Chain-of-thought Reviewing and Correction for Time Series Question Answering

Anonymous ACL submission

Abstract

With the advancement of large language models (LLMs), diverse time series analysis tasks are reformulated as time series question answering (TSQA) through a unified natural language interface. However, existing LLM-based approaches largely adopt general natural language processing techniques and are prone to reasoning errors when handling complex numerical sequences. Different from purely textual tasks, time-series questions admit objective, input-grounded checks (e.g., value comparisons and trend consistency), which enable assessing whether intermediate reasoning remains faithful to the input. Motivated by this property, we propose T3LLM, which performs multi-step reasoning with an explicit correction mechanism for TSQA. The T3LLM framework consists of three LLMs, namely, a worker, a reviewer, and a student. The worker generates step-wise chains of thought (CoT) under structured prompts, the reviewer identifies erroneous steps and provides corrective comments, and the corrected CoT are used to fine-tune the student model to internalize reasoning and self-correction into its parameters. Experiments on multiple real-world TSQA benchmarks demonstrate that T3LLM consistently outperforms strong LLM-based baselines across diverse QA types.[1]

1 Introduction

Time series analysis (TSA) is a conventional practical task for data and pattern analysis, evolving from statistical models (Newbold, 1983; De Livera et al., 2011; Mondal et al., 2014; Liu et al., 2016) to deep learning approaches (Lai et al., 2018; Han et al., 2019; Lim and Zohren, 2021; Zhou et al., 2021; Liu et al., 2022; Nie et al., 2022), driven by the increasing complexity of temporal data and continuous methodological advances (Jin et al., 2023a; Liu et al., 2024b). As a representative task of TSA, time series question answering (TSQA) frames problems in time series data as natural language questions and answers (QA), where each question typically corresponds to time series forecasting, imputation, classification, or anomaly detection (Wang et al., 2025a,d; Chen et al., 2025). Beyond numerical outputs, TSQA often requires textual explanations of the reasoning process and evidence to support reliable decision-making.

With large language models (LLMs) emerging as general-purpose reasoners (Xue and Salim, 2023; Huang et al., 2024; Shu et al., 2025), researchers explore LLMs for TSA since text and time series share one-dimensional chain structures (Zhou et al., 2023; Jin et al., 2023b; Ansari et al., 2024), enabling a unified interface that better leverages LLM reasoning. Existing LLM-based TSA approaches (Rasul et al., 2023; Jin et al., 2023b; Luo et al., 2025) are broadly fall into two categories. One introduces an external encoder to produce time series representations and fuses them with text for LLM input (Jin et al., 2023b; Hu et al., 2025; Zhao et al., 2025), which easily leads to mismatched semantic spaces across encoders and LLMs. The other directly textualizes time series data (Xue and Salim, 2023; Ansari et al., 2024; Luo et al., 2025), often by discretizing values or numeric sequences into tokens (Kojima et al., 2022; Sun et al., 2023), so that allowing LLMs to model TSA tasks within natural language processing (NLP) techniques (Ansari et al., 2024; Chow et al., 2024; Wang et al., 2025b; Luo et al., 2025) especially utilizing QA as a direct interface for information retrieval in TSA, even though with simple solutions (Xie et al., 2024; Wang et al., 2025a; Kong et al., 2025). However, long numeric sequences make models sensitive to subtle fluctuations, especially for complex questions. To avoid falling into such scenario, a reasonable attempt is to enhance the reasoning capabilities of LLMs, enabling them to break down a task into

[1] Code will be released in the final version of the paper.

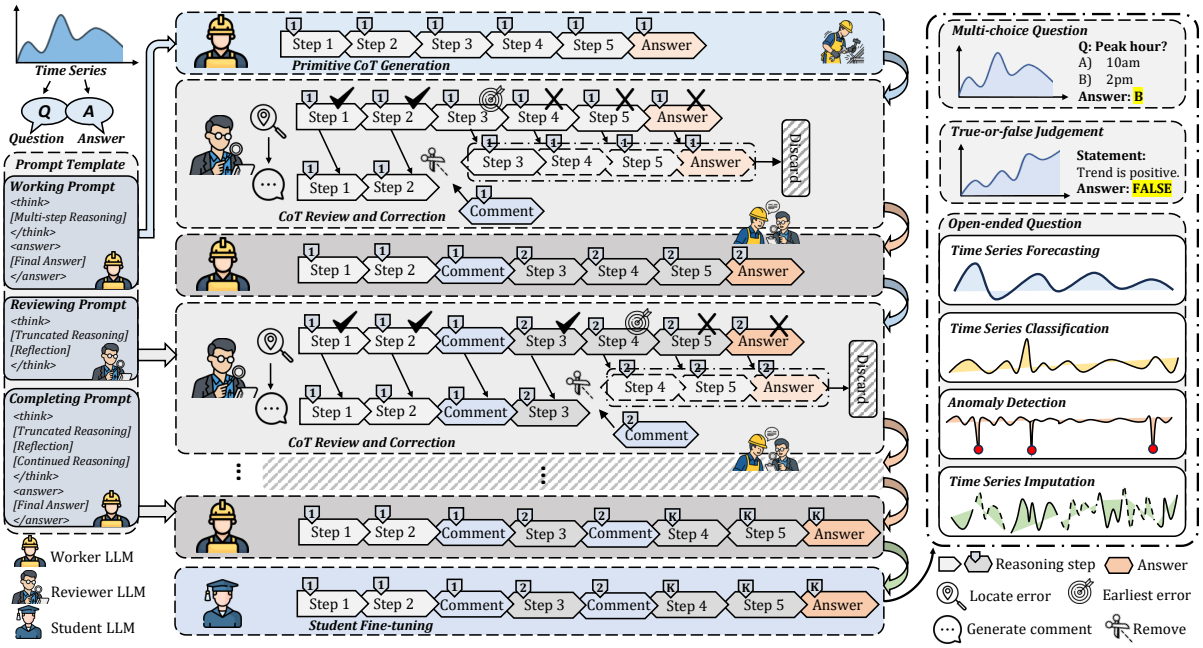


Figure 1: The illustration of our approach. The left side defines the TSQA task and the working, reviewing, and completing prompt templates. The middle depicts the overall process of T3LLM. From top to bottom, they represent the primitive CoT generation, the correction loop, and the fine-tuning based on the CoTs. The right side shows the three types of TSQA tasks (i.e., multi-choice questions, true-or-false judgments, and open-ended questions).

smaller and manageable sub-tasks (Wei et al., 2022; Zhou et al., 2022; Sprague et al., 2024), therefore, driving some studies to use chain-of-thought (CoT) in TSA to force LLMs to explicitly generate intermediate reasoning steps and prove its effectiveness accordingly (Chow et al., 2024; Wang et al., 2025b; Luo et al., 2025). Yet, they still omit the unique characteristic in TSA that time series values are verifiable, allowing consistency checking between reasoning steps and the original input, which is less accessible in purely textual reasoning. To this end, strategically tailoring LLMs' utilization to the nature of time series data is essential in effectively applying LLMs to TSA, especially TSQA.

In this paper, we propose an enhanced CoT framework named T3LLM for TSQA, which explicitly utilizes a self-corrected reasoning process. In our framework, three LLMs are assigned with distinct roles: a worker, a reviewer, and a student, respectively. We adopt a unified CoT prompt that guides the worker model to decompose the time series and question into step-wise reasoning steps, producing primitive chains of thought. The reviewer model inspects the primitive CoT, identifies the earliest incorrect reasoning step, retains only the verified correct steps, and generates corresponding comments. The worker model then completes the reasoning based on the retained correct steps and the comments to update the CoT. This review-and-completion process is repeated until the reasoning is correct or a predefined iteration limit is reached. Finally, we collect all output corrected CoTs from the reviewer LLM and utilize them to fine-tune the student LLM so that it is implicitly enhanced with CoT training and thus able to produce logically consistent multi-step reasoning for TSQA. Extensive experiments on two representative TSQA benchmark datasets demonstrate that T3LLM consistently outperforms strong LLM-based baselines across diverse QA types.

2 The Approach

Given a time series data $\mathcal{T}$, and a set of questions $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_N\}$ with their corresponding answers $\mathcal{A} = \{A_1, A_2, \ldots, A_N\}$ derived from $\mathcal{T}$, TSQA predicts an answer $\widehat{A}_n$ based on a question $Q_n \in \mathcal{Q}$ and the corresponding $\mathcal{T}$. To perform TSQA, the proposed T3LLM builds a TSQA-specific LLM with multi-step reasoning and a review-correction procedure, with the overall framework illustrated in the Figure 1. Specifically, the T3LLM framework contains three models with different roles, namely, a worker LLM f_{w}, a reviewer LLM f_{r}, and a student LLM f_{s}. The worker LLM takes an individual question and its corresponding time series as input, producing a step-wise structured primitive CoT and an initial predicted answer. The reviewer LLM examines the primitive CoT, identifies the earliest incorrect reasoning step, removes this step and all subsequent

reasoning, and generates comments for correction. Guided by the comments, the worker LLM completes generating new reasoning steps, resulting in an updated CoT and a new predicted answer. The generation and review processes alternate iteratively, forming a progressive correction loop until a predefined stopping criterion is met. The high-quality corrected CoT collected during this process is used to fine-tune the student LLM. After fine-tuning, the student LLM is able to produce logically consistent multi-step reasoning without relying on the reviewer LLM, and it is subsequently used as the final deployed model for TSQA. In the following text, we present the details of the process.

2.1 Primitive CoT Generation

In the primitive CoT generation stage, we adopt a working prompt $\mathcal{P}_W$ to regulate the reasoning structure and output format of the worker LLM. The working prompt guides the worker LLM to generate a structured primitive CoT based on a given question and its corresponding time series. The prompt introduces a set of predefined special tokens to explicitly separate reasoning content from final answer content. One category of special tokens (e.g., `<think>...</think>`) indicates the region in which the model produces step-wise reasoning. Another category of special tokens (e.g., `<answer>...</answer>`) indicates the region reserved for generating the final predicted answer. Under this unified prompt template, the primitive CoT is organized as a sequence of discrete and ordered reasoning steps, which is formulated as

$$\mathcal{R}^{(0)} = \left[R_1^{(0)}, R_2^{(0)}, \ldots, R_L^{(0)}\right] \qquad (1)$$

where $\left[\,\cdot\,\right]$ denotes the concatenation of reasoning steps, L denotes the length of the CoT, and $R_l^{(0)}$ is the l-th CoT step. At this stage, the worker LLM generates the primitive CoT together with an initial predicted answer conditioned on the question, the time series, and the generation prompt. This step-wise reasoning formulation provides a clear and systematic structural basis for subsequent reasoning review and fine-grained error localization.

2.2 CoT Review and Correction

In the CoT review and correction stage, we employ the reviewer LLM to conduct a step-by-step inspection of the primitive CoT generated by the worker LLM, with the goal of identifying the earliest reasoning error. Specifically, the reviewer LLM takes as input the question Q_n, the time series $\mathcal{T}$, the golden standard answer A_n, the primitive CoT $\mathcal{R}^{(0)}$, the corresponding predicted answer $\widehat{A}_n^{(0)}$, and a reviewing prompt $\mathcal{P}_R$, and examines the reasoning steps in a sequential manner. The reviewer LLM is prompted to locate the earliest reasoning step that conflicts with the time series evidence or the task definition. Once the erroneous reasoning step is identified, the reviewer LLM removes this step together with all subsequent reasoning steps, retaining only the prefix consisting of verified correct reasoning. The reviewer LLM then generates a comment that highlights the issues in the current reasoning and provides guidance for correction, without revealing any information about the golden standard answer. This comment is appended to the truncated CoT, forming an intermediate but corrected reasoning state, which is formulated as

$$\widetilde{\mathcal{R}}^{(1)} = f_{\mathrm{r}}\big(Q_n, \mathcal{T}, A_n, \mathcal{R}^{(0)}, \widehat{A}_n^{(0)}, \mathcal{P}_R\big) \qquad (2)$$

where $\widetilde{\mathcal{R}}^{(1)}$ denotes the truncated CoT augmented with the reviewer-generated comment. It is important to note that this intermediate CoT is still incomplete, and the remaining reasoning steps need to be regenerated by the worker LLM under the guidance of the appended comment. In the completion stage, we use a completing prompt $\mathcal{P}_C$ to instruct the worker LLM to generate new reasoning steps and an updated predicted answer conditioned on the truncated CoT. This review-truncate-comment-complete process is repeated until a predefined maximum number of correction rounds is reached. For different types of time series questions, the best corrected CoT is selected based on answer correctness or proximity to the golden standard answer. Finally, the final selected corrected CoT is used as a consistent and high-quality target supervision signal for fine-tuning the student LLM.

2.3 Student Fine-tuning

In the student fine-tuning stage, training is performed using individual corrected CoT samples as supervision. For each training instance, the student model is conditioned on the question, the time series, and the corresponding corrected CoT. The student model is trained to autoregressively predict the next token in the CoT given the preceding reasoning context. The CoT loss $\mathcal{L}_{\text{cot}}$ is computed accordingly following the standard cross-entropy loss function. After generating the CoT, the student model further predicts the token sequence of the final answer conditioned on the completed reasoning, where an answer prediction loss $\mathcal{L}_{\text{ans}}$ is com-

Dataset	Type		Train	Val	Test
CTQA	MCQ	–	26,880	6,720	14,400
TMQA	MCQ	–	6,352	1,589	3,332
	T/F	–	3,828	957	2,131
	OPE	Classification	20,719	5,180	11,101
		Anomaly	20,720	5,180	11,100
		Forecasting	23,831	5,958	12,768
		Imputation	21,647	5,412	11,598
Total	–		123,978	30,995	66,430

Table 1: Statistics of the experiment datasets. MCQ, T/F, and OPE stand for multiple-choice, true-or-false, and open-ended QA types, respectively. The open-ended QA further includes time series classification, anomaly detection, forecasting, and imputation tasks.

puted following the standard process. The overall training objective of the student model is to jointly minimize the CoT loss and the answer prediction loss as follows:

$$\mathcal{L} = \mathcal{L}_{\text{cot}} + \mathcal{L}_{\text{ans}} \tag{3}$$

By repeating this process over all training instances, the student model internalizes the multi-step reasoning structure and becomes capable of independently generating consistent reasoning for TSQA.

3 Experiment Settings

3.1 Datasets

We use two datasets in our experiments, namely, CTQA (Wang et al., 2025a) and TMQA (Kong et al., 2025), spanning multiple application domains such as finance, weather, and healthcare, etc. Both datasets consist of question–answer pairs in which the associated time series information appears in the texts. CTQA contains multiple-choice questions (MCQ) constructed from trend, volatility, seasonality, and outlier patterns. TMQA includes MCQ, true-or-false (T/F), and open-ended (OPE) QA pairs derived from typical TSA tasks, including forecasting, imputation, classification, and anomaly detection. Its time series come from public real-world data and LLM-synthesized data, and each question provides additional contextual information (e.g., domain background and series descriptions). We follow the standard protocol and split all datasets into training, validation, and test sets with a ratio of 6:1:3. Statistics of these datasets according to QA types are reported in Table 1.

3.2 Baselines and Comparing Approaches

To evaluate T3LLM, we construct four baselines on the same backbone, namely TSEnc, TSTxt, TSCoT, and TSCoT+, representing three representative LLM-based TSQA paradigms. TSEnc incorporates an independent time series encoder and concatenates its representations with text tokens as LLM input, evaluating TSQA under separate multimodal modeling. TSTxt textualizes time series as normal text and feeds them into the LLM under the standard language modeling pipeline, using the same textualization scheme as T3LLM for fair comparison. TSCoT applies CoT on top of TSTxt using the same prompt as T3LLM to generate multi-step reasoning. TSCoT+ extends TSCoT with an additional correction hint to encourage on-the-fly review of previous steps. These CoT baselines assess standard prompt-based reasoning in TSQA and serve as a direct contrast to T3LLM.

To further demonstrate the superiority of our approach, we compare three existing LLM-based TSQA studies under the same experiment setting, namely ChatTS (Xie et al., 2024), ChatTime (Wang et al., 2025a), and Time-MQA (Kong et al., 2025). ChatTS encodes time series into continuous representations and inserts them into text token sequences, thus preserving temporal context information during language modeling. ChatTime treats discretized time series values as foreign language tokens so that the numerical sequence and text are modeled jointly as if they are in a code-mixture language environment. Time-MQA directly fine-tunes LLMs with LoRA (Hu et al., 2022) on TSQA data under the standard NLP paradigm, thereby enabling the model to perform TSQA tasks.

3.3 Implementation Details

In preprocessing, we follow prior TSQA formatting (Kong et al., 2025; Wang et al., 2025b) by keeping numerical values in their original scale and placing them inside square brackets "[...]".[2] The working, reviewing, and completing prompts (i.e., $\mathcal{P}_W$, $\mathcal{P}_R$, and $\mathcal{P}_C$) used in our approach are presented in Appendix A. Since input representation plays an essential role in data processing (Mikolov et al., 2013; Devlin et al., 2019; Yang et al., 2025a), we utilize state-of-the-art LLMs in the experiments. Specifically, for the worker LLM and reviewer LLM, we use DeepSeek-R1[3] (Guo et al., 2025). For the student LLM, we employ Qwen2.5-14B-Instruct[4]. We

[2] Although some approaches normalize values (Wang et al., 2025a), we keep the original scale to better preserve magnitude, slope, and local fluctuations.

[3] https://huggingface.co/deepseek-ai/DeepSeek-R1

[4] https://huggingface.co/Qwen/Qwen2.5-14B-Instruct

Approach	CTQA	TMQA									
	MCQ	MCQ	T/F	OPE							
				Classification		Anomaly		Forecasting		Imputation	
	Acc ↑	Acc ↑	Acc ↑	Acc ↑	mF1 ↑	Acc ↑	F1 ↑	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓
TSEnc	0.399	0.483	0.518	0.320	0.274	0.286	0.246	24682.469	943.354	4668.367	94.359
TSTxt	0.365	0.279	0.539	0.387	0.257	0.398	0.368	15273.367	666.316	3008.981	75.715
TSCoT	0.390	0.574	0.635	0.495	0.293	0.513	0.393	12321.533	640.241	2900.958	72.611
TSCoT+	0.393	0.589	0.629	0.483	0.317	0.526	0.405	12359.176	658.964	2887.367	70.851
ChatTS	0.467	0.556	0.637	0.393	0.266	0.313	0.299	26543.138	934.681	4356.167	90.348
ChatTime	0.366	0.546	0.619	0.439	0.271	0.489	0.417	11926.354	628.367	2791.376	64.913
Time-MQA†	0.371	0.483	0.564	0.478	0.287	0.432	0.371	17659.672	776.128	3122.176	78.911
T3LLM	**0.665**	**0.635**	**0.766**	**0.572**	**0.336**	**0.532**	**0.494**	**7586.280**	**545.337**	**1320.905**	**50.204**

Table 2: Performance comparison of baselines and existing TSQA studies on the benchmark datasets. For the MCQ and T/F QAs, the accuracy is reported. For the classification in OPE, we report accuracy and macro-F1 (mF1). For the anomaly detection in OPE, we report accuracy and F1. For the forecasting and imputation in OPE, we report RMSE and MAE. **Boldface** and underlines denote the best and second-best results, respectively, across all baselines. "†" indicates the results of our implementation of the corresponding approach following the original paper.

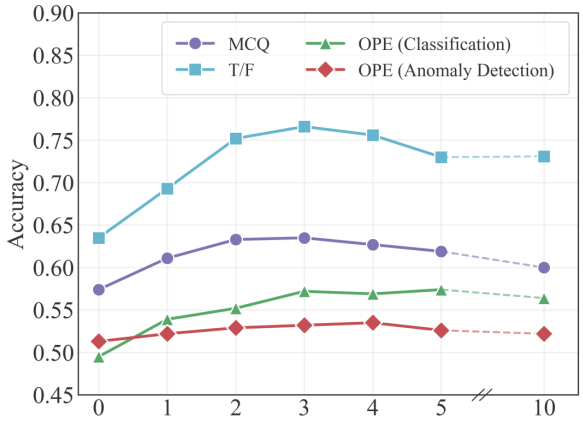
fine-tune the student LLM using LoRA (Hu et al., 2022) with LoRA rank 64, LoRA alpha 128 and LoRA dropout 0.05. We use AdamW (Loshchilov and Hutter, 2017) with a learning rate of $2e{-}5$ for 10 epochs, and set the maximum correction round ($\mathcal{MCR}$) as 3 to avoid excessively long reasoning. In evaluation, we use specific metrics for different QA types, and report the average results over three runs with different random seeds. Specifically, for MCQ, T/F, and classification and anomaly detection in OPE QA, we parse answers into discrete labels and report accuracy and macro-F1 (mF1). For OPE forecasting and imputation, we report mean absolute error (MAE) and root mean squared error (RMSE) computed on the time-series values extracted directly from the final answer text.

4 Results and Analysis

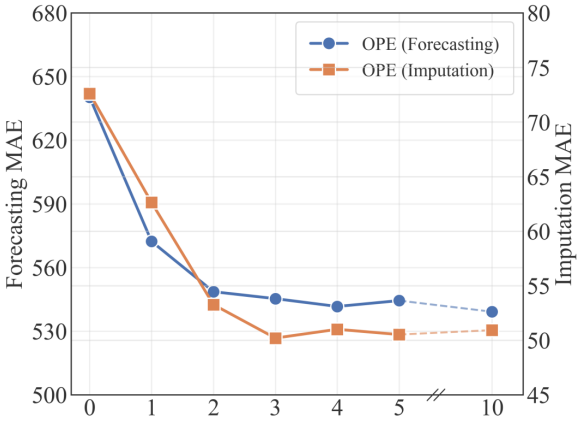
4.1 Overall Results

Table 2 reports the results of different approaches, with several observations. First, TSEnc performs well on MCQ and T/F questions, as well as classification and anomaly detection in OPE QA, but is inferior on forecasting and imputation. This suggests that an independent time series encoder provides reliable representations for comprehension QA, while semantic mismatch across encoders limits its performance when generating time-series values required by the question. Second, TSTxt shows a more balanced performance where it improves forecasting and imputation over TSEnc but is slightly worse on the other QA types. This indicates that unified textual modeling of time series and text is effective for questions requiring joint reasoning over values and language. Third, TSCoT and TSCoT+ further improve TSTxt on comprehension QA and on forecasting and imputation in OPE QA, confirming that CoT strengthens time-series reasoning. Meanwhile, TSCoT+ performs similarly to TSCoT, suggesting that prompt-only reflection is insufficient for effective self-correction. Compared to these baselines, T3LLM achieves the best performance across all QA types, indicating that enhanced CoT with evidence-based validation corrects reasoning with time series evidence.

Table 2 also reports the results of existing representative LLM-based TSQA approaches. We observe that ChatTS performs well in MCQ, T/F QA types, and the classification, anomaly detection tasks in OPE questions. This observation confirms again that using a separate encoder for time series inputs is an effective solution for comprehension QA tasks, which corresponds to that presented by TSEnc. In contrast, ChatTime and Time-MQA offer better performance in forecasting and imputation tasks, which also corresponds to TSTxt, TSCoT, and TSCoT+ in the way that unifying time series and text modeling has the advantage of being able to perceive the most original input values. Furthermore, ChatTime performs better than Time-MQA, which is because ChatTime learns time series values as foreign language tokens, so that simplifies the process of aligning time series information with texts, other than directly mixing them in LLM fine-tuning. Similar to the comparison with baseline models, T3LLM also



(a) Performance on MCQ, T/F, and OPE classification and anomaly detection tasks.



(b) Performance on OPE forecasting and imputation tasks.

Figure 2: The performance of our approach with different numbers of maximum correction rounds $\mathcal{MCR} = \{1, 2, 3, 4, 5, 10\}$ on the TMQA dataset. Figure (a) presents the results on MCQ, T/F and OPE classification and anomaly detection tasks. Figure (b) displays the performance on OPE forecasting and imputation tasks.

achieves the best performance in all QA types than existing studies, confirming its effectiveness with a carefully tailored CoT training process.

4.2 Effect of Different $\mathcal{MCR}$

To investigate the effect of the maximum correction rounds $\mathcal{MCR}$, we evaluate $\mathcal{MCR} \in \{1, 2, 3, 4, 5, 10\}$ and compare with TSCoT as the no-review CoT baseline in Figure 2. We observe that T3LLM already significantly outperforms TSCoT at $\mathcal{MCR} = 1$, and the performance further increases when raising $\mathcal{MCR}$ to 3. The gains on classification and anomaly detection in OPE gradually converge with larger $\mathcal{MCR}$, while forecasting and imputation benefit more since they require broader reasoning space and additional review to correct early biases. Still, when further increasing $\mathcal{MCR}$ to 5, the performance stays stable, which suggests that the benefit of larger $\mathcal{MCR}$ saturates all possible enhancements at its earlier stage, so that more rounds do not provide new information to help TSQA. We also test the case with 10 maximum correction rounds to explore performance under extremely round settings. We found that the performance of 10 setting is even slightly lower than that of 5 correction rounds. This indicates that long correction rounds create too many comments, making the LLM give more conservative answers and reducing performance. As a result, this analysis offers a reasonable trade-off in setting $\mathcal{MCR}$ between performance and computation cost.

4.3 Effect of Different Reviewer LLMs

We evaluate different LLMs as reviewer LLMs in our approach and conduct experiments on the TMQA dataset. We use DeepSeek-V3[5] (Liu et al., 2024a), DeepSeek-R1 (Guo et al., 2025), Qwen2.5-14B-instruct (Yang et al., 2025a), and Qwen2.5-14B-Distill[6] as reviewers, while keeping DeepSeek-R1 as the worker in all cases. Among these LLMs, DeepSeek-R1 has the strongest reasoning ability, followed by Qwen2.5-14B-Distill, while DeepSeek-V3 and Qwen2.5-14B-instruct are relatively weaker (Guo et al., 2025). Results are reported in Table 3. Replacing the reviewer with DeepSeek-V3 degrades the student's performance, particularly for forecasting and imputation, indicating that the reviewer's reasoning ability directly affects corrected CoTs and downstream TSQA. Using Qwen2.5-14B-Distill yields similar MCQ and T/F performance to DeepSeek-V3, suggesting that smaller reviewers can suffice for comprehension QA once equipped with reasoning ability. In contrast, Qwen2.5-14B-instruct further degrades forecasting and imputation, confirming that strong reasoning ability is crucial for complex QA, especially when generation is utilized in OPE questions.

4.4 Effect of Different Worker LLMs

We further evaluate different LLMs as the worker LLMs in our approach and conduct the experiment on the TMQA dataset. Similar to the analysis of review LLMs, we also select DeepSeek-V3, DeepSeek-R1, Qwen2.5-14B-instruct, and Qwen2.5-14B-distill as worker LLMs, while keeping the review LLM fixed with DeepSeek-R1. The

[5] https://huggingface.co/deepseek-ai/DeepSeek-V3

[6] https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-14B

Reviewer LLM	MCQ	T/F	OPE							
			Classification		Anomaly		Forecasting		Imputation	
	Acc↑	Acc↑	Acc↑	mF1↑	Acc↑	F1↑	RMSE↓	MAE↓	RMSE↓	MAE↓
DeepSeek-R1	**0.635**	**0.766**	**0.572**	**0.336**	**0.532**	**0.494**	**7586.280**	**545.337**	**1320.905**	**50.204**
DeepSeek-V3	0.603	0.680	0.539	0.325	0.530	0.445	8934.836	588.148	2141.449	58.174
Qwen2.5-14B-Distill	0.594	0.661	0.518	0.312	0.528	0.439	9926.251	617.422	2473.362	62.492
Qwen2.5-14B-Instruct	0.589	0.648	0.499	0.309	0.524	0.434	11874.432	622.892	2843.963	66.748

Table 3: Performance comparison of different reviewer LLMs on the TMQA dataset.

Worker LLM	MCQ	T/F	OPE							
			Classification		Anomaly		Forecasting		Imputation	
	Acc↑	Acc↑	Acc↑	mF1↑	Acc↑	F1↑	RMSE↓	MAE↓	RMSE↓	MAE↓
DeepSeek-R1	**0.635**	**0.766**	**0.572**	**0.336**	**0.532**	**0.494**	**7586.280**	**545.337**	**1320.905**	**50.204**
DeepSeek-V3	0.624	0.742	0.566	0.331	0.529	0.490	7835.424	566.142	1533.242	51.821
Qwen2.5-14B-Distill	0.591	0.639	0.525	0.324	0.525	0.474	8439.348	574.334	2214.422	64.334
Qwen2.5-14B-Instruct	0.573	0.618	0.496	0.248	0.520	0.432	12421.224	624.422	2794.996	70.382

Table 4: Performance comparison of different worker LLMs on the TMQA dataset.

results are reported in Table 4. We observe that utilizing DeepSeek-V3 as the worker model leads to a substantial performance degradation of T3LLM across all QA tasks, which indicates that strong reasoning capability is also crucial for the worker LLM in our approach, because the initial CoT is crucial in starting our review and correction process. When using Qwen2.5-14B-Distill and Qwen2.5-14B-instruct as the worker LLM, the performance on all types of question further decreases, and we find that such smaller LLMs often fail to produce correct CoT solutions in the first place, so the later correction process is unable to stop the student LLM from learning erroneous CoTs.

4.5 Case Study

In order to have a more detailed understanding of how T3LLM performs on TSQA, Figure 3 visualizes its review and correction process with a few example instances and their generated answers on the T/F question (Q1) and time series anomaly detection (Q2) and imputation (Q3) tasks in OPE questions. As a comparison, the figure also shows the responses of TSCoT+ and the baseline TSTxt on the same examples. Across all cases, T3LLM clearly shows its power on correcting critical reasoning steps to resolve inconsistencies, thus yielding more accurate answers. In contrast, TSTxt often produces limited explanatory texts, so that shows obvious deviations from ground truth on questions that require fine-grained numerical reasoning. While TSCoT+ is also able to produce multi-step reasoning, it yet lacks effective self-correction, thus often causes mistakes in the early reasoning steps that lead to subsequent error propagation later.

5 Related Work

TSA plays an important role in data and pattern analysis (Newbold, 1983; Williams et al., 1998; Hyndman et al., 2008; De Livera et al., 2011; Liu et al., 2016; Han et al., 2019; Wen et al., 2022; Jin et al., 2023a; Liu et al., 2024b), with methodologies spanning from early statistical approaches such as autoregressive integrated moving average (ARIMA) (Box and Jenkins, 1968) and exponential smoothing (Holt, 2004) that model variation patterns (Zhang, 2003; Hyndman et al., 2008; De Livera et al., 2011; Mondal et al., 2014), to deep learning solutions using convolutional or recurrent networks for nonlinear temporal relationships (Oreshkin et al., 2019; Han et al., 2019; Lim and Zohren, 2021), and more recently Transformer-based models for longer inputs (Zhou et al., 2021; Wen et al., 2022; Nie et al., 2022). With the rise of LLMs (Wei et al., 2022; Guo et al., 2025; Yang et al., 2025a), TSA further transfers to LLM-based approaches (Xie et al., 2024; Alnegheimish et al., 2024; Chow et al., 2024; Wang et al., 2025a; Kong et al., 2025; Wang et al., 2025c). Along this line, some studies replace the input and output layers of LLMs with TSA-specific modules (Rasul et al., 2023; Zhang et al., 2023; Zhou et al., 2023; Das et al., 2024; Chow et al., 2024; Xie et al., 2024; Wang et al., 2025d), while others incorporate multimodal context (e.g., text and images) via external encoders (Jin et al., 2023b; Zhong et al., 2025; Hu et al., 2025; Cheng et al., 2025). In contrast, direct solutions treat time series as text under the stan-

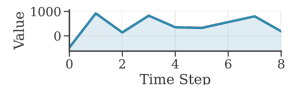
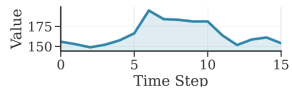
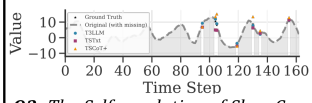
Question	T3LLM	TSCoT+	TSTxt
T/F  ***Q1***: *This dataset captures gait freezing using wearable accelerometers ... (ankle/thigh/hip, 3D acceleration, 64 Hz) ... The input time series is [-484, 921, 138, 827, 351, 323, 563, 800, 184]. Please judge whether this data corresponds to No freeze or Freeze ...* ***GT***: *No freeze.*	***[Step 1]*** *Understand the task: classify the 9-dim accelerometer snapshot into No freeze vs Freeze ...* ***[Step 2]*** *Map values to sensors/axes: ankle (−484, 921, 138), thigh (827, 351, 323) ...* ***[Step 3]*** *Consider patterns ...* ***[Reflection]*** *Magnitude-only reasoning is over-simplified; freezing patterns may not correlate directly with intensity ...* ***[Step 5]*** *I avoid relying solely on magnitude comparisons ...* ***[Answer]*** *Conclude No freeze ...*	***[Step 1]*** *Classify the 9-dim ...* ***[Step 2]*** *Map values to sensors/axes* ***[Step 3]*** *The ankle forward acceleroation is negative ...* ***[Step 4]*** *. Such cross-sensor disagreement can indicate gait instability ... which may align with a freezing-like event ...* ***[Step 5]*** *The vertical accelerations are high, potentially reflecting abrupt loading or jittery stepping, which can appear during freezing episodes ...* ***[Answer]*** *Freeze.*	*Given the strong and inconsistent acelerations across ankle, thigh, hip (e.g., ankle forward is negative while thigh/hip forward are strongly positive, and vertical components are high), this pattern is more consistent with an unstable gait event rather than smooth walking ... Therefore, the answer is freeze.*
OPE (Anomaly Detection)  ***Q2***: *This data originates from monitoring systems ... IOPS metrics ... determine whether anomalies occur-red ... The input time series is [155.88, 152.74, 148.81, 151.97, 157.49, 166.18, 194.73, 184.03, 183.2, 181.11, 181.16, 163.75, 151.68, 158.57, 161.11, 153.98] ...* ***GT***: *Anomaly Point (anomaly exists)*	***[Step 1]*** *Decide whether the series contains anomalies (binary) ...* ***[Step 2]*** *Analyze the time series ...* ***[Step 3]*** *Compute basic statistics to ...* ***[Reflection]*** *A specific thresholding method (e.g., IQR fences) is not specified; reasoning should rely more on contextual pattern rather than arbitrary rules ...* ***[Step 5]*** *Consider the context of IOPS metrics ...* ***[Step 6]*** *.. extreme spike could be indicative of an anomaly ...* ***[Answer]*** *Anomaly exists.*	***[Step 1]*** *Determine whether the given IOPS time series contains anomalies ...* ***[Step 2]*** *The series ranges from ...* ***[Step 3]*** *The series shows a gradual increase from ...* ***[Step 4]*** *This looks like a work-load shift ...* ***[Step 5]*** *... the system recovers smoothly without abrupt oscillations or collapse-like behavior ...* ***[Answer]*** *False (No anomaly).*	*The IOPS values rise to a peak and then remain elevated for several points before gradually returning, which resembles a normal workload increase rather than an isolated abnormal outlier ... The predicted answer is False (No anomaly).*
OPE (Imputation)  ***Q3***: *The Self-regulation of Slow Cortical Potentials dataset. The input Time Series are [5.28, 6.0, ..., 6.56, 8.84, ..., 6.56, 'X', 10.16, ..., 12.31, 'X', 'X', 4.81, ..., -5.69, 'X', ..., 9.88, 'X', 6.06, ..., 2.38, 'X', 3.5, ... The missing values are represented by 'X'."* ***GT***: *... imputed are [5.28, 6.0, ...*	***[Step 1]*** *The question asks to provide the full time* ***[Step 2]*** *The time series is from ...* ***[Step 3]*** *I'll use linear interpolation ...* ***[Reflection]*** *The data exhibits non-linear trends ... involve analyzing the local patterns ...* ***[Step 4]*** *use a local averaging that considers multiple neighboring points to ...* ***[Step 5]*** *Apply the local averaging method to ...* ***[Answer]*** *[..., 9.30, ..., 11.91, 8.99, ...*	***[Step 1]*** *The question asks to impute missing values marked as 'X' ...* ***[Step 2]*** *Identify each 'X' in the sequence and treat them ...* ***[Step 3]*** *Use linear interpolation between the nearest observed values surrounding each missing point ...* ***[Step 4]*** *For a single X in ...* ***[Step 5]*** *... output the completed sequence.* ***[Answer]*** *[..., 6.66, ..., 4.91, 4.73, ...*	*I fill each missing 'X' using linear interpolation (evenly spaced values for consecutive gaps), then return the completed series ... The completed series is [..., 7.41, ..., 13.52, 15.18, ...*

Figure 3: Case study with four examples on MCQ and T/F questions, and time series forecasting and classification tasks in OPE questions from the TMQA dataset. The left side presents the question and the corresponding time series. The right side shows Time-MQA, TSCoT, and T3LLM's reasoning process and their corresponding answer.

dard language modeling paradigm (Ansari et al., 2024; Jin et al., 2024; Wang et al., 2025a; Kong et al., 2025), and further adopt advanced NLP techniques such as retrieval augmentation (Yang et al., 2025b; Xiao et al., 2025; Han et al., 2025), and CoT (Wang et al., 2025b; Luo et al., 2025). As a natural interface, TSQA is proposed to tackle TSA with natural language interactions. Xie et al. (2024) construct TSQA data for time series trend prediction and causal inference, aligning time series with textual descriptions for diverse QA capabilities. Wang et al. (2025a) introduces a vocabulary for time-series numerical values and inserts them in LLM training, mainly focusing on MCQ questions. Kong et al. (2025) finetunes an LLM with self-constructed TSQA data via varying prompts to handle forecasting, anomaly detection, and related tasks. These studies rely on relatively simple QA solutions and exhibit poor adaptability to complex time series questions. Compared to them, T3LLM offers a more reliable framework in utilizing LLMs to mitigate the vulnerability of their operations more effectively with decomposition of reasoning steps and correction of them.

6 Conclusion

In this paper, we propose T3LLM, reviewing and correcting multi-step reasoning to improve LLMs' ability for TSQA. T3LLM first generates primitive CoTs with a worker LLM, then iteratively refines them via a reviewer LLM, and uses the corrected CoTs to fine-tune a compact student LLM, internalizing self-correction into its parameters. Experiments on two TSQA benchmarks show that T3LLM consistently and substantially outperforms existing LLM-based approaches across multiple tasks, validating review-based CoT correction.

Limitations

While T3LLM delivers strong performance on TSQA, several practical considerations remain. First, T3LLM involves multiple LLM roles (worker, reviewer, and student), which imposes higher computational resource requirements during CoT generation and correction. Second, although the context window is sufficient in our experiments, where the average CoT length is 2,479 tokens and the maximum is 11,395 tokens and both are below the LLM's maximum context length of 32,768 tokens, the context limit could still theoretically become a potential bottleneck in more demanding settings.

References

Sarah Alnegheimish, Linh Nguyen, Laure Berti-Equille, and Kalyan Veeramachaneni. 2024. Large language models can be zero-shot anomaly detectors for time series? *arXiv preprint arXiv:2405.14755*.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, and Shubham Kapoor. 2024. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*.

George EP Box and Gwilym M Jenkins. 1968. Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 17(2):91–109.

Jialin Chen, Aosong Feng, Ziyu Zhao, Juan Garza, Gaukhar Nurbek, Cheng Qin, Ali Maatouk, Leandros Tassiulas, Yifeng Gao, and Rex Ying. 2025. Mtbench: A multimodal time series benchmark for temporal reasoning and question answering. *arXiv preprint arXiv:2503.16858*.

Mingyue Cheng, Yiheng Chen, Qi Liu, Zhiding Liu, Yucong Luo, and Enhong Chen. 2025. Instructime: Advancing time series classification with multimodal language modeling. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pages 792–800.

Winnie Chow, Lauren Gardiner, Haraldur T Hallgrímsson, Maxwell A Xu, and Shirley You Ren. 2024. Towards time series reasoning with llms. *arXiv preprint arXiv:2409.11376*.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. 2024. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*.

Alysha M De Livera, Rob J Hyndman, and Ralph D Snyder. 2011. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American statistical association*, 106(496):1513–1527.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, and Xiao Bi. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Sungwon Han, Seungeon Lee, Meeyoung Cha, Sercan O Arik, and Jinsung Yoon. 2025. Retrieval augmented time series forecasting. *arXiv preprint arXiv:2505.04163*.

Zhongyang Han, Jun Zhao, Henry Leung, King Fai Ma, and Wei Wang. 2019. A review of deep learning models for time series prediction. *IEEE Sensors Journal*, 21(6):7833–7848.

Charles C Holt. 2004. Forecasting seasonals and trends by exponentially weighted moving averages. *International journal of forecasting*, 20(1):5–10.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Yuxiao Hu, Qian Li, Dongxiao Zhang, Jinyue Yan, and Yuntian Chen. 2025. Context-alignment: Activating and enhancing llm capabilities in time series. *arXiv preprint arXiv:2501.03747*.

Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. 2024. Vtimellm: Empower llm to grasp video moments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14271–14280.

Rob Hyndman, Anne Koehler, Keith Ord, and Ralph Snyder. 2008. *Forecasting with exponential smoothing: the state space approach*. Springer.

Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. 2023a. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE transactions on knowledge and data engineering*, 36(10):5388–5408.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, and Shirui Pan. 2023b. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*.

Ming Jin, Yifan Zhang, Wei Chen, Kexin Zhang, Yuxuan Liang, Bin Yang, Jindong Wang, Shirui Pan, and Qingsong Wen. 2024. Position: What can large language models tell us about time series analysis. In *41st International Conference on Machine Learning*. MLResearchPress.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Yaxuan Kong, Yiyuan Yang, Yoontae Hwang, Wenjie Du, Stefan Zohren, Zhangyang Wang, Ming Jin, and Qingsong Wen. 2025. Time-mqa: Time series multi-task question answering with context enhancement. *arXiv preprint arXiv:2503.01875*.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104.

Bryan Lim and Stefan Zohren. 2021. Time-series forecasting with deep learning: a survey. *Philosophical transactions of the royal society a: mathematical, physical and engineering sciences*, 379(2194).

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, and Chong Ruan. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Chenghao Liu, Steven CH Hoi, Peilin Zhao, and Jianling Sun. 2016. Online arima algorithms for time series prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Prabhakar Kamarthi, Aditya Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, and Chao Zhang. 2024b. Time-mmd: Multi-domain multimodal dataset for time series analysis. *Advances in Neural Information Processing Systems*, 37:77888–77933.

Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. 2022. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yucong Luo, Yitong Zhou, Mingyue Cheng, Jiahao Wang, Daoyu Wang, Tingyue Pan, and Jintao Zhang. 2025. Time series forecasting as reasoning: A slow-thinking approach with reinforced llms. *arXiv preprint arXiv:2506.10630*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Prapanna Mondal, Labani Shit, and Saptarsi Goswami. 2014. Study of effectiveness of time series modeling (arima) in forecasting stock prices. *International Journal of Computer Science, Engineering and Applications*, 4(2):13.

Paul Newbold. 1983. Arima model building and the time series analysis approach to forecasting. *Journal of forecasting*, 2(1):23–35.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2022. A time series is worth 64words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.

Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2019. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, and Nadhir Hassen. 2023. Lag-llama: Towards foundation models for probabilistic time series forecasting. *arXiv preprint arXiv:2310.08278*.

Fangxun Shu, Lei Zhang, Hao Jiang, and Cihang Xie. 2025. Audio-visual llm for video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4246–4255.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *arXiv preprint arXiv:2409.12183*.

Chenxi Sun, Hongyan Li, Yaliang Li, and Shenda Hong. 2023. Test: Text prototype aligned embedding to activate llm's ability for time series. *arXiv preprint arXiv:2308.08241*.

Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun, Zirui Zhuang, Jinming Wu, Lei Zhang, and Jianxin Liao. 2025a. Chattime: A unified multimodal time series foundation model bridging numerical and textual data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12694–12702.

Jiahao Wang, Mingyue Cheng, and Qi Liu. 2025b. Can slow-thinking llms reason over time? empirical studies in time series forecasting. *arXiv preprint arXiv:2505.24511*.

Xinlei Wang, Mingtian Tan, Jing Qiu, Junhua Zhao, and Jinjin Gu. 2025c. Toward reasoning-centric time-series analysis. *arXiv preprint arXiv:2510.13029*.

Yilin Wang, Peixuan Lei, Jie Song, Yuzhe Hao, Tao Chen, Yuxuan Zhang, Lei Jia, Yuanxiang Li, and Zhongyu Wei. 2025d. Itformer: Bridging time series and natural language for multi-modal qa

with large-scale multitask dataset. *arXiv preprint arXiv:2506.20093*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2022. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*.

Billy M Williams, Priya K Durvasula, and Donald E Brown. 1998. Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models. *Transportation Research Record*, 1644(1):132–141.

Mengxi Xiao, Zihao Jiang, Lingfei Qian, Zhengyu Chen, Yueru He, Yijing Xu, Yuecheng Jiang, Dong Li, Ruey-Ling Weng, and Min Peng. 2025. Retrieval-augmented large language models for financial time series forecasting. *arXiv preprint arXiv:2502.05878*.

Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. 2024. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *arXiv preprint arXiv:2412.03104*.

Hao Xue and Flora D Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6851–6864.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, and Haoran Wei. 2025a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Silin Yang, Dong Wang, Haoqi Zheng, and Ruochun Jin. 2025b. Timerag: Boosting llm time series forecasting via retrieval-augmented generation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

G. Peter Zhang. 2003. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175.

Yunkai Zhang, Yawen Zhang, Ming Zheng, Kezhen Chen, Chongyang Gao, Ruian Ge, Siyuan Teng, Amine Jelloul, Jinmeng Rao, and Xiaoyuan Guo. 2023. Insight miner: A large-scale multimodal model for insight mining from time series. In *NeurIPS 2023 AI for Science Workshop*.

Taibiao Zhao, Xiaobing Chen, and Mingxuan Sun. 2025. Enhancing time series forecasting via multi-level text alignment with llms. *arXiv preprint arXiv:2504.07360*.

Siru Zhong, Weilin Ruan, Ming Jin, Huan Li, Qingsong Wen, and Yuxuan Liang. 2025. Time-vlm: Exploring multimodal vision-language models for augmented time series forecasting. *arXiv preprint arXiv:2502.04395*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, and Quoc Le. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115.

Tian Zhou, PeiSong Niu, Xue Wang, Liang Sun, and Rong Jin. 2023. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*.

A Prompt Templates

We provide three complementary prompt templates used in the T3LLM framework: the working template (Figure 4), the reviewing template (Figure 5), and the completing template (Figure 6).

Working Prompt Template

```
System prompt:
You are an expert in time series analysis and time series question answering (TSQA).

For every query:
- Carefully read the question, including the dataset description, task description, label or answer definitions, and the given time series.
- Perform explicit step-by-step reasoning ONLY inside the <think>...</think> block.
- Ground your reasoning in the numeric time series and the task definitions, not in external knowledge or vague intuition.
- After finishing the reasoning, output the final answer ONLY inside the <answer>...</answer> block.
- The final answer must strictly follow the format requested in the question (e.g., a single label, one of the given options, True/False, or a list of numbers), and should be as concise as possible.
- Do NOT include the final answer text inside <think>. Do NOT repeat the reasoning inside <answer>.

User prompt:
You are given a time series question answering (TSQA) instance.

[Question]
{add question here}

Your task:
1. Understand the task type and the required answer format from the question.
2. Carefully analyze the given time series (level, trend, variation, local anomalies, missing values, etc.) together with the context in the question.
3. Use step-by-step reasoning, grounded in the time series and task definitions, to decide the correct answer.
4. At the end, give your final answer in exactly the format requested by the question (for example: a single label, one of the given options, True/False, or a list of numbers).

Output format:
<think>
[Step 1] ...
[Step 2] ...
[Step 3] ...
[Step 4] ...
(Use as many steps as needed for your reasoning. Do NOT output the final answer here.)
</think>
<answer>
[Final answer only, strictly following the required format.]
</answer>
```

Figure 4: The working prompt template used in T3LLM.

Reviewing Prompt Template

```
System prompt:
You are a rigorous reviewer of time-series reasoning traces.

For each case, you will receive:
- A question (including the time series and task description),
- The ground-truth answer,
- A model's previous output containing a reasoning chain in <think>...</think> and a final answer
in <answer>...</answer>.

Your job is to:
- Decide whether the reasoning chain contains any incorrect, unsupported, or misleading step, given
the question, the time series, and the ground-truth answer.
- If there is an error, you must locate the FIRST problematic step, insert a self-reflection after
that step, and truncate all later steps. You must NOT produce any final answer.
- If there is no error and the final answer matches the ground-truth answer, you must NOT modify
the reasoning at all and you must NOT reveal or hint at the ground-truth answer. In this case, just
output a special marker indicating that no change is needed.

Important constraints:
- Never reveal or explicitly mention the ground-truth answer in your output.
- Never hint which label, option, or numeric values are correct.
- Do not output any <answer> block.

User prompt:
[Question]
{original question}

[Ground-truth Answer]
{gold answer}

[Original Model Output]
<think>
{working llm's reasoning chain}
</think>
<answer>
{working llm's final answer}
</answer>

Your reviewing task:

1. Carefully read the question, the time series, and the ground-truth answer.
2. Check whether the final answer in <answer> matches the ground-truth answer exactly.
3. Read the reasoning steps inside <think> in order, from the beginning to the end.
4.  Identify the FIRST step that is logically wrong, inconsistent with the given time series,
inconsistent with the task definitions, or that leads toward an incorrect or unjustified conclusion.
    - A step is "wrong" if it mis-describes the numeric pattern, misapplies a definition, or
supports a conclusion that contradicts the correct answer.
5. If you find such an incorrect or problematic step, construct a revised thinking trace as follows:
    - Copy all correct steps BEFORE the first wrong step without changing their content (you may
keep existing step markers like [Step 1], [Step 2], etc.).
    - Copy the first wrong step as well.
    - Immediately AFTER this wrong step, insert a new line starting with [Reflection] that:
        * explicitly explains why the previous step is incorrect, imprecise, or misleading,
        * briefly indicates in abstract terms what kind of reasoning direction would be more
appropriate, WITHOUT revealing or hinting at the ground-truth answer (no explicit labels, options,
or exact numeric targets).
    - DELETE all remaining steps after this reflection and DELETE the final answer. Do NOT continue
the reasoning beyond the reflection.
6.  If you CANNOT find any real error in the reasoning and the final answer already matches the
ground-truth answer:
    - Do NOT modify, repeat, or summarize the original reasoning.
    - Do NOT output any explanation, reflection, or hint about why it is correct.
    - Simply output a special marker indicating that no change is needed.
Notes:
- Never output any <answer> block.
-  Never reveal or hint at the ground-truth answer (no explicit correct label, option, or full
numeric sequence).
```

Figure 5: The reviewing prompt template used in T3LLM.

Completing Prompt Template

```
System prompt:
You are an expert in time series analysis and time series question answering (TSQA).

For every query:
- Carefully read the question, including the dataset description, task description, label or answer
definitions, and the given time series.
- You will receive a partially corrected reasoning trace inside <think>...</think>. This trace may
already contain several steps and one or more [Reflection] lines that point out previous errors or
adjustments.
- You must treat the existing content inside <think> as fixed. Do NOT modify, delete, or reorder
any existing steps or reflections.
- Your job is to CONTINUE the reasoning from the end of the given <think> block, adding new steps
that follow the corrections suggested by the latest [Reflection].
- Explicit step-by-step reasoning must appear ONLY inside the <think>...</think> block.
- After finishing the reasoning, output the final answer ONLY inside the <answer>...</answer> block.
- The final answer must strictly follow the format requested in the question (e.g., a single label,
one of the given options, True/False, or a list of numbers), and should be as concise as possible.
- Do NOT include the final answer text inside <think>. Do NOT repeat the reasoning inside <answer>.

User prompt:
You are given a time series question answering (TSQA) instance together with a partially corrected
reasoning trace.

[Question]
{original question}

[Partially corrected reasoning trace]
<think>
{revised reasoning chain}
</think>

Your task:
1. Carefully read the question and understand the task type and the required answer format.
2. Read the existing reasoning inside <think>. Earlier steps and [Reflection] lines indicate which
directions were previously problematic and how the reasoning should be adjusted in general.
3. WITHOUT changing any existing content inside <think>, continue the reasoning by adding new steps
AFTER the last existing line inside <think>.
    - Your new steps should respect the corrections suggested by the latest [Reflection].
    - Avoid repeating the same errors that were pointed out by any [Reflection].
4. Based on the completed reasoning, decide the correct final answer in the exact format required
by the question.

Output format:
<think>
{First, copy the entire given reasoning trace EXACTLY as it is, including all previous steps and
[Reflection] lines.}
[Next Step] ...
[Next Step] ...
(... add as many new steps as needed; do NOT include the final answer text here.)
</think>
<answer>
[Final answer only, strictly following the required format.]
</answer>
```

Figure 6: The completing prompt template used in T3LLM.