
Are Large Language Models Good Temporal Graph Learners?

Shenyang Huang^{1,2,7} Ali Parviz^{1,6*} Emma Kondrup^{1,2*} Zachary Yang^{1,2}
Zifeng Ding³ Michael Bronstein^{7,8} Reihaneh Rabbany^{1,2,5} Guillaume Rabusseau^{1,4,5}
¹Mila - Quebec AI Institute, ²School of Computer Science, McGill University,
³University of Cambridge, ⁴DIRO, Université de Montréal, ⁵CIFAR AI Chair,
⁶New Jersey Institute of Technology, ⁷University of Oxford, ⁸AITHYRA * Equal contribution

Abstract

Large Language Models (LLMs) have driven significant advancements in Natural Language Processing. While a broad range of literature has explored the graph-reasoning capabilities of LLMs, including their use as predictors on graphs, the application of LLMs to dynamic graphs, real-world evolving networks, remains relatively unexplored. Recent work studies synthetic temporal graphs generated by random graph models, but applying LLMs to real-world temporal graphs remains an open question. To address this gap, we introduce Temporal Graph Talker (TGTalker), a novel temporal graph learning framework designed for LLMs. TGTalker utilizes the recency bias in temporal graphs to extract relevant structural information, converted to natural language for LLMs, while leveraging temporal neighbors as additional information for prediction. TGTalker demonstrates competitive link prediction capabilities compared to existing Temporal Graph Neural Network (TGNN) models. Across five real-world networks, TGTalker performs competitively with state-of-the-art temporal graph methods while consistently outperforming popular models such as TGN and HTGN. Furthermore, TGTalker generates textual explanations for each prediction, thus opening up exciting new directions in explainability and interpretability for temporal link prediction. Our code is publicly available at <https://github.com/shenyangHuang/TGTalker>.

1 Introduction

The field of artificial intelligence has witnessed remarkable progress through the development of Large Language Models (LLMs), beginning with the foundational Transformer architecture [54] and evolving through significant milestones including BERT [15], GPT-3 [5], and ChatGPT [41]. These models have demonstrated unprecedented capabilities across diverse domains, showcasing their versatility and potential for real-world applications. They have achieved remarkable levels on computer vision tasks including multi-modal understanding [36, 42], on complex medical reasoning tasks [52, 52], and shown promising results in scientific research [48], legal analysis [6], and creative tasks [22], highlighting their transformative potential across professional and academic fields.

A key advancement in LLM capabilities is the emergence of In-Context Learning (ICL), which enables models to adapt to new tasks without explicit fine-tuning [17, 10, 51, 5]. This paradigm encompasses three fundamental approaches: zero-shot learning, where models solve tasks directly from natural language instructions; one-shot learning, which adds a single example; and few-shot learning, which leverages multiple examples to illustrate patterns. This flexibility in learning approaches has opened new avenues for applying LLMs to complex structured data. To harness these capabilities for graph-structured data, recent research has explored various approaches to integrate LLMs with graph

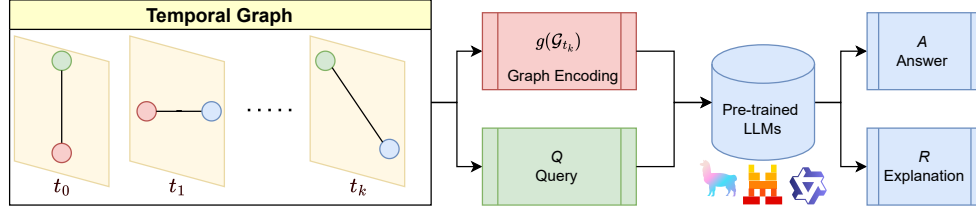


Figure 1: Overview of TGTalker framework.

reasoning tasks. This integration is particularly relevant given that many real-world systems—from molecular structures [2] and social networks [24] to transportation systems [68] and web data [4]—are naturally represented as graphs. Prior work has investigated methods for encoding graph structure for LLMs [21], combining graph encoders with LLMs [20], and fine-tuning LLMs for graph tasks [62].

However, a significant gap exists in applying LLMs to Temporal Graphs (TGs): evolving networks, such as the Internet and social media, where entities and relations change over time. While the pioneering LLM4DyG [66] demonstrated preliminary LLM spatial-temporal understanding capabilities on synthetic temporal graphs, it is limited to small-scale graphs with approximately 20 nodes. Their potential for real-world TGs with thousands or millions of edges and exhibiting complex temporal dynamics thus remains largely unexplored.

To bridge this gap, we introduce Temporal Graph Talker (TGTalker), the first comprehensive framework that leverages LLMs for predictions on real-world temporal graphs. TGTalker is based on a pre-trained LLM, and intelligently translates TG structures into natural language representations tailored to specific tasks. Figure 1 shows an overview of the TGTalker framework, which consists of four key components: (1) a *background set* that captures temporal context by incorporating relevant temporal links near the target link, (2) an *example set* that provides in-context learning samples through carefully selected question-answer pairs, (3) a *query set* that formulates prediction tasks in natural language, and (4) a *temporal neighbor sampling* mechanism that ensures efficient and relevant context selection. Our framework aligns with standard temporal link prediction tasks in Temporal Graph Learning (TGL) [45, 64, 26, 23] while introducing novel capabilities for explainability and adaptability through LLM-based reasoning. Our main contributions are as follows:

- **LLM for real-world temporal graphs.** We present TGTalker, a novel framework that leverages LLMs for predictions on real-world TGs. Our approach combines the recency bias inherent in TGs with temporal neighbor sampling to extract most relevant temporal subgraph as textual representations for LLM.
- **Strong link prediction performance.** Through empirical experiments across three open-source LLM families and five diverse real-world TGs, we demonstrate that TGTalker achieves competitive performance with state-of-the-art Temporal Graph Neural Networks (TGNs), consistently outperforming some across datasets.
- **Temporal link explanations.** TGTalker introduces a novel paradigm of temporal link explanation by leveraging the natural language generation capabilities of LLMs. This enables the generation of human-readable explanations for model predictions, opening new directions for explainable TG methods. We validate explanations’ correctness through a human annotation experiment.
- **Diverse explanation categories.** From LLM-generated explanations, we identify ten diverse explanation categories, which are often related to patterns detected by existing TG algorithms. These categories shed light on the diverse link reasoning capability of LLMs.

2 Preliminary on Temporal Graph Learning

Following the formulation of Continuous Time Dynamic Graphs (CTDGs) in [26, 45, 44, 38], we represent temporal graphs as timestamped streams of edges and formally defined as:

Definition 1 (Temporal Graph). A temporal graph $\mathcal{G}$ can be represented as $\mathcal{G} = \{(s_1, d_1, t_1), (s_2, d_2, t_2), \dots, (s_T, d_T, T)\}$, where the timestamps are ordered ($0 \leq t_1 \leq t_2 \leq \dots \leq T$) and s_i, d_i, t_i denote the source node, destination node and timestamp of the i -th edge.

Note that $\mathcal{G}$ can be directed or undirected, and weighted or unweighted. $\mathcal{G}_t$ is the cumulative graph constructed from all edges in the stream before time t with nodes $\mathbf{V}_t$ and edges $\mathbf{E}_t$. We consider a fixed chronological split to form the training, validation and test set. We follow the standard *streaming setting* [45] for evaluation in CTDGs. The test set information is available to update the model representation (i.e. via a memory module or temporal neighbor sampling); however, no gradient or weight updates are allowed at test time. In this way, models can adapt to incoming data without expensive gradient updates.

Definition 2 (k -hop neighborhood in a temporal graph.). Given a timestamp of interest t , denote $\mathcal{G}_t$ as the cumulative graph constructed by all temporal links before t . Given a node u , the k -hop neighborhood of u before time t is denoted by $\mathcal{N}_u^{t,k}$ and it is defined as the set of all nodes v where there exists at least a single walk of length k from u to v in $\mathcal{G}_t$.

3 Methodology

3.1 Applying LLMs on Temporal Graph

LLM Notation. Consider a pre-trained Large Language Model (LLM) as an interface function f where discrete tokens are received as input and generated as output via the token space $\mathbf{W}$, i.e. $f : \mathbf{W} \rightarrow \mathbf{W}$. Therefore, many tasks with LLM can be formulated as Q&A pairs, i.e., $\mathbf{A} = f(\mathbf{Q})$ where $\mathbf{Q} \in \mathbf{W}$ is the question or query and $\mathbf{A} \in \mathbf{W}$ is the answer. The TGTalker framework can be applied to any pre-trained LLM model. We consider 5 families of LLM models include Qwen3 [60], Qwen2.5 [60, 59], Mistral [7], Llama3 [18] and GPT-4.1-mini [1]. With TGTalker, LLMs can be directly adapted to new TG datasets without any training or fine-tuning steps, significantly reducing the overhead for manual model selection and hyperparameter tuning seen in TGNs.

Temporal Link Prediction with LLMs. Given a TG observed until time t , i.e. $\mathcal{G}_t$, the goal of temporal link prediction is to predict the destination node d of a given source node s at a future timestamp t' . Therefore, we can formulate temporal link prediction as a question-and-answer task for LLMs as follows:

$$\mathbf{A} = f(g(\mathcal{G}_t), q(\mathbf{Q}_{s,?,t'})) \quad (1)$$

where $\mathbf{Q}_{s,?,t'}$ is the link query, $g : \mathcal{G} \rightarrow \mathbf{W}$ is a TG encoding function and $q : \mathbf{W} \rightarrow \mathbf{W}$ is the query encoding function. Thus, it is important to identify the appropriate TG encoding function g and query encoding function q such that the LLMs can provide the correct answer $\mathbf{A}$ while respecting the constraints from the LLM architecture such as maximum input size $l_{\max}$. Note that in practice, standard TGL methods treat the temporal link predicting problem as a ranking problem [26, 63] where the model outputs a probability for each considered node pair and then selects the node pair with the highest probability. With LLMs, it is possible to directly ask the model which destination node is the most probable and avoid the need to predict probability for all node pairs for ranking. Therefore, in TGTalker, the LLM directly outputs the destination node.

Temporal Link Explanation with LLMs. LLMs have been shown as powerful tools for explainability for tasks such as medical diagnosis analysis [67], financial decision transparency [35] and educational feedback [49]. With LLMs directly interfacing with discrete language tokens, it is possible to generate temporal link explanations in natural language. The temporal link explanation task can also be formulated as a question-and-answer task for LLMs as follows:

$$\mathbf{R} = f^*(\mathbf{A}, g(\mathcal{G}_t), q(\mathbf{Q}_{s,?,t'})) \quad (2)$$

where $\mathbf{A} = f(g(\mathcal{G}_t), q(\mathbf{Q}_{s,?,t'}))$ is the answer to query $q(\mathbf{Q}_{s,?,t'})$ and $\mathbf{R}$ is the explanation or the reasoning for the answer. Note that the LLM f^* used for the explanation $\mathbf{R}$ does not necessarily have to be the same LLM f used for prediction. Explaining predictions on TGs is inherently challenging due to the spatial and temporal dependencies between entities [56, 9]. Unlike standard TGNs where only a prediction probability, LLMs provide an alternative approach for contextualizing structural interactions via natural language.

Temporal Graph Talker (TGTalker) TGTalker encodes temporal graphs for LLMs by providing an effective TG encoding function g and a query encoding function q . Importantly, LLMs each have a fixed maximum input size $l_{\max}$, which limits the amount of TG structural information that an LLM can receive. To tackle this issue, we leverage the strong recency bias in TGs [12, 53, 26, 44] and subsample the edges that are closest to the time of interest. Figure 2 shows the overview of TGTalker

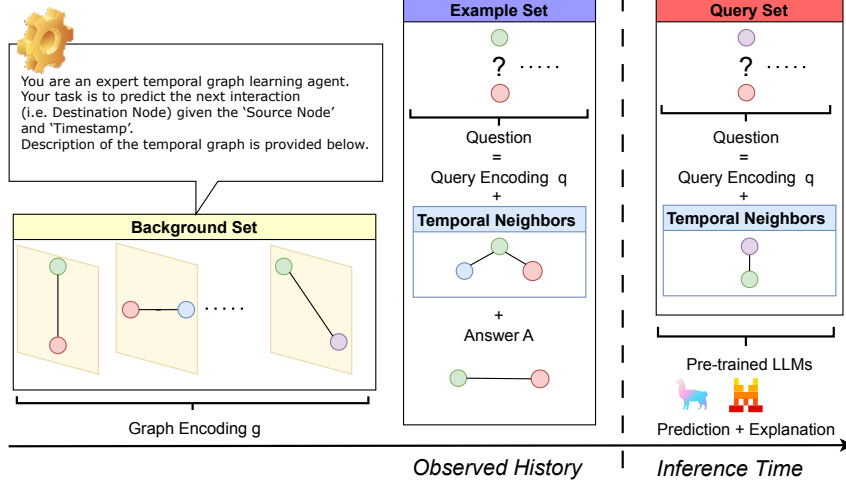


Figure 2: TGTalker framework has four core components: a background set, an example set, a query set and temporal neighbor sampling.

which contains four core components: a *background set*, an *example set*, a *query set*, and *temporal neighbor sampling*. The background set contains recent temporal graph structures, acting as the context for the LLM. The example set shows matching question and answer pairs that instructs the LLM to follow the text correctly. The query set contains the current batch of edges for prediction and then converts them to LLM queries via the query encoding function q . Importantly, temporal neighbor sampling identifies the most recent neighbors for each source node in the query and retrieves them to provide the LLM with additional context. Detailed prompts in TGTalker are provided in Appendix D.

Both TGNNs [58, 45] and TG transformers [64] utilize temporal neighbor sampling as a key component in architecture design. In TGTalker, we also augment LLMs with temporal neighborhood information, following Definition 2. For a given node of interest u , we implement a one-hop neighbor sampler which tracks its neighborhood information $\mathcal{N}_u^{t,1}$ and includes the most recent m neighbors in relation to the current timestamp t . In practice, we observe that the inclusion of temporal neighbor information significantly boosts LLM performance as it directly retrieves the most relevant information for LLM. The *background set* contains b most recent edges in $\mathcal{G}_t$ where t is the timestamp of prediction and acts as the TG encoding function $g(\mathcal{G}_t)$. The LLM is instructed to be a TGL agent, with each node encoded by its integer node ID. Edges are represented with parentheses as (src node ID, dst node ID, timestamp), where timestamps are also integer-encoded. Note that on continuous-time dynamic graphs, novel nodes are slowly introduced over time via their first edge interaction. Thus, assigning node IDs to nodes based on their chronological sequence of appearance provides a principled way to encode nodes with integer encoding.

Following the convention in [26, 45], TGTalker predicts test edges in a fixed-size batch. Each test edge is then converted to text via the query encoding function $q(\mathbf{Q}_{s,?,t'})$ where s and t' are the source node and timestamp of interest, respectively. Similar to the background set, we encode the edges with parentheses, and encode nodes by integer node ID. To provide additional context on the source node, we utilize *temporal neighbor sampling* to also include m most recent neighbors of the source node in the prompt. The *example set* provides in-context learning examples for LLMs to follow. Each example is a Q&A pair, i.e. $(\mathbf{A}, \mathbf{Q})$, instructing the model to follow the same Q&A patterns. For our experiments, we consider the example set size of 5, thus conducting 5-shot prompting on LLM.

4 Temporal Link Prediction

In this section, we evaluate the efficacy of TGTalker for temporal link prediction against established TGL methods. Table 1 presents the test Mean Reciprocal Rank (MRR) for all models across five temporal networks. As TGTalker leverages pre-trained LLMs without task-specific fine-tuning, we report its results from a single inference run. For conventional TGL methods, which are trained from scratch for each dataset, we report the mean and standard deviation across five random seeds.

Table 1: Test MRR comparison for temporal link prediction, results reported from 5 runs for TGNs and a single run for pre-trained LLMs. All LLMs are instruct models. Top results are highlighted by **first**, **second**, **third**.

	Method	tgbl-wiki	Reddit	LastFM	UCI	Enron
TGTalker (ours)	Qwen3-1.7B	0.607	0.606	0.066	0.166	0.166
	Qwen3-8B	0.651	0.626	0.069	0.189	0.192
	Qwen2.5-7B	0.648	0.617	0.079	0.220	0.200
	Mistral-7B-v0.3	0.604	0.612	0.067	0.212	0.184
	Llama3-8B	0.604	0.613	0.069	0.213	0.193
	GPT-4.1-mini	0.679	0.623	0.078	0.232	0.207
Event	TGN [46]	0.396 ± 0.060	0.499 ± 0.011	0.053 ± 0.015	0.051 ± 0.011	0.130 ± 0.066
	TNCN [65]	0.711 ± 0.007	0.696 ± 0.020	0.156 ± 0.002	0.245 ± 0.040	0.379 ± 0.081
	GraphMixer [11]	0.118 ± 0.002	0.136 ± 0.078	0.087 ± 0.005	0.034 ± 0.062	0.014 ± 0.002
	EdgeBank $_{\infty}$ [44]	0.495	0.485	0.020	0.079	0.101
	EdgeBank $_{tw}$ [44]	0.600	0.589	0.026	0.222	0.141
Snapshot	HTGN (UTG) [61, 27]	0.464 ± 0.005	0.533 ± 0.007	0.027 ± 0.007	0.038 ± 0.005	0.107 ± 0.009
	GCLSTM (UTG) [8, 27]	0.374 ± 0.010	0.467 ± 0.004	0.019 ± 0.001	0.013 ± 0.001	0.157 ± 0.019
	GCN (UTG) [30, 27]	0.336 ± 0.009	0.242 ± 0.005	0.024 ± 0.001	0.022 ± 0.001	0.232 ± 0.005

The deterministic EdgeBank baseline is reported by a single run. In Table 1, TGTalker consistently achieves top-two performance on three out of five datasets, and top-three on the Enron and LastFM datasets, when compared with SOTA TGL methods. This highlights the significant potential of leveraging LLMs for complex temporal graph prediction tasks. In addition, TGTalker requires no fine-tuning or training, unlike existing methods. Another key observation is that increased model size within the same LLM family boosts performance. For instance, Qwen3-8B generally outperforms Qwen3-1.7B across multiple datasets. Among the LLM models, GPT-4.1-mini achieves the best overall performance. TGTalker processes only a recent subset of the temporal graph’s history due to maximum token length limitations on LLMs. In contrast, conventional TGL methods are typically exposed to the entire graph history up to the prediction time. The competitive performance of TGTalker suggests that LLMs utilize the most salient recent structural and temporal information from the background set, example set and temporal neighbors in the TGTalker prompts.

5 Temporal Link Explanation

Designing Explanation Categories. To understand the range of link explanations, we first establish reasoning categories. Qwen3-8B and GPT-4.1-mini are used to generate example explanations for categorization. We then perform human annotation to establish ten reasonable and comprehensive categories. These categories, along with their detailed descriptions, are shown in Table 2. The ‘Other’ category is left open for the LLM to propose novel categories if desired. The category of ‘Most Recent Interaction’ is similar to the well-known EdgeBank algorithm [44] where the model simply uses recent past neighbors of a node for prediction. The ‘Default or Most Common Node’ category is analogous to the TG algorithm named PopTrack [14] which captures recently globally popular destination nodes. Another interesting set of explanations, such as ‘Pattern Continuation’ and ‘sequence logic’ involves pattern extrapolation and analyzing alternating interactions from a given source node to various destination nodes. These explanations show that LLM can naturally discover algorithms on a temporal graph and provide insight into link pattern discovery on temporal networks.

Quantitative Analysis of Explanation Categories. To quantify the prevalence of these reasoning categories, we prompted the same LLM that generated the prediction to classify its explanation into one of the ten established categories, including an ‘Others’ option for unclassifiable explanations. The same LLM is used for prediction and explanation classification because only the prediction LLM knows the reasoning for its prediction, while other LLMs might not give the same reasoning. We study the first 5,000 test predictions for Llama3-8B and GPT-4.1-mini on the tgbl-wiki and UCI datasets. Figure 3 top row illustrates the distribution of these explanation categories. A key finding is the adaptability of LLM reasoning to dataset characteristics: the composition of explanations varies significantly across datasets. For example, when comparing GPT-4.1-mini’s explanations on tgbl-wiki versus UCI, there is a marked increase in the ‘Lack of Data’ category for UCI. This

Table 2: Categories of Temporal Link Explanations of LLMs with example explanations.

Category & Description	GPT-4.1-mini Example
Most Recent Interaction: the model predicts the destination node as the one with which the source node had its most recent interaction before (or closest to) the given timestamp.	Considering the latest timestamp 2249408 is close in temporal sequence to the previous interactions, the most likely destination node remains 9180.
Repeated Interaction Pattern: if a source node has repeatedly interacted with the same destination node at multiple timestamps, the model predicts that this pattern will continue.	The source node 415 has two past interactions, both with the destination node 9180 at timestamps 2129703 and 2218092. This indicates a strong and consistent pattern of interaction with the destination node 9180.
Most Frequent Past Destination: when multiple past interactions exist, the explanation chooses the destination node that appears most frequently in the interaction history.	Identify the most frequent Destination Nodes that 4946 has interacted with close to the query timestamp 2237653. Based on past interaction frequency and recency, predict the next likely Destination Node for Source Node 4946 at the given timestamp 2237653.
Pattern Continuation: The model infers the next likely destination by extrapolating from observed interaction patterns, even when the exact match isn't present.	There is no direct information about interactions involving Source Node 969 in the given temporal graph dataset. However, analyzing the dataset, node 1546 connected to node 2093 repeatedly suggests that node 2093 is a likely target for nodes connected with 1546, an adjacent node to 969 in the provided history. Given no other direct interactions for 969, the best estimate based on similar patterns is that 969 will interact with destination node 2093 at timestamp 7554157, consistent with historical interactions of the related node 1546.
Sequence or Alternation Logic: the model uses the order of interactions (e.g., alternating between nodes) to predict the next likely destination.	The most recent interactions for node 431 are at timestamps 9168862 (to 3662) and 9168862 (approx) to 2461. Both are recent and close in time, and both destination nodes appear recurrently. Given that node 431 alternates between these two destination nodes, and there is no recent interaction indicating a new destination, the most likely next destination node at timestamp 9259228 is one of these two. Between 2461 and 3662: Given that there are slightly more occurrences of interactions to 3662, including one of the most recent timestamps (9168862), it is more likely the next interaction will also be to node 3662.
Default or Most Common Node: in the absence of a clear match, the explanation may default to the most common or logical node, or state that any node could be chosen.	Since no direct interactions for Source Node 1741 are present, and no pattern can be derived, we consider the most frequent Destination Node overall or a default prediction. The most frequent Destination Node connected to various sources in the temporal graph is 3692.
Lack of Data: when no clear pattern or sufficient data is available, the model defaults to a plausible guess, sometimes stating the lack of information.	No interactions are shown in the provided temporal graph for node 1624, so we have no historical interactions for node 1624 to infer from.
New Node: the model infers that the next interaction might be with a new node that hasn't appeared in the source node's history, especially if all previous interactions are exhausted.	The 'Source Node' 997 is not present in the provided TEMPORAL GRAPH. Thus, we must infer based on the context: The 'Source Node' 997 likely behaves similarly to node 1546, as their historical interactions from the user mention are similar.
Ambiguous Candidates: the explanation discusses more than one plausible destination (e.g., similar timestamps), and may use additional heuristics to select among them.	Analyze the frequency and recency of destination nodes for Source Node 1543. The latest interactions for Source Node 1543 before timestamp 10383318 are (1543, 3115, 10286093) and (1543, 2539, 10286406). Among these, 3115 and 2539 are recent and likely candidates.
Others: use this only if none of the above apply. Include a proposed new category name and brief justification in the required format.	Examine the provided past interactions for Source Node 2115. Both interactions are from node 6545 to 9180 at different timestamps. There's no direct interaction from 2115 in the given data, but observing the pattern for 6545 might help. Since there's no direct data on 2115 in the provided temporal graph, we look at the past interactions' pattern for node 6545 to infer the destination node for 2115 at the given timestamp.

aligns with UCI's known characteristic of introducing more novel nodes and interactions over time, contrasting with `tgbl-wiki`'s higher prevalence of repeating edges. This demonstrates that LLMs can tailor their reasoning strategies to the underlying data distribution. Further comparison between LLMs in Figure 3 reveals distinct reasoning profiles. For instance, `LLama3-8B` explanations feature a higher proportion of the 'Pattern Continuation' category compared to `GPT-4.1-mini`, which, in turn, exhibits a greater percentage of explanations in the 'Repeated Interaction Pattern' category. These differences offer insights into the varying intrinsic reasoning capabilities and biases of different LLM architectures when applied to temporal graphs. Interestingly, this categorization also surfaces model-specific behaviors and potential limitations. For example, `LLama3-8B` puts none of the explanations in the 'Sequence or Alternation Logic' and 'Ambiguous Candidates' categories, and in some cases, explanations classified under 'Most Common Node' appeared to be hallucinations. Conversely, `GPT-4.1-mini` occasionally produced explanations categorized as 'Others' that suggested novel reasoning patterns, such as 'Analogy-Based Inference from Similar Node,' highlighting the potential for LLMs to uncover unforeseen reasoning strategies.

Correlating Explanation Categories with Predictive Performance. To assess whether the explanation was hallucinated or aligned with the predictive outcomes, we analyzed the test MRR performance within each category in Figure 3 bottom row. For `GPT-4.1-mini`, the results largely align with expectations: the 'Lack of Data' category consistently shows low MRR, validating that the LLM recognizes instances with insufficient evidence. Conversely, categories such as 'Most Recent Interaction' and 'Most Frequent Past Destination' exhibit high MRR, reinforcing the importance of recency and frequency heuristics in these temporal graphs. In comparison, `LLama3-8B` shows

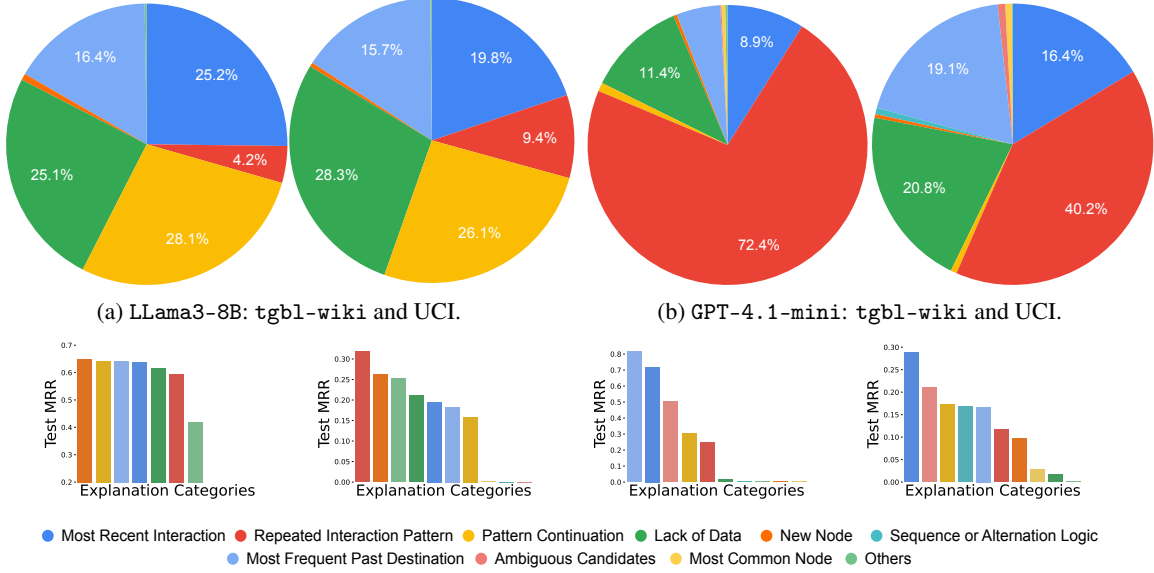


Figure 3: Explanation category composition (top row) and test MRR per category (bottom row) plots for Llama3-8B and GPT-4.1-mini on tgb1-wiki and UCI datasets. Note that empty explanation categories are not shown in bar plots.

the strongest performance in ‘Repeated Interaction Pattern’ and ‘Ambiguous Candidates’ as well as surprisingly high performance in ‘Lack of Data’ categories, suggesting that Llama3-8B is less capable of correlating its explanation with likely predictive outcomes.

Human Annotation Experiment. To ensure the validity of LLM-generated explanations, we conduct a human annotation experiment where we asked four independent annotators to examine TGTalker output explanations, as detailed in Appendix F. This showed that LLM explanations are generally very correct, with low hallucination counts and a strong correlation between human annotators’ assigned categories and that of LLMs. Overall, we observed high intra-annotator agreement to support these findings.

6 Conclusion

In this work, we introduced TGTalker, a novel framework that leverages LLMs for prediction on real-world temporal graphs. TGTalker utilizes the recency bias inherent in temporal graphs and temporal neighbor sampling to extract the most relevant temporal graph structures to textual representations for LLM reasoning. TGTalker achieves competitive performance with state-of-the-art TGNNs across five temporal networks without any fine-tuning. Beyond prediction accuracy, our framework introduces novel temporal link explanations via LLMs’ natural language generation capabilities, providing human-readable explanations for predicted temporal links. The framework’s ability to identify various reasoning patterns - from temporal relations to repeated patterns and sequence identification - opens new possibilities for explainability in temporal graph learning.

Acknowledgments and Disclosure of Funding

This research was 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. Shenyang Huang was 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. Zifeng Ding receives funding from the European Research Council (ERC) under the European Union’s Horizon 2020 Research and Innovation programme grant AVeriTeC (Grant agreement No. 865958). This research was also enabled in part by compute resources provided by Mila (mila.quebec).

References

- [1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [2] D. Beaini, S. Huang, J. A. Cunha, Z. Li, G. Moisescu-Pareja, O. Dymov, S. Maddrell-Mander, C. McLean, F. Wenkel, L. Müller, et al. Towards foundational models for molecular learning on large-scale multi-task datasets. In *ICLR*, 2024.
- [3] A. Behrouz, A. Parviz, M. Karami, C. Sanford, B. Perozzi, and V. S. Mirrokni. Best of both worlds: Advantages of hybrid graph sequence models. *ArXiv*, abs/2411.15671, 2024.
- [4] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer Networks*, 33(1-6):309–320, 2000.
- [5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.
- [6] G. Calzolari, A. Ichino, M. Nardotto, and G. Zanella. Large language models as tax attorneys: a case study in legal analysis. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 382(2284):20230159, 2024.
- [7] D. S. Chaplot et al. Mistral 7b. <https://arxiv.org/abs/2310.06825>, 2023. Version 0.1, Mistral AI. For instruct fine-tuned version see <https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3>.
- [8] J. Chen, X. Wang, and X. Xu. Gc-lstm: Graph convolution embedded lstm for dynamic network link prediction. *Applied Intelligence*, pages 1–16, 2022.
- [9] J. Chen and R. Ying. Tempme: Towards the explainability of temporal graph neural networks via motif discovery. *Advances in Neural Information Processing Systems*, 36:29005–29028, 2023.
- [10] J. Coda-Forno, M. Binz, Z. Akata, M. Botvinick, J. X. Wang, and E. Schulz. Meta-in-context learning in large language models. In *Advances in Neural Information Processing Systems*, 2023.
- [11] 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*, 2022.
- [12] F. Cornell, O. Smirnov, G. Z. Gandler, and L. Cao. On the power of heuristics in temporal graphs. *arXiv preprint arXiv:2502.04910*, 2025.
- [13] X. Dai, Q. Wen, Y. Shen, H. Wen, D. Li, J. Tang, and C. Shan. Revisiting the graph reasoning ability of large language models: Case studies in translation, connectivity and shortest path. *arXiv preprint arXiv:2408.09529*, 2024.
- [14] M. Daniluk and J. Dabrowski. Temporal graph models fail to capture global temporal dynamics. In *Temporal Graph Learning Workshop@ NeurIPS 2023*.
- [15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

- [16] Z. Ding, H. Cai, J. Wu, Y. Ma, R. Liao, B. Xiong, and V. Tresp. zrlm: 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.
- [17] H. Dong, Z. Zhang, Y. Li, Z. Li, Y. Wang, Y. Liu, W. Chen, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2023.
- [18] A. Dubey, L. Van Der Maaten, et al. The llama 3 herd of models, 2024. Version 3.
- [19] V. P. Dwivedi, L. Rampásek, M. Galkin, A. Parviz, G. Wolf, A. T. Luu, and D. Beaini. Long range graph benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340, 2022.
- [20] B. Fatemi, J. Halcrow, and B. Perozzi. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560*, 2023.
- [21] B. Fatemi, J. Halcrow, and B. Perozzi. Talk like a graph: Encoding graphs for large language models. In *International Conference on Learning Representations (ICLR)*, 2024.
- [22] G. Franceschelli et al. On the creativity of large language models. *arXiv preprint arXiv:2304.00008*, 2025.
- [23] 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. *Advances in Neural Information Processing Systems*, 2024.
- [24] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [25] Q. Huang, H. Ren, P. Chen, G. Kržmanc, D. Zeng, P. S. Liang, and J. Leskovec. Prodigy: Enabling in-context learning over graphs. *Advances in Neural Information Processing Systems*, 36:16302–16317, 2023.
- [26] S. Huang, F. Poursafaei, J. Danovitch, M. Fey, W. Hu, E. Rossi, J. Leskovec, M. Bronstein, G. Rabusseau, and R. Rabbany. Temporal graph benchmark for machine learning on temporal graphs. *Advances in Neural Information Processing Systems*, 36:2056–2073, 2023.
- [27] 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. In *The Third Learning on Graphs Conference*.
- [28] Z. Ji, J. Lee, T. Sun, Z. Du, Y. Zhang, Z. Lin, X. Deng, Y. Song, Y. Tan, Q. Zhou, et al. Siren’s song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.
- [29] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, 21(70):1–73, 2020.
- [30] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2016.
- [31] S. Kumar, X. Zhang, and J. Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278, 2019.
- [32] 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.

- [33] J. Li, R. Wu, Y. Zhu, H. Zhang, L. Chen, and Z. Zheng. Are large language models in-context graph learners? *arXiv preprint arXiv:2502.13562*, 2025.
- [34] 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.
- [35] L. Limonad, F. Fournier, J. M. V. Díaz, I. Skarbovsky, S. Gur, and R. Lazcano. Monetizing currency pair sentiments through llm explainability. In *European Conference on Artificial Intelligence*, 2024.
- [36] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, pages 26113–26129, 2023.
- [37] A. Longa, V. Lachi, G. Santin, M. Bianchini, B. Lepri, P. Liò, F. Scarselli, A. Passerini, et al. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *TRANSACTIONS ON MACHINE LEARNING RESEARCH*, 2023.
- [38] Y. Luo and P. Li. Neighborhood-aware scalable temporal network representation learning. In *Learning on Graphs Conference*, pages 1–1. PMLR, 2022.
- [39] N. McKenna, T. Li, L. Cheng, M. J. Hosseini, M. Johnson, and M. Steedman. Sources of hallucination by large language models on inference tasks. *arXiv preprint arXiv:2305.14552*, 2023.
- [40] M.-V. Nguyen, L. Luo, F. Shiri, D. Phung, Y.-F. Li, T.-T. Vu, and G. Haffari. Direct evaluation of chain-of-thought in multi-hop reasoning with knowledge graphs. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2862–2883, Bangkok, Thailand, Aug. 2024. Association for Computational Linguistics.
- [41] OpenAI. Chatgpt (mar 14 version) [large language model]. <https://chat.openai.com/chat>, 2023.
- [42] OpenAI et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [43] B. Perozzi, B. Fatemi, D. Zelle, A. Tsitsulin, M. Kazemi, R. Al-Rfou, and J. J. Halcrow. Let your graph do the talking: Encoding structured data for llms. *ArXiv*, abs/2402.05862, 2024.
- [44] F. Poursafaei, S. Huang, K. Pelrine, and R. Rabbany. Towards better evaluation for dynamic link prediction. *Advances in Neural Information Processing Systems*, 35:32928–32941, 2022.
- [45] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.
- [46] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.
- [47] C. Sanford, B. Fatemi, E. Hall, A. Tsitsulin, S. M. Kazemi, J. J. Halcrow, B. Perozzi, and V. S. Mirrokni. Understanding transformer reasoning capabilities via graph algorithms. *ArXiv*, abs/2405.18512, 2024.
- [48] E. Schulz, E. M. Bender, M. Marelli, M. Botvinick, and S. J. Gershman. How should the advancement of large language models affect the practice of science? *Proceedings of the National Academy of Sciences*, 122(5):e2401227121, 2025.
- [49] H. Seo, T. Hwang, J. Jung, H. Kang, H. Namgoong, Y. Lee, and S. Jung. Large language models as evaluators in education: Verification of feedback consistency and accuracy. *Applied Sciences* (2076-3417), 15(2), 2025.
- [50] K. Shamsi, F. Poursafaei, S. Huang, B. T. G. Ngo, B. Coskunuzer, and C. G. Akcora. Graphpulse: Topological representations for temporal graph property prediction. In *The Twelfth International Conference on Learning Representations*, 2024.

- [51] S. Sia, D. Mueller, and K. Duh. Where does in-context learning happen in large language models? In *NeurIPS*, 2024.
- [52] K. Singhal, S. Azizi, T.-J. Tu, S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole, J. Lee, et al. Large language models encode clinical knowledge. *Nature*, 620(7973):172–180, 2023.
- [53] A. Souza, D. Mesquita, S. Kaski, and V. Garg. Provably expressive temporal graph networks. *Advances in neural information processing systems*, 35:32257–32269, 2022.
- [54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, pages 5998–6008, 2017.
- [55] 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.
- [56] W. Xia, M. Lai, C. Shan, Y. Zhang, X. Dai, X. Li, and D. Li. Explaining temporal graph models through an explorer-navigator framework. In *The Eleventh International Conference on Learning Representations*, 2022.
- [57] Y. Xia, D. Wang, Q. Liu, L. Wang, S. Wu, and X. Zhang. Chain-of-history reasoning for temporal knowledge graph forecasting. 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 16144–16159. Association for Computational Linguistics, 2024.
- [58] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations (ICLR)*, 2020.
- [59] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang, G. Dong, H. Wei, H. Lin, J. Tang, J. Wang, J. Yang, J. Tu, J. Zhang, J. Ma, J. Xu, J. Zhou, J. Bai, J. He, J. Lin, K. Dang, K. Lu, K. Chen, K. Yang, M. Li, M. Xue, N. Ni, P. Zhang, P. Wang, R. Peng, R. Men, R. Gao, R. Lin, S. Wang, S. Bai, S. Tan, T. Zhu, T. Li, T. Liu, W. Ge, X. Deng, X. Zhou, X. Ren, X. Zhang, X. Wei, X. Ren, Y. Fan, Y. Yao, Y. Zhang, Y. Wan, Y. Chu, Y. Liu, Z. Cui, Z. Zhang, and Z. Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.
- [60] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- [61] M. Yang, M. Zhou, M. Kalander, Z. Huang, and I. King. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1975–1985, 2021.
- [62] R. Ye, C. Zhang, R. Wang, S. Xu, and Y. Zhang. Language is all a graph needs. *arXiv preprint arXiv:2308.07134*, 2023.
- [63] J. You, T. Du, and J. Leskovec. Roland: graph learning framework for dynamic graphs. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2358–2366, 2022.
- [64] L. Yu, L. Sun, B. Du, and W. Lv. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36:67686–67700, 2023.
- [65] X. Zhang, Y. Wang, X. Wang, and M. Zhang. Efficient neural common neighbor for temporal graph link prediction. *arXiv preprint arXiv:2406.07926*, 2024.

- [66] 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 *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4350–4361, 2024.
- [67] H. Zhao, Z. Wang, X. Wang, Y. Ma, Y. Lai, Z. Lin, Y. Zhang, and X. Cheng. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):20:1–20:38, 2024.
- [68] Z. Zheng and A. K. Bashir. Graph-enabled intelligent vehicular network data processing. *IEEE Transactions on Intelligent Transportation Systems*, 23(5):4726–4735, 2022.

A Limitations

First, the process of converting temporal graphs into textual representations is inherently constrained by the context window size of the underlying LLM. Given that the context window of LLMs are often limited to around 32k (small models) or 128k (larger models) for common open-source models such as Qwen [60, 59], it is not feasible to use entire temporal graphs with millions of edges as input. TGTalker is designed with this limitation in mind where we provide recent edges to the LLM in the input prompt. However, this strategy may overlook long-range dependencies[19], which may lead to lower performance on datasets where relevant links span longer durations. Second, TGTalker is a framework for adapting LLMs for predictions on temporal graphs. Thus, the speed and efficiency of TGTalker are inherently tied to the characteristics of the base LLM model it employs. In addition, the choice of the base model affects link prediction performance as well as the explanations generated. Therefore, more LLM base models can be explored for evaluation as well.

B Broader Impact

Impact on Temporal Graph Learning. As the first work to demonstrate the effectiveness of LLM on real world temporal graph predictions, we expect TGTalker to raise significant discussion in the community and spark future work that continue to explore the use of LLM for TGL, with TGTalker serving as a solid foundation for more advanced and specific works to follow. As shown in experiments, the link explanations provided by TGTalker can also contribute to future efforts on explainability in this field. More broadly, the advancement in temporal graph learning can be translated to improvements in recommendation systems, epidemic modeling, biological network analysis and more. As such, improving the modeling and analysis of such data opens up a promising avenue for innovation.

Potential Negative Impact. It is well-known that LLMs can hallucinate and provide unreliable explanations [28, 39]. As TGTalker framework utilizes pre-trained LLM, any hallucination or errors from the base LLM might also affect the output from TGTalker. Therefore, practitioners should be aware of the limitations of LLMs when adapting TGTalker. In addition, if the link explanation categories become widely-used in the field, it might limit novel categories to be discovered. We also expect that with novel temporal networks, more explanation categories will be discovered as LLMs adapts to different link patterns.

C Related Work

Temporal Graph Learning. Temporal Graph Learning (TGL) focuses on modeling spatial and temporal dependencies in evolving networks. Following the taxonomy in [29], methods are typically classified as either Continuous-Time Dynamic Graphs (CTDGs) or Discrete-Time Dynamic Graphs (DTDGs). CTDG methods [37], such as TGN [45], EdgeBank [44], TNCN [65], and Graph-Mixer [11], operate on irregular event streams, capturing fine-grained node interactions important for domains like social networks [31] and financial transactions [50]. In contrast, DTDG methods process graph snapshots at fixed intervals, combining GNNs with recurrent models [61, 30, 8]. While DTDGs offer computational efficiency by processing full snapshots and retaining historical context, they may suffer from reduced temporal fidelity [27]. The UTG framework [27] addresses this gap by adapting snapshot-based models to event-based datasets. In our experiments, TGTalker is evaluated against representative CTDG and DTDG methods.

In-Context Learning on Graphs. Recent efforts have explored applying in-context learning (ICL) to graph-structured data, where challenges arise from translating complex relational information into linear prompts suitable for LLMs. Unlike conventional NLP tasks, effective graph ICL requires careful encoding of substructures and relational patterns. Frameworks such as PRODIGY [25] propose pretraining objectives and model architectures tailored for graph-based prompting. Other work investigates the innate ability of LLMs to infer patterns from structured examples with minimal adaptation [33]. A key challenge addressed by TGTalker lies in how to strategically select, verbalize, and sample relevant temporal graph substructures to maximize ICL effectiveness.

Graph Reasoning with Large Language Models. Beyond in-context learning, a growing body of research examines the capacity of LLMs for more general graph reasoning tasks [21, 47, 3, 43].

Table 3: Example Prompt Construction in TGTalker. Background set describes recent links in the temporal graph in text form.

Prompt	Example
System Prompt	You are an expert temporal graph learning agent. Your task is to predict the next interaction (i.e. Destination Node) given the ‘Source Node’ and ‘Timestamp’.
TG Prompt	Description of the temporal graph is provided below, where each line is a tuple of (‘Source Node’, ‘Destination Node’, ‘Timestamp’).
Background Set	(0,8227,0), (1,8228,36), (1,8228,77), (2,8229,131) ...
Example Set	‘Source Node’ 1 has the following past interactions: (1,8228,36), (1,8228,77). Please predict the most likely ‘Destination Node’ for ‘Source Node’ 1 at ‘Timestamp’ 150. Answer: ‘Destination Node’ is 8228.
Query Set	‘Source Node’ 1 has the following past interactions: (1,8228,36), (1,8228,77). Please predict the most likely ‘Destination Node’ for ‘Source Node’ 1 at ‘Timestamp’ 217.

While LLMs demonstrate preliminary spatial and temporal understanding [66], they often struggle with dynamic graphs and tasks requiring complex multi-hop reasoning [13, 40]. Moreover, shortcut learning behaviors—where models produce correct outputs via flawed reasoning—have been widely observed. Nonetheless, the ability of LLMs to reason without task-specific fine-tuning positions them as attractive alternatives to specialized temporal graph neural networks (TGNNs). TGTalker builds on these insights by proposing methods to structure temporal information in ways that enhance LLMs’ robustness and reasoning depth.

LLMs for Temporal Knowledge Graphs. A third line of research adapts LLMs to Temporal Knowledge Graphs (TKGs), leveraging either ICL [32], semantic knowledge of temporal dynamics [16, 57, 55], or fine-tuning strategies [34]. However, most TKGs encode rich textual descriptions for nodes and edges, allowing LLMs to rely on surface-level semantics rather than deeper structural reasoning. In contrast, TGTalker focuses on general temporal graphs where entities are represented abstractly (e.g., by identifiers), requiring models to reason over raw temporal-structural patterns without external semantic clues. This distinction is crucial for evaluating the true relational reasoning abilities of LLMs at fine temporal granularities.

D Prompt Details

Here we show the detailed prompts we used in TGTalker in Table 3. The prompts include system prompt, temporal graph prompt, background set prompt, example set prompt and query set prompt.

E Explanation Categories

Here we provide further details on the temporal link explanations generated by LLMs. Figure 4 illustrates the distribution of explanations generated by Qwen2.5-7B, highlighting the model’s preference for the *Repeated Interaction Pattern* category. Example explanations from Llama3-8B-instruct, GPT-4.1-mini, and Qwen2.5-7B are shown in Table 2, Table 7, and Table 6, respectively. Notably, Llama3-8B-instruct omits explanations in certain categories such as ‘Ambiguous Candidates’ and exhibits hallucinations in the ‘Default or Most Common Node’ category. Similarly, Qwen2.5-7B often misclassifies its ‘Default’ hallucinations, which more accurately reflect ‘Lack of Data’ descriptions.

F Human Annotation Experiment

To empirically validate the correctness of generated explanations, we conducted a human annotation experiment. Four independent annotators were asked to assess the correctness of the explanations and categorize them, as well as flag any instances of hallucinations. Table 4 details the results of this experiment. Notably we find LLM-generated explanations are rated very correct with a low hallucination rate, and considerable agreement between human-attributed categories and those generated by LLMs. We overall find high intra-annotator agreement to support these findings, though it is lower for category attribution, which we attribute to the high number of categories which induce higher variance.

Table 4: Human Annotation Experiment for Wiki and UCI Datasets

Metric	wiki	UCI
Avg. Correctness Rating (1–3)	2.854	2.835
Human–LLM Category: Raw Agreement	70.83%	72.00%
Intra-Annotator Agreement (Correctness Rating)	75.00%	74.00%
Intra-Annotator Agreement (Category Attribution)	45.83%	62.00%
Avg. Count of hallucination behaviour instances	3%	5.5%

G Dataset Details

In this work, we evaluate TGTalker across five continuous-time dynamic graph datasets. Table 5 shows the statistics for each dataset. As seen in Table 5, our experiments cover a wide range of datasets with up to 1.2 million edges and timestamps. Dataset details are described in Appendix G. The number of nodes in our experiments are up to 500x larger than the synthetic graphs used in prior work [66]. The surprise index is defined as $surprise = \frac{|E_{test} \setminus E_{train}|}{E_{test}}$ [44] which measures the proportion of unseen edges in the test set when compared to the training set. The `tgbl-wiki` dataset is from the Temporal Graph Benchmark (TGB) [26] while the rest of datasets are from [44]. We follow the evaluation protocol in TGB where the link prediction task is treated as a ranking problem and the goal is to rank the true positive edge higher than all negative samples. For `tgbl-wiki`, we use the TGB negative samples, while for other datasets, we follow TGB procedure [26] to generate negative samples for evaluation, including 50% historical negatives and 50% random negatives.

- **tgbl-wiki** [26] is a bipartite interaction network that captures temporal editing activity on Wikipedia over one month. The nodes represent Wikipedia pages and their editors, and the edges indicate timestamped edits. Each edge is a 172-dimensional LIWC feature vector derived from the edited text.
- **Reddit** [44] models user-subreddit posting behavior over one month. Nodes are users and subreddits, and edges represent posting requests made by users to subreddits, each associated with a timestamp. Each edge is also a 172-dimensional LIWC feature vector based on post contents.
- **LastFM** [44] is a bipartite user–item interaction graph where nodes represent users and songs. Edges indicate that a user listened to a particular song at a given time. The dataset includes 1000 users and the 1000 most-listened songs over a one-month period. It contains no node or edge attributes.
- **UCI** [44] is an anonymized online social network from the University of California, Irvine. Nodes represent students, and edges represent timestamped private messages exchanged within an online student community. The dataset does not contain node or edge attributes.
- **Enron** [44] is a temporal communication network that is based on email correspondence over a period of three years. Nodes represent employees of the ENRON energy company, while edges correspond to timestamped emails. The dataset does not include node or edge features.

Tgbl-wiki is obtained from the Temporal Graph Benchmark [26], where the dataset can be downloaded along with the package, see TGB website for details. Reddit, LastFM, UCI and Enron

Table 5: Dataset statistics.

Dataset	# Nodes	# Edges	# Unique Edges	# Unique Steps	Surprise	Duration
tgbl-wiki	9,227	157,474	18,257	152,757	0.108	1 month
Reddit	10,984	672,447	78,516	669,065	0.069	1 month
LastFM	1,980	1,293,103	154,993	1,283,614	0.35	1 month
UCI	1,899	26,628	20,296	58,911	0.535	196 days
Enron	184	10,472	3,125	22,632	0.253	3 years

are obtained from [44] and can be downloaded at <https://zenodo.org/records/7213796#.Y8Qic0zMJB2>.

H Variants and Baselines

We test five families of LLMs with varying sizes as pre-trained LLMs in TGTalker, including the following open-source LLM models: Qwen3-1.7B, Qwen3-8B, Qwen2.5-7B-Instruct [60, 59], Mistral-7B-Instruct-v0.3 [7] and Llama3-8B-instruct [18]. We also include the closed-source model: GPT-4.1-mini [1]. All of the LLMs are instruct models thus better fit for in-context learning. We drop the instruct suffix later for brevity. By default, we use background set size $b = 300$, batch size 200, example set size 5, and two 1-hop temporal neighbors for each source node during prediction.

I Compute Resources

For our experiments, we used the following compute resources. The TGTalker (LLM experiments) were conducted on a single NVIDIA A100-SXM4 GPU (80GB memory) paired with 4 AMD Milan 7413 CPU nodes (2.65 GHz, 128MB L3 cache), each equipped with 128GB RAM. For CTDGs and DTDGs experiments, we used a Quadro RTX 8000 GPU paired with 4 CPU nodes, each with 64GB RAM. Each experiment had a five-day time limit and was repeated five times, with results reported as averages and standard deviations across runs. For GPT-4o-mini and GPT-4.1-mini experiments, we used OpenAI’s batch processing API <https://platform.openai.com/docs/guides/batch>. Aside from methods based on PyTorch Geometric, several baseline models—tested using their original code or the DyGLib repository—failed with out-of-memory or timeout errors on larger datasets, even under generous configurations.

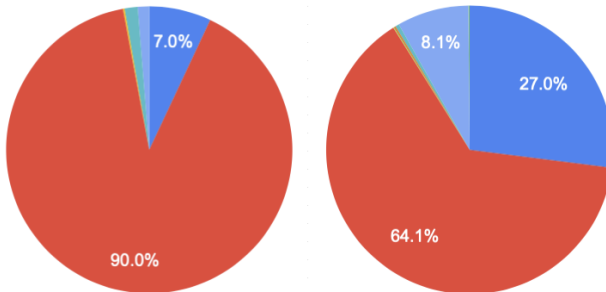


Figure 4: Explanation category composition plots for Qwen2.5-7B on tgbl-wiki and UCI datasets.

J Code and Model Access

Code for reproducing TGTalker is available at <https://github.com/shenyangHuang/TGTalker>. We also attach our code in the supplementary materials as backup. We use various pre-trained LLMs in TGTalker. For reproducibility, the Hugging Face link to all the open-source LLMs are available below:

- Qwen3-1.7B: [link](#).

- Qwen3-8B: [link](#).
- Qwen2.5-7B-Instruct: [link](#).
- Mistral-7B-Instruct-v0.3: [link](#).
- Llama3-8B-instruct: [link](#).

Table 6: Examples of Qwen2.5-7B explanations for each category.

Category Description	Qwen2.5-7B Example
Most Recent Interaction: The model predicts the destination node as the one with which the source node had its most recent interaction before (or closest to) the given timestamp.	I used the temporal graph provided to analyze the movement patterns between Node 775 and its recorded neighbors in the graph. Given that Node 775's movement has been predominately towards Node 8529, and aiming to predict next interaction's node, the Node 8529 is predicted as the most plausible neighbor based on past interactions.
Repeated Interaction Pattern: If a source node has repeatedly interacted with the same destination node at multiple timestamps, the model predicts that this pattern will continue.	I used the temporal graph provided to analyze the movement patterns between Node 7326, Node 400, and Node 2517. Based on previous interactions, I determined the two most frequent Destination Nodes for each Source Node. Therefore, my prediction for Node 7326 is 8491, and my prediction for Nodes 400 and 2517 is 8817 respectively.
Most Frequent Past Destination: When multiple past interactions exist, the explanation chooses the destination node that appears most frequently in the interaction history.	I predict that the most likely 'Destination Node' for 'Source Node' 2196 at 'Timestamp' 2220367 is '8471' because it has been connected to this node more frequently in the temporal graph observations. The adjacency relationships between these nodes is the key factor in my prediction.
Pattern Continuation: The model infers the next likely destination by extrapolating from observed interaction patterns, even when the exact match isn't present.	I predict that the most likely destination node for Source Node 1282 at Timestamp 7206287 is 3185 based on its previous interactions. It has visited two other nodes, 1285 and 2315, around 79 times in Temporal Graph. This visit distribution across these three nodes is discovered by revisiting the Temporal Graph which fits well with a triadic closure of nodes 1282, 1285 and 2315 and highlights potential strength of three nodes to communicate with each other, indicating that 1282 might be looking to create new connections. Therefore, considering the node co-occurrence, node strength and high triadic closure, I predict Render Tree to be the destination node of 1282 at Timestamp 7206287.
Sequence or Alternation Logic: The model uses the order of interactions (e.g., alternating between nodes) to predict the next likely destination.	I used the temporal graph provided to analyze the movement patterns between Node 775 and its recorded neighbors in the graph. Given that Node 775's movement pattern changes between Node 9180 and Node 8529, I predict Node 8529 as the most likely next destination because it represents a different pattern from the one represented by Node 9180.
Default or Most Common Node: In the absence of a clear match, the explanation may default to the most common or logical node, or state that any node could be chosen.	The only explanation generated in this category is a hallucination confounded with the <i>Lack of Data</i> category.
Lack of Data: When no clear pattern or sufficient data is available, the model defaults to a plausible guess, sometimes stating the lack of information.	I used the temporal graph provided to analyze the movement patterns between Node 312 and its recorded neighbors in the graph. Given that Node 390 has existed in previous time-steps but is not seen connected to Node 2032, it can be inferred that Nodes 390 and 2032 are not directly connected to one another at this point. Therefore, predicting a connection between 312 and 2032 might not be accurate, and choosing them as a destination could be a random guess without further context.
New Node: The model infers that the next interaction might be with a new node that hasn't appeared in the source node's history, especially if all previous interactions are exhausted.	I predict that the most likely destination node for Source Node 1285 at Timestamp 7198235 is 3213 based on its previous interactions. It has visited two other nodes, 1282 and 3547, around 7091387. Since 3213 is not discovered in these past interactions and its occurrence in Temporal Graph is less than these two nodes, I believe it has not been visited yet.
Ambiguous Candidates: The explanation discusses more than one plausible destination (e.g., similar timestamps), and may use additional heuristics to select among them.	No explanations fall within this category (None).
Others: Use this only if none of the above apply. Include a proposed new category name and brief justification in the required format.	I predict that the most likely destination node for Source Node 447 at Timestamp 7138932 is 3567 due to my analysis of the temporal graph data, in which Source Node 447 has not interacted with any other Node in the provided data. Since Temporal Graph Learning is based on historical patterns, historical data discovered in the past, and interactions between nodes over time, I predict that no further interaction is likely to occur. Please note that this prediction is based purely on the pattern of absence in the provided data and Temporal Graph Learning best practices, and not on any other external factors. This prediction should be taken as a stochastic exercise and may not be entirely accurate always.

Table 7: Examples of ‘Llama3-8B-instruct’ explanations for each category.

Category Description	Llama3-8B-instruct Example
Most Recent Interaction: The model predicts the destination node as the one with which the source node had its most recent interaction before (or closest to) the given timestamp.	The source node 131 has a past interaction with destination node 220, which is a relatively strong signal. Since there is no other strong signal in the data, I predict that the most likely destination node for source node 131 at timestamp 94823623 is node 276, which is the node that is closest to node 220 in the graph.
Repeated Interaction Pattern: If a source node has repeatedly interacted with the same destination node at multiple timestamps, the model predicts that this pattern will continue.	The temporal graph model learns patterns and relationships based on the past interactions. When we look at the past interactions for ‘Source Node’ 6, we see that it typically interacts with nodes 237 and 314. Considering the timestamp 94969339, it’s possible that the pattern of interactions is maintained, and the most likely destination node for ‘Source Node’ 6 would be ‘Node 314’. However, please note that this prediction is based on a limited set of past interactions and may not generalize to all possible future interactions.
Most Frequent Past Destination: When multiple past interactions exist, the explanation chooses the destination node that appears most frequently in the interaction history.	Based on the given temporal graph, which includes multiple interactions across different timestamps, I can see that ‘Source Node’ 147 has a history of interactions with several nodes, including 280, 293, 314, and 367. The predicted timestamp (95227379) is in the same time range as the majority of interactions. Therefore, I predict that the most likely ‘Destination Node’ for ‘Source Node’ 147 at ‘Timestamp’ 95227379 is node 367, given that this node has a high frequency of interactions with node 147 in the past.
Pattern Continuation: The model infers the next likely destination by extrapolating from observed interaction patterns, even when the exact match isn’t present.	The temporal graph model learns to predict the likelihood of a future interaction based on past interactions. Since ‘Source Node’ 117 has only two past interactions, both with ‘Destination Node’ 64, which is not present in the general interaction list, it’s difficult to determine a pattern or relationship. However, the closest pair to ‘Source Node’ 117 is ‘Source Node’ 125 with a time difference of 18, and its closest pair is a repetition of the same interaction. Therefore, I predicted ‘Destination Node’ 299 as it is similar in structure to the past interaction of ‘Source Node’ 125, even though it may not be the exact same node.
Sequence or Alternation Logic: The model uses the order of interactions (e.g., alternating between nodes) to predict the next likely destination.	No explanations fall within this category. (None)
Default or Most Common Node: In the absence of a clear match, the explanation may default to the most common or logical node, or state that any node could be chosen.	The only explanation in this category is a hallucination.
Lack of Data: When no clear pattern or sufficient data is available, the model defaults to a plausible guess, sometimes stating the lack of information.	Unfortunately, since there is no past interaction data available for ‘Source Node’ 2134, I cannot provide a reliable prediction. The graph learning model relies heavily on past interactions to make predictions, so without any relevant data, it’s challenging to make an accurate prediction. If more data was available, I could train the model to predict the most likely ‘Destination Node’ for ‘Source Node’ 2134 at ‘Timestamp’ 2479023.
New Node: The model infers that the next interaction might be with a new node that hasn’t appeared in the source node’s history, especially if all previous interactions are exhausted.	The source node 1546 is not available in the previous network as we moved to source node 1548. The name sequences of the source node may differ but all other procedures remain same.
Ambiguous Candidates: The explanation discusses more than one plausible destination (e.g., similar timestamps), and may use additional heuristics to select among them.	No explanations fall within this category. (None)
Others: Use this only if none of the above apply. Include a proposed new category name and brief justification in the required format.	Unfortunately, since you provided a random input with fake interactions, I don’t have enough real data to accurately predict the destination node. The interactions you provided have no existing pattern or correlation with any actual node or timestamp, so I would recommend trying with real data for a more accurate prediction.

K Ablation Study

Here we conduct ablation studies on the components of TGTalker. Table 8 shows the link prediction performance of the Qwen3.8b model with respect to *in context learning*, the number of *temporal neighbors*, and the size of the *background set* – as well as the performance of TGTalker with none of these components. Notably, the exclusion of temporal neighbours has a significant reduction in model performance – and the exclusion of all three components drastically nullifies the model’s

performance. This emphasizes the importance of extracting relevant structural information specific to the node of interest (i.e. the recent neighbours of the source node).

Table 8: Ablation Study Results for TGTalker with Qwen3.8b as base model where test MRR is reported.

Configuration	tgbl-wiki	Reddit
TGTalker (All components)	0.649	0.613
– w/o In Context Learning	0.645	0.612
– w/o Temporal Neighbours	0.322	0.122
– w/o Background Set	0.648	0.618
– with no components	0.008	0.002

In comparison, removing either the In-context learning and background set seems to have minor impact on the model performance. Their importance is however put forth by the drop in performance with no components. Table 9 offers more detailed insights into the effect of temporal neighbours. While the absence of such neighbours drastically plummets TGTalker’s performance, the increase in number of neighbours also has a strong positive effect.

Table 9: Ablation Study Results for TGTalker with Qwen3.8b on the number of temporal neighbors

Configuration	tgbl-wiki	Reddit
TGTalker (nbr = 2)	0.649	0.613
– Number of temporal neighbours = 0	0.322	0.122
– Number of temporal neighbours = 1	0.646	0.520
– Number of temporal neighbours = 5	0.652	0.635
– Number of temporal neighbours = 10	0.656	0.643

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: The claims made in the abstract and introduction are centered around the TGTalker's temporal link prediction performance and explanations for predictions. Both are supported by experimental results in Section 4. TGTalker is compared of state-of-the-art TGNNs and achieved competitive performance. The explanations from TGTalker is also discussed and analyzed accordingly. We expect the strong performance of TGTalker to generalize to other temporal graph datasets based on the performance of TGTalker across five datasets tested in this work.

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: The limitations of this work is discussed in the limitations section 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: The paper does not present theoretical results.

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: The TGTalker architecture is explained in detail as the central part of this paper, and supporting elements, including datasets, negative sampling strategies, and batching, are detailed in Sections 3 and 4.

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: [Yes]

Justification: The code used in this work is accessible at the following repository: <https://github.com/shenyangHuang/TGTalker> with instructions in its README files. All datasets used are publicly available from prior work. `tgb1-wiki` is obtained from the Temporal Graph Benchmark (https://tgb.complexdatalab.com/docs/dataset_overview/) [26] and the rest of the datasets are obtained from [44] via <https://zenodo.org/records/7213796#.Y8QicOzMJB2>.

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: Data processing and other relevant details are presented in Sections 3 and 4.

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: [Yes]

Justification: In this work, the TGNN results are reported across multiple seeds with mean and standard deviation values, as they are required to train from scratch for each individual dataset. In contrast, TGTalker results are based on pre-trained LLM models where the weights are directly downloaded from public repositories. Therefore, we report one run for each LLM.

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: Yes, we report the compute resources we used in Appendix I. For gpt-4o-mini and gpt-4.1-mini, we use OpenAI's batch processing api <https://platform.openai.com/docs/guides/batch>.

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: The presented work utilizes publicly available data and models, without collecting sensitive information or affecting human subjects. It thus poses no foreseeable ethical risks and adheres to the 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: The societal impacts expected from this work are detailed in the Broader Impact section 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 TGTalker only utilizes pre-trained language models and no new pre-trained model is released. Thus, the presented work does not pose such risk.

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: The datasets and models used in this work are publicly available and open-sourced. For the closed-source LLMs used, their license is respected. They are all properly credited.

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: [Yes]

Justification: Such assets includes the proposed TGTalker method, which is well explained both in this paper and in the accompanying documentation. We provide the code implementation for reproducibility in <https://anonymous.4open.science/r/TGTalker-668D>.

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: [NA]

Justification: The work does not involve human subjects or crowdsourcing.

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: The present work does not involve crowdsourcing nor research with human subjects thus no approval is needed.

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 TGTalker framework uses LLM as an important component. LLMs are used in the TGTalker framework as link predictors and explainers via natural language text. We test our results across five families of LLMs and show that TGTalker results extends to different LLMs.

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.