https://doi.org/10.1007/sxxxxx-yyy-zzzz-1

RESEARCH ARTICLE

Reasoning Pattern Matters: Learning to Reason without Human Rationales

Chaoxu PANG[1], Yixuan CAO[1] ✉, Ping LUO[1] ✉

1. Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences.

Received month dd, yyyy; accepted month dd, yyyy

E-mail: {caoyixuan, luop}@ict.ac.cn.

Abstract

Large Language Models (LLMs) have demonstrated remarkable reasoning capabilities under the widely adopted SFT+RLVR paradigm, which first performs Supervised Fine-Tuning (SFT) on human-annotated reasoning trajectories (rationales) to establish initial reasoning behaviors, then applies Reinforcement Learning with Verifiable Rewards (RLVR) to further optimize the model using verifiable reward signals without golden rationales. However, annotating high-quality rationales for the SFT stage remains prohibitively expensive. This paper investigates when and how the cost of rationale annotations can be substantially reduced without compromising reasoning performance. We identify a broad class of problems, termed *patterned reasoning tasks*, where the reasoning process follows a fixed, procedural solution strategy that remains consistent across all instances of the same task. Although individual instances vary in their content,such as domain knowledge, factual information, or numeric values, the solution is obtained by applying a shared reasoning pattern to instance-specific inputs. We argue that the success of SFT+RLVR on such tasks primarily stems from its ability to enable the model to internalize the underlying reasoning patterns. Using numerical semantic matching as a representative task, we provide two complementary lines of evidence: (i) from a *cause-side* perspective, controlled experiments demonstrate that reasoning patterns, rather than the quantity or quality of rationales, are the dominant factor driving performance; and (ii) from an *effect-side* perspective, forking-token analysis reveals that models trained with SFT+RLVR exhibit more task-relevant reasoning behaviors, indicating stronger alignment with the task's inherent reasoning pattern. Building on these insights, we propose **PARO** (**P**attern-**A**ware LLMs as **R**ationale Ann**O**tators), a simple yet effective framework that enables LLMs to generate rationales aligned with task-specific reasoning patterns *without requiring human rationale annotations*. Experiments on two patterned reasoning tasks show that PARO-generated rationales achieve comparable SFT+RLVR performance to human rationales that are 10× larger. These results suggest a paradigm shift: for patterned reasoning tasks, large-scale human rationale annotations can be replaced with LLM-based automatic annotations, requiring only limited human supervision over reasoning patterns.

Key words

Large Language Models, Reinforcement Learning, Numerical Semantic Matching

1 Introduction

Large Language Models (LLMs) have recently achieved remarkable progress on complex reasoning-intensive tasks such as mathematics [1–4] and coding [2,5]. A key driver of this progress is the standard two-stage training paradigm of **SFT+RLVR**. In the first stage, Supervised Fine-Tuning (SFT), models are exposed to high-quality reasoning trajectories (rationales) to encourage explicit reasoning behaviors. In the second stage, Reinforcement Learning with Verifiable Rewards (RLVR) [6], reinforcement learning algorithms such as GRPO [1] or PPO [7] further optimize the model using rule-based reward signals derived from verifiable (question, answer) pairs, without requiring golden rationales. However, annotating large-scale, high-quality rationales for the SFT stage remains prohibitively expensive [6, 8], raising a fundamental question: *when and how can rationale annotation costs be reduced without compromising reasoning performance?*

We address this question for a broad class of problems that we term **patterned reasoning tasks**—tasks where the reasoning process follows a fixed, procedural solution strategy that remains consistent across all instances of the same task, while individual instances vary in their content such as domain knowledge, factual information, or numeric values. The solution is obtained by applying a shared reasoning pattern to instance-specific inputs. Many criterion-driven problems fall into this category. Examples include text classification (e.g., topic classification [9] with detailed category definitions), verification tasks that typically follow fixed verification routines [10], and information extraction problems governed by predefined schemas [11, 12]. Such tasks are particularly prevalent in professional domains where decision-making workflows are well-defined, such as medical diagnosis pipelines [13], financial auditing processes [14], and scientific information extraction [15]. In each case, different inputs invoke the

same decision or extraction procedure, so model performance is determined largely by how well the model learns and applies the shared pattern to variable content. We show more details and example tasks in Section 3. By contrast, open-ended or adaptive reasoning tasks, such as heterogeneous mathematical problem solving [1], competitive programming challenges [2, 5], and complex planning problems [16], require selecting or adapting solution strategies on a per-instance basis, and therefore cannot be captured by a single fixed reasoning pattern.

We posit that the effectiveness of SFT+RLVR on patterned reasoning tasks arises from its ability to enable the model to internalize the underlying reasoning patterns. Using numerical semantic matching as a representative task, we present two complementary lines of evidence:

1. **Reasoning pattern, rather than rationale quantity or quality, is the dominant factor driving SFT+RLVR performance.** From a *cause-side* perspective, controlled experiments reveal that: (i) reducing human-annotated rationales for SFT by an order of magnitude (10× fewer samples) while preserving the reasoning pattern leads to negligible performance degradation; and (ii) randomly corrupting a substantial portion of rationales (e.g., 25%) while maintaining the pattern yields minimal impact. These findings suggest that LLMs primarily learn *how* to reason—the procedural pattern of thought—rather than memorizing instance-specific rationale content.
2. **Models trained with SFT+RLVR exhibit more task-relevant reasoning behaviors, indicating stronger alignment with the task's inherent reasoning pattern.** From an *effect-side* perspective, we analyze model reasoning behavior through *forking tokens* [17]—key tokens that mark decision points in reasoning trajectories and serve as indicators of reasoning pattern comprehension [17, 18]. Models trained with SFT+RLVR produce forking tokens that are substantially more task-relevant, demonstrating deeper alignment with the task's inherent reasoning pattern. In contrast, models trained solely with RLVR or hint-based methods [19] tend to generate generic discourse connectors (e.g., "but," "because"), reflecting a lack of focus on the core reasoning patterns essential for task completion.

Building on these insights, we introduce **P**attern-**A**ware LLMs as **R**ationale Ann**O**tators (**PARO**), a simple yet effective framework that explicitly instructs strong LLMs to generate rationales following task-specific reasoning patterns. We evaluate PARO on two representative patterned reasoning tasks—numerical semantic matching and transaction purpose classification. PARO achieves comparable performance to human rationale datasets that are 10× larger, while not requiring human rationale annotations, and outperforms approaches that distill internal reasoning trajectories from large reasoning models such as Qwen3-235B-A22B-Thinking [20].

Taken together, our findings empirically demonstrate that for patterned reasoning tasks, the central challenge lies not in collecting more high-quality rationales, but in defining and enforcing clear reasoning patterns. Through extensive experimental analysis, we show that this insight preserves model performance while dramatically reducing annotation costs, providing a practical and scalable pathway for reasoning supervision in LLMs.

2 Preliminaries

Reinforcement Learning from Verifiable Rewards (RLVR) [6] has emerged as a widely adopted paradigm for enhancing the reasoning capabilities of LLMs. Prior to RLVR, models are typically warm-started with a **Supervised Fine-Tuning (SFT)** stage, which encourages explicit, human-readable reasoning trajectories.

Concretely, the SFT stage requires a small collection of question–rationale–answer triples $\mathcal{D}_r = \{(q, r, a)\}$, where q, r, and a denote the question, rationale, and answer, respectively. To ensure high quality, rationales are typically annotated by human experts. The SFT objective fine-tunes the model by maximizing the likelihood of generating a concatenated rationale–answer sequence $f_e(r, a)$ conditioned on the question q:

$$\theta^{(1)} = \arg\max_{\theta} \mathbb{E}_{(q,r,a)\sim\mathcal{D}_r} \left[\log \pi_\theta(f_e(r, a) \mid q)\right]. \tag{1}$$

Subsequently, the **RLVR stage** initializes from $\theta^{(1)}$ and further optimizes the model on a larger dataset of question–answer pairs $\mathcal{D}_d = \{(q, a)\}$ using the RLVR objective:

$$\max_{\theta} \ \mathbb{E}_{(q,a)\sim\mathcal{D}_d} \ \mathbb{E}_{y\sim\pi_\theta(\cdot|q)} \left[R_{\text{RLVR}}(q, a, y)\right], \tag{2}$$

where y denotes the response sampled from the policy π_θ. The core principle underlying the RLVR objective is straightforward: the model receives reward only when its generated response is *verifiably correct*. Formally, the RLVR reward function can be expressed as:

$$R_{\text{RLVR}}(q, a, y) = v(y, a) - \beta\, \text{KL}\big(\pi_\theta(y \mid q) \,\|\, \pi_{\text{ref}}(y \mid q)\big), \tag{3}$$

where v is a verifiable reward function. A commonly used instantiation compares the extracted final answer from y against the ground truth a:

$$v(y, a) = \begin{cases} 1, & \text{if extract}(y) = a, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Here, extract$(\cdot)$ denotes an answer-extraction function that parses the final prediction from the generated rationale. This objective is typically optimized using policy-gradient algorithms such as PPO [7] or GRPO [1].

3 Patterned Reasoning Tasks

In this section, we formally characterize patterned reasoning tasks and distinguish them from adaptive reasoning tasks, establishing the conceptual framework that defines the scope of our study.

3.1 Definition and Characteristics

We define **patterned reasoning tasks** as tasks where the overall reasoning process follows a fixed, procedural solution strategy across all instances. While the reasoning content may vary between instances and require diverse knowledge (e.g., commonsense, domain-specific expertise, or deductive logic), the underlying *reasoning pattern* remains invariant. In contrast, **adaptive reasoning tasks** require the reasoning process to flexibly adapt to instance-specific characteristics,

resulting in diverse solution strategies that cannot be captured by a single fixed pattern.

Formally, a patterned reasoning task can be characterized by:

- **Stable reasoning pattern** $\mathcal{P}$: a consistent procedural framework applicable across all task instances.
- **Instance-specific content** C_i: variable knowledge, facts, or values required for each instance i.
- **Pattern execution function** $f(\mathcal{P}, C_i) \rightarrow y_i$: the application of $\mathcal{P}$ to C_i to produce output y_i.

The key distinction lies in whether the reasoning pattern $\mathcal{P}$ depends on the instance. For patterned reasoning tasks, $\mathcal{P}$ remains consistent across instances, whereas adaptive reasoning tasks require $\mathcal{P}_i$ to be flexibly chosen or adapted for each instance i.

3.2 Task Categories

We provide representative examples of the two reasoning tasks to illustrate their typical task types.

- Adaptive reasoning tasks.
  - **Mathematical problem solving** [1]: Diverse problems requiring distinct solution strategies.
  - **Programming tasks** [2,5]: Coding problems demanding varied algorithmic designs and implementation approaches.
  - **Planning tasks** [16]: Sequential decision-making problems where different initial states or goals necessitate distinct strategic approaches.
- Patterned reasoning tasks.
  - **Criterion-driven problems**: Tasks with predefined categorical definitions or explicit discrimination criteria:
    - *Classification problems*: Tasks such as sentiment analysis [21], topic classification [22], and intent recognition [23], governed by fixed label sets and decision criteria. These tasks rely on consistent feature patterns and boundary conditions to assign inputs to predefined categories.
    - *Verification problems*: Tasks such as fact-checking [24, 25], following clear verification protocols. The criteria involve cross-referencing claims against reliable sources and applying logical consistency checks to determine truthfulness.
    - *Extraction problems*: Tasks such as relation extraction [26], event extraction [11], and table information extraction [27, 28], which follow consistent annotation schemas. These tasks operate under predefined entity types, relationship categories, and structural templates that guide the identification and extraction process.
  - **Deductive reasoning**: Tasks applying established rules to derive conclusions:
    - *Logical reasoning*: Tasks such as symbolic reasoning [29].
    - *Mathematical calculations*: Tasks such as geometric or algebraic computations [30].
    - *Algorithmic simulation*: Tasks such as executing well-defined stepwise procedures [31].

Patterned reasoning tasks are particularly prevalent in professional and specialized domains, where decision-making workflows are typically well-defined and domain experts follow relatively standardized procedures (e.g., medical diagnosis pipelines [13], financial auditing processes [14], or scientific information extraction [15]).

In this work, we focus on two representative patterned reasoning tasks from the financial domain: numerical semantic matching and transaction purpose classification. We construct new datasets with carefully annotated labels and rationales. Unlike many widely studied benchmarks, these tasks have limited exposure in current pre-training corpora, which helps mitigate both data contamination (where models memorize benchmark samples) and task contamination (where models implicitly learn task-specific strategies from repeated exposure to public benchmarks). We present detailed analyses of the task definitions and their underlying reasoning patterns in the following subsections.

3.3 Numerical Semantic Matching

Numerical mentions are pervasive in documents such as financial reports, regulatory filings, and web pages, where they play a central role in statistical reporting and disclosure. **Numerical Semantic Matching (NSM)** is the task of determining whether two numerical mentions are *semantically equivalent*, i.e., whether they refer to the same underlying numerical fact.

As shown in Figure 1, two distinct mentions may appear in different parts of a company's annual report, yet both correspond to the identical fact that *the treasury share component within BHP Group Limited's closing number of shares at the end of fiscal year 2024 is 1,257,674*. Accurately detecting such equivalence has broad applications such as fact checking [32], knowledge base construction [33], and numerical reasoning [34–36].

- Task formulation.

Formally, given two numerical mentions n_1 and n_2 with their associated contexts c_1 and c_2, the goal of NSM is to determine whether n_1 and n_2 refer to the same fact (binary classification). Each context may be a paragraph or a table, and we standardize them into textual form (with tables linearized using Markdown).

- Reasoning pattern.

Solving NSM can be generally decomposed into four steps:

1. **Numerical grounding:** Locate each numerical mention in the given context.
2. **Semantic interpretation:** Identify the semantics of the numerical mention within its context (e.g., time, subject, indicator).
3. **Entity alignment:** Link contextual references to consistent entities or events (e.g., "BHP Group Limited" vs. "the company"; "FY2024" vs. "as of June 2024").
4. **Equivalence decision:** Compare the semantic frames of both mentions and decide whether they denote the same fact.

This reasoning process demonstrates that NSM goes beyond simple string matching or numerical comparison, requiring the integration of quantitative values with domain-specific contextual cues to determine semantic equivalence.

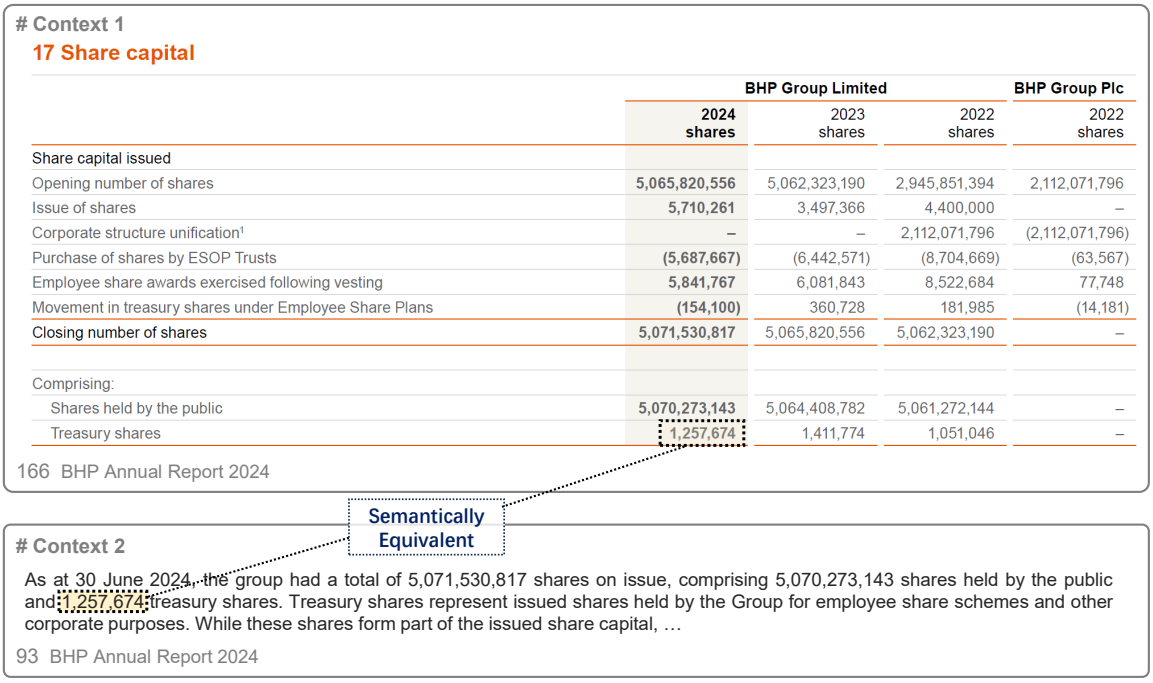


Fig. 1 Two document excerpts from the BHP Annual Report 2024. The numerical mentions highlighted in dashed boxes are semantically equivalent.

3.4 Transaction Purpose Classification

Banking systems generate vast volumes of transaction data, and accurately identifying the underlying purpose of each transaction is crucial for compliance auditing, financial analysis, and fraud detection. The **Transaction Purpose Classification (TPC)** task addresses this need by assigning a single transaction record to one of 62 predefined categories, covering both corporate-related and personal-related purposes.

- Task formulation.

Formally, given a structured record consisting of: (i) account holder type (enterprise/individual), (ii) transaction direction (credit/debit), (iii) free-text transaction memo, (iv) counterparty identity, and (v) contextual metadata (e.g., time, channel, amount), the task is to predict the correct purpose category from a fixed taxonomy of 62 classes.

- Reasoning pattern.

Solving TPC can be generally decomposed into four steps:

1. **Identify key attributes:** Determine account holder type and transaction direction.
2. **Extract salient cues:** Parse informative keywords, amounts, organizations, and dates from the memo and counterparty fields.
3. **Apply taxonomy rules:** Match the extracted cues to the taxonomy, applying priority rules for authoritative signals (e.g., payments to tax authorities, legal institutions, or banks).
4. **Finalize decision:** Select the most plausible category and, if required, generate a concise rationale citing the decisive cues.

More details of the TPC dataset and taxonomy are provided in Appendix D.

4 Pilot Comparison Experiments

In this section, we use the NSM task as a case study to investigate the following research question: *Does SFT+RLVR achieve the best performance in incentivizing the reasoning capabilities of LLMs?*

4.1 Experimental Setups

4.1.1 Dataset

Training set. We collect 110k numerical semantic matching samples from 544 annual reports of Chinese companies. Annual reports provide comprehensive overviews of companies' yearly performance, operations, and future outlook, serving to inform shareholders and stakeholders. Our dataset encompasses substantial diversity across two key dimensions: (1) *Industry coverage*: The companies span multiple sectors including finance, manufacturing, technology, and real estate; (2) *Temporal coverage*: The reporting periods range from 2018 to 2024, capturing temporal diversity across varying market conditions and economic cycles. For each sample, we annotate the ground-truth answer following the protocol established in previous work [37]. Additionally, we engage 8 professional annotators to provide rationales for 10k samples, with subsequent validation performed by 2 financial experts. Statistical analysis reveals that the first-round annotation accuracy reaches approximately 97%, which is further improved after expert validation. We denote these 10k samples with rationales as **RatQA-10k**, while the remaining 100k samples without rationales are denoted as **DirQA-100k**.

Test set. Following a similar data construction pipeline, we collect 20k samples each from annual reports and IPO prospectuses to form our evaluation datasets. Unlike annual reports, IPO prospectuses contain comprehensive information about a company's financials, business

model, and risk factors to ensure transparency and regulatory compliance during public offerings. The IPO prospectus evaluation set serves as a more challenging cross-domain benchmark, better reflecting the model's generalization capabilities across different document types and contexts.

More details about the dataset construction and statistics are provided in Appendix A.

4.1.2 Metrics

We adopt accuracy, precision, recall, and F1-score as evaluation metrics, following prior work [32, 37]. Specifically, given a set of golden semantically equivalent numerical pairs g and a set of predicted pairs p, we define the metrics as follows:

$$\text{Precision (P)} = \frac{|g \cap p|}{|p|}, \tag{5}$$

$$\text{Recall (R)} = \frac{|g \cap p|}{|g|}, \tag{6}$$

$$\text{F1-score} = \frac{2 \cdot \text{P} \cdot \text{R}}{\text{P} + \text{R}}. \tag{7}$$

4.1.3 Implementation Details

We select Qwen3-8B [38] as our backbone model due to its exceptional Chinese language understanding capabilities [39] and its popularity as a foundation for RLVR [17, 40].

For SFT, we employ the Hugging Face Transformers library [41] and DeepSpeed ZeRO [42] for efficient distributed training. The model is trained for 2 epochs with a learning rate of 2e-5, a batch size of 20 per GPU. We use a cosine annealing scheduler with warmup for the first 1% of training steps. The maximum gradient norm is clipped at 1.0 to ensure training stability. For RLVR, we adopt DAPO [4] with the GRPO advantage estimator to optimize the RLVR objective, and employ VERL [43] for efficient distributed training. The sampling temperature is set to 1.0, and we generate 16 responses per prompt during rollout. We use a constant learning rate of 1e-6 with a training batch size of 192 prompts. To stabilize policy updates, we apply asymmetric clipping with ratios of 0.2 and 0.28, and aggregate losses using token-level mean. No KL regularization is applied, either as an explicit loss or as a reward term. The maximum prompt and response lengths are set to 4096 and 8192 tokens, respectively.

All training procedures are conducted on a cluster of 24 H100 GPUs with 80GB memory each. We utilize vLLM [44] for efficient inference during evaluation. For both training and inference, the maximum input and output sequence length is set to 4096 and 1024 tokens, respectively.

4.1.4 Baselines

We evaluate SFT+RLVR against the following training strategies:

- **SFT-direct**: Supervised fine-tuning directly on DirQA-100k, i.e., training only on answers without rationales. This is the standard SFT baseline, which reduces rationale annotation cost but does not explicitly guide the reasoning process.
- **SFT-rationales**: Supervised fine-tuning only on RatQA-10k, where the model is trained to generate rationales before predicting the final answers. This provides explicit reasoning supervision but requires costly rationale annotations.
- **pure-RLVR** [2]: Reinforcement learning with verifiable rewards, optimized purely on RatQA-10k. The model is trained only with correctness feedback on final answers, without any rationale supervision.
- **UFT** [19]: unified fine-tuning, which concatenates rationales as hints into the prompts and applies curriculum learning to dynamically control the proportion and length of hints. This guides the model toward correct reasoning paths while maintaining flexibility.

4.2 Experimental Results

Table 1 summarizes the results across different training strategies on the numerical semantic matching task. Several key observations emerge:

Rationales provide substantial benefits over direct supervision. The SFT-rationales baseline, trained on only 10k samples, achieves 79.2% average accuracy and 57.6% F1, whereas SFT-direct, trained on 100k samples, only reaches 79.8% accuracy and 52.2% F1. Despite using ten times less data, SFT-rationales delivers comparable accuracy and notably higher F1, demonstrating the effectiveness of explicit reasoning supervision. This highlights that NSM requires reasoning rather than surface-level pattern learning, and well-annotated rationales significantly improve model generalization.

SFT+RLVR achieves the strongest overall performance. Among reinforcement-based strategies, pure-RLVR surpasses both SFT baselines by leveraging verifiable reward signals. UFT further improves performance by leveraging rationales as hints and applying curriculum learning strategies. However, SFT+RLVR achieves the best results overall, with 90.3% average accuracy and 78.4% F1, consistently outperforming both pure reinforcement and unified hint-based training.

Cross-domain consistency. The evaluation across both IPO Prospectus and Annual Report domains reveals remarkably consistent relative performance rankings, with SFT+RLVR maintaining its superiority in both domains (IPO: 88.3% accuracy, 73.6% F1; Annual Report: 92.3% accuracy, 83.2% F1). This cross-domain consistency validates the robustness of SFT+RLVR and suggests that the reasoning capabilities developed through SFT+RLVR generalize effectively across different financial document types. The consistent performance gaps between methods across domains further confirm that the observed improvements are not artifacts of specific data characteristics but reflect genuine enhancements in NSM reasoning ability.

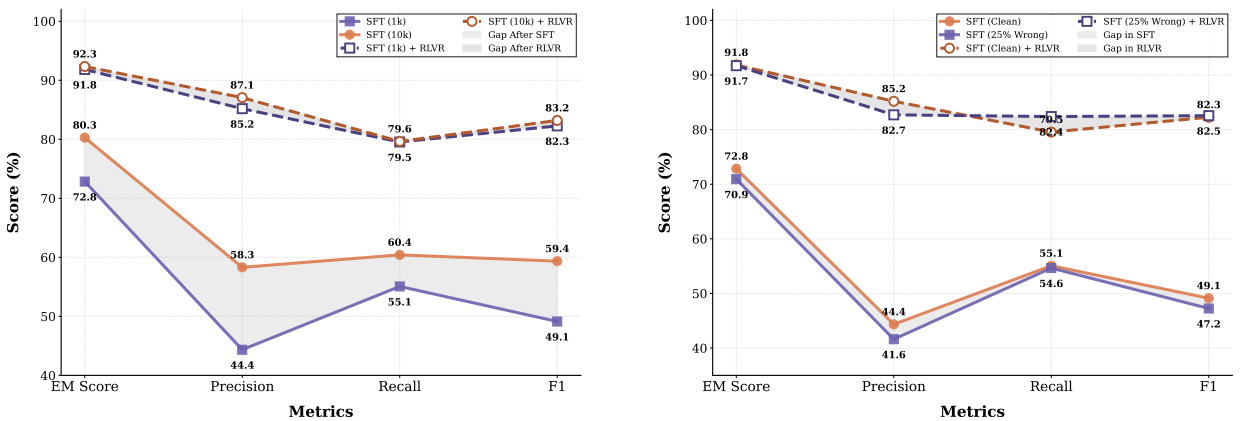
Conclusion. Taken together, these results highlight two key insights: (i) reasoning capabilities play a pivotal role in numerical semantic matching, and (ii) integrating rationale-based supervised fine-tuning with reinforcement learning (SFT+RLVR) provides the most effective strategy among all compared methods for stimulating and enhancing the model's reasoning ability.

5 Analysis of the Importance of Reasoning Patterns

In this section, we present a comprehensive analysis to explain why SFT+RLVR substantially outperforms other training strategies. We hypothesize that SFT+RLVR enables models to more effectively acquire reasoning patterns by providing direct supervision on rationales during

Strategy	RatQA-10k	DirQA-100k	IPO Prospectus				Annual Report				Average	
			Acc.	P.	R.	F1.	Acc.	P.	R.	F1.	Acc.	F1.
SFT-direct		✓	76.8	51.7	33.8	40.8	82.8	64.2	63.0	63.6	79.8	52.2
SFT-rationales	✓		78.0	53.5	58.2	55.7	80.3	58.3	60.4	59.4	79.2	57.6
pure-RLVR [2]		✓	87.1	75.2	68.2	71.5	89.1	77.0	77.1	77.1	88.1	74.3
UFT [19]	✓	✓	87.5	76.6	68.3	72.2	90.7	83.5	75.9	79.5	89.1	75.9
SFT+RLVR	✓	✓	**88.3**	**79.2**	**68.8**	**73.6**	**92.3**	**87.1**	**79.6**	**83.2**	**90.3**	**78.4**
Controlled Experiments												
SFT+RLVR	1k	✓	87.7	77.7	67.3	72.1	91.8	85.2	79.5	82.3	89.8	77.2
SFT+RLVR	1k, 25% wrong	✓	88.2	80.6	66.3	72.7	91.7	82.7	82.4	82.6	90.0	77.7

Table 1 Comparison of different training strategies.



(a) Impact of rationale quantity on the performance of SFT and RLVR stages. Results are reported on the annual report subset.

(b) Impact of rationale quality on the performance of SFT and RLVR stages. Results are reported on the annual report subset.

Fig. 2 Performance comparison of SFT and RLVR stages under varying rationale quantity (a) and quality (b).

fine-tuning. To validate this hypothesis, we provide two complementary lines of evidence: (1) from a *cause-side* perspective, through controlled experiments (Section 5.1), we demonstrate that it is the reasoning pattern, rather than the mere quantity or quality of rationales, that plays the decisive role in the success of SFT+RLVR; and (2) from a *effect-side* perspective, through forking token analysis (Section 5.2), we show that models trained with SFT+RLVR capture task-specific reasoning patterns more accurately, leading to more focused problem-solving behavior.

5.1 Evidence 1: Effect of Rationale Quantity and Quality

In this section, we design controlled experiments to demonstrate that reasoning pattern serves as the key factor when leveraging rationales, outweighing two alternative factors: the quantity (Section 5.1.1) and quality (Section 5.1.2) of rationales.

5.1.1 Reasoning Pattern over Rationale Quantity

To achieve this, we reduce the size of the RatQA-10k dataset by 10-fold, randomly retaining only 1k out of 10k training samples. Previous studies [45] have shown that training dataset size is highly correlated with the volume of task-specific knowledge acquired. This reduction results in significantly lower task-specific knowledge while preserving the overall reasoning pattern present in each training sample.

We employ the same training recipe as SFT+RLVR described in Section 4.1.3. The results of SFT+RLVR (only 1k rationale samples) are presented in Table 1. The performance remains comparable to the full SFT+RLVR, with F1 decreasing by merely 1.2 points. This is particularly notable when compared to other reasoning-enhanced methods, which lag significantly behind SFT+RLVR by 2.5-20.8 points. These results indicate that training dataset size is not the primary factor driving the performance gains of SFT+RLVR, thereby supporting our hypothesis that reasoning patterns are the crucial component.

5.1.2 Reasoning Pattern over Rationale Quality

Previous work [46] has indicated that golden demonstrations are not strictly required for in-context learning—randomly replacing labels in demonstrations barely hurts the performance across a range of classification and multiple-choice tasks. In this section, we investigate whether this phenomenon extends to SFT+RLVR: specifically, whether the correctness of rationales significantly impacts performance. To achieve this, we randomly replace 25% of human-annotated rationales with incorrect rationales while maintaining the underlying reasoning

pattern unchanged. We prompt GPT-4.1[1] to modify 25% of manually-annotated rationales into incorrect rationales. The detailed prompt is provided in Appendix C.

We employ the same training setup as SFT+RLVR described in Section 4.1.3. The results of SFT+RLVR (1k, 25% wrong) are shown in Table 1. Remarkably, its performance remains comparable to that of the full SFT+RRFT, with F1 decreasing by only 0.7 points. Interestingly, we observe that SFT+RLVR (1k, 25% wrong) slightly outperforms SFT+RLVR (1k). We hypothesize that this improvement arises because model-modified responses introduce greater response diversity than human rationales, thereby enhancing the overall performance [47]. We leave a detailed investigation of this phenomenon for future work.

These results demonstrate that the quality of rationales is not the primary factor driving the performance gains of SFT+RLVR. Instead, they provide compelling evidence that reasoning patterns serve as the key component, consistent with findings from the in-context learning literature, and further support our central hypothesis.

5.1.3 Analysis of Performance Gaps Between Training Phases

To better understand the roles of SFT and RLVR stages, we conduct a analysis of intermediate performance under different rationale data conditions from Section 5.1.1 and Section 5.1.2. Figure 2a and Figure 2b present the performance comparison after each training phase across different evaluation metrics.

The results reveal two key insights about our SFT+RLVR training pipeline:

SFT stage: Quantity and quality of rationales are critical. During the SFT stage, both the quantity and quality of rationales significantly impact model performance. The 10k SFT model substantially outperforms the 1k SFT model across all metrics (e.g., 59.4% vs. 49.1% F1 score), demonstrating that abundant rationale data provides richer task-relevant knowledge for establishing stronger initial reasoning capabilities. A similar pattern emerges in Figure 2b when comparing models trained with correct versus incorrect rationales (25% wrong), where data quality directly translates to a 2% F1 score degradation. These substantial gaps underscore the importance of rationale quantity and quality in building a solid foundation for reasoning.

RLVR Stage: Performance Convergence via Self-Refining Reasoning. After RLVR training, the performance gaps across settings almost vanish, with final models achieving highly similar results, differences of only 0.9% and 0.2% in F1 score for the rationale quantity and quality experiments, respectively. This convergence suggests that RLVR effectively compensates for initial weaknesses by acquiring task-relevant knowledge from large-scale (question, answer) pairs, even when starting from suboptimal SFT baselines. The underlying mechanism behind this robustness is that once the SFT stage establishes basic reasoning patterns—regardless of data limitations or quality issues—the model can continue to self-improve during RLVR by generating diverse rationales, reinforcing reasoning paths that lead to correct answers, and suppressing erroneous ones.

[1] gpt-4.1-2025-04-14

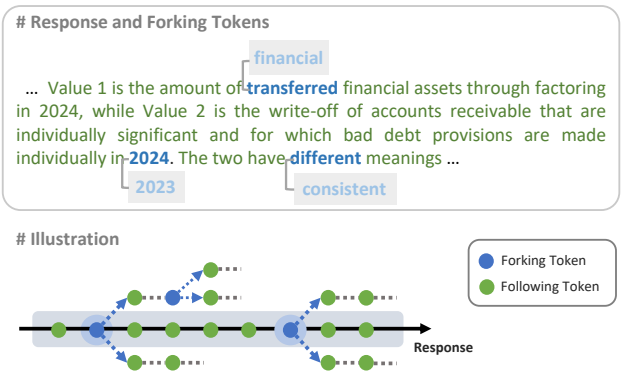


Fig. 3 An Example of forking tokens in LLM-generated responses.

This complementary learning dynamic highlights the synergistic relationship between SFT and RLVR stages: SFT provides essential reasoning patterns and templates, while RLVR enables scalable knowledge acquisition through self-exploration of the reasoning space. This property makes the approach robust to limited availability of high-quality rationale data, a common challenge in real-world applications.

5.2 Evidence 2: Analysis on Reasoning Behaviors

In the previous section, we provided evidence from an *cause-side* perspective through controlled experiments. However, this analysis still lacks the *effect-side* perspective, focusing on how reasoning patterns are concretely manifested in model responses. To address this gap, we identify and analyze the forking tokens [17] within model responses—critical decision points that fundamentally shape the reasoning trajectory and final outcomes. As illustrated in Figure 3, the LLM exhibits three pivotal decision points (i.e., forking tokens) when generating this response: (1) *transferred* vs *financial*, which determines the semantic interpretation of the accounting subject for value 1; (2) *2024* vs *2023*, which specifies the temporal context of value 2; and (3) *different* vs *consistent*, which governs the ultimate decision on semantic equivalence between the two values. In contrast, the model demonstrates high certainty at other token positions, which exert minimal influence on the final outcome. Forking tokens thus serve as crucial indicators of the model's underlying reasoning patterns. By analyzing these forking tokens, we can obtain deeper insights into the model's reasoning behaviors and decision-making processes.

5.2.1 Rollout-based Forking Token Detection

Traditional entropy-based methods [17, 18] for identifying forking tokens typically consider token positions with high entropy as forking tokens. However, these approaches suffer from a fundamental limitation: *they misclassify positions where candidate tokens are semantically similar but probabilistically diverse as genuine forking tokens*. For instance, a position may contain synonymous candidate tokens such as `company` and `company's`, which carry nearly identical semantic meaning and exhibit high entropy, yet do not represent a critical decision point in the reasoning process. We present a detailed case study in Appendix B. This leads to the erroneous identification of positions that do not substantially influence the reasoning trajectory, thereby reducing the accuracy of forking token detection. To address

Algorithm 1 Rollout-based Forking Token Detection (RFTD)

Require: Response $R = \{t_1, \ldots, t_L\}$, probability distributions $\{P_i\}_{i=1}^{L}$, threshold α, hyperparameters k, m, n
Ensure: Forking token set F

$F \leftarrow \emptyset$ {Initialize empty forking token set}
$T \leftarrow \text{TopK}(\{H(P_i) \mid i \in [1, L]\}, k)$ {Select top-k positions with highest entropy}
for $t_c \in T$ **do**
  $S \leftarrow \text{TopK}(\{P(v|t_1, \ldots, t_{c-1}) \mid v \in V \setminus \{t_c\}\}, m)$ {Select top-m alternative tokens}
  for $s_j \in S$ **do**
    $R' \leftarrow R[:c-1] \oplus s_j$ {Replace original token with substitute}
    $\{C_1, C_2, \ldots, C_n\} \leftarrow \text{Generate}(R', n)$
    $\rho_j \leftarrow \frac{1}{n} \sum_{l=1}^{n} \text{Divergent}(C_l, R)$
  end for
  if $\max_j \rho_j > \alpha$ **then**
    $F \leftarrow F \cup \{t_c\}$ {Add to forking token set}
  end if
end for
return F

this limitation, we propose **Rollout-based Forking Token Detection (RFTD)**, a novel approach that transcends probability-based heuristics by empirically evaluating the actual downstream impact of token substitutions through controlled generation rollouts.

The RFTD method operates by first identifying candidate positions based on entropy, then empirically testing their importance through token substitutions and continuation generation. Let $R = \{t_1, t_2, \ldots, t_L\}$ denote a response sequence of length L. We define the entropy at position i as $H(t_i) = -\sum_{v \in V} P(v|t_1, \ldots, t_{i-1}) \log P(v|t_1, \ldots, t_{i-1})$, where V is the vocabulary. For each candidate position, we substitute it with alternative tokens and measure the proportion of rollouts that produce significantly different outcomes. A position is classified as a forking token if this proportion exceeds threshold α.

The algorithm is presented in Algorithm 1, where $\text{Generate}(R', n)$ prompts the LLM to produce n continuation sequences from prefix R', and $\text{Divergent}(C, R)$ returns 1 if the final answer of continuation C differs from response R, 0 otherwise. This approach effectively distinguishes between tokens that merely exhibit high uncertainty in the probability distribution and those that genuinely influence the subsequent reasoning trajectory, thereby providing more accurate identification of critical decision points in responses.

5.2.2 Analysis of Forking Tokens

In this section, we leverage our RFTD method to characterize and compare the reasoning behaviors of models trained under various learning strategies. Specifically, we compare SFT+RLVR with alternative training strategies, including pure-RLVR and UFT, across the entire test set (refer to Appendix B for comprehensive experimental configurations). By identifying and visualizing the frequency of forking tokens for these models (Figures 11, 12, 13), we underscore the distinct differences in their reasoning processes. A detailed computational analysis of the RFTD method is provided in Appendix G.

SFT+RLVR produces task-relevant forking tokens. The forking tokens of SFT+RLVR show strong alignment with the reasoning pattern of the NSM task. Tokens such as *different*, *unanimous*, *not*, and *no* directly correspond to semantic equivalence judgment, while others like *annual*, *firm*, *main_business*, and *operating_income* reflect reasoning about the contextual meaning of numerical values.

Other methods exhibit meta-reasoning or exploratory behaviors. In contrast, models trained with alternative strategies (e.g., UFT, pure-RLVR) display forking tokens dominated by logical connectors and meta-reasoning indicators such as *but*, *according_to*, *if*, *because*, and *possible*. These tokens suggest that such models tend to rely on generic reasoning behaviors like hypothesis generation (*if*, *possible*), causal explanation (*because*), and discourse linking (*but*, *however*) rather than task-specific reasoning.

Overall, these observations demonstrate that SFT+RLVR more effectively internalizes the reasoning pattern required by the NSM task. By explicitly learning from rationale supervision, SFT+RLVR encourages the model to focus on these essential steps, leading to more accurate and interpretable reasoning behavior.

6 Application: Pattern-Aware LLMs as Rationale Annotators

Building on the insights from the above controlled experiments and analysis, we introduce **P**attern-**A**ware LLMs as **R**ationale Ann**O**tators (**PARO**), a framework designed to reduce the rationale annotation cost in SFT+RLVR. Motivated by our finding that reasoning patterns are more critical than the quantity or quality of rationales, PARO aims to lessen the dependence on large-scale human-annotated rationale datasets by leveraging LLMs to automatically generate high-quality rationales in place of costly human annotation.

Specifically, we prompt state-of-the-art LLMs with reasoning pattern priors to synthesize rationales for given (question, answer) pairs. The reasoning pattern prior is encoded within the prompt instructions. For each task, we provide step-wise reasoning guidance along with two manually-annotated exemplar rationales to guide the model's reasoning process. Importantly, we do not provide the final answer as part of the prompt to prevent the model from generating shortcut rationales that bypass proper reasoning.

6.1 Experimental Setup

We apply PARO to two representative reasoning tasks: Numerical Semantic Matching (NSM) and Transaction Purpose Classification (TPC), as detailed in Section 3.3 and Section 3.4. We evaluate four rationale annotation strategies: (1) **SFT(1k, Human)+RLVR**, where SFT is trained on 1k human-annotated rationales; (2) **SFT(10k, Human)+RLVR**, which uses the full human-annotated dataset (10k samples for NSM only); (3) **SFT(1k, Distill)+RLVR**, where rationales are distilled directly from the internal reasoning traces of a large reasoning model; and (4) **SFT(1k, PARO)+RLVR**, where rationales are synthesized by PARO based on reasoning pattern priors. To generate rationales for Distill and PARO, we employ Qwen3-235B-A22B-Thinking [20] (Qwen3-235B), a state-of-the-art open-source reasoning

Strategy	Acc.	P.	R.	F1
Qwen3-8B				
SFT(1k, Human)+RLVR	91.8	85.2	79.5	82.3
SFT(10k, Human)+RLVR	92.3	**87.1**	79.6	83.2
SFT(1k, Distill)+RLVR	90.4	83.1	76.0	79.4
SFT(1k, PARO)+RLVR	**92.4**	84.4	**82.9**	**83.6**
Qwen3-4B				
SFT(1k, Human)+RLVR	90.1	78.7	**80.4**	79.5
SFT(1k, Distill)+RLVR	88.6	**83.7**	75.0	73.2
SFT(1k, PARO)+RLVR	**90.7**	81.7	78.5	**80.1**
Llama3.1-8B-Instruct				
SFT(1k, Human)+RLVR	88.9	76.9	76.2	76.5
SFT(1k, Distill)+RLVR	87.8	**80.8**	63.7	71.3
SFT(1k, PARO)+RLVR	**89.1**	78.2	**76.8**	**77.5**

Table 2 Performance comparison of rationale annotation strategies (SFT data source and size) across different base LLMs on the NSM task.

Strategy	Acc.	P.	R.	F1
SFT(1k, Human)+RLVR	87.9	87.6	87.9	87.2
SFT(1k, Distill)+RLVR	85.7	86.9	85.7	85.6
SFT(1k, PARO)+RLVR	**88.2**	**89.0**	**88.2**	**87.9**

Table 3 Performance comparison of rationale annotation strategies on the TPC tasks, using Qwen3-8B as the base model.

LLM, as the rationale annotator. We present the prompt template for these two tasks in Appendix E (Figure 9 and Figure 10).

For the NSM task, the dataset is the same as described in Section 4.1.1. For the TPC task, we collect 1k (q, r, a) samples with manually annotated rationales for SFT training and 40k (q, a) samples for RLVR training. The training recipes for both tasks follow the same configuration described in Section 4.1.3.

6.2 Results and Analysis

As shown in Table 2 and Table 3, PARO achieves strong and consistent performance across different model families and scales, including Qwen3-4B, Qwen3-8B, and Llama3.1-8B-Instruct. In all settings, **SFT(1k, PARO)+RLVR** matches or outperforms both human-annotated and distillation-based baselines.

For smaller models such as Qwen3-4B, PARO attains the best overall accuracy and F1 score, indicating that reasoning pattern priors are especially beneficial when model capacity is limited. On larger Qwen3-8B models, PARO achieves an F1 score comparable to or exceeding the fully human-annotated **SFT(10k, Human)+RLVR** baseline while using an order of magnitude less annotation. Similar trends are observed on Llama3.1-8B-Instruct, where PARO yields the highest accuracy and F1, with notably improved recall over distillation-based supervision. We present a detailed case study of PARO in Appendix F.

Overall, these results show that the benefits of PARO are consistent across model series and scales, reinforcing its effectiveness as a scalable and cost-efficient approach for reasoning supervision.

7 Related Work

7.1 Reinforcement Learning with Verifiable Rewards

Reinforcement Learning with Verifiable Rewards (RLVR) [6] has proven highly effective for enhancing the reasoning abilities of LLMs across diverse domains. A key advantage of RLVR lies in its ability to optimize models using rule-based rewards derived solely from verifiable (question, answer) pairs, eliminating the need for costly human-annotated reasoning trajectories and thereby enabling scalable training on large datasets [2]. In mathematical reasoning [1], such rewards are typically defined through symbolic or numerical verification rules, whereas in code generation [2, 5], executable program outputs provide objective feedback signals. Prior to the RLVR stage, models are typically warmed up through Supervised Fine-Tuning (SFT) on human-annotated reasoning trajectories (rationales) to establish initial reasoning behaviors [2]. Recent research has explored various strategies for integrating human reasoning trajectories into the RLVR framework. Some efforts aim to enhance interpretability by generating human-readable reasoning paths [2], whereas others attempt to unify supervised and reinforcement learning under joint optimization frameworks [19, 48, 49]. However, the substantial cost of annotating large-scale, high-quality rationales remains largely overlooked. In contrast, we identify a broad class of problems, *patterned reasoning tasks*, for which rationale annotation can be reliably automated by LLMs without compromising final performance.

7.2 Reasoning Pattern Analysis

Understanding how models learn and apply reasoning patterns has recently attracted significant research attention. Recent work on forking token analysis [17, 18] has provided valuable insights into the mechanisms underlying model reasoning behavior. However, these studies typically employ entropy-based detection methods, treating high-entropy tokens as forking tokens. Such approaches may misclassify positions where candidate tokens are semantically similar but probabilistically diverse as genuine forking points, potentially leading to inaccurate analyses of reasoning patterns. Recent research [47] has also explored how SFT and RL jointly shape reasoning behaviors by analyzing reasoning trajectories and constructing reasoning graphs, revealing that SFT expands correct reasoning paths while RL compresses incorrect ones, concentrating reasoning functionality into fewer steps. While their study focuses on how training dynamics restructure reasoning processes, our work instead investigates, under the SFT+RL paradigm, how the reasoning capabilities established during the SFT stage can be efficiently supervised, emphasizing the central role of reasoning patterns.

7.3 Numerical Semantic Matching

Numerical semantic matching (NSM) has emerged as a critical task with significant overlap with document understanding [50] and information extraction [11]. Early approaches [37] primarily focused on training models to perform binary classification directly, without explicitly modeling the underlying reasoning process. However, these approaches overlook the sophisticated reasoning capabilities required to understand and compare the multi-faceted semantics of numerical

mentions. While recent work [32] has begun exploring more sophisticated LLM-based approaches for enhanced numerical understanding, the reasoning-intensive nature of NSM remains largely underexplored. This work uses NSM as a representative patterned reasoning task and provide the first exploration of the reasoning nature inherent in the NSM task.

8 Conclusion and Future Work

This paper revisits the role of rationales in the standard SFT+RLVR training pipeline for large language models. Through a systematic analysis of patterned reasoning tasks such as numerical semantic matching, we demonstrate that reasoning patterns—rather than the quantity or quality of rationales—serve as the primary determinant of reasoning performance. Building on these insights, we introduce **PARO** (**P**attern-**A**ware LLMs as **R**ationale Ann**O**tators), a cost-efficient annotation framework that leverages pattern-aware LLMs to automatically generate high-quality rationales. PARO achieves comparable or even superior performance to models trained on large human-annotated datasets, while eliminating the need for human rationale annotation, thereby offering a practical and scalable approach to reasoning supervision in LLMs.

Looking ahead, several promising directions merit further exploration: (1) Generalization to broader reasoning domains. Extending PARO to tasks involving logical, temporal, or spatial reasoning [51–56] could reveal the broader applicability of the reasoning-pattern perspective. (2) Automated discovery of reasoning patterns. Developing methods that automatically extract or infer reasoning structures from unlabeled data would further minimize human involvement and enable large-scale, self-improving reasoning supervision. (3) Adaptive reasoning supervision. Future work could investigate hybrid strategies that dynamically balance pattern enforcement with exploratory reasoning, bridging the gap between patterned reasoning and adaptive domains such as mathematics or code generation.

Overall, our findings highlight a paradigm shift: for patterned reasoning tasks, effective reasoning supervision lies not in more or better rationales, but in teaching models *how to reason*—through explicit reasoning patterns.

Appendixes

Appendix A: NSM Dataset Details

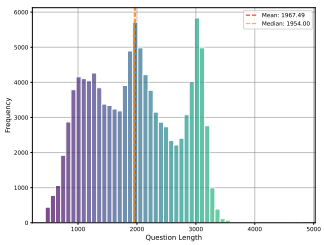


Fig. 4 Frequency distribution of question lengths in the dataset.

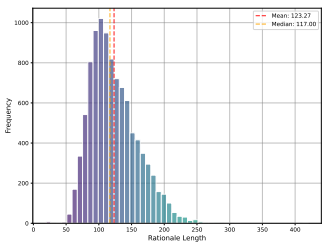


Fig. 5 Frequency distribution of rationale lengths in the dataset.

We present the frequency distributions of question lengths and rationale lengths in Figure 4 and Figure 5, respectively. Length is measured by the number of tokens obtained after tokenizing each string with the Qwen3 tokenizer [38]. Table 4 summarizes the statistics of context types. The majority of samples fall under the Table-Table category, which is likely due to most numerical mentions appearing in tables rather than paragraphs.

Context Type	Count	Ratio
Table-Table	98,155	89.2%
Table-Paragraph	8,811	8.0%
Paragraph-Paragraph	3,034	2.8%

Table 4 Statistics of context types for pairs of numerical mentions in each sample.

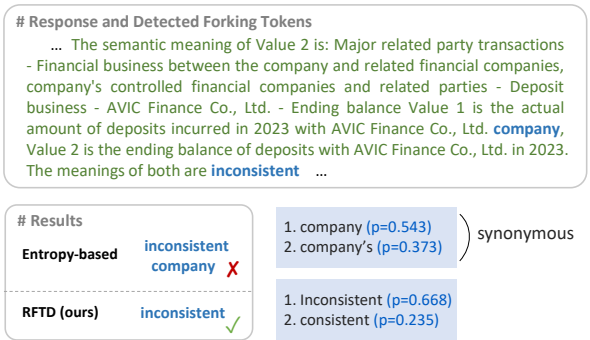


Fig. 6 Case study: Traditional entropy-based methods incorrectly identify tokens with synonymous candidates as forking tokens. We display the top-2 candidate tokens for each position along with their corresponding probabilities.

Appendix B: Details of Forking Token Analysis

Experimental details. We use consistent experimental settings across pure-RLVR, UFT, and SFT+RLVR. To generate the original responses, we first employ the model to predict on the full test set using greedy decoding (temperature=0). We then apply the RFTD algorithm 1 for forking token detection. The hyperparameters are set as follows: k=5, m=3, n=10, $\alpha = 0.5$. For continuous generation, we use the corresponding LLM with a relatively high temperature of 0.7. Note that we report the chosen token at the forking position in the original response. We use vLLM [44] for efficient inference.

Case study. As presented in Figure 6, we demonstrate the limitations of previous entropy-based methods [17, 18] in accurately detecting forking tokens. These approaches typically identify token positions with high entropy in the output distribution as forking tokens. However, they frequently misclassify positions where candidate tokens are synonymous or semantically equivalent, such as when distinguishing between `company` and `company's`. In contrast, our rollout-based forking token detection method empirically evaluates the actual downstream impact of token substitutions through multiple rollouts, making it more robust to synonymous equivalence.

Appendix C: Prompts

We provide the detailed instructions for generating incorrect rationales in Figure 8.

 # Task Instructions
{task instruction}
Please note that this sample provides manually annotated hints before the Output. You may refer to the Hint content, but be aware that the Hint may not be complete.
 # Input
{input}
 # Hint
{hint}
 # Output

Fig. 7 Prompt template for the numerical semantic matching task with hint guidance.

 # Task Instructions
Given a Question and an Answer, please output a `Modified Answer` that transforms the Answer into an incorrect response.
Requirements:
- Ensure that the final answer in `Modified Answer` is **different** from the original Answer.
- If the Answer's final conclusion is "yes", then the `Modified Answer`'s final conclusion should be "no", and vice versa.
- Output the `Modified Answer` directly without additional explanations.

 # Question
{question}
 # Answer
{answer}
 # Modified Answer

Fig. 8 Prompt template for generating incorrect rationales

Appendix D: Details of the TPC Task

- Task Definition

The *Transaction Purpose Classification* (TPC) task aims to classify a single bank transaction into one of 62 predefined purpose categories, including 42 corporate-related and 20 personal-related categories. Each transaction record provides multiple fields: account holder, transaction direction, transaction memo, counterparty, and contextual information. The objective is to predict the correct purpose category based on these inputs.

Each input is a structured record containing the following fields:

- **Account holder type:** Enterprise or Individual
- **Transaction direction:** Credit or Debit
- **Transaction memo:** Free-text description of the transaction
- **Counterparty:** The entity on the other side of the transaction
- **Contextual cues:** Additional metadata such as time, channel, or amount

The expected output is:

- A **category label** chosen from the taxonomy (62 classes in total).

Original Prompt: ... If they are semantically equivalent, please output "yes", otherwise please output "no".
Prompt with Reasoning Pattern Prior: ... If they are semantically equivalent, please output "yes", otherwise please output "no". Please first provide your reasoning process in `<rationale>` and `</rationale>` tags, including: (1) Analyze the semantics of Value 1 and Value 2; (2) Compare the similarities and differences between their semantics in terms of time, subject, scope, entity, etc. If there is a difference in any aspect, then the output should be "no", otherwise output "yes". Please follow the format below for output and do not output any other content: {**two-shot manually-annotated rationales**}

Fig. 9 Prompt design with reasoning pattern prior for NSM.

- Illustrative Example.

  Input:
  Account holder: Xima Intelligent Technology Co., Ltd.
  Account number: 88010122000085759
  Transaction date: 2019-03-11
  Credit: 10,000,000.0
  Debit: 0.0
  Balance: 10,018,196.76
  Transaction memo: "Structured deposit principal"
  Counterparty: Xima Intelligent Technology Co., Ltd.
  Counterparty account: 88010122000173077

  Output:
  Label: `Non-operating Income--Other Income`

- Example Subset of Labels

Since the full taxonomy contains 62 categories, we list a representative subset here:

- `Corporate--Tax Payment`
- `Corporate--Salary Distribution`
- `Corporate--Loan Repayment`
- `Corporate--Supplier Payment`
- `Personal--Credit Card Repayment`
- `Personal--Utility Bill Payment`
- `Personal--E-commerce Purchase`
- `Personal--Peer-to-Peer Transfer`

Appendix E: Prompt Templates of PARO

We show the prompt template of PARO for the NSM and TPC task in Figure 9 and Figure 10, respectively.

Appendix F: Case Study of PARO

we conducted a detailed case study on the NSM task by manually analyzing **204 misclassified samples** produced by the **SFT(1k, PARO) + RLVR** model. Based on this analysis, we identified several recurring error categories, summarized in Table 5.

Overall, the dominant failure modes fall into three major categories: (1) *lack of domain knowledge*, where the model fails to correctly apply basic financial or accounting principles; (2) *reasoning hallucination*,

Original Prompt:
... Please classify the purpose of the given bank transaction into one of the predefined categories.
Prompt with Reasoning Pattern Prior:
... Please classify the purpose of the given bank transaction into one of the predefined categories. Please first provide your reasoning process in `<rationale>` and `</rationale>` tags, following these structured steps:

1. **Entity Identification:** Determine whether the account holder is an *enterprise* (e.g., company, corporation) or an *individual* (personal name).
2. **Direction Determination:** Identify the transaction direction — whether it represents *income (credit)* or *expense (debit)*.
3. **Information Matching:** Prioritize transaction keyword matching, then analyze the counterparty information:
   - Financial institutions → investment / wealth management / loan categories
   - Tax authorities → tax-related categories
   - Judicial authorities → penalty / compensation categories
   - Government departments → subsidy / tax-related categories
4. **Refined Classification:** Combine the subject type and transaction nature to select the most appropriate purpose category.

Please follow the output format below and do not output any other content:
{**two-shot manually-annotated rationales**}

Fig. 10 Prompt design with reasoning pattern prior for TPC.

where the model introduces unsupported assumptions or produces logically inconsistent conclusions; and (3) *semantic misinterpretation*, particularly of numerical expressions or financial semantics. Together, these account for more than 85% of the observed errors. The remaining cases are due to intrinsically ambiguous inputs with insufficient evidence, or minor issues in annotation or task definition.

This analysis indicates that we do not observe systematic failures due to a lack of understanding of the task's reasoning pattern. Instead, errors primarily arise during the execution of individual reasoning steps, including insufficient domain knowledge, reasoning hallucinations, and unclear interpretation of numerical or financial semantics, as well as inherent ambiguity in the input. This error breakdown provides useful insight into the limitations of pattern-based reasoning supervision and highlights promising directions for future improvement, such as incorporating stronger domain knowledge, reducing hallucinations during intermediate reasoning steps, and more explicitly handling under-specified inputs.

Appendix G: Computational Analysis of RFTD

In this section, we discuss the computational characteristics of the proposed Rollout-based Forking Token Detection (RFTD) and justify its design choices relative to simpler heuristics.

While RFTD requires generating n continuations for the top-k candidate positions, making it computationally more demanding than entropy-based alternatives, this approach is necessitated by several factors:

1. **Identification Accuracy.** RFTD provides substantially more reliable forking-token identification than entropy-based heuristics. Our manual evaluation indicates that RFTD achieves an accuracy of **85.7%**, whereas entropy-based methods reach only **32.3%**. The primary limitation of entropy-based approaches is their inability to distinguish between semantic branching and mere linguistic variability; tokens with multiple synonymous candidates often exhibit high entropy and are thus incorrectly flagged as forking tokens. A representative case study of this failure mode is illustrated in Figure 6.
2. **Offline Analytical Framework.** RFTD is designed as an *offline diagnostic tool* to characterize model behavior rather than a component for real-time inference. Within this context, the robustness and precision of forking-token detection are prioritized over per-sample latency.
3. **Practical Scalability.** The cost of generating $n \times k$ continuations is significantly mitigated in practice through prefix caching techniques implemented in modern inference engines such as vLLM [44]. In our empirical setup, analyzing an 8B model over 10,000 samples (with $n = 10, k = 5$) using eight NVIDIA RTX 4090 GPUs required approximately 10 hours. Given that this diagnostic process is typically performed once per model checkpoint, the computational overhead remains well within practical limits for research.

In summary, the substantial gain in detection accuracy provided by RFTD justifies the additional computational investment, particularly for the rigorous offline analysis required to understand LLM reasoning patterns.

Error Type	Typical Description	Count	Ratio (%)
Lack of Domain Knowledge	Failure to apply fundamental accounting or financial principles, e.g., treating registered capital as paid-in capital, misunderstanding credit loss rates versus provisions, or confusing gross profit, net profit, and profit attributable to shareholders.	75	36.8
Reasoning Hallucination	Generating conclusions not supported by the input text, including fabricated assumptions (e.g., inferring missing financial items), or internal inconsistencies where the final decision contradicts the preceding analysis.	62	30.4
Semantic Misinterpretation	Incorrect interpretation of numerical expressions or financial semantics, such as treating ratios or percentages as absolute values, confusing sub-items with aggregate totals, misreading year-end balances as period flows, or overlooking table headers and hierarchical structure.	41	20.1
Insufficient Evidence	The input text is under-specified or ambiguous, making it impossible to determine equivalence without introducing external knowledge or unverifiable assumptions, e.g., missing subject alignment, unclear temporal scope, or incomplete table context.	14	6.9
Annotation or Task Definition Issues	Errors arising from inconsistencies or ambiguities in annotation guidelines or task formulation, e.g., unclear equivalence criteria, borderline cases between not equivalent and indeterminable, or mismatches between labeling rules and financial semantics.	12	5.9

Table 5 Distribution of major error types identified in the case study analysis of the NSM task.

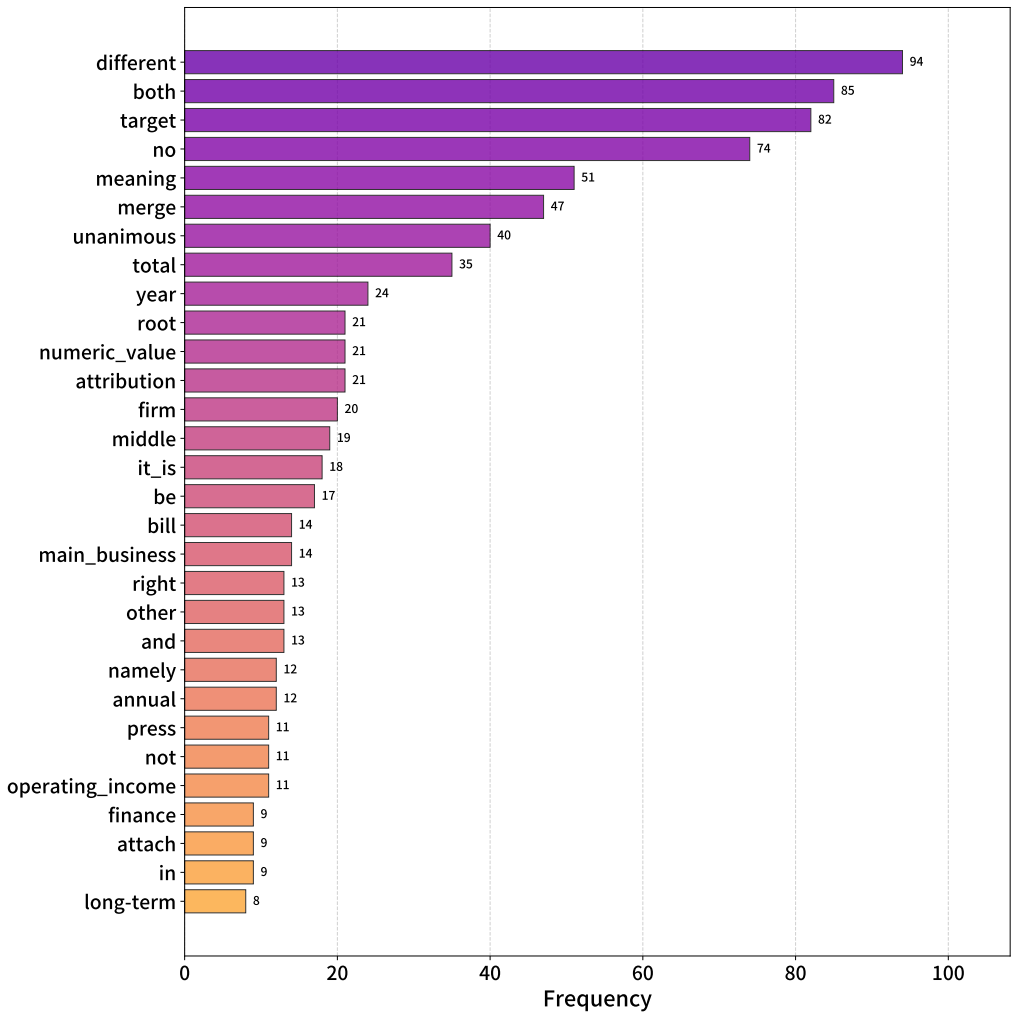


Fig. 11 Top forking token frequencies for **SFT+RLVR**.

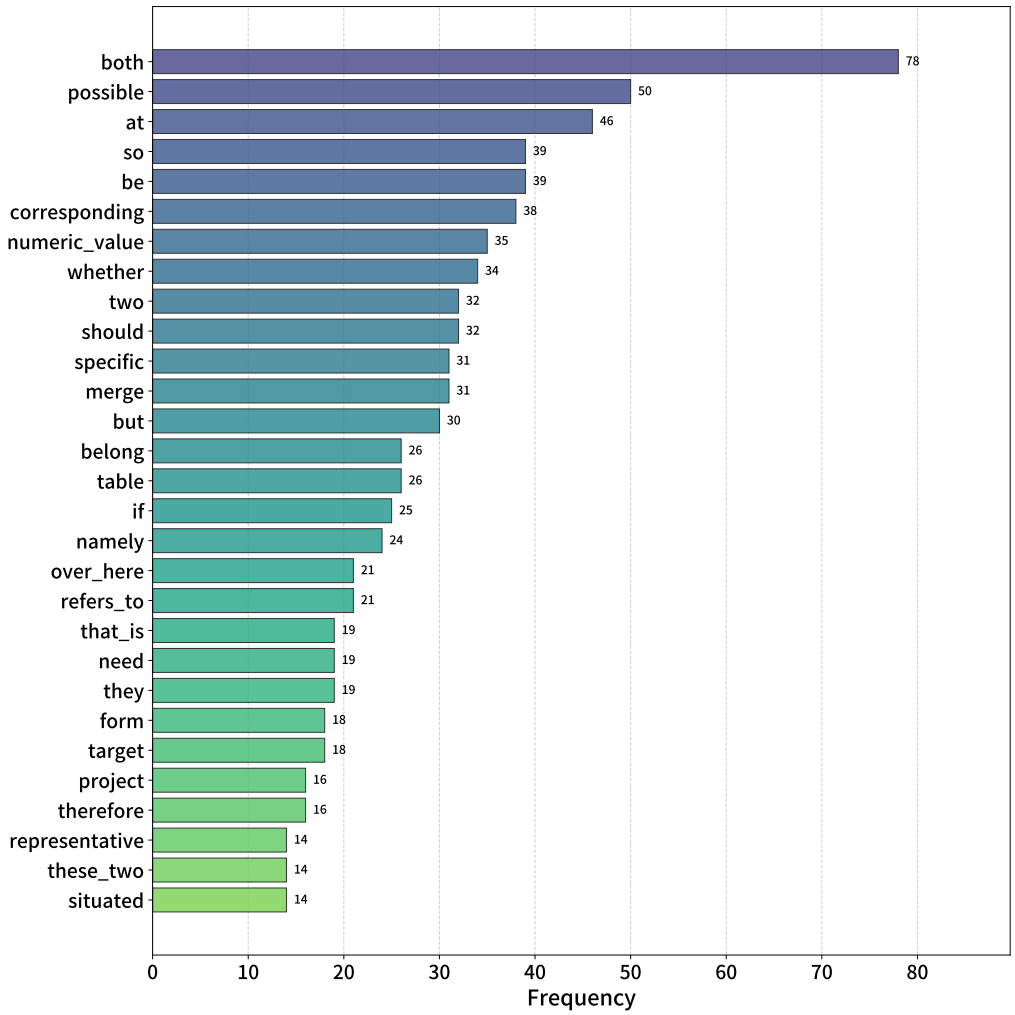


Fig. 12 Top forking token frequencies for **UFT**.

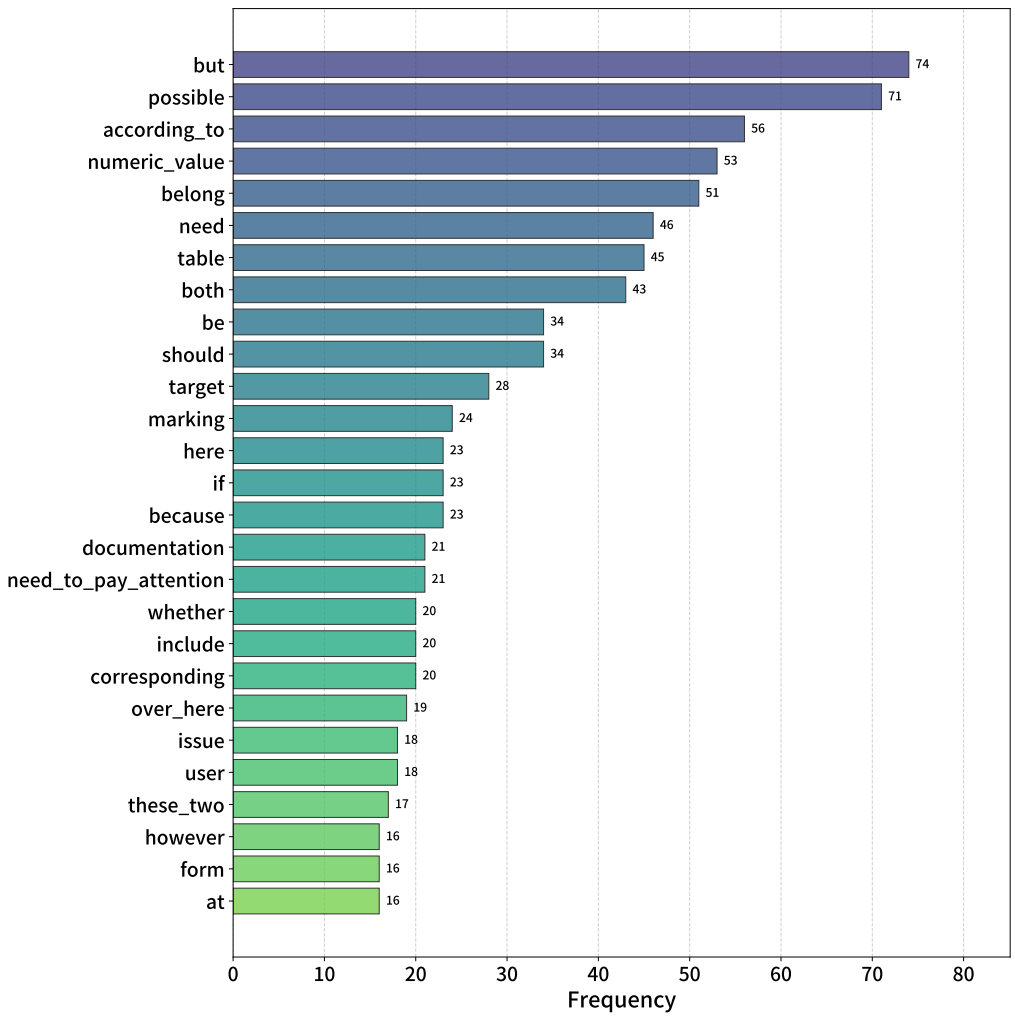


Fig. 13 Top forking token frequencies for **pure-RLVR**.

References

[1] Shao Z, Wang P, Zhu Q, Xu R, Song J, Bi X, Zhang H, Zhang M, Li Y, Wu Y, others . Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024

[2] Guo D, Yang D, Zhang H, Song J, Zhang R, Xu R, Zhu Q, Ma S, Wang P, Bi X, others . Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948, 2025

[3] Chen H, Zheng K, Zhang Q, Cui G, Cui Y, Ye H, Lin T Y, Liu M Y, Zhu J, Wang H. Bridging supervised learning and reinforcement learning in math reasoning. arXiv preprint arXiv:2505.18116, 2025

[4] Yu Q, Zhang Z, Zhu R, Yuan Y, Zuo X, Yue Y, Dai W, Fan T, Liu G, Liu L, others . Dapo: An open-source llm reinforcement learning system at scale. arXiv preprint arXiv:2503.14476, 2025

[5] Jiang J, Wang F, Shen J, Kim S, Kim S. A survey on large language models for code generation. arXiv preprint arXiv:2406.00515, 2024

[6] Lambert N, Morrison J, Pyatkin V, Huang S, Ivison H, Brahman F, Miranda L J V, Liu A, Dziri N, Lyu S, Gu Y, Malik S, Graf V, Hwang J D, Yang J, Bras R L, Tafjord O, Wilhelm C, Soldaini L, Smith N A, Wang Y, Dasigi P, Hajishirzi H. TÜLU 3: Pushing frontiers in open language model post-training. CoRR, 2024, abs/2411.15124

[7] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017

[8] Kandpal N, Raffel C. Position: The most expensive part of an llm should be its training data. arXiv preprint, 2025, arXiv:2504.12427

[9] Zhang W, others . Sentiment analysis in the era of large language models. In: Findings of the North American Chapter of the Association for Computational Linguistics (NAACL) – Findings. 2024. Comprehensive evaluation of LLMs on sentiment tasks; discusses prompting and evaluation.

[10] Vykopal I, others . Generative large language models in automated fact-checking: A survey. arXiv preprint, 2024. Survey of LLM applications and limits in fact-checking, discussing model capabilities and reliance on human fact-checkers.

[11] Pang C, Cao Y, Ding Q, Luo P. Guideline learning for in-context information extraction. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023, 15372–15389

[12] Dagdelen J, others . Structured information extraction from scientific text with pretrained large language models. Nature Communications, 2024, 15: 45563. Shows how LLMs can be fine-tuned for joint NER/RE and structured extraction in scientific domains.

[13] Zuo K, Jiang Y, Mo F, Lio P. Kg4diagnosis: A hierarchical multi-agent llm framework with knowledge graph enhancement for medical diagnosis. In: AAAI Bridge Program on AI for Medicine and Healthcare. 2025, 195–204

[14] Hillebrand L, Berger A, Deußer T, Dilmaghani T, Khaled M, Kliem B, Loitz R, Pielka M, Leonhard D, Bauckhage C, others . Improving zero-shot text matching for financial auditing with large language models. In: Proceedings of the ACM Symposium on Document Engineering 2023. 2023, 1–4

[15] John L, Wittenborg T, Auer S, Karras O. Human-in-the-loop workflow for neuro-symbolic scholarly knowledge organization. arXiv preprint arXiv:2506.03221, 2025

[16] Hao Y, Chen Y, Zhang Y, Fan C. Large language models can solve real-world planning rigorously with formal verification tools. In: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Long Papers). 2025, 3434–3483

[17] Wang S, Yu L, Gao C, Zheng C, Liu S, Lu R, Dang K, Chen X, Yang J, Zhang Z, others . Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. arXiv preprint arXiv:2506.01939, 2025

[18] Cheng D, Huang S, Zhu X, Dai B, Zhao W X, Zhang Z, Wei F. Reasoning with exploration: An entropy perspective. arXiv preprint arXiv:2506.14758, 2025

[19] Liu M, Farina G, Ozdaglar A. Uft: Unifying supervised and reinforcement fine-tuning. arXiv preprint arXiv:2505.16984, 2025

[20] Team Q. Qwen3 technical report, 2025

[21] Zhang W, Deng Y, Liu B, Pan S J, Bing L. Sentiment analysis in the era of large language models: A reality check. In: Findings of the Association for Computational Linguistics: NAACL 2024. 2024, 246–259

[22] Gretz S, Halfon A, Shnayderman I, Toledo-Ronen O, Dankin L, Katsis Y, Arviv O, Katz Y, Slonim N, Ein-Dor L. Zero-shot topical text classification with llms - an experimental study. In: Findings of the Association for Computational Linguistics: EMNLP 2023. 2023, 9647–9676

[23] Arora G, Jain S, Merugu S. Intent detection in the age of llms. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (Industry Track). 2024, 1559–1570

[24] Li M, Peng B, Galley M, Gao J, Zhang Z. Self-checker: Plug-and-play modules for fact-checking with large language models. In: Findings of the Association for Computational Linguistics: NAACL 2024. 2024, 163–181

[25] Leippold M, Vaghefi S A, Stammbach D, Muccione V, Bingler J, Ni J, Senni C C, Wekhof T, Schimanski T, Gostlow G, others . Automated fact-checking of climate claims with large language models. npj Climate Action, 2025, 4(1): 17

[26] Wan Z, Cheng F, Mao Z, Liu Q, Song H, Li J, Kurohashi S. Gpt-re: In-context learning for relation extraction using large language models. arXiv preprint arXiv:2305.02105, 2023

[27] Pang C, Cao Y, Yang C, Luo P. Uncovering limitations of large language models in information seeking from tables. In: Findings of the Association for Computational Linguistics ACL 2024. 2024, 1388–1409

[28] Zhao Y, Zhang H, Si S, Nan L, Tang X, Cohan A. Investigating table-to-text generation capabilities of large language models in real-world information seeking scenarios. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track. 2023, 160–175

[29] Pan L, Albalak A, Wang X, Wang W. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In: Findings of the Association for Computational Linguistics: EMNLP 2023. 2023, 3806–3824

[30] Weng S, Wang Z, Zhou Y, Lu R, Liu T, Teng Z, Liu X, Liu H. Geosketch: A neural-symbolic approach to geometric multimodal reasoning with auxiliary line construction and affine transformation, 2025

[31] Ma P, Wang T H, Guo M, Sun Z, Tenenbaum J B, Rus D, Gan C, Matusik W. Llm and simulation as bilevel optimizers: a new paradigm to advance physical scientific discovery. In: Proceedings of the 41st International Conference on Machine Learning. 2024, 33940–33962

[32] Pang C, Cao Y, Zhou G, Li H, Luo P. Document-level tabular numerical cross-checking: A coarse-to-fine approach. arXiv preprint arXiv:2506.13328, 2025

[33] Zheng H, Wang S, Huang L. A survey of document-level information extraction. arXiv preprint arXiv:2309.13249, 2023

[34] Ran Q, Lin Y, Li P, Zhou J, Liu Z. NumNet: Machine reading comprehension with numerical reasoning. In: Proceedings of EMNLP. 2019

[35] Akhtar M, others . Exploring the numerical reasoning capabilities of language models: A comprehensive analysis on tabular data. arXiv preprint arXiv:2311.02216, 2023

[36] Wang D, others . Enhancing numerical reasoning with the guidance of chain-of-thought (encore). In: Proceedings of ACL (Long). 2024

[37] Li H, Yang Q, Cao Y, Yao J, Luo P. Cracking tabular presentation diversity for automatic cross-checking over numerical facts. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2020, 2599–2607

[38] Yang A, Li A, Yang B, Zhang B, Hui B, Zheng B, Yu B, Gao C, Huang C, Lv C, others . Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025

[39] Blog G. Beyond gpt: How qwen is reshaping ai. 2025. "The inclusion of substantial Chinese-language content gives Qwen an advantage in understanding Chinese cultural contexts, idioms, and specialized terminology."

[40] Wu J, Liao C, Feng M, Zhang S, Wen Z, Shao P, Xu H, Tao J. Thought-augmented policy optimization: Bridging external guidance and internal capabilities. arXiv preprint arXiv:2505.15692, 2025

[41] Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, others . Transformers: State-of-the-art natural language processing. In: Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations. 2020, 38–45

[42] Rajbhandari S, Rasley J, Ruwase O, He Y. Zero: Memory optimizations toward training trillion parameter models. In: SC20: International Conference for High Performance Computing, Networking, Storage and Analysis. 2020, 1–16

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

[44] Kwon W, Li Z, Zhuang S, Sheng Y, Zheng L, Yu C H, Gonzalez J, Zhang H, Stoica I. Efficient memory management for large language model serving with pagedattention. In: Proceedings of the 29th symposium on operating systems principles. 2023, 611–626

[45] Jiang C, Zhang M, Ye J, Fan X, Cao Y, Sun J, Xi Z, Dou S, Dong Y, Shen Y, Tong J, Wang Z, Liang T, Fei Z, Wan M, Ma G, Zhang Q, Gui T, Huang X. Predicting large language model capabilities on closed-book qa tasks using only information available prior to training. arXiv preprint arXiv:2502.04066, 2025

[46] Min S, Lyu X, Holtzman A, Artetxe M, Lewis M, Hajishirzi H, Zettlemoyer L. Rethinking the role of demonstrations: What makes in-context learning work? In: EMNLP. 2022

[47] Matsutani K, Takashiro S, Minegishi G, Kojima T, Iwasawa Y, Matsuo Y. Rl squeezes, sft expands: A comparative study of reasoning llms. arXiv preprint arXiv:2509.21128, 2025

[48] Xi Z, Chen W, Hong B, Jin S, Zheng R, He W, Ding Y, Liu S, Guo X, Wang J, others . Training large language models for reasoning through reverse curriculum reinforcement learning. In: International Conference on Machine Learning. 2024, 54030–54048

[49] Fu Y, Chen T, Chai J, Wang X, Tu S, Yin G, Lin W, Zhang Q, Zhu Y, Zhao D. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. arXiv preprint arXiv:2506.19767, 2025

[50] Ma Y, Zang Y, Chen L, Chen M, Jiao Y, Li X, Lu X, Liu Z, Ma Y, Dong X, others . Mmlongbench-doc: Benchmarking long-context document understanding with visualizations. Advances in Neural Information Processing Systems, 2024, 37: 95963–96010

[51] Wan Y, Wang W, Yang Y, Yuan Y, Huang J t, He P, Jiao W, Lyu M. Logicasker: Evaluating and improving the logical reasoning ability of large language models. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024, 2124–2155

[52] Liu T, Xu W, Huang W, Zeng Y, Wang J, Wang X, Yang H, Li J. Logic-of-thought: Injecting logic into contexts for full reasoning in large language models. In: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). 2025, 10168–10185

[53] Cheng A C, Yin H, Fu Y, Guo Q, Yang R, Kautz J, Wang X, Liu S. Spatialrgpt: grounded spatial reasoning in vision-language models. In: Proceedings of the 38th International Conference on Neural Information Processing Systems. 2024, 135062–135093

[54] Zhang Y, Xu Z, Shen Y, Kordjamshidi P, Huang L. Spartun3d: Situated spatial understanding of 3d world in large language model. In: Yue Y, Garg A, Peng N, Sha F, Yu R, eds, International Conference on Representation Learning. 2025, 73388–73406

[55] Xiong S, Payani A, Kompella R, Fekri F. Large language models can learn temporal reasoning. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024, 10452–10470

[56] Wang Y, Zhao Y. Tram: Benchmarking temporal reasoning for large language models. In: Findings of the Association for Computational Linguistics ACL 2024. 2024, 6389–6415

is currently a fourth-year PhD student at the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences (ICT, CAS), under the supervision of Professor Ping Luo. Before joining ICT, he received his bachelor's degree in information and communication engineering from Beijing University of Posts and Telecommunications. His research interests lie in natural language processing, large language models, and table understanding. He has published innovative works in top-tier conferences such as ACL and EMNLP.

is currently an associate professor in the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences (ICT, CAS). Before joining ICT, he received his Ph.D. in computer science from the University of Chinese Academy of Sciences (Institute of Computing Technology) in 2020. His research interests lie in natural language processing, document intelligence, and trustworthy AI. He has published papers in top-tier journals and conferences, such as KDD, WWW, CIKM, NeurIPS, and AAAI.

is currently an associate professor at the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences (ICT, CAS), and the University of Chinese Academy of Sciences. Before joining ICT, he served as a senior research scientist and research manager at Hewlett-Packard Labs, China. His research interests include data mining and machine learning, with a particular focus on document AI. Dr. Luo has published over 100 research papers in top-tier journals and conferences, such as IEEE TKDE, KDD, CIKM, WSDM, ICDM, and NeurIPS. He has received several prestigious awards, including the ACM CIKM Best Student Paper Award (2012), ACM CIKM Best Paper Candidate Award (2010), SDM Best Paper Candidate Award (2010), and the Doctoral Dissertation Award from the China Computer Federation (2009).