LLMTM: Benchmarking and Optimizing LLMs for Temporal Motif Analysis in Dynamic Graphs

Bing Hao[1,2*], **Minglai Shao** [1,3*], **Zengyi Wo** [1], **Yunlong Chu** [1], **Yuhang Liu** [1], **Ruijie Wang** [2†]

[1]School of New Media and Communication, Tianjin University, China
[2]School of Computer Science and Technology, Beihang University, China
[3]Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education, Guangxi Normal University, China
ruijiew@buaa.edu.cn, {haobing, shaoml, wozengyi1999, cyl2024245030, liuyuhang_13}@tju.edu.cn

Abstract

The widespread application of Large Language Models (LLMs) has motivated a growing interest in their capacity for processing dynamic graphs. Temporal motifs, as an elementary unit and important local property of dynamic graphs which can directly reflect anomalies and unique phenomena, are essential for understanding their evolutionary dynamics and structural features. However, leveraging LLMs for temporal motif analysis on dynamic graphs remains relatively unexplored. In this paper, we systematically study LLM performance on temporal motif-related tasks. Specifically, we propose a comprehensive benchmark, LLMTM (Large Language Models in Temporal Motifs), which includes six tailored tasks across nine temporal motif types. We then conduct extensive experiments to analyze the impacts of different prompting techniques and LLMs (including nine models: openPangu-7B, the DeepSeek-R1-Distill-Qwen series, Qwen2.5-32B-Instruct, GPT-4o-mini, DeepSeek-R1, and o3) on model performance. Informed by our benchmark findings, we develop a tool-augmented LLM agent that leverages precisely engineered prompts to solve these tasks with high accuracy. Nevertheless, the high accuracy of the agent incurs a substantial cost. To address this trade-off, we propose a simple yet effective structure-aware dispatcher that considers both the dynamic graph's structural properties and the LLM's cognitive load to intelligently dispatch queries between the standard LLM prompting and the more powerful agent. Our experiments demonstrate that the structure-aware dispatcher effectively maintains high accuracy while reducing cost.

Benchmark, Code and Extended version — https://github.com/Wjerry5/LLMTM

1 Introduction

The success of Large Language Models (LLMs) has motivated exploration into their capabilities on complex structured data, such as web data (Mao et al. 2024). A key frontier is the application of LLMs to dynamic graphs, aiming at capturing evolution patterns of temporal graphs. Recent works study the LLMs' spatial-temporal understanding abilities on dynamic graphs, highlighting the immense potential of LLMs as a new paradigm for dynamic graph analysis. (Zhang et al. 2024; Huang et al. 2025)

Temporal motifs, as elementary units reflecting important local properties of dynamic graphs (Paranjape et al. 2017; Liu et al. 2021), are typically defined as a set of nodes that interact in a specific temporal sequence within a short period of time. Therefore, temporal motifs play a critical role in revealing the functionality and characterizing the key features of dynamic graphs (Seshadhri et al. 2013; Jha et al. 2014; Pinar et al. 2016; Bressan et al. 2017; Jain and Seshadhri 2018). Thus, mining temporal motifs is essential for numerous real-world applications, such as fraud detection (Zhang et al. 2025), friendship prediction (Qiu et al. 2023), vendor identification (Liu et al. 2025), knowledge graph reasoning (Wang et al. 2024b, 2023, 2022; Liu 2025; Liu and Shu 2025; Liu, Wang, and Tong 2025), among others.

Traditional temporal motif detection methods are typically designed for specific motifs and cannot handle diverse motifs in a unified manner (Cai et al. 2024). Deep learning-based approaches, which often require supervised training, perform poorly on this task (experiments in Appendix C.4). However, the capability of LLMs to solve temporal motif-related problems in dynamic graphs remains underexplored. Different from other existing benchmarks (Table 1), this paper starts by exploring the following research question:

RQ1: Can Large Language Models Solve Temporal Motif Problems on Dynamic Graphs?

Addressing this question is non-trivial and presents three key challenges:

- How to design a benchmark that can rigorously assess an LLM's understanding and reasoning on temporal motifs?
- How to generate dynamic graph datasets with a balanced distribution of positive and negative motif instances for fair evaluation?
- How to formulate prompting scheme that can precisely instruct LLMs to understand and process the complex spatio-temporal characteristics of temporal motifs?

The Benchmark. To address these challenges, we introduce **LLMTM**, a comprehensive benchmark for evaluating LLMs on temporal motif problems (Table 1). Unlike prior work that only considers incremental graph changes (Zhang et al. 2024), our benchmark uses a quadruplet representation, (u, v, t, op), to fully capture both edge appearance (add) and

*Equal contribution.
†Corresponding author.

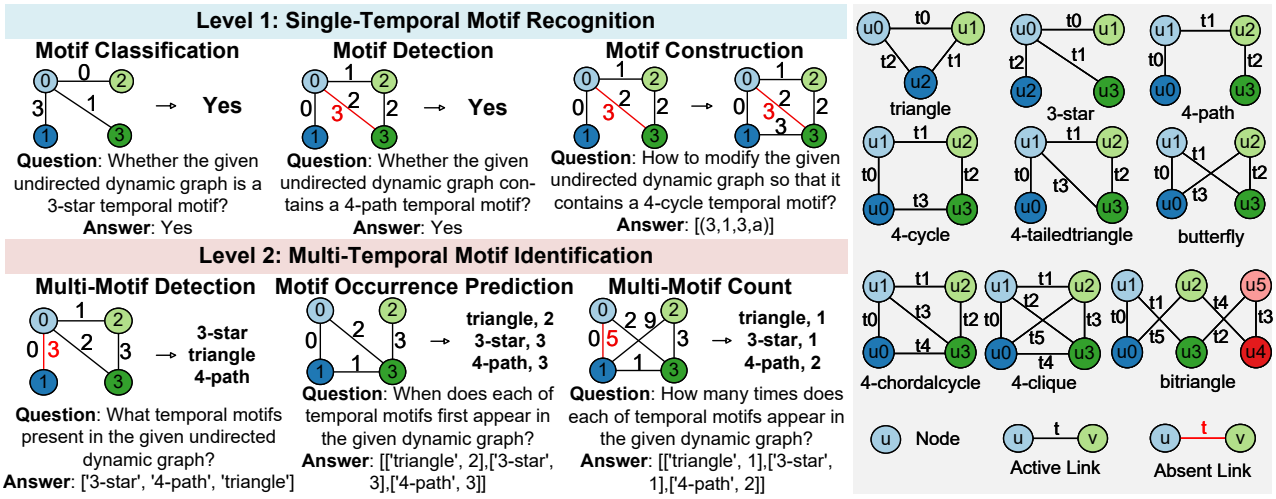

Figure 1: An overview of our LLMTM Benchmark, which includes six tasks (the question is a shortened version due to space) and nine temporal motifs. The tasks (left) are organized into two levels of increasing complexity: (1) single-temporal motif recognition and (2) multi-temporal motif identification. The nine motifs are illustrated on the right.

Bench.	LLM	Dyn. Graph	Temp. Motif	#Ev. LLMs
LLMGP (Dai et al. 2025)	✓	×	×	7
LLM4DyG (Zhang et al. 2024)	✓	✓	×	5
TNM (Liu et al. 2021)	×	✓	✓	0
LLMTM (Ours)	✓	✓	✓	**9**

Table 1: A comparison of related benchmarks by their core research areas. Prior work has focused on LLMs for static graphs (LLMGP), general dynamic graph problems (LLM4DyG), or temporal motifs without considering LLMs (TNM). In contrast, our LLMTM benchmark is the first to specifically evaluate the capabilities of LLMs on temporal motifs, thereby filling a critical research gap.

disappearance (delete). It comprises six tailored tasks organized into two levels of increasing complexity: (1) single-temporal motif recognition, and (2) multi-temporal motif identification (Figure 1).

Furthermore, by analyzing the relationship between temporal motifs frequency, time window size, and dynamic graph scale, we determined random dynamic graph generation settings that ensure our datasets are balanced. To comprehensively assess the capabilities of LLMs, we designed a well-defined prompting scheme (Appendix A.1) and conducted extensive experiments, evaluating four prompting techniques (including zero/one-shot prompting, zero/one-shot chain-of-thought prompting (Wei et al. 2023)) and nine LLMs (including closed-source o3, DeepSeek-R1, GPT-4o-mini and open-source openPangu-7B (Shi et al. 2025), DeepSeek-R1-Distill-Qwen-7B, 14B, 32B (DeepSeek-AI et al. 2025), Qwen2.5-32B-Instruct (Yang et al. 2024), QwQ-32B (Qwen et al. 2025)).

Our extensive experiments reveal a key limitation of LLMs. We observe that LLMs perform poorly on complex tasks such as "Motif Detection" and all Level 2 multi-motif tasks, primarily due to excessive cognitive load (The long-context reasoning capability required for LLMs to extract dynamic graph and temporal motif from natural language) (Observations 3 & 5). This suggests that the reasoning depth of current LLMs remains shallow, and they are likely to fail on problems that require complex, multi-step reasoning.

Tool learning with large language models (LLMs) has emerged as a promising paradigm for augmenting the capabilities of LLMs to tackle highly complex problems (Qu et al. 2025). We further consider:

RQ2: How can agentic capability help to solve temporal motif problems on dynamic graphs?

A Tool-Augmented LLM Agent. Motivated by this, we design a tool-augmented LLM agent, which leverages five algorithms and precisely engineered prompts to solve all six tasks in our benchmark with high accuracy (Table 7). Our experiments show that the proposed agent achieves exceptional performance across all tasks. However, despite high accuracy, the agent incurs greater costs (longer token length and slower response time). This trade-off makes it critical to determine when the agent's intervention is truly necessary, where some simple query can be processed by direct LLM prompt without tool usage. This leads to a key question:

RQ3: How to automatically determine agentic tool usage and direct LLM prompting to balance the trade-off between accuracy and cost?

Trade-off Between Accuracy and Cost. To this goal, we propose the **structure-aware dispatcher** (Figure 2), an agent that strategically routes a given problem to either a standard LLM or a tool-augmented agent. The core of this dispatcher is to predict a problem's intrinsic difficulty using five novel features extracted based on dynamic graph properties and LLM cognition capability: cyclomatic complexity, number of edges, edge locality score, and the proportion of nodes with specific degrees. Empirically, we demonstrate

that the structure-aware dispatcher accurately predicts problem difficulty, determines the optimal path to balance accuracy and cost, and exhibits strong generalization to unseen motif-related tasks (Table 6).

In summary, our key findings are threefold:

- **LLMs have a performance bottleneck.** While they perform well on simple temporal motif tasks, their performance is bottlenecked by cognitive load on more complex ones. Among mainstream LLMs, DeepSeek-R1 essentially achieves the best performance.
- **Agents are accurate but costly.** The tool-augmented agent solves all temporal motif tasks with high accuracy, but at a significant computational cost.
- **The Structure-Aware Dispatcher effectively balances this trade-off.** Our structure-aware dispatcher strategy works by first predicting a problem's intrinsic difficulty and then intelligently routing it to either the standard LLM or the tool-augmented agent, thereby optimizing the accuracy-cost balance.

2 Formulations and Background

2.1 Notations

Dynamic Graph. Prior representations of dynamic graphs often use triplet, (u, v, t), that overlooks edge deletions, a common event in real-world networks. To address this limitation, we introduce a quadruplet representation, (u, v, t, op), to fully capture both edge appearance and disappearance. Here, u, v are nodes, t is the timestamp, and the operator $op \in \{a, d\}$ denotes an "add" or "delete" operation.

Temporal Motif. A temporal motif is a sequence of interconnected edge events that form a specific structural pattern within a constrained time duration. Formally, a (k, l, δ)-temporal motif is a time-ordered sequence of l distinct edge events, $M = \{(u_1, v_1, t_1, a), \ldots, (u_l, v_l, t_l, a)\}$, involving k unique nodes that satisfies the following four constraints:

- **Structural Constraint:** The set of static edges corresponding to the events must form a predefined pattern (e.g., a star or a triangle).
- **Temporal Constraint:** The event timestamps must be strictly increasing, i.e., $t_1 < t_2 < \cdots < t_l$.
- **Duration Constraint:** The entire sequence must occur within a time window of duration δ, meaning $t_l - t_1 \leq \delta$.
- **Connectivity Constraint:** Consecutive events must be connected, meaning each event e_{i+1} shares at least one node with the set of nodes involved in the preceding events $\{e_1, \ldots, e_i\}$.

3 The LLMTM Benchmark

3.1 Dataset Construction

Aiming to create robust benchmarks with balanced positive and negative temporal motif instances, we first analyzed how graph scale (N), time span (T), and window size (W) collectively influence motif distribution to inform our data generation. (Appendix B.1)

Generation Process. We first generate a static graph using the Erdős–Rényi (ER) model, $G_0 = \text{ER}(N, p)$, assigning each edge a random timestamp uniformly distributed over the total time span T. To model edge dynamics, all edges are initially designated as "add" operations (a), with "delete" operations (d) subsequently introduced for a random subset. A fixed random seed ensures full reproducibility.

Task-Specific Settings. For Level 1 tasks, we generate 20 instances per motif type, with two tasks being further refined: "Motif Classification" matches the graph scale to the query motif's size, while "Motif Detection" uses settings that ensure an approximate 50% presence rate. For Level 2 multi-motif tasks, we generate 100 instances with settings ($N = 20, p = 0.3, T = 15$) chosen to ensure a non-zero probability for all motif types. Detailed settings in Appendix B.2.

Real-world dataset. We randomly sample ego-graphs from real graphs for evaluation. Experiments in Appendix C.3.

3.2 Task Definitions

Temporal Motifs. We investigate nine distinct types of temporal motifs: the 3-star, triangle, 4-path, 4-cycle, 4-chordalcycle, 4-tailedtriangle, 4-clique, bitriangle, and butterfly. Figure 1 illustrates the structural patterns and valid temporal orderings for each of these motifs.

Benchmark Tasks. Building on these nine motifs, we design six tailored tasks to systematically evaluate the capabilities of LLMs on temporal motif problems. As illustrated in Figure 1, these tasks are organized into two levels of increasing complexity: (1) single-temporal motif recognition and (2) multi-temporal motif identification.

Level 1: Single-Temporal Motif Recognition. This level assesses the ability of LLMs to recognize and reason about individual temporal motif instances within a dynamic graph.

- **Motif Classification.** Requires the model to classify if a given dynamic graph, as a whole, is an exact instance of a temporal motif. The balanced input data includes positive cases and negative cases derived by individually violating the structural, temporal, or duration constraints.
- **Motif Detection.** Tasks the model with detecting if a dynamic graph contains a specific temporal motif as a subgraph. A positive instance is true if any subset of the graph's events satisfies all four motif constraints.
- **Motif Construction.** Requires the model to complete a nearly-formed temporal motif by adding the single missing edge. The input graph is constructed to contain a pattern missing only its final (l-th) edge; the model must identify and output the complete quadruplet for this edge.

Level 2: Multi-Temporal Motif Identification. This level challenges the LLM's ability to comprehend and resolve problems involving multiple, co-occurring temporal motifs in a dynamic graph.

- **Multi-Motif Detection.** Requires the model to determine, for each of the nine predefined motif types, whether it is contained as a subgraph in a given dynamic graph. The evaluation uses a fine-grained scoring rule that awards partial credit but penalizes false positives.
- **Motif Occurrence Prediction.** Asks the model to detect all present motif types and report the first occurrence

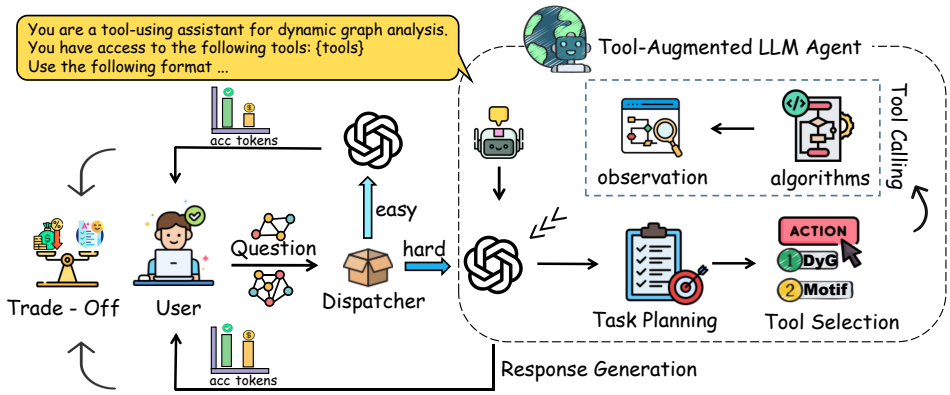

Figure 2: An overview of our framework for balancing the accuracy-cost trade-off. A "Structure-Aware Dispatcher" first extracts the dynamic graph from natural language, predicts the query's difficulty, and then strategically routes simple queries to a standard LLM (for low cost); complex ones to our tool-augmented agent. The agent follows a workflow of task planning, tool selection, tool calling, and response generation to achieve high accuracy, albeit at a greater computational cost.

time for each. The time is defined as the timestamp of the final edge completing the motif's first instance. Partial credit is awarded for incomplete but correct responses.

- **Multi-Motif Count.** Tasks the model with detecting and counting the total number of occurrences for each of the nine predefined motif types. For evaluation, the final score is calculated by summing the ratio of the predicted count to the ground-truth count for each motif type.

We also conduct a broader evaluation covering four fundamental dynamic graph tasks in the Appendix E.

3.3 Prompt and Key Findings

Prompt Structure. Our prompts comprise six sequential components: dynamic graph, temporal motif, task-specific , and answer format instructions, exemplars, and the final question. To prevent ambiguity, we use a distinct notation for motifs (e.g., $(u0, u1, t0, a)$), although both dynamic graphs and motifs are represented by quadruplets. (Appendix A.1)
Prompting Strategies. We investigate the effect of four prompting strategies on model performance: zero-shot, few-shot (Brown et al. 2020), chain-of-thought (CoT) (Wei et al. 2023), and a combination of few-shot with CoT. For CoT-based methods, we use custom-designed prompts with reasoning steps carefully tailored to each task.
Bottleneck. LLMs perform poorly on "Motif Detection" and all Level 2 multi-motif tasks, due to excessive cognitive load (Observations 3 & 5). Strikingly, some LLMs self-diagnose this limitation, responding that the problems exceed their text-based capabilities and require specialized algorithmic tools. This finding directly motivates our work on the tool-augmented agent presented in the next section.

4 Exploring Agentic Capability on Temporal Motif-Related Tasks

Our methodology addresses the limitations of standard LLMs in two stages. We first introduce a tool-augmented agent that, while highly accurate, reveals a significant accuracy-cost trade-off. To resolve this, we then present our primary contribution: the **Structure-aware Dispatcher**. This predictive framework assesses a problem's intrinsic difficulty using a novel set of metrics that quantify graph structure and LLM cognitive load. It then strategically routes each query to either a standard LLM or the powerful agent, creating a hybrid system that optimally balances accuracy and computational cost with strong generalization.

4.1 Tool-Augmented LLM Agent

To overcome the limitations of standard LLMs, we introduce a tool-augmented agent designed to solve the tasks in our benchmark. It is equipped with a specialized suite of algorithmic tools and guided by a precisely engineered prompt.
Tools. The agent utilizes a suite of five algorithmic tools for motif reasoning (e.g., Motif_Detection). The core implementation identifies valid motifs via a two-step process: (1) we use the classic GraphMatcher algorithm (Cordella et al. 2004) to find all subgraph isomorphisms satisfying the **Structural** and **Connectivity** constraints. (2) Each resulting topological mapping is then verified against the **Temporal** and **Duration** constraints. Detailed in Appendix B.3.
Agent Workflow and Prompting. Our agent adapts the classical Reason-Act (ReAct) paradigm in a four-stage workflow (Figure 2): **Task Planning** (interpreting the query and system prompt), **Tool Selection** (choosing a tool and parameters), **Tool Calling** (executing the tool), **Response Generation** (synthesizing the output). A key challenge is designing a single, universal prompt that can reliably steer this process. Through iterative refinement, we developed a unified prompt that compels the model to follow: (1) a structured reasoning framework and (2) precise formatting for robust tool calls, as detailed in Appendix A.3.
Trade-Off. While our tool-augmented agent consistently outperforms direct LLM inference, it is costly, consuming, on average, at least three times more tokens, as shown in Table 7 and Figure 3. This clear trade-off between the agent's high accuracy and its significant computational cost motivates our subsequent investigation into how to automatically determine agentic tool usage and direct LLM prompting to

Motif Classification	3-star	triangle	4-path	4-cycle	4-chordalcycle	4-tailedtriangle	4-clique	bitriangle	butterfly
openPangu-7B	55%	60%	70%	70%	55%	75%	65%	70%	60%
DeepSeek-Qwen-7B	50%	50%	65%	50%	65%	60%	65%	65%	50%
DeepSeek-R1-Distill-Qwen-14B	**100%**	90%	95%	**100%**	90%	**100%**	**100%**	**100%**	90%
DeepSeek-R1-Distill-Qwen-32B	95%	95%	95%	90%	95%	**100%**	**100%**	**100%**	85%
Qwen2.5-32B-Instruct	**100%**	80%	90%	90%	95%	90%	75%	70%	90%
QwQ-32B	**100%**	95%	95%	90%	**100%**	90%	95%	90%	85%
GPT-4o-mini	**100%**	90%	**100%**	90%	95%	90%	90%	85%	85%
DeepSeek-R1	**100%**	95%	**100%**	95%	90%	**100%**	**100%**	**100%**	**100%**
o3	**100%**	**100%**	**100%**	**100%**	95%	90%	90%	**100%**	**100%**
DeepSeek-Qwen-7B-One-shot	60%	50%	65%	50%	65%	60%	35%	50%	55%
DeepSeek-Qwen-7B-Zero-shot+CoT	50%	50%	65%	50%	65%	60%	65%	65%	50%
DeepSeek-Qwen-7B-One-shot+CoT	90%	95%	95%	65%	65%	90%	45%	80%	90%

Table 2: Performance comparison on the "Motif Classification" task against the random baseline. 5-run average results are reported. The best and second-best performance in each column are marked in **bold** and underlined, respectively.

balance the trade-off between accuracy and cost.

4.2 Structure-Aware Dispatcher

To address the inherent accuracy-cost trade-off in using tool-augmented agents, we introduce our primary contribution: the **Structure-Aware Dispatcher** (Figure 2), an agent designed to strategically route a given problem to either a standard LLM prompting or a tool-augmented agent by first predicting the problem's intrinsic difficulty.

Informed by our LLMTM findings (Observation 3) that an LLM's reasoning is primarily affected by graph structure and cognitive load, we focused on the "Motif Detection" task and designed five novel metrics to form the core of our structure-aware dispatcher. We group these into three conceptual levels: (1) **Structural Scale**, measured by the number of edges; (2) **Structural Complexity**, measured by cyclomatic complexity and node degree proportions (ratio_nodes_eq_2 / ratio_nodes_ge_3); and (3) **LLM Cognitive Load** is the long-context reasoning capability required for LLMs to extract dynamic graph and temporal motif from natural language, measured by a new metric, the edge locality score, which calculates the dispersion of edges in their sequential representation. Detailed formulas in Appendix D.

Based on the metrics described above, We train a lightweight XGBoost classifier, encapsulated within the Structure-Aware Dispatcher. The classifier is trained on a dataset with varying graph scales and difficulties across five motif types: 3-star, 4-cycle, 4-clique, 4-chordalcycle, and bitriangle. As shown in Table 6, our approach is highly effective: the structure-aware dispatcher accurately predicts problem difficulty, determines the optimal path to balance accuracy and cost, and exhibits strong generalization.

5 Experiments

5.1 LLMTM Benchmark (RQ1)

We conduct extensive experiments to evaluate the capabilities of LLMs (Large Language Models) on temporal motif problems. Beyond quantitative performance, we also analyze the LLMs' reasoning processes, finding that while they perform well on simple tasks, their performance is bottlenecked by cognitive load on more complex ones.

Models. We evaluate a diverse set of LLMs, including closed-source models (GPT-4o-mini, DeepSeek-R1, o3) and several open-source models. These include openPangu-7B-DeepDiver (Shi et al. 2025), the DeepSeek-R1-Distill-Qwen series at 7B, 14B, and 32B parameter sizes (DeepSeek-AI et al. 2025), Qwen2.5-32B-Instruct (Yang et al. 2024), and QwQ-32B (Qwen et al. 2025). Across all experiments, we set the decoding temperature to $\tau = 0$ for reproducibility and use accuracy as evaluation metric.

Abbreviations. For presentation, in certain locations, we use "Multi Detect", "Motif Occur", and "Multi Count" as shorthand for the Multi-Motif Detection, Motif Occurrence Prediction, and Multi-Motif Count tasks; and the DeepSeek-R1-Distill-Qwen-7B, 14B, and 32B and Qwen2.5-32B-Instruct models are hereafter referred to as the DeepSeek-Qwen-7B, 14B, 32B, and Qwen2.5-32B models, respectively.

Observation 1: LLMs identify motifs via textual pattern matching and are challenged by multiple constraints. LLM's primary mechanism is textual matching. For motifs with intuitive names (e.g., "4-cycle" or "3-star"), the LLM also leverages its semantic understanding to guide the matching process, leading to higher accuracy on these types. (Table 2). Nevertheless, the requirement to concurrently satisfy four constraints remains the core challenge. Without explicit guidance, the LLM tends to overlook some constraints, leading to errors.

Observation 2: Chain-of-Thought (CoT) improves performance by providing semantic guidance and enforcing step-by-step reasoning. We designed CoT prompts that guide the LLM to first describe a motif's structural characteristics in words (activating semantic understanding), and then sequentially check each constraint (pattern, temporal order, and so on). As shown in Table 2, this strategy is highly effective because it leverages the model's semantic strengths to aid its weaker structural reasoning, while task decomposition mitigates the difficulty of handling multiple constraints simultaneously. Detailed examples in Appendix A.2.

Observation 3: Cognitive Load is the key bottleneck for

Motif Detection	3-star	triangle	4-path	4-cycle	4-chordalcycle	4-tailedtriangle	4-clique	bitriangle	butterfly
openPangu-7B	25%	35%	15%	20%	20%	20%	35%	30%	10%
DeepSeek-R1-Distill-Qwen-7B	50%	20%	35%	30%	15%	20%	30%	5%	0%
DeepSeek-R1-Distill-Qwen-14B	**90%**	80%	75%	30%	30%	35%	25%	35%	40%
DeepSeek-R1-Distill-Qwen-32B	**90%**	85%	80%	45%	10%	55%	40%	35%	35%
Qwen2.5-32B-Instruct	75%	70%	75%	50%	55%	45%	20%	45%	35%
QwQ-32B	85%	85%	70%	50%	5%	25%	5%	10%	15%
GPT-4o-mini	60%	45%	45%	30%	55%	30%	45%	**55%**	40%
DeepSeek-R1	**90%**	**95%**	**90%**	**75%**	**65%**	**85%**	**55%**	**55%**	**80%**
o3	**90%**	**95%**	80%	65%	20%	55%	50%	50%	55%

Table 3: Performance comparison on the 'Motif Detection' task against the random baseline. 5-run average results are reported.

motif detection in complex graphs. The performance drop from the simple "Motif Classification" task (Table 2) to the more complex "Motif Detection" task (Table 3) highlights this limitation. The latter task's core challenge stems from numerous irrelevant "distractor" edges, notably increasing its cognitive load, thereby degrading detection accuracy.

Observation 4: Generative tasks are far more challenging than discriminative ones. The "Motif Construction" task highlights this gap. Despite demonstrating a conceptual understanding, models perform poorly because their inability to perform direct, algorithmic construction forces a highly inefficient trial-and-error strategy that fails in larger search spaces, as shown in Table 4.

Motif Construction	4-chordal cycle	4-clique	4-cycle	4-tailed triangle	bitriangle
openPangu-7B	60%	45%	50%	50%	45%
DeepSeek-Qwen-7B	5%	15%	10%	5%	25%
DeepSeek-Qwen-14B	40%	65%	70%	35%	80%
DeepSeek-Qwen-32B	55%	65%	80%	60%	50%
Qwen2.5-32B	65%	20%	30%	60%	55%
QwQ-32B	45%	45%	70%	40%	30%
GPT-4o-mini	**90%**	90%	80%	75%	80%
DeepSeek-R1	**90%**	**95%**	95%	**100%**	85%
o3	85%	**95%**	**100%**	95%	**90%**

Table 4: Performance comparison on "Motif Construction" task across various temporal motifs.

Observation 5: LLMs exhibit a "Capability Collapse" on multi-motif identification tasks. On Level 2 tasks, we observe a sharp performance decline. While LLMs understand the high-level procedure—sequentially checking each motif type, they fail due to a compounding cognitive load. This requires simultaneously tracking multiple, independent constraints for all nine motif types, which causes a severe combinatorial explosion and a diffusion of attention. Strikingly, the Qwen2.5-32B-Instruct model self-diagnosed the limitation, stating the "Multi-Motif Count" task required a "specialized algorithm". This metacognitive observation, where the model recognizes that the task exceeds its text-based limitations, directly explains the poor accuracies in Table 5.

Observation 6: DeepSeek-R1 achieves the best performance. Across all tasks, DeepSeek-R1 essentially performs best, indicating superior long-range logical reasoning.

	Multi Detect	Motif Occur	Multi Count
openPangu-7B	10.71%	0.75%	0.19%
DeepSeek-Qwen-7B	11.19%	0.99%	2.00%
DeepSeek-Qwen-14B	23.67%	10.20%	6.08%
DeepSeek-Qwen-32B	24.91%	8.73%	5.40%
Qwen2.5-32B	23.88%	11.48%	**19.69%**
QwQ-32B	23.19%	4.02%	0.40%
GPT-4o-mini	18.80%	8.44%	13.07%
DeepSeek-R1	32.14%	17.43%	13.41%
o3	**39.64%**	**25.92%**	1.94%

Table 5: Performance comparison on Level 2 tasks.

(a) accuracy (b) average token consumption

Figure 3: Performance of the tool-augmented Agent versus GPT-4o-mini on the "Motif Detection" task, comparing (a) accuracy and (b) average token consumption. See Appendix C.2 for results on all other tasks.

Observation 7: openPangu-7B and DeepSeek-R1-Distill-Qwen-7B perform comparably. The performance of openPangu-7B and DeepSeek-R1-Distill-Qwen-7B tends to be consistent, except on Motif Construction. Interestingly, on this task, openPangu-7B "smartly" outputs a complete motif, artificially inflating its performance.

Notes that we present selected experimental results in this section. The full results are available in Appendix C.1.

5.2 Tool-Augmented LLM Agent (RQ2)

We compare our tool-augmented LLM agent against the baseline GPT-4o-mini on the six LLMTM tasks in Table 7 and Figure 3, which shows that while the agent solves temporal motif tasks with high accuracy, it incurs a significant computational cost.

Backbone	GPT-4o-mini										openPangu-7B	
Task	GPT-4o-mini		Random		Single		**Ours**		Agent		**Ours**	
	Acc	Tokens	Acc	Tokens	Acc	Tokens	Acc	Tokens	Acc	Tokens	Acc	Tokens
3-star	68.5%	**1685**	80.0%	4430	84.5%	3788	88.0%	4165	**100.0%**	6689	78.0%	5906
4-cycle	57.5%	**1669**	75.5%	3893	88.5%	3807	92.5%	4002	**100.0%**	5651	77.0%	7042
4-clique	50.0%	**1580**	71.0%	3746	93.0%	3653	97.5%	4141	**100.0%**	5551	86.5%	7248
bitriangle	53.0%	**1630**	71.0%	3900	93.0%	3776	95.0%	4105	**100.0%**	5685	80.5%	7117
4-chordalcycle	51.0%	**1647**	72.5%	3970	90.5%	3832	96.0%	4309	**100.0%**	5835	86.0%	7452
4-tailedtriangle	53.0%	**1642**	75.0%	3843	85.5%	3978	88.6%	3705	**99.8%**	5883	84.0%	6713
triangle	60.0%	**1652**	77.8%	3864	86.5%	3715	84.8%	3405	**100.0%**	5932	79.80%	6206

Table 6: Performance comparison of the Structure-Aware Dispatcher against several baselines. "Random" refers to randomly assigning instances, "Single" refers to training and applying on the same temporal motif. Detailed results of the Structure-Aware Dispatcher using openPangu-7B as the LLM in the Appendix C.5. This highlights how our methods consistently achieve the second-best accuracy while being significantly more cost-effective than the top-performing "Agent". We held out the "4-tailedtriangle" and "triangle" motifs from the training set to specifically evaluate generalization.

Tasks	Agent		GPT-4o-mini	
	Acc.	Avg. Tokens	Acc	Avg. Tokens
Multi Detect	**98%**	12298.63	18.80%	**2716.59**
Motif Occur	**100%**	11778.15	8.44%	**3023.36**
Multi Count	**99%**	9190.19	13.07%	**2790.10**

Table 7: Performance comparison between our tool-augmented agent and a baseline LLM. We report accuracy (Acc.) and average token consumption (Avg. Tokens) as a measure of cost. The best accuracy and lowest token count in each row are marked in **bold**.

5.3 Structure-Aware Dispatcher(RQ3)

We train a lightweight XGBoost classifier using a dataset of 1500 instances with varying scales and difficulties across five motif types (e.g., 3-star, 4-cycle), which tools LLMs to form an Agent acting as a **"Structure-Aware Dispatcher"**. Evaluated on a test set of 200 instances per motif, structure-aware dispatcher proves highly cost-effective, significantly outperforming baselines. To further assess its generalization, we created new test sets for two unseen motif types ("4-tailedtriangle" and "triangle"), each containing 500 instances. As shown in Table 6, the structure-aware dispatcher remains effective at balancing accuracy and cost even on these novel motifs, underscoring its strong generalizability.

6 Related Work

LLMs on Dynamic Graphs. The application of Large Language Models (LLMs) to dynamic graphs began with feasibility studies: for instance, LLM4DyG (Zhang et al. 2024) benchmarked the spatio-temporal understanding of LLMs and GraphArena (Tang et al. 2025) benchmarked the graph computation of LLMs. Subsequent work developed more specialized frameworks: LLM-DA (Wang et al. 2024a) extracts adaptable temporal logic rules; TGL-LLM (Chang et al. 2025) aligns graph and language representations; and All in One (Sun et al. 2023) leans towards machine learning.

Temporal Motif Discovery. Temporal motifs, critical local property of dynamic graphs, have a long history of study in fields from biology (Chechik et al. 2008; Faisal and Milenkovic 2013) to social science (Xuan et al. 2015; Kosyfaki et al. 2018), and possess classical algorithms for motif discovery and efficient counting (Sun et al. 2019a,b; Liu et al. 2019). More recently, deep learning methods have leveraged motifs for specific tasks, exemplified by a GCN-based approach that fuses local motif information with global properties for fraud detection (Li et al. 2023).

Tool Learning. Tool learning has emerged as a promising paradigm to augment LLMs for complex problems (Qu et al. 2025). Studies show that LLMs can effectively use tools via prompting without fine-tuning (Paranjape et al. 2023; Zhang 2023; Li et al. 2024). For example, HuggingGPT (Shen et al. 2023) leverages a sophisticated prompt design, while TPTU (Ruan et al. 2023) introduces a structured framework with one-step and sequential agents.

7 Conclusion

In this paper, we introduced the LLMTM benchmark to systematically evaluate the capabilities of Large Language Models (LLMs) on temporal motif problems in dynamic graphs, a previously unexplored domain. The benchmark comprises six tasks across two levels of increasing complexity: single-temporal motif recognition and multi-temporal motif identification. Our extensive experiments yielded seven fine-grained observations, illuminating the core reasoning mechanisms and performance bottlenecks of LLMs.

Building on these insights, we demonstrated that a tool-augmented agent can achieve strong performance across all benchmark tasks. To address the significant cost of this approach, we then introduced our primary contribution: a **Structure-Aware Dispatcher**. This framework uses a novel set of metrics to predict a problem's intrinsic difficulty, enabling the selective deployment of the powerful but expensive agent. Our results show that this method successfully balances the accuracy-cost trade-off and exhibits strong generalization.

Acknowledgments

This work is partially supported by NSFC program (No. 62272338) and Research Fund of Key Lab of Education Blockchain and Intelligent Technology, Ministry of Education (EBME25-F-06). Ruijie Wang is supported by the Fundamental Research Funds for the Central Universities.

References

Bressan, M.; Chierichetti, F.; Kumar, R.; Leucci, S.; and Panconesi, A. 2017. Counting Graphlets: Space vs Time. In de Rijke, M.; Shokouhi, M.; Tomkins, A.; and Zhang, M., eds., *WSDM*, 557–566. ACM. ISBN 978-1-4503-4675-7.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165.

Cai, X.; Ke, X.; Wang, K.; Chen, L.; Zhang, T.; Liu, Q.; and Gao, Y. 2024. Efficient Temporal Butterfly Counting and Enumeration on Temporal Bipartite Graphs. arXiv:2306.00893.

Chang, H.; Wu, J.; Tao, Z.; Ma, Y.; Huang, X.; and Chua, T.-S. 2025. Integrate Temporal Graph Learning into LLM-based Temporal Knowledge Graph Model. arXiv:2501.11911.

Chechik, G.; Oh, E.; Rando, O.; Weissman, J.; Regev, A.; and Koller, D. 2008. Activity motifs reveal principles of timing in transcriptional control of the yeast metabolic network. *Nat. Biotechnol.*, 26: 1251–1259.

Cordella, L.; Foggia, P.; Sansone, C.; and Vento, M. 2004. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10): 1367–1372.

Dai, X.; Qu, H.; Shen, Y.; Zhang, B.; Wen, Q.; Fan, W.; Li, D.; Tang, J.; and Shan, C. 2025. How Do Large Language Models Understand Graph Patterns? A Benchmark for Graph Pattern Comprehension. arXiv:2410.05298.

DeepSeek-AI; Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; Zhang, X.; Yu, X.; Wu, Y.; Wu, Z. F.; Gou, Z.; Shao, Z.; Li, Z.; Gao, Z.; Liu, A.; Xue, B.; Wang, B.; Wu, B.; and Feng, B. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948.

Faisal, F. E.; and Milenkovic, T. 2013. Dynamic networks reveal key players in aging. arXiv:1307.3388.

Huang, S.; Parviz, A.; Kondrup, E.; Yang, Z.; Ding, Z.; Bronstein, M.; Rabbany, R.; and Rabusseau, G. 2025. Are Large Language Models Good Temporal Graph Learners? arXiv:2506.05393.

Jain, S.; and Seshadhri, C. 2018. A Fast and Provable Method for Estimating Clique Counts Using Turán's Theorem. arXiv:1611.05561.

Jha, M.; Seshadhri, C.; Pinar, A.; and et al. 2014. Path Sampling: A Fast and Provable Method for Estimating 4-Vertex Subgraph Counts. arXiv:1411.4942.

Kosyfaki, C.; Mamoulis, N.; Pitoura, E.; and Tsaparas, P. 2018. Flow Motifs in Interaction Networks. arXiv:1810.08408.

Li, C.; Yang, R.; Li, T.; Bafarassat, M.; Sharifi, K.; Bergemann, D.; and Yang, Z. 2024. STRIDE: A Tool-Assisted LLM Agent Framework for Strategic and Interactive Decision-Making. arXiv:2405.16376.

Li, S.; Zhou, J.; MO, C.; LI, J.; Tso, G. K. F.; and Tian, Y. 2023. Motif-aware temporal GCN for fraud detection in signed cryptocurrency trust networks. arXiv:2211.13123.

Liu, L. 2025. HyperKGR: Knowledge Graph Reasoning in Hyperbolic Space with Graph Neural Network Encoding Symbolic Path. In Christodoulopoulos, C.; Chakraborty, T.; Rose, C.; and Peng, V., eds., *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 25188–25199. Suzhou, China: Association for Computational Linguistics. ISBN 979-8-89176-332-6.

Liu, L.; and Shu, K. 2025. Unifying Knowledge in Agentic LLMs: Concepts, Methods, and Recent Advancements. Technical report, TechRxiv. Available at https://www.techrxiv.org/users/964779/articles/1333370-unifying-knowledge-in-agentic-llms-concepts-methods-and-recent-advancements?commit=bc73a26e68087ea82a763ea741af036a2ee2e0c5.

Liu, L.; Wang, Z.; and Tong, H. 2025. Neural-Symbolic Reasoning over Knowledge Graphs: A Survey from a Query Perspective. *SIGKDD Explor. Newsl.*, 27(1): 124–136.

Liu, P.; Altun, B.; Acharyya, R.; Tillman, R. E.; Kimura, S.; Masuda, N.; and Sarıyüce, A. E. 2025. Temporal Motifs for Financial Networks: A Study on Mercari, JPMC, and Venmo Platforms. arXiv:2301.07791.

Liu, P.; Guarrasi, V.; Sarıyüce, A. E.; and et al. 2021. Temporal Network Motifs: Models, Limitations, Evaluation. arXiv:2005.11817.

Liu, P.; R, A.; et al. Benson; and Charikar, M. 2019. Sampling Methods for Counting Temporal Motifs. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*.

Mao, Q.; Liu, Z.; Liu, C.; Li, Z.; and Sun, J. 2024. Advancing Graph Representation Learning with Large Language Models: A Comprehensive Survey of Techniques. arXiv:2402.05952.

Paranjape, A.; Benson, A. R.; Leskovec, J.; and et al. 2017. Motifs in Temporal Networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM 2017, 601–610. ACM.

Paranjape, B.; Lundberg, S.; Singh, S.; Hajishirzi, H.; Zettlemoyer, L.; and Ribeiro, M. T. 2023. ART: Automatic multi-step reasoning and tool-use for large language models. arXiv:2303.09014.

Pinar, A.; Seshadhri, C.; Vishal, V.; and et al. 2016. ESCAPE: Efficiently Counting All 5-Vertex Subgraphs. arXiv:1610.09411.

Qiu, Z.; Wu, J.; Hu, W.; Du, B.; Yuan, G.; and Yu, P. S. 2023. Temporal Link Prediction With Motifs for Social Networks. *IEEE Transactions on Knowledge and Data Engineering*, 35(3): 3145–3158.

Qu, C.; Dai, S.; Wei, X.; Cai, H.; Wang, S.; Yin, D.; Xu, J.; and Wen, J.-r. 2025. Tool learning with large language models: a survey. *Frontiers of Computer Science*, 19(8).

Qwen; Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Tang, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Qiu, Z. 2025. Qwen2.5 Technical Report. arXiv:2412.15115.

Rossi, E.; Chamberlain, B.; Frasca, F.; Eynard, D.; Monti, F.; and Bronstein, M. 2020. Temporal Graph Networks for Deep Learning on Dynamic Graphs. arXiv:2006.10637.

Ruan, J.; Chen, Y.; Zhang, B.; Xu, Z.; Bao, T.; Du, G.; Shi, S.; Mao, H.; Li, Z.; Zeng, X.; and Zhao, R. 2023. TPTU: Large Language Model-based AI Agents for Task Planning and Tool Usage. arXiv:2308.03427.

Seshadhri, C.; Pinar, A.; Kolda, T. G.; and et al. 2013. Triadic Measures on Graphs: The Power of Wedge Sampling. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, 10–18. Society for Industrial and Applied Mathematics.

Shen, Y.; Song, K.; Tan, X.; Li, D.; Lu, W.; and Zhuang, Y. 2023. HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face. arXiv:2303.17580.

Shetty, J.; and Adibi, J. 2004. The Enron Email Dataset Database Schema and Brief Statistical Report.

Shi, W.; Tan, H.; Kuang, C.; Li, X.; Ren, X.; Zhang, C.; Chen, H.; Wang, Y.; Hou, L.; and Shang, L. 2025. DeepDiver: Adaptive Search Intensity Scaling via Open-Web Reinforcement Learning. arXiv:2505.24332.

Sun, X.; Cheng, H.; Li, J.; Liu, B.; and Guan, J. 2023. All in One: Multi-task Prompting for Graph Neural Networks. arXiv:2307.01504.

Sun, X.; Tan, Y.; Wu, Q.; Chen, B.; and Shen, C. 2019a. TM-Miner: TFS-Based Algorithm for Mining Temporal Motifs in Large Temporal Network. *IEEE Access*, 7: 49778–49789.

Sun, X.; Tan, Y.; Wu, Q.; Wang, J.; and Shen, C. 2019b. New Algorithms for Counting Temporal Graph Pattern. *Symmetry*, 11: 1188.

Tang, J.; Zhang, Q.; Li, Y.; Chen, N.; and Li, J. 2025. GraphArena: Evaluating and Exploring Large Language Models on Graph Computation. arXiv:2407.00379.

Wang, J.; Sun, K.; Luo, L.; Wei, W.; Hu, Y.; Liew, A. W.-C.; Pan, S.; and Yin, B. 2024a. Large Language Models-guided Dynamic Adaptation for Temporal Knowledge Graph Reasoning. arXiv:2405.14170.

Wang, R.; Li, Z.; Sun, D.; Liu, S.; Li, J.; Yin, B.; and Abdelzaher, T. 2022. Learning to sample and aggregate: few-shot reasoning over temporal knowledge graphs. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NeurIPS '22.

Wang, R.; Li, Z.; Yang, J.; Cao, T.; Zhang, C.; Yin, B.; and Abdelzaher, T. 2023. Mutually-paced Knowledge Distillation for Cross-lingual Temporal Knowledge Graph Reasoning. In *Proceedings of the ACM Web Conference 2023*, WWW '23.

Wang, R.; Zhang, Y.; Li, J.; Liu, S.; Sun, D.; Wang, T.; Wang, T.; Chen, Y.; Kara, D.; and Abdelzaher, T. 2024b. MetaHKG: Meta Hyperbolic Learning for Few-shot Temporal Reasoning. In *Proceedings of the 47th International Conference on Research and Development in Information Retrieval*, SIGIR '24.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; and Zhou, D. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? arXiv:1810.00826.

Xuan, Q.; Fang, H.; Fu, C.; and Filkov, V. 2015. Temporal motifs reveal collaboration patterns in online task-oriented networks. *Phys. Rev. E*, 91: 052813.

Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; Dong, G.; Wei, H.; Lin, H.; Tang, J.; Wang, J.; Yang, J.; Tu, J.; Zhang, J.; Ma, J.; Yang, J.; Xu, J.; Zhou, J.; Bai, J.; He, J.; Lin, J.; Dang, K.; Lu, K.; Chen, K.; Yang, K.; Li, M.; Xue, M.; Ni, N.; Zhang, P.; Wang, P.; Peng, R.; Men, R.; Gao, R.; Lin, R.; Wang, S.; Bai, S.; Tan, S.; Zhu, T.; Li, T.; Liu, T.; Ge, W.; Deng, X.; Zhou, X.; Ren, X.; Zhang, X.; Wei, X.; Ren, X.; Liu, X.; Fan, Y.; Yao, Y.; Zhang, Y.; Wan, Y.; Chu, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; Guo, Z.; and Fan, Z. 2024. Qwen2 Technical Report. arXiv:2407.10671.

Zhang, J. 2023. Graph-ToolFormer: To Empower LLMs with Graph Reasoning Ability via Prompt Augmented by ChatGPT. arXiv:2304.11116.

Zhang, Z.; Song, L.; Bao, E.; Lv, X.; and Wang, X. 2025. ATM-GAD: Adaptive Temporal Motif Graph Anomaly Detection for Financial Transaction Networks. arXiv:2508.20829.

Zhang, Z.; Wang, X.; Zhang, Z.; Li, H.; Qin, Y.; and Zhu, W. 2024. LLM4DyG: Can Large Language Models Solve Spatial-Temporal Problems on Dynamic Graphs? arXiv:2310.17110.