal graphs (TGs) represent node interactions as links annotated with timestamps [22], making them well-suited for modeling a wide range of real-world scenarios such as social and transaction networks [19]. This expressiveness has fueled the growing interest in TG reasoning, which focuses on capturing the dynamic graphical structures within TGs to support various downstream tasks. A widely studied task in TG reasoning is future link prediction, also known as link forecasting. It aims to predict future interactions between nodes based on historical node interactions, which is particularly useful in practical applications such as recommendation systems [9], community discovery [35] and

*Equal contribution.
†Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Graph Machine Learning.

financial analysis [37]. Mainstream methods for link forecasting train neural-based models such as temporal graph neural networks (TGNNs) [52, 31, 47, 13], memory networks [36, 29], and sequence modeling units [55, 42, 6] on the training set of a TG, and then apply the trained model to the test set of the same TG. While effective, they suffer from two key limitations. First, most neural-based models lack human-readable explanations for their predictions, making it difficult for users to assess the trustworthiness of the results. Second, they typically require retraining when adapted to a new TG, and therefore cannot seamlessly generalize to unseen graphs.

Recently, the rapid scaling of language models has made them increasingly effective at generating coherent text, leading to their widespread adoption in question answering (QA) tasks across diverse domains. Building on this progress, an emerging line of research investigates whether large language models (LLMs) can also reason over graphs by prompting them to answer graph-related (such as link prediction) questions. Compared with traditional graph reasoning methods, LLMs naturally provide human-readable explanations and exhibit strong zero-shot generalization, suggesting the potential to handle previously unseen graphs without retraining. Nevertheless, most existing studies focus on static graphs [1, 32, 10, 2, 15], and only a few have investigated TGs. Among these, several efforts focus on TGs with textual attributes and demonstrate strong performance [25, 28, 45, 50]. However, such settings carry a risk of data leakage, since textual features—including those directly relevant to prediction and even the correct answers to the questions—may already have been seen during pre-training [5]. In contrast, LLM4DyG [57] evaluates LLMs on TG reasoning using fully synthetic graphs anonymized from text, thereby avoiding leakage. However, its experiments are restricted to very small scales (up to 20 nodes), limiting the applicability of the findings to realistic scenarios. Moreover, existing studies largely overlook the evaluation of LLMs' reasoning outputs. Strong performance on link prediction metrics such as accuracy does not necessarily imply that the underlying reasoning traces are correct. In practice, LLMs may generate flawed reasoning or introduce hallucinations that still lead to the right prediction label, raising concerns about their reliability.

Building on these observations, we propose **Rea**soning-Enhanced **L**earning for **T**emporal **G**raphs (ReaL-TG), a reinforcement learning (RL) framework that fine-tunes LLMs to do perform link forecasting over TGs. Unlike prior works that rely on textual attributes or synthetic toy datasets, Real-TG is developed and evaluated on anonymized real-world TGs (where nodes are represented with numerical IDs without any semantic information) provided by the popular Temporal Graph Benchmark (TGB) [19], making it both practical and aligned with real application needs. By removing semantic information from textual attributes, anonymized graphs prevent data leakage and require the model to reason solely over TG topology, leading to reasoning patterns focusing on the intrinsic dynamics of TG evolution. During RL, we choose Qwen3 [53] as the base model and adopt Grouped Regularized Policy Optimization (GRPO) [38] together with an outcome-based reward tailored to TG link forecasting. This setup not only encourages the model to self-explore reasoning strategies through its own textual outputs without process-level supervision, but also compels it to produce human-readable explanations justifying its predictions. In this way, the model is pushed to achieve both strong predictive accuracy and sound reasoning that supports its answers. To comprehensively evaluate LLMs in TG link forecasting, we further propose a new evaluation protocol tailored to this setting. First, we formulate the task as QA, where an LLM must directly generate the set of nodes it predicts as correct answers. On top of this formulation, we introduce penalized mean reciprocal rank (pMRR), an extension of MRR [44] that discounts the score when predicted nodes fall outside the ground-truth set, thereby discouraging over-generation. Second, to assess the quality of LLM-generated reasoning traces , we design an LLM-as-a-Judge [58] evaluation focusing on three criteria: faithfulness, logical consistency and answer–explanation alignment.

We summarize our contributions as follows: (i) we propose ReaL-TG, the first framework that enables LLMs to perform explainable and effective link forecasting on real-world TGs via RL; (ii) we introduce a new evaluation protocol for TG link forecasting with LLMs that assesses not only prediction accuracy but also reasoning quality and the impact of hallucinations; (iii) our fine-tuned model ReaL-TG-4B outperforms much larger frontier LLMs, including GPT-5 mini, on both seen and unseen graphs. In addition, it produces high-quality explanations.

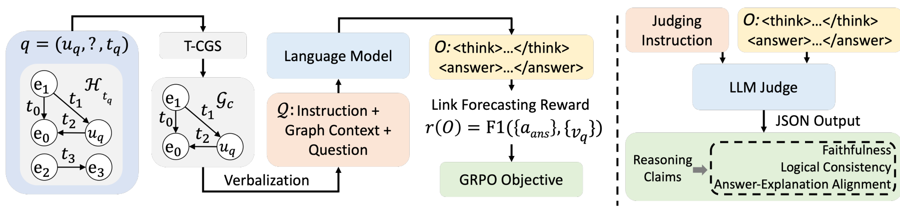

Figure 1: Left: The ReaL-TG framework, which enables RL fine-tuning of LLMs to improve TG forecasting (see Sec. 3). Right: The proposed LLM-as-a-Judge system, which provides a comprehensive evaluation of LLM reasoning quality in TG link forecasting (see Sec. 4).

2 Preliminaries

A detailed discussion of related works is provided in App. D. We first define TG as follows. Note that, in this work, we deliberately exclude node and edge features, focusing instead on how LLMs can reason over TGs solely from their topological structure.

Definition 1 (Temporal Graph) *Let $\mathcal{N}$ and $\mathcal{T}$ denote a set of nodes and timestamps, respectively. A TG can be represented as a sequence of $|\mathcal{G}|$ chronological interactions $\mathcal{G} = \{(u_i, v_i, t_i)\}_{i=1}^{|\mathcal{G}|} \subseteq \mathcal{N} \times \mathcal{N} \times \mathcal{T}$ with $0 \leq t_1 \leq t_2 \leq ... \leq t_{|\mathcal{G}|}$, where $u_i, v_i \in \mathcal{N}$ are the source and destination node of the i-th interaction happening at $t_i \in \mathcal{T}$, respectively.*

Inspired by Huang et al. [18], we then define TG link forecasting as a QA task, making it naturally adaptable to LLMs. We discuss the advantages of this formulation over the traditional one in App. F.

Definition 2 (TG Link Forecasting with LLMs) *Assume a TG $\mathcal{G} \subseteq \mathcal{N} \times \mathcal{N} \times \mathcal{T}$ containing all ground-truth interactions, and let $f(\cdot)$ denote the inference process of an LLM. Given a prediction query $q = (u_q, ?, t_q)$ with source node $u_q \in \mathcal{N}$ and timestamp $t_q \in \mathcal{T}$, together with its history $\mathcal{H}_{t_q} = \{(u_i, v_i, t_i) \mid t_i < t_q, (u_i, v_i, t_i) \in \mathcal{G}\}$, TG link forecasting requires the model to produce a text-based answer A specifying the ground truth missing node(s) $v_q \subseteq \mathcal{N}$ as the predicted missing destination(s). The answer is obtained by $A = f\big(\psi(\mathcal{H}_{t_q}, q)\big)$, where $\psi(\cdot, \cdot)$ is a function that converts $\mathcal{H}_{t_q}$ and q into a prompt consisting of historical graph context and a natural language question asking about the missing destination node(s).*

3 ReaL-TG

The left part of Fig. 1 illustrates our ReaL-TG framework. Given a query $q = (u_q, ?, t_q)$ and its history $\mathcal{H}_{t_q}$ before query timestamp t_q, we first apply the Temporal Context Graph Selection (T-CGS) algorithm to construct a subgraph $\mathcal{G}_c$ that is most relevant to q based on $\mathcal{H}_{t_q}$. $\mathcal{G}_c$ serves as the graph context from which the LLM extracts information to make predictions. We then verbalize all links in $\mathcal{G}_c$ and combine them with a natural language question derived from q into a prompt template, denoted as $\mathcal{Q}$. The prompt $\mathcal{Q}$ is fed into an LLM for inference, from which we extract the prediction answer. We compute a link forecasting reward for each prediction with a customized reward function, and through RL the model self-explores reasoning patterns to improve forecasting in TGs.

Temporal Context Graph Selection. We input graph context as text into the LLM to ensure explainability, since we require the output reasoning trace to explicitly justify predictions in natural language (see App. G for further discussion on why we represent graph context as text). We aim to include as much relevant graph information as possible while excluding redundant details that do not contribute to prediction. To this end, we propose T-CGS, an algorithm that selects a temporal context graph for each query $q = (u_q, ?, t_q)$. Inspired by Li et al. [27], we construct $\mathcal{G}_c$ centered around a temporal query node (u_q, t_q). Starting from (u_q, t_q), we perform an α-temporal random walk, where at each step the walk terminates at the current temporal node (e, t) with probability $\alpha \in (0, 1)$, and with probability $1 - \alpha$ it continues to a node in the historical temporal neighborhood $Nei_{(e,t)} = (e', t') \mid (e, e', t')$ or $(e', e, t') \in \mathcal{H}_t, t' < t$ of (e, t). If the walk continues, the transition probability from (e, t) to each $(e', t') \in Nei_{(e,t)}$ is given by $P_{(e,t)}(e', t') = \beta^{|\{(e'', t'') | (e'', t'') \in Nei_{(e,t)}, t'' \geq t'\}|} / \sum_{z=1}^{|Nei_{(e,t)}|} \beta^z$, where $\beta \in (0, 1)$ is a decay factor.

The intuition behind it is to assign higher transition probabilities to temporal neighbors that are closer in time to the current node (e, t), since recent interactions are generally more influential in information propagation on TGs, as shown in prior works [30, 7, 27]. Based on this setting, we compute the probability of an α-temporal random walk starting from the query node (u_q, t_q) and terminating at one of its k-hop historical neighbors. We then rank all visited temporal nodes by their termination probabilities and select the top-ranked nodes $\mathcal{N}_q$ as the most relevant for answering query q. To construct the context graph $\mathcal{G}_c$, we retrieve all links in the ground-truth graph that involve nodes in $\mathcal{N}q$ and collect them into $\mathcal{G}_c$. In practice, we set $|\mathcal{N}q|$ to 100 and limit the random walk to at most 2 steps, yielding a $\mathcal{G}_c$ that contains temporal neighbors of (u_q, t_q) up to 3 hops away.

Prompt Construction. Given $\mathcal{G}_c$ and query q, we construct the prompt $\mathcal{Q}$ shown in Fig. 2, which embeds the graph context and instructs the LLM to produce both predictions and explanatory reasoning traces. To facilitate extraction, we require the reasoning to be enclosed within <think> </think> tags and the final predictions within <answer> </answer> tags.

Training Data Collection. We collect 1,000 link forecasting queries from 4 TGB datasets: `tgbl-wiki`, `tgbl-subreddit`, `tgbl-coin`, and `tgbl-flight` to construct the training data. Since each query $(u_q, ?, t_q)$ may have multiple ground-truth nodes as answers, the total number of involved links is larger than 1,000. Specifically, we sample 225 queries each from `tgbl-wiki` and `tgbl-subreddit`, and 275 queries each from `tgbl-coin` and `tgbl-flight`. The latter two datasets are empirically shown to be more challenging in the original TGB benchmark [19], so we allocate more training examples to them. For all datasets, queries are sampled in reverse chronological order from the last training timestamp until the desired size is reached, ensuring richer histories for constructing temporal context graphs. We skip queries where (i) the T-CGS–selected temporal context graph does not contain all ground-truth answers or (ii) the temporal context graph exceeds 600 links. This avoids cases where the LLM cannot observe the answer within its prompt, making fine-tuning meaningless, or where the temporal context graph is so large that it consumes most of the context window, leaving limited space for reasoning. Finally, for each query we construct a $\mathcal{Q}$ prompt and pair it with its ground-truth missing nodes $\{v_q\}$ to form a training example.

Fine-tuning LLMs with RL. We use GRPO with a customized reward to fine-tune models. For each query $(u_q, ?, t_q)$ with a set of ground-truth missing nodes $\{v_q\}$, the LLM aims to predict as many ground-truths as possible without introducing spurious nodes. To achieve this, we design a reward function based on the F1 score, balancing precision (whether all predicted nodes are correct) and recall (whether all ground-truth nodes are retrieved). Specifically, let the contents between <answer </answer> tags in the LLM output O be denoted as $A_{<ans>}$. We parse $A_{<ans>}$ into a set $A = \{a_{<ans>}\}$ of predicted nodes and compute a *link forecasting reward* as $r(O) = \text{F1}(\{a_{<ans>}\}, \{v_q\})$. This reward depends solely on model outputs, encouraging LLMs to discover transferable reasoning patterns across graphs without constraining their reasoning traces. Moreover, it is non-parametric, requiring no additional cost for training a separate reward model. Given the reward, we update model parameters by maximizing the GRPO objective. We provide more details in App. H.

4 Evaluation Protocol

Prediction Label Evaluation. We first follow [19] to evaluate models with Mean Reciprocal Rank (MRR). Assume we have M evaluation examples, each consisting of a prompt Q_m, a query $(u_{q_m}, ?, t_{q_m})$, and a ground-truth set $\eta_m^{\text{gt}} = \{v_{q_m}\}$. The corresponding prediction set is $\eta_m^{\text{pred}} = \{v'_{q_m}\}$, which contains all nodes the LLM predicts as belonging to η_m^{gt}. We compute MRR as $\text{MRR} = \frac{1}{\sum_{m=1}^{M} \eta_m^{\text{gt}}} \sum_{m=1}^{M} \sum_{s=1}^{\eta_m^{\text{gt}}} \frac{1}{\text{rank}_m^s}$. rank_m^s denotes the rank of the s-th node in η_m^{gt}. The ranking is computed as follows. We first assign a score of 0 to all nodes in the dataset, and then set the score to 1 for nodes included in η_m^{pred}. Following prior works [16, 12], we use filtered MRR, where the influence of other correctly predicted nodes is excluded by resetting their scores to 0 when evaluating a given node. Finally, for each node we compute the mean of its optimistic rank (treating equally scored nodes as ranked lower) and pessimistic rank (treating them as ranked higher), which gives rank_m^s. Although MRR is a widely used and robust metric for evaluating link prediction, it does not capture the risk of *over-generation* in LLMs when the task is framed as QA-style generation. During reasoning, LLMs often predict all nodes they believe belong to η_m^{gt}, sometimes accompanied by supporting reasoning. While not always undesirable, this behavior can be problematic when accurate link forecasting is required. To better capture the over-generation phenomenon, we introduce penalized MRR (pMRR), which follows MRR's definition but slightly modifies the computation of rank_m^s. Specifically, for all nodes in $\eta_m^{\text{pred}} \setminus \eta_m^{\text{gt}}$, we assign a score of 1.1 instead of 1. This ensures that incorrectly predicted nodes are ranked above correctly predicted ones, thereby penalizing over-generation. The more such nodes appear, the stronger the penalty, resulting in a lower pMRR.

Reasoning Trace Evaluation. LLMs naturally benefit from their text generation ability, making them well-suited for explainable link forecasting. However, no prior work has systematically evaluated their reasoning traces, i.e., how prediction labels are derived. Such evaluation is crucial because a trustworthy forecaster should not only produce accurate predictions but also provide reasonable justifications. Moreover, predictions outside the ground truth are not always undesirable if they are supported by strong reasoning. In real-world forecasting, ground-truth labels are unavailable before *events actually occur*, unlike in experimental setups where metrics such as MRR can be computed. This makes the evaluation of an LLM forecaster's reasoning quality even more important. The most reliable way to assess LLM reasoning is to do human evaluation, however, it is not scalable. Motivated by the recent success of LLM-as-a-Judge [58], we adopt this approach for quicker and more scalable assessment, focusing on three criteria: faithfulness, logical consistency, and answer–explanation alignment. For faithfulness, we evaluate whether the LLM's reasoning is supported by the input context graph $\mathcal{G}_c$. The Judge first splits a reasoning trace into a series of atomic claims, each describing some aspect of the graph context. It then determines the proportion of claims that are faithful to $\mathcal{G}_c$, i.e., contain no factual errors in describing it. This proportion is defined as the faithfulness score δ_f. For logical consistency, we assess whether the reasoning follows a coherent and valid chain. Here, the Judge disregards faithfulness and focuses solely on whether the LLM's reasoning proceeds in a logically sound manner without self-contradiction. The Judge assigns a score from $\{0, 1, 2\}$, with higher values indicating better consistency. This score is then normalized to $[0, 1]$ and defined as the consistency score δ_{lc}. For answer–explanation alignment, we assess whether the predicted answers are justified by the model's own reasoning. A predicted node is considered justified if (i) the reasoning trace contains explicit supporting claims for it, and (ii) those claims are judged as faithful in the faithfulness evaluation. We define the alignment score δ_a as the proportion of predicted nodes that are well-justified. From another perspective, these three scores can also be viewed as capturing the impact of different types of hallucinations in LLM reasoning. δ_f targets factual hallucinations, where the model introduces hallucinated claims about the context graph. δ_{lc} addresses logical inconsistency hallucinations, where the model produces contradictory or incoherent logic chains. δ_a reflects justification hallucinations, where predictions are made without being grounded in faithful reasoning. By jointly evaluating these dimensions, our system provides a more comprehensive assessment of LLMs' reasoning quality in explainable link forecasting. We use GPT-4.1 mini as Judge. See Fig. 3 for the complete prompt, i.e., instruction, for Judge. See the right part of Fig. 1 for an illustration of the system. We compute the aggregated scores $\overline{\delta_f}$, $\overline{\delta_{lc}}$, and $\overline{\delta_a}$ by averaging over all evaluation examples, providing an overall measure of reasoning quality.

5 Experiments

We fine-tune a Qwen3-4B with ReaL-TG and name it ReaL-TG-4B. We compare it with baselines on both seen and unseen graphs using our proposed evaluation protocol. We report the comparative results of prediction accuracy and reasoning quality among LLMs, along with a performance comparison between ReaL-TG-4B and traditional TG link forecasting methods as well as the human evaluation of reasoning traces generated by ReaL-TG-4B. Additional results are provided in appendices: (i) human evaluation of our LLM-as-a-Judge system (App.L); and (ii) qualitative analysis with two case studies illustrating how RL improves LLM-based link forecasting (App.M)

Experimental Setup. We collect evaluation data from the test sets of 4 TGB datasets used during training and from the test sets of 2 unseen datasets (`tgbl-uci`, `tgbl-enron`) to assess models' transferability to unseen graphs To control evaluation cost, we curate a moderately sized dataset for assessing LLMs in TG link forecasting. We first select the last 1,000 queries from each of the 6 TGB datasets in reverse chronological order, ensuring that test data are accompanied by abundant historical information. For each query, we then extract the temporal context graph using T-CGS. Finally, we filter out queries following the same principles adopted in query skipping when we construct training

Table 1: Results on prediction label. We omit the `tgbl` prefix in dataset names. The top two results are highlighted by **first** and <u>second</u>. We use pMRR to represent penalized MRR.

Dataset	Seen								Unseen				Combined	
	`wiki`		`subreddit`		`coin`		`flight`		`uci`		`enron`		Overall	
Model	MRR	pMRR	MRR	pMRR	MRR	pMRR	MRR	pMRR	MRR	pMRR	MRR	pMRR	MRR	pMRR
Qwen3-0.6B	0.338	0.331	0.245	0.238	0.111	0.107	0.121	0.111	0.114	0.108	0.089	0.084	0.171	0.164
Qwen3-4B	0.721	0.682	0.678	0.639	0.368	0.333	0.090	0.072	0.300	0.239	0.174	0.137	0.375	0.339
Qwen3-8B	0.763	0.721	<u>0.731</u>	<u>0.688</u>	<u>0.380</u>	<u>0.343</u>	0.109	0.087	0.364	0.293	0.300	0.243	0.436	0.391
Gemma 3 4B	0.698	0.673	0.686	0.650	0.290	0.235	0.159	0.121	0.328	0.268	0.274	0.223	0.407	0.364
Gemma 3 12B	<u>0.782</u>	<u>0.738</u>	0.718	0.671	0.376	0.302	<u>0.315</u>	**0.249**	0.390	0.298	<u>0.469</u>	<u>0.381</u>	0.520	<u>0.452</u>
GPT-5 mini	0.714	0.630	0.674	0.596	0.288	0.201	0.286	0.180	0.355	0.266	0.333	0.215	0.456	0.351
Llama3.3-70B	0.759	0.687	0.716	0.644	0.372	0.257	**0.323**	<u>0.245</u>	<u>0.422</u>	<u>0.347</u>	0.441	0.328	<u>0.521</u>	0.423
ReaL-TG 4B	**0.824**	**0.792**	**0.765**	**0.726**	**0.431**	**0.401**	0.198	0.175	**0.607**	**0.523**	**0.492**	**0.435**	**0.552**	**0.508**

data and get in total 4,246 evaluation data. We provide the evaluation data statistics in Table 4. For baselines, we evaluate several frontier models, including non-reasoning models (Gemma 3 4B/12B, Llama 3.3 70B) and reasoning models (Qwen3-0.6B/4B/8B, GPT5-mini). All models are tested with the same prompts for fair comparison. For non-reasoning models, we use greedy decoding, while reasoning models are run with their default configurations. See App. I for implementation details.

Results: Prediction Accuracy. We report the results of MRR and pMRR in Table 1. Our main findings are as follows: (i) within the same model family (e.g., Qwen3-0.6B/4B/8B), larger model size generally leads to better performance on TG link forecasting; (ii) larger LLMs tend to predict more nodes as answers (with larger difference between MRR and pMRR), likely because their stronger capacity allows them to consider more candidate predictions, although this behavior is not always beneficial for link forecasting; (iii) ReaL-TG-4B outperforms all baselines, including GPT-5 mini and Llama 3.3 70B, across nearly all datasets on both seen and unseen graphs, demonstrating the effectiveness of the ReaL-TG framework. Although ReaL-TG-4B trails some baselines on `tgbl-flight`, we attribute this to the limitations of its base model Qwen3-4B on this dataset; (iv) ReaL-TG-4B achieves substantial gains over its base model, confirming the effectiveness of our RL-based training framework. To supplement, we also provide a comparison among ReaL-TG-4B and several traditional link forecasting methods in App. 5.

Results: Reasoning Quality. We report the reasoning evaluation results in Table 2. The comparison includes Qwen3-4B/8B, the Gemma 3 family, and Llama 3.3-70B. We exclude GPT-5 mini for two reasons: (i) our Judge is GPT-4.1 mini, which may introduce family-bias [41], i.e., assigning higher judgment scores to other OpenAI models; and (ii) the GPT-5 series restricts access to full reasoning traces, providing only a summary of its reasoning, which prevents accurate evaluation of its actual reasoning behavior. We summarize our key findings as follows: (i) within the same model family, larger models are more robust to hallucinations and achieve higher reasoning quality, suggesting a correlation between prediction accuracy and reasoning quality. (ii) ReaL-TG-4B demonstrates substantial improvements over its base model Qwen3-4B in reasoning quality, validating the effectiveness of RL fine-tuning and showing that the ReaL-TG framework enables LLMs to discover meaningful reasoning patterns useful for TG link forecasting. (iii) despite these gains, ReaL-TG-4B lags behind larger models in logical consistency and answer–explanation alignment. We attribute this to the natural advantage of larger models in producing more robust reasoning traces, particularly in providing consistent logic and sufficient supporting evidence for predictions. This indicates that applying ReaL-TG to larger base models would be a promising direction in the future. We give a more detailed discussion to show how the reasoning quality is influenced by base model size and ReaL-TG in App. K. We also give a human evaluation on the quality of our LLM-based judging system in App. L.

Table 2: Results on the quality of reasoning traces.

Model	$\bar{\delta_f}$	$\bar{\delta_{lc}}$	$\bar{\delta_a}$
Qwen3-4B	0.683	0.700	0.653
Qwen3-8B	0.792	0.808	0.770
Gemma 3 4B	0.595	0.666	0.558
Gemma 3 12B	0.867	0.928	0.771
Llama 3.3 70B	0.878	0.950	0.820
ReaL-TG-4B	0.885	0.880	0.732

ReaL-TG-4B vs. Traditional TG Link Forecasting Methods. Table 3 reports results of 3 strong TGNNs: TGN [36], DyGFormer [55] and TNCN [56], together with the widely used EdgeBank baseline [33]. We train TGNNs separately on the original training set of each involved dataset on TGB with their default implementation settings and evaluate all models using MRR. TGNNs formulate TG link forecasting as a binary classification task, where models are trained to decide whether a potential link exists, which makes ranking metrics computationally expensive since obtaining a rank requires a

Table 3: MRR comparison among ReaL-TG-4B and traditional TG link forecasting methods.

Dataset	wiki	subreddit	coin	flight	uci	enron
EdgeBank	0.425	0.271	0.153	0.179	0.202	0.129
TGN	0.464	0.698	Timeout	Timeout	0.050	0.281
DyGFormer	**0.847**	0.659	Timeout	Timeout	0.011	0.341
TNCN	0.732	0.739	Timeout	Timeout	0.049	0.263
ReaL-TG 4B	0.824	**0.765**	**0.431**	**0.198**	**0.607**	**0.492**

forward pass over every node in the node set (see App. F for details). Besides, it is impossible to evaluate binary classification-based TGNNs with pMRR because they do not return node IDs directly as answers. To avoid excessive cost, we control the budget for evaluation with a timeout constraint of 24 hours. Note that for ReaL-TG-4B, `tgbl-uci` and `tgbl-enron` are treated as unseen graphs, whereas for TGNNs, they are trained exclusively on these datasets and are therefore considered seen graphs. Our results show that the fine-tuned model outperforms strong traditional methods while providing explicit reasoning to justify its predictions, demonstrating strong potential. Moreover, by formulating TG link forecasting as QA, our framework enables low-cost prediction in real-world applications and eliminates the need to train a model from scratch for new TGs.

Human Evaluation: Quality of Reasoning Traces. We recruit 5 annotators to evaluate the quality of reasoning traces generated by ReaL-TG-4B. A random sample of 50 data examples is selected, and annotators provide judgment scores for the three criteria following the same instructions given to the LLM judge. Averaging their annotations yields high scores of 0.885/0.872/0.839 for $\bar{\delta_f}/\bar{\delta_{lc}}/\bar{\delta_a}$ (maximum score 1), which closely align with the judge's scores of 0.909/0.890/0.787. This strong correlation validates our LLM-as-a-Judge system and also demonstrates the substantial reasoning capability gained through ReaL-TG fine-tuning. Further annotation details are provided in App. L.

6 Conclusion

In summary, we present ReaL-TG, the first RL-based framework that enables LLMs to perform explainable and effective link forecasting on TGs. We further introduce a new evaluation protocol, featuring a new automated ranking metric coupled with a dedicated LLM-as-a-Judge system. Our experiments show that ReaL-TG allows LLMs to self-explore reasoning strategies for TG link forecasting, achieving improvements both in prediction accuracy and in generating well-grounded reasoning traces. We also conduct human evaluation of both the LLM-as-a-Judge system and the fine-tuned model, validating the effectiveness of our framework and evaluation methodology.

Acknowledgment

Zifeng Ding, Zhangdie Yuan and Andreas Vlachos are supported by the ERC grant AVeriTeC (GA 865958). Andreas Vlachos is also supported by the DARPA program SciFy. Shenyang Huang is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) Postgraduate Scholarship Doctoral (PGS D) Award and Fonds de recherche du Québec - Nature et Technologies (FRQNT) Doctoral Award. This research is supported by the Canadian Institute for Advanced Research (CIFAR AI chair program), the EPSRC Turing AI World-Leading Research Fellowship No. EP/X040062/1 and EPSRC AI Hub No. EP/Y028872/1. All authors thank Zeyu Cao for providing computational resources.

References

[1] Z. Chai, T. Zhang, L. Wu, K. Han, X. Hu, X. Huang, and Y. Yang. Graphllm: Boosting graph reasoning ability of large language model. *CoRR*, abs/2310.05845, 2023.

[2] R. Chen, T. Zhao, A. K. Jaiswal, N. Shah, and Z. Wang. Llaga: Large language and graph assistant. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[3] Z. Chu, J. Chen, Q. Chen, W. Yu, H. Wang, M. Liu, and B. Qin. Timebench: A comprehensive evaluation of temporal reasoning abilities in large language models. In L. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 1204–1228. Association for Computational Linguistics, 2024.

[4] W. Cong, S. Zhang, J. Kang, B. Yuan, H. Wu, X. Zhou, H. Tong, and M. Mahdavi. Do we really need complicated model architectures for temporal networks? In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[5] Z. Ding, H. Cai, J. Wu, Y. Ma, R. Liao, B. Xiong, and V. Tresp. zrllm: Zero-shot relational learning on temporal knowledge graphs with large language models. In K. Duh, H. Gómez-Adorno, and S. Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 1877–1895. Association for Computational Linguistics, 2024.

[6] Z. Ding, Y. Li, Y. He, A. Norelli, J. Wu, V. Tresp, M. M. Bronstein, and Y. Ma. DyGMamba: Efficiently modeling long-term temporal dependency on continuous-time dynamic graphs with state space models. *Transactions on Machine Learning Research*, 2025.

[7] Z. Ding, Y. Ma, B. He, Z. Han, and V. Tresp. A simple but powerful graph encoder for temporal knowledge graph completion. In *NeurIPS 2022 Temporal Graph Learning Workshop*, 2022.

[8] Z. Ding, S. Yan, M. Yuan, X. Hu, F. Lin, and A. Vlachos. TCP: a benchmark for temporal constraint-based planning. In *The 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

[9] Z. Fan, Z. Liu, J. Zhang, Y. Xiong, L. Zheng, and P. S. Yu. Continuous-time sequential recommendation with temporal graph collaborative transformer. In G. Demartini, G. Zuccon, J. S. Culpepper, Z. Huang, and H. Tong, editors, *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 433–442. ACM, 2021.

[10] B. Fatemi, J. Halcrow, and B. Perozzi. Talk like a graph: Encoding graphs for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[11] B. Fatemi, M. Kazemi, A. Tsitsulin, K. Malkan, J. Yim, J. Palowitch, S. Seo, J. Halcrow, and B. Perozzi. Test of time: A benchmark for evaluating llms on temporal reasoning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

[12] J. Gastinger, S. Huang, M. Galkin, E. Loghmani, A. Parviz, F. Poursafaei, J. Danovitch, E. Rossi, I. Koutis, H. Stuckenschmidt, R. Rabbany, and G. Rabusseau. TGB 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[13] A. Gravina, G. Lovisotto, C. Gallicchio, D. Bacciu, and C. Grohnfeldt. Long range propagation on continuous-time dynamic graphs. In *Forty-first International Conference on Machine Learning*, 2024.

[14] A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023.

[15] X. Guo, A. Li, Y. Wang, S. Jegelka, and Y. Wang. G1: teaching llms to reason on graphs with reinforcement learning. *CoRR*, abs/2505.18499, 2025.

[16] Z. Han, Z. Ding, Y. Ma, Y. Gu, and V. Tresp. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In M. Moens, X. Huang, L. Specia, and S. W. 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.

[17] H. Huang, X. Bu, H. Zhou, Y. Qu, J. Liu, M. Yang, B. Xu, and T. Zhao. An empirical study of llm-as-a-judge for LLM evaluation: Fine-tuned judge model is not a general substitute for GPT-4. In W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, editors, *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 5880–5895. Association for Computational Linguistics, 2025.

[18] S. Huang, A. Parviz, E. Kondrup, Z. Yang, Z. Ding, M. M. Bronstein, R. Rabbany, and G. Rabusseau. Are large language models good temporal graph learners? *CoRR*, abs/2506.05393, 2025.

[19] S. Huang, F. Poursafaei, J. Danovitch, M. Fey, W. Hu, E. Rossi, J. Leskovec, M. M. Bronstein, G. Rabusseau, and R. Rabbany. Temporal graph benchmark for machine learning on temporal graphs. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[20] S. Huang, F. Poursafaei, R. Rabbany, G. Rabusseau, and E. Rossi. UTG: towards a unified view of snapshot and event based models for temporal graphs. *CoRR*, abs/2407.12269, 2024.

[21] D. S. Islakoglu and J. Kalo. Chronosense: Exploring temporal understanding in large language models with time intervals of events. *CoRR*, abs/2501.03040, 2025.

[22] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart. Representation learning for dynamic graphs: A survey. *J. Mach. Learn. Res.*, 21:70:1–70:73, 2020.

[23] E. Kondrup. Base3: a simple interpolation-based ensemble method for robust dynamic link prediction. *CoRR*, abs/2506.12764, 2025.

[24] S. Kumar, X. Zhang, and J. Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi, and G. Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 1269–1278. ACM, 2019.

[25] D. Lee, K. Ahrabian, W. Jin, F. Morstatter, and J. Pujara. Temporal knowledge graph forecasting without knowledge using in-context learning. In H. Bouamor, J. Pino, and K. 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.

[26] J. Li, R. Wu, Y. Zhu, H. Zhang, L. Chen, and Z. Zheng. Are large language models in-context graph learners? *CoRR*, abs/2502.13562, 2025.

[27] Y. Li, Y. Shen, L. Chen, and M. Yuan. Zebra: When temporal graph neural networks meet temporal personalized pagerank. *Proc. VLDB Endow.*, 16(6):1332–1345, 2023.

[28] R. Liao, X. Jia, Y. Li, Y. Ma, and V. Tresp. Gentkg: Generative forecasting on temporal knowledge graph with large language models. In K. Duh, H. Gómez-Adorno, and S. 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.

[29] Y. Liu, J. Ma, and P. Li. Neural predicting higher-order patterns in temporal networks. In F. Laforest, R. Troncy, E. Simperl, D. Agarwal, A. Gionis, I. Herman, and L. Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 1340–1351. ACM, 2022.

[30] Y. Liu, Y. Ma, M. Hildebrandt, M. Joblin, and V. 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.

[31] Y. Ma, Z. Guo, Z. Ren, J. Tang, and D. Yin. Streaming graph neural networks. In J. X. Huang, Y. Chang, X. Cheng, J. Kamps, V. Murdock, J. Wen, and Y. Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 719–728. ACM, 2020.

[32] B. Perozzi, B. Fatemi, D. Zelle, A. Tsitsulin, S. M. Kazemi, R. Al-Rfou, and J. Halcrow. Let your graph do the talking: Encoding structured data for llms. *CoRR*, abs/2402.05862, 2024.

[33] F. Poursafaei, S. Huang, K. Pelrine, and R. Rabbany. Towards better evaluation for dynamic link prediction. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[34] Y. Qiu, Z. Zhao, Y. Ziser, A. Korhonen, E. M. Ponti, and S. B. Cohen. Are large language model temporally grounded? In K. Duh, H. Gómez-Adorno, and S. Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 7064–7083. Association for Computational Linguistics, 2024.

[35] G. Rossetti and R. Cazabet. Community discovery in dynamic networks: A survey. *ACM Comput. Surv.*, 51(2):35:1–35:37, 2018.

[36] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. *CoRR*, abs/2006.10637, 2020.

[37] K. Shamsi, F. Victor, M. Kantarcioglu, Y. R. Gel, and C. G. Akcora. Chartalist: Labeled graph datasets for UTXO and account-based blockchains. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[38] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. K. Li, Y. Wu, and D. Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024.

[39] G. Sheng, C. Zhang, Z. Ye, X. Wu, W. Zhang, R. Zhang, Y. Peng, H. Lin, and C. Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

[40] J. Skalse, N. H. R. Howe, D. Krasheninnikov, and D. Krueger. Defining and characterizing reward hacking. *CoRR*, abs/2209.13085, 2022.

[41] E. Spiliopoulou, R. Fogliato, H. Burnsky, T. Soliman, J. Ma, G. Horwood, and M. Ballesteros. Play favorites: A statistical method to measure self-bias in llm-as-a-judge. *arXiv preprint arXiv:2508.06709*, 2025.

[42] Y. Tian, Y. Qi, and F. Guo. Freedyg: Frequency enhanced continuous-time dynamic graph model for link prediction. In *The Twelfth International Conference on Learning Representations*, 2024.

[43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[44] E. M. Voorhees and D. M. Tice. The TREC-8 question answering track. In *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association, 2000.

[45] J. Wang, K. Sun, L. Luo, W. Wei, Y. Hu, A. W. Liew, S. Pan, and B. Yin. Large language models-guided dynamic adaptation for temporal knowledge graph reasoning. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

[46] L. Wang, X. Chang, S. Li, Y. Chu, H. Li, W. Zhang, X. He, L. Song, J. Zhou, and H. Yang. TCL: transformer-based dynamic graph modelling via contrastive learning. *CoRR*, abs/2105.07944, 2021.

[47] X. Wang, D. Lyu, M. Li, Y. Xia, Q. Yang, X. Wang, X. Wang, P. Cui, Y. Yang, B. Sun, and Z. Guo. APAN: asynchronous propagation attention network for real-time temporal graph embedding. In G. Li, Z. Li, S. Idreos, and D. Srivastava, editors, *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*, pages 2628–2638. ACM, 2021.

[48] Y. Wang and Y. Zhao. TRAM: benchmarking temporal reasoning for large language models. In L. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 6389–6415. Association for Computational Linguistics, 2024.

[49] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019.

[50] F. Wu, V. P. Dwivedi, and J. Leskovec. Large language models are good relational learners. In W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 7835–7854. Association for Computational Linguistics, 2025.

[51] S. Xiong, A. Payani, R. Kompella, and F. Fekri. Large language models can learn temporal reasoning. In L. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 10452–10470. Association for Computational Linguistics, 2024.

[52] D. Xu, C. Ruan, E. Körpeoglu, S. Kumar, and K. Achan. Inductive representation learning on temporal graphs. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[53] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, C. Zheng, D. Liu, F. Zhou, F. Huang, F. Hu, H. Ge, H. Wei, H. Lin, J. Tang, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Bao, K. Yang, L. Yu, L. Deng, M. Li, M. Xue, M. Li, P. Zhang, P. Wang, Q. Zhu, R. Men, R. Gao, S. Liu, S. Luo, T. Li, T. Tang, W. Yin, X. Ren, X. Wang, X. Zhang, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Zhang, Y. Wan, Y. Liu, Z. Wang, Z. Cui, Z. Zhang, Z. Zhou, and Z. Qiu. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025.

[54] J. You, T. Du, and J. Leskovec. ROLAND: graph learning framework for dynamic graphs. In A. Zhang and H. Rangwala, editors, *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*, pages 2358–2366. ACM, 2022.

[55] L. Yu, L. Sun, B. Du, and W. Lv. Towards better dynamic graph learning: New architecture and unified library. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[56] X. Zhang, Y. Wang, X. Wang, and M. Zhang. Efficient neural common neighbor for temporal graph link prediction. *CoRR*, abs/2406.07926, 2024.

[57] Z. Zhang, X. Wang, Z. Zhang, H. Li, Y. Qin, and W. Zhu. Llm4dyg: Can large language models solve spatial-temporal problems on dynamic graphs? In R. Baeza-Yates and F. Bonchi, editors, *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pages 4350–4361. ACM, 2024.

[58] L. Zheng, W. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

A Limitations

The capabilities of LLMs fine-tuned with ReaL-TG are inherently limited by the input temporal context graph. If key predictive signals lie outside the k-hop historical neighborhood considered in T-CGS, ReaL-TG may struggle to identify the correct solution. Similar limitations are observed in many TGNN models, which also rely on temporal neighbor sampling to select the most informative neighbors for aggregation [36, 52]. In addition, LLMs are constrained by their context window size, which limits the amount of temporal graph information they can process. For instance, the base model used in our work, Qwen3-4B, has a context window of 32k tokens, making it infeasible to provide entire real-world TGs as input. We also provide a more detailed discussion about this problem in App. G.

B Broader Impact

Positive Societal Impact. In this work, we present ReaL-TG, a framework that fine-tunes LLMs for explanatory TG link forecasting. It is among the first methods to showcase the ability of LLMs to generate textual explanations for TG-related tasks, which we expect will stimulate broader interest in adopting LLMs for explainability in temporal graph research. Beyond academic value, explainability is crucial for industrial practitioners, particularly in high-stakes domains such as fraud detection and network attack analysis. By enabling LLMs to deliver both accurate predictions and meaningful explanations, ReaL-TG offers novel insights into the dynamics of evolving networks and opens promising directions for future work.

Potential Negative Impact. Our work applies LLMs to TG link forecasting, and thus inherits the well-known risks associated with LLMs. For instance, LLMs are prone to hallucination, often producing responses that appear plausible but are factually incorrect. While we show that ReaL-TG can mitigate hallucination to some extent, it cannot eliminate it entirely. Therefore, practitioners adopting ReaL-TG should remain aware of these behaviors and exercise caution in fully trusting LLM outputs, especially in safety-critical applications where misuse or overreliance could lead to adverse outcomes.

C Dataset Access

All datasets used in this work is obtained from the Temporal Graph Benchmark Github repository[3]. The TGB package provides download links along with the processed files. Some datasets used in this work was added in recent updates to TGB such as `subreddit`, `uci` and `enron`. The download links for the datasets from TGB are as follows: `tgbl-wiki`[4], `tgbl-subreddit`[5], `tgbl-coin`[6], `tgbl-flight`[7], `tgbl-uci`[8], `tgbl-enron`[9].

[3] https://github.com/shenyangHuang/TGB
[4] https://object-arbutus.cloud.computecanada.ca/tgb/tgbl-wiki-v2.zip
[5] https://object-arbutus.cloud.computecanada.ca/tgb/tgbl-subreddit.zip
[6] https://object-arbutus.cloud.computecanada.ca/tgb/tgbl-coin-v2.zip
[7] https://object-arbutus.cloud.computecanada.ca/tgb/tgbl-flight-v2.zip
[8] https://object-arbutus.cloud.computecanada.ca/tgb/tgbl-uci.zip
[9] https://object-arbutus.cloud.computecanada.ca/tgb/tgbl-enron.zip

D Related Work

Here we provide a detailed discussion of related works.

Traditional Link Forecasting Methods. Traditional approaches to TG link forecasting span several modeling paradigms. Memory-based methods such as TGN [36] maintain evolving node memories to capture temporal dynamics, often combined with a Graph Neural Network (GNN) to aggregate graph information. Another line of works, including JODIE [24], TCL [46], DyGFormer [55], and DyGMamba [6], leverages sequence modeling units such as recurrent neural networks, Transformers [43], and Mamba layers [14] to model temporal dynamics. Heuristic-based approaches like EdgeBank [33] and Base 3 [23] avoid learnable parameters altogether, instead relying on carefully designed algorithms to extract relevant information from past interactions. Pure MLP-based methods such as GraphMixer [4] have also shown promise by directly encoding link information. Finally, snapshot-based methods like ROLAND [54] and UTG [20] adapt standard GNN architectures to TGs by modifying their training and inference procedures. While effective on standard benchmarks, these methods require retraining from scratch (often with hyperparameter tuning) when applied to new datasets, and they provide no explanations for their predictions, limiting their applicability in settings where interpretability is critical.

LLMs for Graph Reasoning. A growing body of research explores LLMs' reasoning abilities on graph-related tasks. Fatemi et al. [10] show that appropriate graph encodings can improve performance. Methods such as GraphToken[32], GraphLLM [1], and LLaGA [2] enhance reasoning by jointly training LLMs with graph representations, while G1 [15] further demonstrates that RL improves reasoning on static graphs. Recent works have started to examine LLMs' capabilities on TGs. LLM4DyG [57] shows that LLMs capture basic spatio-temporal dependencies but struggle with multi-hop reasoning, and its evaluation is limited to small synthetic TGs. Li et al. [26] explore in-context learning (ICL) on TGs, showing that performance is highly sensitive to prompt design and subgraph selection. Concurrently, TGTalker [18] investigates ICL-based link forecasting on real-world TGs. Despite these advances, none of the existing works addresses how to systematically evaluate LLMs' reasoning quality or how to guide them, through training, towards more effective reasoning strategies for link forecasting on real-world TGs.

LLMs for Temporal Reasoning. Since the rise of LLMs, numerous benchmarks have been proposed to evaluate their temporal reasoning capabilities across a broad range of skills [34, 3, 48, 21]. However, many of these benchmarks partly depend on real-world knowledge, enabling LLMs to answer by recalling memorized facts rather than reasoning, which undermines the accuracy of the evaluation. To address this, recent works introduce benchmarks with anonymized entities to decouple temporal reasoning from factual knowledge [51, 11, 8]. Following this practice, we leverage anonymized real-world TGs in our work, removing the influence of textual attributes and potential risk of data leakage to more accurately reflect LLMs' temporal reasoning abilities on the intrinsic dynamics of TG evolution.

E Data Statistics

Table 4 presents the statistics of the evaluation data.

Table 4: Evaluation data statistics. All data are taken from TGB [19] and thus we omit the prefix in dataset names. Inv. means involved, and T means timestamps. Note that we do not reassign node or timestamp IDs; instead, we directly use the anonymized IDs provided in TGB.

Dataset	# Inv. Nodes	# Queries	# Inv. Links	# Inv. T
`wiki`	2,844	914	914	17,419
`subreddit`	8,097	888	888	44,716
`coin`	9,194	457	482	19,792
`flight`	5,449	488	952	387
`uci`	1,227	660	660	8,738
`enron`	296	839	1,283	3,802

F Advantages of QA Formulation for TG Link Forecasting

Previous studies typically formulate TG link forecasting as a binary classification task, where models are trained to determine whether a potential link (u_q, v_q, t_q) exists.

Definition 3 *Given a TG $\mathcal{G}$, a source node $u_q \in \mathcal{N}$, a destination node $v_q \in \mathcal{N}$, a timestamp $t_q \in \mathcal{T}$, together with the history $\mathcal{H}_{t_q} = \{(u_i, v_i, t_i) \mid t_i < t_q, (u_i, v_i, t_i) \in \mathcal{G}\}$, TG link forecasting aims to predict whether the interaction (u_q, v_q, t_q) exists.*

This makes the computation of ranking metrics such as MRR highly costly. To obtain the rank of a node $e \in \mathcal{N}$, the model must perform a forward pass for every candidate node in $\mathcal{N}$, resulting in a total of $|\mathcal{N}|$ passes that scale with $|\mathcal{N}|$ linearly. In contrast, by formulating TG link forecasting as a QA problem, the model can directly output the predicted nodes in a single forward pass, substantially reducing computational cost for real-world TGs with large $|\mathcal{N}|$. In TGB [19], for each existing *positive* link in the evaluation data, Huang et al. sample a set of *negative* links with false destination nodes and compare the model scores assigned to them. Their evaluation does not consider all nodes in $|\mathcal{N}|$. In contrast, in this work, both MRR and pMRR are computed against the entire node set $|\mathcal{N}|$, which ensures evaluation completeness and efficiency.

G Can We Inject Graph Context in Other Ways?

A limitation of our approach of injecting graph context purely as text is that the amount of information is constrained by the LLM's context window. Several works instead compress graphs into low-dimensional representations and jointly fine-tune them with language models [1, 2]. While effective for downstream tasks, this strategy faces a key limitation for explainable link forecasting. In principle, one could compress more graph information—including the entire historical graph—into such representations, giving LLMs maximal input coverage. Although this offers an advantage over our text-based method, overly compressed representations make it difficult for LLMs to distinguish relevant information for prediction from redundant details. Furthermore, explainable forecasting requires human-readable reasoning traces that depend directly on the input graph context. If the graph is not provided as text, the LLM must also learn to reconstruct graphs from encoded representations back into natural language during inference, which is possible but would require substantial methodological advances. We regard the problem of optimally providing graph context for LLMs as outside the scope of this work, but an important open direction for future research.

H GRPO Details

Given the computed link forecasting reward, we update model parameters by maximizing the GRPO objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{\mathcal{Q}\sim P(\mathcal{Q}),\{O_i\}_{i=1}^{g}\sim\pi_{\theta_{\text{old}}}(O|\mathcal{Q})} \frac{1}{g}\sum_{i=1}^{g}\frac{1}{|O_i|}\sum_{j=1}^{|O_i|}$$
$$\left(\min\left(\frac{\pi_\theta(O_{i,j}|\mathcal{Q},O_{i,<j})}{\pi_{\theta_{\text{old}}}(O_{i,j}|\mathcal{Q},O_{i,<j})}Adv_{i,j},\ \text{clip}\left(\frac{\pi_\theta(O_{i,j}|\mathcal{Q},O_{i,<j})}{\pi_{\theta_{\text{old}}}(O_{i,j}|\mathcal{Q},O_{i,<j})},\ 1-\epsilon,\ 1+\epsilon\right)Adv_{i,j}\right) - \gamma D_{KL}(\pi_\theta \,\|\, \pi_{\text{ref}})\right), \tag{1}$$

where $P(\mathcal{Q})$ is the prompt sampling distribution. π_θ and $\pi_{\theta_{\text{old}}}$ denote the current and old policy models[10], respectively. ϵ is a constant that clips the objective to prevent the policy from changing too drastically in a single update step. γ is a weighting factor for the KL-divergence D_{KL} between π_θ and the pre-trained reference model π_{ref}, ensuring the fine-tuned model does not diverge excessively from the original base model. For each prompt $\mathcal{Q}$, g rollouts $\{O_i\}_{i=1}^{g}$ are sampled, each being a full response, and the objective averages over all $|O_i|$ tokens per rollout. $Adv_{i,j}$ denotes the advantage of the j-th token in the i-th rollout relative to the group of g rollouts, and is defined as

$$Adv_{i,j} = \frac{r(O_i) - \mu(\{r(O_i)\}_{i=1}^{g})}{\sigma(\{r(O_i)\}_{i=1}^{g})}. \tag{2}$$

We refer readers to [38] for more details of GRPO.

[10] In RL, we treat the LLM as a policy model, with the old policy model being the checkpoint before the current update.

I Implementation Details

Training. We train ReaL-TG-4B with Qwen3-4B as the base model. We develop ReaL-TG on top of verl [39], a strong framework for post-training on language models. Our training is performed on a compute node with 96 Intel(R) Xeon(R) Platinum 8469C CPU cores and 4 × Nvidia H100 GPU each with 80GB memory. We provide the training hyperparameters in Table 5.

Table 5: Hyperparameter configurations of ReaL-TG-4B during training.

Model	# Epoch	Batch Size	Mini-Batch Size	Learning Rate	γ	Max Response Length	# Rollout (g)
ReaL-TG-4B	3	32	16	$2e^{-6}$	0.001	16,384	5

Evaluation. All evaluations are conducted on the same compute node as used for training. For the Qwen3 family, we generate responses using verl, following their official repositories: Qwen3-0.6B[11], Qwen3-4B[12], and Qwen3-8B[13]. The Gemma 3 family is run via Hugging Face Transformers [49], using their official repositories: Gemma-3-4B-it[14] and Gemma-3-12B-it[15]. We also evaluate Llama-3.3-70B[16] under the same setting. For GPT-5-mini, we use OpenAI's openai-python API. The specific release we use in our experiments is gpt-5-mini-2025-08-07. All reasoning models are executed three times with default hyperparameters, and we report the mean results. Non-reasoning models are run with temperature fixed to 0 for greedy decoding, while all other hyperparameters follow their default configurations.

Judge Model. We employ GPT-4.1-mini for our LLM-as-a-Judge system, implemented via OpenAI's openai-python API. Specifically, we use the gpt-4.1-mini-2025-04-14 release in our experiments. To ensure reproducibility, the model's temperature is set to 0, and outputs are constrained to JSON format for reliable parsing of judgment information.

J Full Prompts

```
<|system|>
You are a temporal graph learning expert.
<|user|>
You will be asked to predict the next interaction (i.e. `Query Destination Node`) given the `Query Source Node` and `Query
Timestamp`.
You will also be given a number of historical interactions extracted from a temporal subgraph, where each of them is
represented as a tuple of (`Source Node`, `Destination Node`, `Timestamp`). Use this information to predict the most likely
`Query Destination Node`s for `Query Source Node` at `Query Timestamp`.
You will only receive information available before `Query Timestamp`. No information at or after this timestamp will be
provided. The user instruction is correct and contains no mistakes or typos.
INSTRUCTIONS:
1. You must FIRST think about the reasoning process as an internal monologue and then provide the final answer.
2. The reasoning process MUST BE enclosed within <think> </think> tags.
3. The final answer MUST BE put within <answer> </answer> tags.
4. If the answer contains multiple `Query Destination Node`s, please provide all of them and put them in a list in sorted
order, e.g., <answer>[0, 1, 2]</answer>, otherwise, please show the answer as a list with only one element, e.g.,
<answer>[0]</answer>.

Question:
Given the following historical interactions:
{Links in G_c}
Could you list all plausible `Query Destination Node`s for `Query Source Node` {u_q} at `Query Timestamp` {t_q}?
```

Figure 2: Prompt template for LLM to do TG link forecasting in ReaL-TG.

[11] https://huggingface.co/Qwen/Qwen3-0.6B
[12] https://huggingface.co/Qwen/Qwen3-4B
[13] https://huggingface.co/Qwen/Qwen3-8B
[14] https://huggingface.co/google/gemma-3-4b-it
[15] https://huggingface.co/google/gemma-3-12b-it
[16] https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

```
<|system|>
You are a meticulous evaluator for temporal graph QA with explanations.
You will receive: (q) the question, (g) a temporal subgraph as lines of (src, dst, ts) strictly before the query timestamp
and a model response R that contains an explanation inside <think>...</think> and a final answer list inside
<answer>...</answer>
Your job is to output ONLY valid JSON matching the JSON Schema provided in the instructions.
You should follow the evaluation procedure as follows:

---
### Evaluation Procedure

1. Parse response
- Extract the answer list A as the JSON array inside <answer> ... </answer>. If parsing fails, set A=[] and record a note
in alignment.notes.
- Extract the explanation E as the natural-language content inside <think> ... </think>. Judge only what is explicitly
stated in E.

2. Break explanation into atomic claims
- Split E into minimal atomic claims about edges, nodes, times, paths, counts, or membership related to graph.
- Produce a list of claims = [c1, c2, ...]. Use short, verifiable sentences.
- Also return the number of claims as #claims.

3. Faithfulness to g
- For each claim ci, label one of:
  "Supported" (entailed by g),
  "Contradicted" (g states the opposite),
  "Not-in-g" (cannot be verified from g; count as unsupported).
- faithfulness.score = #Supported / max(1, #claims).
- Return all Supported claims. For non-Supported claims, return objects with fields: claim, reason ("Contradicted"|"Not-
in-g"), and pointer (cite/summarize lines in g).

4. Logic Consistency (internal reasoning soundness; independent of g’s truth)
- Use 0–2 scale:
  2 Excellent – steps are necessary & sufficient; no contradictions; valid transitions; no major gaps.
  1 Good – slight gap or mild unstated assumption; mostly valid.
  0 Poor/Invalid – The reasoning is unconvincing or fundamentally flawed. It may have significant gaps, make speculative
leaps, contain inconsistencies, or include clear formal fallacies like contradictions or circular reasoning.
- Return a rationale with a concise one-sentence summary.

5. Answer–Explanation Alignment
- An answer a ∈ A is justified iff:
  (1) E explicitly argues for a, and
  (2) those supporting claims are Supported in step 3.
- alignment.score = |justified_answers| / max(1, |A|).
- Return the justified_answers.
- Return the justification_notes that explicitly indicates why the answers are justified. This part will be used to
classify the reasoning patterns of models, so be clear and concise.
- Return the unjustified_answers (in A but not justified).

6. Output
- Return ONLY a JSON object with fields: claims, faithfulness, logic, alignment.
- Do not include any text outside the JSON object.
---

Score three aspects: (1) Faithfulness to g, (2) Logic Consistency, (3) Answer–Explanation Alignment.

IMPORTANT INSTRUCTIONS:
1. Please be VERY CAUTIOUS when you are asked to extract claims and calculate the number of claims.
2. When you are asked to extract claims, DO NOT include any claim making conclusions about the final answer.
3. In many cases, model will correct its previous claims with new claims during reasoning. When you are asked to extract
claims, ALWAYS consider this situation and ONLY include the claims that are not corrected by the model in later steps.
4. When you are asked to evaluate logic consistency, you should evaluate the explanation as a whole reagrdless of the
result of faithfulness.
5. The timestamps with larger numbers are later than the ones with smaller numbers.
6. When judging whether answers are justified or writing justification_notes, remain strictly objective and evaluate only
against the model’s own explanation. Consider an answer justified if the explanation explicitly supports it, even if you
personally disagree with the reasoning. DO NOT mark an answer as unjustified simply because you think it should be
justified in another way.

<|user|>
### JSON Schema
Your output must be a single JSON object that validates against this schema:
{schema_json}

### Inputs
- q:
Could you list all plausible `Query Destination Node`s for `Query Source Node` {u_q} at `Query Timestamp` {t_q}?

- g (historical interactions; all timestamps < {ts}):
{Links in 𝒢_c}

- Metadata:
  - Query Source Node: {u_q}
  - Query Timestamp: {t_q}
  - Ground-truth answers: {{v_q}}
  - Model's final answer: {{a_<ans>}}

- Model response R:
{O}
```

Figure 3: Prompt template for LLM-as-a-Judge system.

K Effect of Base Model Size

In our experiments, we also train a separate model, ReaL-TG-0.6B, based on Qwen3-0.6B. We evaluate its reasoning traces with our LLM-based Judge and compare them against Qwen3-4B and ReaL-TG-4B in Table 6. We find that training from a much smaller base model results in significantly worse reasoning quality: even with our RL framework, a 0.6B model is outperformed by a 4B model substantially. Moreover, we observe a notable case of reward hacking [40]: in many reasoning traces, the fine-tuned ReaL-TG-0.6B justifies its predictions by claiming *"(u_q, v_q, t_q) has already been seen in the provided graph context"*, which is impossible in a forecasting task. This indicates that the model attempts to maximize the outcome-based reward by guessing correct answers while providing a shallow thinking strategy. One major reason is due to the limited reasoning capacity of a tiny model. During RL training, the fine-tuned model must generate full responses (rollouts) based on its own reasoning, following a trial-and-error process guided by the achieved reward. If the base model is too weak, it cannot effectively self-explore more advanced or reasonable reasoning strategies for TG link forecasting. Our results further suggest that using a larger base model enables much stronger fine-tuned performance. Nonetheless, we also observe that after fine-tuning with ReaL-TG, the 0.6B model reaches reasoning quality comparable to Qwen3-4B, highlighting the effectiveness of our RL framework.

Table 6: Results on the quality of reasoning traces compared with ReaL-TG-0.6B.

Model	$\bar{\delta_f}$	$\bar{\delta_{lc}}$	$\bar{\delta_a}$
ReaL-TG-0.6B	0.702	0.710	0.674
Qwen3-4B	0.683	0.700	0.653
ReaL-TG-4B	0.885	0.880	0.732

L Human Evaluation and Annotation Details

We recruit 5 human annotators to do evaluation on the quality of our LLM-as-a-Judge system as well as the reasoning traces output by our fine-tuned ReaL-TG-4B. All annotators are either PhD students or Postdoctoral Researchers in Computer Science with at least full professional proficiency in English. All of them consent our usage of their data. The annotation guidelines are provided in Fig. 4.

Human Evaluation on the Quality of the LLM-as-a-Judge System. To directly assess the reliability of our LLM-based judging system, we randomly sample 50 evaluation examples and collect both the responses generated by ReaL-TG-4B and the corresponding judgments from the system. We then recruit five human annotators to evaluate the quality of these judgments. For each of the three criteria, annotators assign a score from {0,1,2}, with higher values indicating better judging quality. The resulting average scores are 1.71 for faithfulness, 1.88 for logical consistency, and 1.71 for answer–explanation alignment (maximum 2), demonstrating excellent judgment quality. Due to cost constraints, we employ GPT-4.1 mini as the judge, however, judging quality is strongly tied to the capability of the underlying model [17] and can be further enhanced by switching to a more advanced judge, such as Gemini 2.5 Pro.

M Qualitative Analysis: How Does RL Help?

From Table 1 and 2, we observe consistent improvements of the ReaL-TG-trained model over its base model. To illustrate what the model has learned through RL that leads to these gains, we provide a qualitative analysis based on two case studies, comparing ReaL-TG-4B and Qwen3-4B. In Case 1 (Fig. 5 and 6), we observe that after RL, the model no longer exhausts the context window by repeating the same content. Instead, it predicts the most plausible destination node by leveraging interaction recency. In Case 2 (Fig. 5 and 7), we observe that after RL, the model is less prone to getting stuck in iterative self-reflection and demonstrates greater confidence and effectiveness in applying reasoning strategies to support its predictions. To summarize, exploration during RL, in which an LLM tries different strategies for forecasting links depending on the observed graph context,

```
ReaL-TG Human Annotation Guideline

Data: Please download it and fill it out locally
Background:
You will be given multiple data examples. Each consists of:
- A prompt input into a language model (LM).
- The LM's response.
- The judgement of LM's response produced by an automated judging system according to the
   following  procedure:

---
### Evaluation Procedure...
---

Your need to perform two tasks:
Task 1: Judge Evaluation
Evaluate whether the judging system's scores (faithfulness, logical consistency, alignment) are reasonable. For each score,
assign one label:
0 - Big mistake / false
- The score is not supported by correct reasoning if reasoning trace is given.
- The score is incorrectly assigned since it does not match the logic behind the judging system's reasoning.

1 - Largely correct (minor fault)
- The score largely reflects the true quality of the LM response.
- If available, the explanation for the score is mostly accurate but may contain small imprecisions, minor omissions, or
slightly unclear reasoning.

2 - Completely correct
- The score and reasoning are fully correct and accurately reflect the LM's response quality.

Task 2: Human Judge
Re-evaluate each example yourself using the same procedure. You only need to output the three scores (faithfulness, logical
consistency, alignment). If the judging system's score is completely correct, you may directly copy it without re-
evaluating.
```

Figure 4: Human Annotation guideline. The detailed evaluation procedure is taken from the prompt template for the LLM-based judging system in Fig. 3.

is essential for improving both prediction accuracy and the quality of reasoning traces. Although base models already show strong abilities in producing plausible reasoning, they still need to learn how to adjust their reasoning style to the specific context in which it is applied.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims are supported by extensive experiments in Section 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, we discuss the limitations of the paper in detail in Appendix A.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: No formal theorems or proofs are present in the paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, we provide detailed instructions for reproducibility of our experiments, data and evaluation settings in Section 3, 4 and 5. Additional details regarding baseline results are reported in Appendix 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The datasets used for experiment are from the publicly available TGB benchmark [19]. The download links to the datasets are also provided in Appendix C. Detailed instructions for implementation are provided in Section 3, 4 and 5.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All experimental details are provided in Section 3, 4 and 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: In ReaL-TG we fine-tune pre-trained open-source LLMs for temporal link prediction and evaluation thus the weights are pre-trained. Therefore, we report our results without error bars.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the detailed compute resources in Appendix I.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, the positive and negative societal impact of the work is discussed in Appendix B.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our proposed ReaL-TG is a RL fine-tuning framework for LLMs on temporal data thus any safeguards on the base LLM used for fine-tuning can be directly applied here as well.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, all datasets are downloaded from TGB and the dataset licenses can be found in the original TGB paper [19], we provide the dataset download links in Appendix C.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Our ReaL-TG framework provides a novel way of using RL to fine-tune LLM for temporal graph link prediction and explanation thus no new asset are released. The details for our method is found in Section 3.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We conducted a human annotation experiment to evaluate the quality of reasoning traces generated by ReaL-TG, five human annotators volunteered for this experiments and provided assistance. The detailed instructions for this experiment is included in Appendix L.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: Our research do not study human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: Yes, our ReaL-TG framework uses a pre-trained LLM as an important component to fine-tune on. LLM used in ReaL-TG are used for temporal link prediction and link explanation.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Case1

$\mathcal{G}_c$:
(3390, 8648, 833529), (3390, 8648, 927657), (4272, 8929, 1027429), (4272, 8929, 1027461), (104, 8648, 1093360), (3390, 8648, 1103097), (3390, 8648, 1103671), (167, 8648, 1266808), (167, 8648, 1266809), (866, 8648, 1278569), (4459, 8648, 1335789), (4459, 8648, 1335874), (3390, 8929, 1344764), (3390, 8648, 1344818), (3390, 8648, 1344868), (4459, 8648, 1353699), (4459, 8648, 1353719), (866, 8648, 1389561), (866, 8648, 1390132), (866, 8648, 1420514), (997, 8929, 1444089), (997, 8929, 1444395), (997, 8929, 1446670), (997, 8929, 1446795), (997, 8929, 1450527), (423, 8648, 1451994), (3390, 8929, 1461814), (3390, 8648, 1463750), (859, 8648, 1504113), (866, 8648, 1517985), (866, 8648, 1518071), (866, 8648, 1518498), (866, 8648, 1519023), (997, 8929, 1522620), (2727, 8648, 1524334), (866, 8648, 1525088), (866, 8648, 1525235), (5522, 8929, 1525556), (2863, 8929, 1533240), (997, 8929, 1534720), (2863, 8929, 1535928), (2863, 8929, 1535943), (4531, 8929, 1536373), (3390, 8929, 1547848), (3390, 8648, 1549002), (233, 8648, 1575061), (4459, 8648, 1590422), (4459, 8648, 1593828), (611, 8648, 1596720), (5937, 8648, 1606417), (5937, 8648, 1606438), (5937, 8648, 1606461), (5938, 8648, 1607964), (5938, 8648, 1608194), (3390, 8648, 1620262), (997, 8929, 1620278), (997, 8929, 1620574), (997, 8929, 1620852), (997, 8929, 1621381), (997, 8929, 1622753), (997, 8929, 1622892), (5522, 8929, 1624366), (5522, 8929, 1624414), (997, 8929, 1624661), (997, 8929, 1628002), (997, 8929, 1657475), (3390, 8929, 1691346), (997, 8929, 1695077), (997, 8929, 1695521), (3390, 8929, 1696857), (6942, 8929, 2061590), (997, 8929, 2062009), (997, 8929, 2133359), (997, 8929, 2133419), (5522, 8929, 2218607), (7458, 8929, 2262998), (7458, 8929, 2264131), (7458, 8929, 2264356), (7458, 8929, 2264753), (7458, 8929, 2265033), (997, 8929, 2283892), (997, 8929, 2283988), (3390, 8929, 2289548), (8173, 8929, 2646640), (8173, 8929, 2646702), (997, 8929, 2648320), (5522, 8929, 2656128), (997, 8929, 2656490), (8192, 8929, 2659851), (8192, 8929, 2659898), (8192, 8929, 2660147), (8192, 8929, 2660185), (17, 8929, 2660187), (997, 8929, 2663130), (997, 8929, 2663161), (8199, 8929, 2664402), (3390, 8929, 2677842)

q: (3390, ?, 2677935)

$\{v_q\}$: {8929}

Case 2

$\mathcal{G}_c$:
(574, 8552, 1419500), (574, 8552, 1419845), (1601, 8552, 1420897), (3458, 8552, 1432139), (5539, 8552, 1448204), (5539, 8552, 1448331), (1726, 8552, 1458033), (5204, 8552, 1502319), (1206, 8552, 1505338), (2466, 8852, 2315899), (221, 9149, 2439895), (7854, 8852, 2460397), (3138, 9149, 2473041), (1206, 9149, 2473942), (499, 9149, 2479422), (1206, 8734, 2481811), (1206, 8852, 2481993), (499, 9149, 2484302), (221, 9149, 2489612), (4096, 8734, 2501385), (5528, 8734, 2501601), (4096, 8734, 2501828), (1942, 8852, 2502029), (1187, 8734, 2508169), (1206, 8734, 2508797), (1206, 8734, 2509084), (1206, 8734, 2509168), (1206, 8734, 2509314), (1206, 8734, 2509471), (221, 9149, 2515672), (221, 9149, 2516310), (221, 9149, 2517110), (221, 9149, 2518569), (7959, 8734, 2522021), (221, 8734, 2526640), (221, 8734, 2528137), (1221, 8734, 2531985), (1221, 8734, 2532364), (1221, 8734, 2532652), (1343, 8734, 2536121), (1369, 8734, 2539035), (1206, 8734, 2539495), (1206, 8734, 2539603), (2466, 8852, 2561406), (2210, 8734, 2564667), (7914, 8734, 2566838), (8035, 8552, 2567081), (2761, 8552, 2575312), (1680, 8734, 2579425), (1206, 9149, 2586472), (1206, 9149, 2586707), (8035, 8552, 2591725), (1680, 8734, 2593063), (1680, 8734, 2593653), (1680, 8734, 2593703), (1680, 8734, 2594042), (1680, 8734, 2594277), (1680, 8734, 2594499), (4554, 8734, 2597331), (2723, 8852, 2603595), (2723, 8852, 2603688), (2723, 8852, 2603764), (2723, 8852, 2603801), (2723, 8852, 2603995), (221, 8734, 2606232), (1343, 8734, 2614416), (4752, 9084, 2626153), (1973, 9084, 2626233), (1973, 9084, 2626264), (1973, 9084, 2626290), (2425, 9084, 2626614), (8135, 9084, 2626712), (480, 9084, 2626802), (480, 9084, 2626960), (480, 9084, 2627053), (480, 9084, 2627129), (1206, 9084, 2627443), (1206, 8734, 2628541), (1206, 8734, 2628595), (4085, 8552, 2644290), (1206, 8552, 2646542), (4638, 8852, 2657228), (4638, 8852, 2657343), (4638, 8852, 2657416), (4638, 8852, 2658765), (4638, 8852, 2659045), (4638, 8852, 2661840), (4638, 8852, 2662118), (1206, 8852, 2670794), (1206, 8852, 2670884)

q: (1206, ?, 2670969)

$\{v_q\}$: {8852}

Figure 5: The verbalized temporal context graph, query, and ground truths in Case 1 and 2. The input prompts to two cases are derived from combining contents in this figure with the prompt template in Fig. 2.

ReaL-TG-4B	**Qwen3-4B**
<think> Okay, let's tackle this problem. The user is asking for the most likely destination node for a query where the source node is 3390 and the timestamp is 2,677,935. **First, I need to look at all the historical interactions involving node 3390 and see if there's a pattern or frequent destination nodes around the query timestamp.** The query timestamp is 2,677,935. Let's check each entry where the source node is 3390... Looking at the list, I see an entry: (3390, 8929, 2677842). The timestamp here is 2,677,842, which is before the query timestamp... So, **this interaction happened recently. The destination node is 8929...** But maybe there are other interactions where source is 3390... For example, earlier entries: like (3390, 8648, 1620262), (3390, 8929, 1691346), (3390, 8929, 2677842). So, three interactions where source is 3390. Let's check their timestamps. The first is at 1,620,262 (before 2.6m), second at 1,691,346 (before 2.6m), third at 2,677,842 (just before query time). **So, the most recent is with 8929 at 2.6m. So, the answer would be 8929.** </think> <answer> [8929] </answer>	<think> Okay, let's tackle this problem. The user wants to predict the next destination node for source node 3390 at timestamp 2677935 based on historical interactions. First, I need to look at all the historical data provided. Looking through the list, I notice that 3390 is the source in several interactions. Let me list those out. The historical entries where the source is 3390 are: (3390, 8648, 833529) (3390, 8648, 927657) (3390, 8929, 1027429)... (3390, 8648, 1344868) **(3390, 8648, 1344868)** **(3390, 8648, 1344868)** **(3390, 8648, 1344868)...**

Figure 6: Comparison of reasoning traces on the same evaluation data generated by ReaL-TG-4B and Qwen3-4B on Case 1. The **black bold** part indicates the effective reasoning strategy used by ReaL-TG-4B to obtain the correct prediction, whereas the **red bold** part marks the low-quality reasoning produced by Qwen3-4B. Qwen3-4B failed to complete its reasoning and instead filled the context window by repeatedly generating the same link.

ReaL-TG-4B	**Qwen3-4B**
<think> Okay, let's tackle this problem. The user is asking for the most likely destination node for a query where the source node is 1206 and the timestamp is 2,670,969. They provided a list of historical interactions between nodes with their timestamps. **First, I need to look at all the historical interactions involving node 1206 and see if there's a pattern or frequent destinations.** The latest interaction with node 1206 is at timestamp 2,670,884 (for destination 8852)... So, the most recent interaction is between 1206 and 8852. **But maybe there are other interactions between 1206 and other nodes that are more frequent? Let me check all the entries where the source node is 1206.** Looking at the list: - (1206, 8852, 2670794) - (1206, 8852, 2670884) **So, the only interaction involving 1206 as the source is with node 8852, at timestamps 2,670,794 and 2,670,884... So, the answer is just node 8852.** </think> <answer> [8852] </answer>	<think> Okay, let's tackle this problem. The user wants to predict the most likely destination nodes for the source node 1206 at timestamp 2670969 based on the historical interactions provided. First, I need to look at all the historical interactions involving source node 1206. Let me go through the list and note down each entry where the source is 1206. Looking at the data, here are the entries with source 1206: - (1206, 8734, 2481811) - (1206, 8852, 2481993) - (1206, 8734, 2484302) **- (221, 9149, 2489612)** - (1206, 8734, 2498082) **Wait, no, that's not in the list. Let me check again...** **Wait, no. Let me go through each entry step by step...** **Wait, no. Wait, let me check again. The user provided a long list. Let me parse it correctly. Looking at the list again, the entries with source 1206:...** **Wait, no, that's not correct. Let me go through the list again. Looking at the list:...** **Wait, no, that's not correct. Let me check again. The original list is:...** **no, I think I need to check again. Let me look at the original data again...** **no, that's not in the list. Wait, maybe I'm miscounting. Actually, the original data includes:...** **no, I think I'm missing some entries. Let me check the original data again...** Wait, I think I'm getting confused. Let me go through the entire list again... Now, based on the historical interactions, what are the most common destination nodes for source 1206? Looking at the historical data for source 1206: - 8734 appears multiple times. - 8852 appears multiple times. - 9149 appears multiple times. I think I need to stop here and proceed with the data I have. Assuming that the destination nodes for source 1206 are 8734, 8852, and 9149, and **given that the query is at timestamp 2670969, which is after the last interaction with 8852 at 2670884, but before the next interactions with 8852 at 2670794, which is earlier...** But this is just a guess. Alternatively, if the user has more entries, but I'm not seeing them. However, the query is at 2670969, which is after the last interaction with 8852 at 2670884, so the next possible destination could be based on the previous interactions. Given that, the most likely destinations are 8734, 8852, and 9149. </think> <answer> [8734, 8852, 9149] </answer>

Figure 7: Comparison of reasoning traces on the same evaluation data generated by ReaL-TG-4B and Qwen3-4B on Case 2. The **black bold** part indicates the effective reasoning strategy used by ReaL-TG-4B to obtain the correct prediction, whereas the **red bold** part marks the low-quality reasoning produced by Qwen3-4B. Qwen3-4B is prone to hallucinate incorrect links in the context graph and becomes entangled in iterative self-reflection, consuming a large number of tokens without making substantive progress. Ultimately, it abandons the reasoning process and resorts to guessing answers independently of the context it was struggling with.