PiERN: Token-Level Routing for Integrating High-Precision Computation and Reasoning

Anonymous authors
Paper under double-blind review

Abstract

Tasks on complex systems require high-precision numerical computation to support decisions. However, current large language models (LLMs), even with enhanced reasoning capabilities, cannot integrate such computations as an intrinsic and interpretable capability with existing architectures. To this end, we propose **Physically-isolated Experts Routing Network** (PiERN), an architecture that directs computation and reasoning at token level, thereby enabling iterative alternation within a single chain of thought. We systematically evaluate PiERN on representative computation–reasoning tasks, including PDEBench and battery management tasks. Results show that PiERN achieves not only higher accuracy than directly finetuning LLMs but also significant improvements in response latency, token usage, GPU energy consumption, and experts routing accuracy compared with mainstream multi-agent approaches, while exhibiting no significant degradation in performance on MMLU and GLUE benchmarks. PiERN offers an efficient, interpretable, and scalable paradigm for interfacing language models with scientific systems.

1 Introduction

Decision-making processes in complex systems arising from scientific research and engineering practice often rely critically on high-precision numerical computation results (Kennedy & O'Hagan, 2002; Hennig et al., 2015). For example, power grid scheduling in modern power systems requires the accurate prediction of system loads and renewable energy generation to support reliable and optimal scheduling decisions (Hasan et al., 2025; Sharifhosseini et al., 2024; Biswal et al., 2024). Although large language models (LLMs) have recently achieved breakthrough progress in language understanding and logical reasoning, they still exhibit significant shortcomings in their intrinsic ability for high-precision numerical computation (Yang et al., 2025a). LLMs can generate seemingly reasonable chains of logic during reasoning, but once high-precision float operations, multi-step calculations, or partial differential equations (PDEs) solving are involved, they often yield incorrect or imprecise results (Huang et al., 2025; Jiang et al., 2025b). This deficiency severely constrains the applicability of LLMs in complex system decision-making scenarios (Alampara et al., 2025).

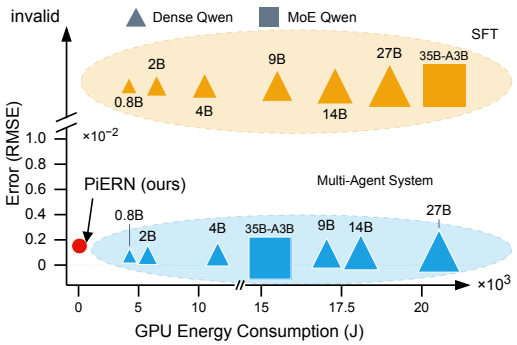

Figure 1: **PiERN achieves high precision with low GPU energy consumption.** The horizontal axis represents GPU energy consumption, and the vertical axis represents normalized RMSE on Diffusion-Sorption.

To compensate for this deficiency, existing studies have mainly pursued two approaches. The first approach is to perform end-to-end finetuning of LLMs, enabling them to directly learn numerical computation capabilities. However, it often remain insufficient when the task inputs and outputs take the form of high-dimensional numerical matrices or spatiotemporal grid solutions (e.g., PDE solution

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

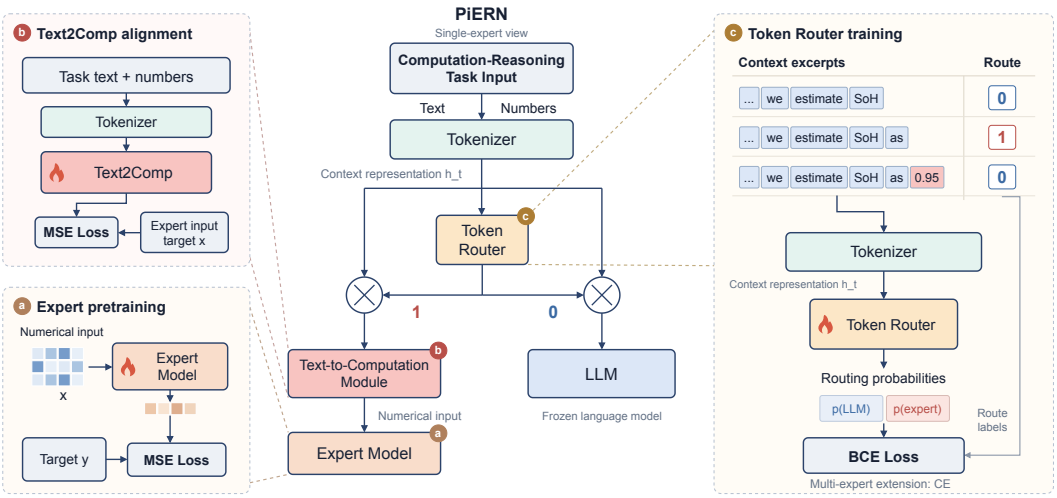

Figure 2: **(a):** Training of Expert Model for specific high-precision numeric computation tasks. **(b):** Training the Text-to-Computation Module for alignment, we adpot LLM as text-to-computation Decoder. **(c):** Training the Token Router to determine LLM for next token prediction or experts for high-precision computational results. **Middle: The overall architecture of PiERN**.

fields) (Bao et al., 2025b; Spathis & Kawsar, 2024a; Yang et al., 2025b), and may face catastrophic forgetting (Li et al., 2024a; Kotha et al., 2024). The second approach is based on multi-agent systems that invoke external experts, where LLMs act as the central brain (Schick et al., 2023; Wu et al., 2023b; Li et al., 2025), responsible for functions such as task understanding and scheduling, while external experts are responsible for executing specific high-precision computations. Although this approach ensures the accuracy, it inevitably introduces additional communication and coordination overhead, leading to low reasoning efficiency, high response latency, large GPU energy consumption (Chen et al., 2024b), and limited scalability in large-scale deployments. Recent work has explored directly integrating high-precision computation capabilities into language modeling (Wu et al., 2024; McLeish et al., 2024). Although these approaches demonstrate promising performance on specific tasks, they still remain difficult to extend to more complex computation–reasoning scenarios such as text-to-computation and multi-step computation, due to the lack of coupling computation with reasoning at the architectural level.

Therefore, how to faithfully and efficiently integrate high-precision numerical computation with language reasoning, has become the core scientific problem in advancing next-generation scientific intelligence systems.

To address this challenge, we propose **Physically-isolated Experts Routing Network** (PiERN), an architecture that integrates high-precision scientific computation experts for complex systems state solving with the language reasoning of LLMs, regardless of whether such experts are realize via neural or non-neural computational patterns. We compare the GPU energy consumption and precision to further highlight the limitations of existing approaches in high-precision computation-reasoning tasks. As shown in Figure 1, when viewed along the x-axis, multi-agent systems are able to achieve relatively high-precision, but cost extremely large GPU energy consumption. Along the y-axis, the finetuned LLMs even fail to reliably follow instructions, far from attaining sufficient computational accuracy, when the inputs and outputs are high-dimensional numerical matrix. In contrast, PiERN simultaneously overcomes both limitations, achieving high-precision with minimal GPU energy consumption, thereby demonstrating clear advantages in efficiency and scalability.

As illustrated in Figure. 2, the core idea of PiERN is to enable dynamic switching and efficient coordination between high-precision computation experts and LLMs reasoning at the token level. In contrast to multi-agent systems through external experts invocation, PiERN adopts a fundamentally different computation–reasoning granularity by internalizing experts coordination within a single chain of thought, thereby enabling coupling between high-precision computation and reasoning with preserved interpretability and efficiency. PiERN consists of three components: physically-

isolated high-precision scientific computation experts, a text-to-computation module, and a token router (Sec. 3). We conducted systematic evaluations of PiERN on a range of representative computation–reasoning tasks, including PDEBench (Takamoto et al., 2022) and battery management tasks (Sec. 4). The results show that PiERN significantly outperforms finetuned LLMs in prediction accuracy, and multi-agent baselines in inference cost; moreover, with the language expert kept frozen, PiERN exhibits no significant degradation on the Massive Multitask Language Understanding (MMLU) (Hendrycks et al., 2021) and the General Language Understanding Evaluation (GLUE) (Wang et al., 2019) benchmarks.

Our contributions are summarized as follows:

1. We introduce PiERN, an architecture that natively integrates physically-isolated high-precision scientific computation experts into LLMs, and under this architecture systematically design a stepwise training method and a token-level dynamic expert routing inference paradigm, thereby achieving the unity of accuracy, interpretability, and inference efficiency.
2. We demonstrate that, by adopting LLM as text-to-computation decoder, PiERN learns semantically conditioned transformations that directly parameterize the numerical input values of high-precision experts, thereby fundamentally extending beyond traditional time-series and scientific patterns that are restricted to purely numerical inputs.
3. We reveal PiERN to support reasoning-driven compositional high-precision computation. The computational results are propagated via reasoning chains and routed to other experts to enable collaborative and iterative computation, providing a new modeling paradigm for multi-stage computation-reasoning tasks.
4. We construct and release standardized computation–reasoning task datasets, including text-conditioned extensions of PDEBench, as well as a battery-domain dataset supporting multi-expert collaborative and iterative computation, enabling reproducible and comparable evaluation of PiERN and related researches.

2 Related Work

Multimodal Capabilities of LLMs Multimodal learning has become a central direction in today's AI, aiming to integrate text, vision, and audio within unified architectures (Zhang et al., 2024a; 2023). This field is systematically reviewed in surveys that summarize key architectures, training strategies, and challenges (Wu et al., 2023a). Recent breakthroughs such as GPT-4o (OpenAI, 2024) and GLM-4.5V (Team et al., 2025) highlight these rapid progress. Mixture-of-Experts (MoE) (Jacobs et al., 1991) has emerged as an efficient paradigm, demonstrating strong performance in multimodal generation, alignment, and controllable content creation (Li et al., 2024b; Chen et al., 2024a; Qin et al., 2023). Despite these advances, their utility in high-precision computation remains limited (Jiang et al., 2025a).

Mathematical and Reasoning Abilities LLMs have achieved impressive results in mathematical reasoning, with comprehensive surveys outlining the field's progression from comprehension to complex answer generation (Wang et al., 2025). To enhance the reliability of multi-step reasoning, research has focused on integrating feedback mechanisms, such as process and outcome rewards, to guide the reasoning process (Wei et al., 2025). Despite these advances, LLMs still fail at improving the fundamentalnumerical understanding and processing abilities (NUPA)(Yang et al., 2025a) via end-to-end finetuning. Multi-agent systems mitigate this gap by leveraging external experts through function calls (Schick et al., 2023; Patil et al., 2025; Wu et al., 2023b), significantly enhancing high-precision capabilities, yet they still suffer from communication overhead and limited scalability (Chen et al., 2024b).

Integrating High-Precision Computation with Language Recent attempts have explored directly embedding numerical representations into models (McLeish et al., 2024; Wu et al., 2024), as well as adapting LLMs to structured numerical domains such as time series through reprogramming or tokenization schemes (Jin et al., 2023; Ansari et al., 2024). A core challenge is the mismatch between the text-generation paradigm and high-dimensional numerical tasks. Standard tokenizers fragment continuous values, undermining numerical integrity (Spathis & Kawsar, 2024b; Zhang et al., 2024b), while autoregressive struggles with long, structured outputs like fine-grained PDE solutions (Bao et al., 2025a).

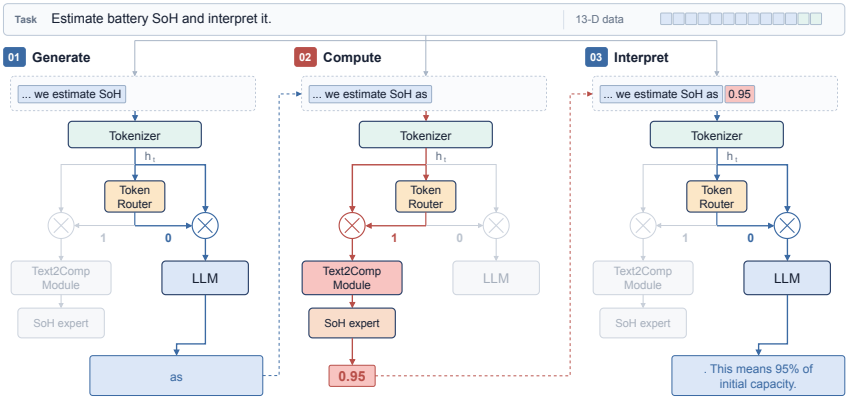

Figure 3: **Token-Level Routing for reasoning-computation inference paradigm in PiERN**. **Left:** Token Router decides to send tokenized inputs into LLM for generating the next token. **Middle:**Token Router decides to send tokenized inputs into the expert model for high-precision computation. **Right:** Token Router decides to send tokenized inputs with computation results to LLM for subsequent reasoning and planning.

3 PIERN METHODOLOGY

In this section, we detail the methodology of PiERN. PiERN integrates high-precision scientific computation experts and LLMs as modules within the same architecture. This integration is interpretable, supporting controllable training and efficient inference. We first provide an overview of the overall PiERN architecture, followed by a stepwise training method for each component module, and finally, we present the inference paradigm of alternating invocation of different experts at the token-level granularity.

3.1 ARCHITECTURE OVERVIEW

As shown in Figure 2, the PiERN architecture consists of three core components: (i) a set of high-precision scientific computation experts, which are trained on domain-specific data; (ii) a text-to-computation module, which aligns the inputs of computation-reasoning task inputs with experts input representations; and (iii) a token router, which dynamically decides whether to invoke an expert or the LLM.

3.2 STEPWISE TRAINING

We propose a stepwise training method that decouples the training processes of different modules in PiERN, reduces the interference between heterogeneous optimization objectives of numerical computation and natural language.

Stage 1: Expert Model Pre-training. As shown in Figure 2(a), for neural network based high-precision scientific computation experts, the first stage involves training on fixed numerical input-output pairs, where the data come from a scientific or industrial domain. It should be noted that this stage can be skipped for non-neural network experts. Let the training data be $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_{\text{exp}}$, where $\mathbf{x}$ denotes the input conditions and $\mathbf{y}$ denotes the corresponding ground-truth high-precision numerical computation results. The expert model f_θ approximates the true mapping by minimizing the mean squared error (MSE):

$$\mathcal{L}_{\text{exp}} = \frac{1}{N} \sum_{i=1}^{N} \| f_\theta(\mathbf{x}_i) - \mathbf{y}_i \|^2. \tag{1}$$

After convergence, the parameters of the expert model are frozen to maintain its high-precision scientific computation capability during subsequent PiERN inference.

Stage 2: Text-to-Computation Module Training. In the second stage, we optimize the text-to-computation module so that it can align inputs of language computation-reasoning task with the

inputs of high-precision scientific computation experts. The training data are $(\mathbf{s}, \mathbf{x}) \in \mathcal{D}_{\text{text2comp}}$, where $\mathbf{s}$ denotes the natural language computation-reasoning task inputs and $\mathbf{x}$ denotes the structured numerical inputs required by the experts. The mapping function g_ϕ learns to project text inputs into numerical input representations that are not only compatible with the experts but also capture how the semantics of the text may inherently influence the appropriate inputs for the experts, as shown in Figure 2(b), by minimizing the MSE loss:

$$\mathcal{L}_{\text{text2comp}} = \frac{1}{N} \sum_{i=1}^{N} \|g_\phi(\mathbf{s}_i) - \mathbf{x}_i\|^2. \tag{2}$$

To further strengthen the alignment between semantics and numerical values, we optionally introduce a contrastive loss (van den Oord et al., 2018), inspired by its successful application in cross-modal representation learning such as CLIP for vision–language alignment (Radford et al., 2021):

$$\mathcal{L}_{\text{contrastive}} = -\sum_{i=1}^{N} \log \frac{\exp(\text{sim}(g_\phi(\mathbf{s}_i), \mathbf{x}_i)/\tau)}{\sum_{j=1}^{N} \exp(\text{sim}(g_\phi(\mathbf{s}_i), \mathbf{x}_j)/\tau)}, \tag{3}$$

where $\text{sim}(\cdot, \cdot)$ denotes the similarity function (e.g., cosine similarity), and τ denotes the temperature coefficient. The final training objective is defined as the weighted sum of (2) and (3):

$$\mathcal{L}_{\text{stage2}} = \mathcal{L}_{\text{text2comp}} + \lambda \mathcal{L}_{\text{contrastive}}, \tag{4}$$

where λ is the balancing coefficient. This joint training objective can distinguish correct and incorrect language–expert pairs, improve training efficiency, and promote the text-to-computation module to accurately regress language tasks to the correct numerical inputs required by experts.

Stage 3: Token Router Training. In the final stage, we train the token router to dynamically decide at each time step whether to invoke a high-precision scientific computation expert or the LLM. Its input is the hidden representation $\mathbf{h}_t$ of all tokens at the current time step, and its output is a probability distribution $p(e \mid \mathbf{h}_t)$ over the set of experts and the LLM $\mathcal{E}$, which indicates which expert model or the LLM should be selected in the next-token prediction process. The training data are $(\mathbf{h}_t, e_t) \in \mathcal{D}_{\text{router}}$, where e_t denotes the corresponding token-level invocation label, as shown in Figure 2(c). The router is optimized using a cross-entropy (CE) loss:

$$\mathcal{L}_{\text{router}} = -\sum_{t} \sum_{e \in \mathcal{E}} y_{t,e} \log p(e \mid \mathbf{h}_t), \tag{5}$$

where $y_{t,e}$ is a one-hot vector. In particular, when $\mathcal{E}$ contains only one expert and the LLM, this objective naturally degenerates into the binary cross-entropy (BCE) loss.

3.3 Inference Paradigm

During the inference paradigm shown in Figure 3, PiERN integrates the pre-trained high-precision experts, text-to-computation module, token router, and the LLM via neural network connections. This integrated model is then used to execute language computation-reasoning tasks, and subsequent inference and planning are carried out based on the high-precision computation results.

Building on this architecture, PiERN dynamically switches between standard language reasoning and high-precision computation of expert model. For example, given inputs “The battery health”, the token router detects no computation requirement (as the next token is likely to be "is") and forwards the sequence to the LLM for ordinary next-token prediction. In contrast, given “The battery health is” when a concrete numeric value is required, the router invokes the text-to-computation module to transform the sequence into expert inputs; then the expert model returns a high-precision value (e.g., 0.95), which is appended to the sequence as “The battery health is 0.95.” The computed value is seamlessly incorporated into the context, enabling the LLM to continue reasoning (e.g., “, which is in a relatively good state for daily use.”). In this way, PiERN preserves numerical results accuracy while maintaining coherent language reasoning, planning, and decision-making.

4 Experiments and Results

4.1 Task Descriptions

We evaluate PiERN on five PDE tasks: Diffusion-Reaction, Diffusion-Sorption, Burgers, Navier-Stokes, and Acoustic Wave (Takamoto et al., 2022; Deng et al., 2021). These tasks ensure a compre-

hensive evaluation of PiERN across a range of scenarios, from linear to nonlinear, and from simple diffusion processes to more complex interactions involving multiple physical phenomena. Further task details are provided in Appendix A.1.

4.2 Language Templates and Data Synthesis

For these five tasks, we construct text-numerical computation-reasoning data by embedding numerical inputs in natural-language instructions. For all five tasks, the text-to-computation data use 80 training templates and 20 unseen test templates. The token-router labels $y \in \{0, 1, 2, 3, 4, 5\}$ indicate the LLM or one of the five task-specific experts. Appendix A.2 describes this construction. The standalone component study below uses a separate paired-template protocol to evaluate generalization independently of the system-level comparisons.

4.3 Generalization of Core Components

Experimental setup. We evaluate the text-to-computation module (text2comp) and Token Router separately on Diffusion-Sorption using seen and unseen instruction templates. We measure input reconstruction RMSE for text2comp and decision-point routing accuracy for the Router. Detailed experimental settings are provided in Appendix B.

Table 1: Generalization of core components to unseen instruction templates.

Component	**Metric**	**Seen templates**	**Unseen templates**
text2comp	Reconstruction RMSE ($\times 10^{-3}$) $\downarrow$	6.607	6.569
Token Router	Routing accuracy $\uparrow$	100%	100%

Experimental results. Table 1 validates the generalization of both core components to unseen instruction-template combinations. text2comp achieves mean reconstruction RMSEs of 6.607×10^{-3} on seen templates and 6.569×10^{-3} on unseen templates. The nearly identical errors show that numerical alignment remains stable under the tested template changes, with no observed increase in mean reconstruction error on the unseen split.

The Token Router achieves 100% test accuracy on both splits, with no false expert invocations or missed invocations at the evaluated decision points. This result also holds separately for Identity, Scaling, and Offset instructions. Thus, the learned decision of when to invoke computation transfers to unseen template combinations rather than being restricted to the complete templates used during training.

Together, these results experimentally validate two complementary capabilities: reconstructing numerical expert inputs and selecting computation-invocation points under unseen instruction-template combinations.

4.4 PiERN

Figure 4 presents a five-expert example of the PiERN architecture. PiERN-2.4B uses Qwen3.5-0.8B as its base language model. The task-specific text2comp module and Token Router are each obtained by finetuning Qwen3.5-0.8B. For each task, these three modules have a nominal total of 2.4B parameters: 0.8B from the base language model and an additional 1.6B from text2comp and the Token Router. Other PiERN variants differ only in the base LLM used for language generation; the text2comp modules, Token Router, and overall architecture remain unchanged. The routing accuracy reaches 100% during training. Simultaneously, the numerical reconstruction MSE of the text-to-computation module demonstrates excellent alignment: training and test errors remain closely aligned, with MSE reaching 10^{-4}. This indicates that PiERN can accurately bridge the gap between natural language instructions and formal computational inputs for experts. It is worth noting that under the PiERN architecture, with a 100% experts routing success rate, the computation error primarily arises from the text-to-computation module and the expert model. Appendix A show an end-to-end running example of PiERN.

Table 2: RMSE comparison between PiERN and base models

Model	Diffusion Reaction	Diffusion Sorption	Burgers	Navier–Stokes	Acoustic Wave
Downsampled Model	0.000334	0.000488	0.000064	0.000503	0.042331
PiERN	**0.001151**	**0.001371**	**0.000516**	**0.001382**	**0.054738**
Fine-tuned Model	**invalid**	**invalid**	**invalid**	**invalid**	**invalid**

4.4.1 PERFORMANCE OVER LLM FINETUNING

Finetuning is a common solution to bring domain knowledge into LLMs, thereby it can also be used to enhance LLMs with high-precision computation capabilities. Across five tasks, we compare RMSE on normalized outputs for the downsampled models, PiERN, and the fine-tuned models. Table 2 shows that PiERN produces numerical predictions across all five tasks, with a limited increase in absolute RMSE over the corresponding downsampled models. The fine-tuned model outputs are marked as invalid. Moreover, the training of LLMs is an end-to-end, data-driven process, leading the fine-tuned models have very low interpretability. In contrast, our PiERN method integrates the pre-trained, and frozen experts, significantly enhancing the interpretability and stability.

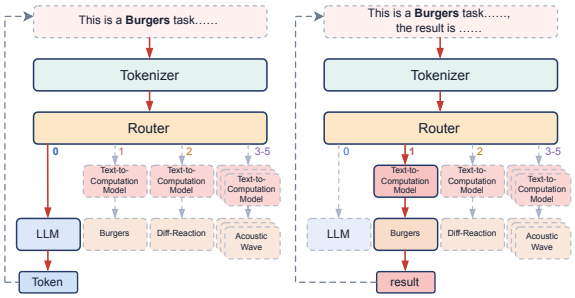

Figure 4: **A five-expert example of the PiERN architecture**. **Left:** Routing to the LLM expert for next token prediction and reasoning. **Right:** Routing to Burgers expert for high-precision computation.

4.4.2 PERFORMANCE OVER MULTI-AGENT SYSTEM

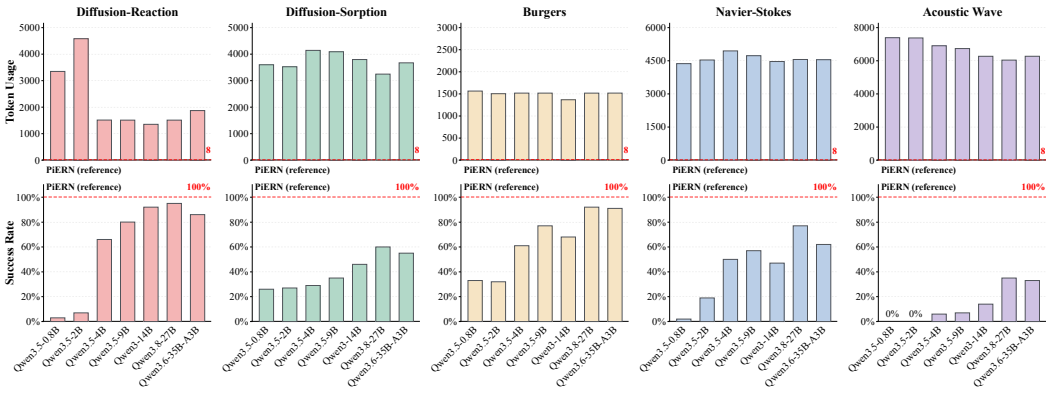

Figure 5: Token usage and success rate across different tasks comparing PiERN with MAF-based multi-agent systems. **The red dashed line denotes PiERN**.

Building on Microsoft Agent Framework (MAF), we constructed multi-agent systems using the same base LLMs of different sizes as PiERN. In these systems, different agents undertake specialized roles: LLMs are responsible for routing and dialogue interaction, while external experts are responsible for executing the corresponding tasks of high-precision numerical computation, and the agents collaborate through explicit communication. To highlight the performance advantages of the PiERN architecture, we design a series of comparative experiments to compare PiERN with these multi-agent systems. We compare performance along four dimensions, which directly reflect the core differences between PiERN and the multi-agent systems, with detailed information of these metrics provided in Appendix C. Results for LangGraph and Programmatic are also provided in Appendix C.

Table 3: Latency and GPU energy consumption of PiERN and MAF across five tasks.

Model (LLM)	Latency (s)					GPU Energy Consumption (kJ)				
	Diffusion Reaction	Diffusion Sorption	Burgers	Navier–Stokes	Acoustic Wave	Diffusion Reaction	Diffusion Sorption	Burgers	Navier–Stokes	Acoustic Wave
PiERN										
Qwen3.5-0.8B	0.62	0.63	0.62	0.64	0.77	0.056	0.078	0.057	0.123	0.171
Qwen3.5-2B	0.64	0.65	0.63	0.67	0.80	0.063	0.092	0.065	0.135	0.185
Qwen3.5-4B	0.74	0.75	0.74	0.82	0.98	0.094	0.109	0.096	0.170	0.229
Qwen3.5-9B	0.71	0.75	0.70	0.85	1.04	0.106	0.127	0.107	0.201	0.271
Qwen3-14B	0.77	0.81	0.77	0.97	1.20	0.135	0.162	0.138	0.257	0.350
Qwen3.8-27B	1.05	1.14	1.05	1.43	1.78	0.209	0.249	0.213	0.398	0.530
Qwen3.6-35B-A3B	1.19	1.25	1.22	1.28	1.44	0.142	0.166	0.151	0.215	0.282
MAF										
Qwen3.5-0.8B	58.85	42.30	10.70	46.26	108.06	5.921	4.249	1.122	4.746	11.917
Qwen3.5-2B	90.07	39.16	8.97	47.19	108.19	13.225	5.763	1.320	6.324	14.398
Qwen3.5-4B	12.50	74.06	12.35	83.06	135.82	1.933	11.592	1.911	12.667	20.728
Qwen3.5-9B	12.61	76.57	12.52	77.49	130.85	2.628	17.037	2.588	14.264	24.284
Qwen3-14B	10.14	65.36	10.51	59.73	101.03	2.697	18.102	2.756	15.141	25.317
Qwen3.8-27B	25.24	87.17	25.24	135.03	199.67	5.959	20.535	5.966	33.939	49.117
Qwen3.6-35B-A3B	42.14	98.09	21.00	114.18	179.55	6.597	15.274	3.284	17.743	29.140

Table 3 presents the comparison of latency and GPU energy consumption across five computation-reasoning tasks for different models. The PiERN series models demonstrate overwhelming speed advantages across all tests, achieving latency reductions of 1 to 2 orders of magnitude compared to the MAF-based systems. Moreover, the PiERN architecture demonstrates exceptional energy efficiency, reducing energy consumption also by 1 to 2 orders. For instance, on Diffusion-Reaction with Qwen3.5-0.8B, PiERN consumes only 0.056 kJ, less than 1% of the MAF-based system's energy consumption (5.921 kJ). The results indicate that PiERN significantly enhances computational efficiency while dramatically reducing computational costs.

Furthermore, as shown in Figure 5, PiERN consumes only 8 tokens across all five tasks, while the MAF-based systems typically require substantially more tokens to perform the same tasks. Moreover, despite consuming a large number of tokens, the MAF-based systems exhibit highly unstable expert invocation success rates, with some models nearing failure. In contrast, PiERN maintains a 100% expert routing success rate. This comparison strongly demonstrates the efficiency and robustness of PiERN.

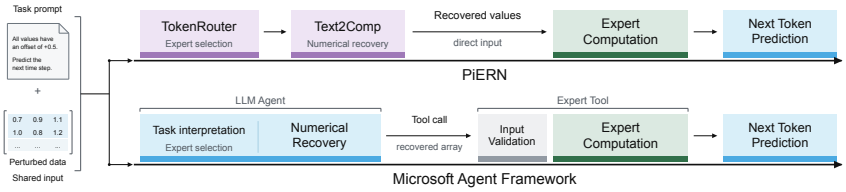

Figure 6: **Execution-flow decomposition for PiERN and the MAF-based system**.

Figure 6 presents the execution-flow decomposition for PiERN and the MAF-based system during the execution of a single task. In PiERN, the inference process alternates between next-token prediction and high-precision expert computation, eliminating redundant overhead. In contrast, multi-agent systems must sequentially perform task analysis, data alignment, code generation, tool invocation, and result analysis, with each step introducing additional costs in terms of CoT (Wei et al., 2023). This decomposition clearly reveals why PiERN can achieve significantly faster inference.

4.4.3 General Language Evaluation

We evaluate the performance of PiERN by using lm-evaluation-harness (Gao et al., 2024) framework, on the MMLU (Hendrycks et al., 2021) and GLUE (Wang et al., 2019) benchmarks. As shown in Table 5 (Appendix D), the accuracy of PiERN on both MMLU and GLUE benchmarks closely matches that of base models, even as it introduces high-precision computation capabilities

for scientific computation-reasoning tasks. The chosen of GLUE and MMLU was based on their diverse language tasks.

4.5 PiERN-BMS Modeling Paradigm

To demonstrate PiERN's capability in coordinating multiple experts for collaborative iterative computations in complex tasks, we introduce the PiERN-BMS (Battery Management System) task. As shown in Figure 7, PiERN-BMS integrates an LLM and two experts: a non-linear neural network expert for battery remaining capacity estimation, and a trained MLP expert for arbitrage profit prediction (More details are provided in Appendix E). During the inference, the token router first invokes the non-linear expert to diagnose the health status; subsequently, the LLM utilizes the state computed by the previous expert and, through language-based reasoning, triggers the linear expert to perform the final profit computation. This iterative "text-computation-reasoning-computation" loop highlights that PiERN can also enable the collaborative iterative computation of outputs from different experts.

5 Discussion

In this study, we propose PiERN, an architecture that unifies high-precision experts' computation with LLMs' reasoning. PiERN goes beyond the traditional workflow paradigm of tool invocation by enabling token-level alternating execution of expert computation and language reasoning within a single CoT. It should be noted that PiERN demonstrates significant potential for large-scale agentic RL training (Gao et al., 2025; Zhang et al., 2025), particularly in terms of scaling along the time dimension. Moreover, the statistical paradigm of current LLMs is mainly embodied in inductive reasoning and analogical reasoning, while PiERN endogenously integrates high-precision computation (deductive reasoning) into LLMs. The integration of inductive, analogical, and deductive reasoning paradigms introduces a new paradigm of intelligence. Nevertheless, the current evaluation of PiERN still remains limited to the specific computation–reasoning tasks.

Future research will focus on three main directions: first, investigating compressed representations and exact reconstruction to enable scalable handling of high-dimensional inputs, including PiERN architecture which based on both LLMs as the numerical encoders and text-to-computation decoders (Dosovitskiy et al., 2020; Wang et al., 2024); second, exploring automated methods for efficient alignment of large-scale domain knowledge, such as physical laws, operational protocols, and scheduling manuals, which encoded in textual form, to enable cross-modal text-to-computation fusion for scientific experts, thus overcoming the inefficiencies associated with manual data curation; finally, promoting the practical application of PiERN in complex system by incentivizing the computation-reasoning ability through reinforcement learning (Guo et al., 2025), such as power grid scheduling, drug discovery, and materials simulation, thereby verifying its feasibility and transformative potential as the infrastructure for next-generation scientific intelligence systems.

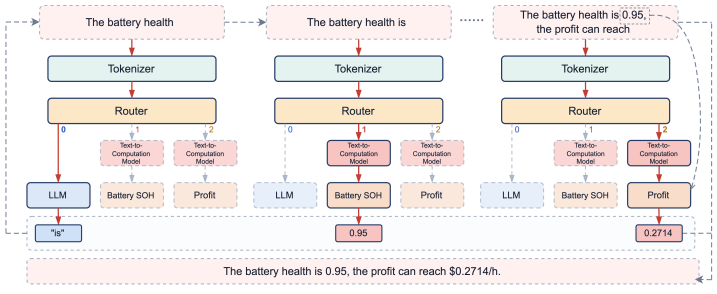

Figure 7: **PiERN-BMS Modeling Paradigm**. **Left:** Routing to the LLM for reasoning. **Middle:** Routing to a scientific expert for capacity estimation. **Right:** Routing to another expert for profit computation **using the computational result from the previous expert**.

AI USE STATEMENT

In our experiments, we used LLMs to assist in implementing parts of the technical pipeline code. We have carefully reviewed and verified all generated codes. In preparing the manuscript, we used LLMs to translate parts of our drafts that had been carefully prepared, and to polish the language. All generated content has been thoroughly checked by us to ensure accuracy. We take full responsibility for the validity of the research results and the final content of the paper.

IMPACT STATEMENT

The PiERN architecture proposed in this study will significantly accelerate scientific research and engineering innovation. First, PiERN enables faster exploration of complex physical systems (e.g., fluid dynamics, power grid scheduling, meteorological prediction) through natural language interactions, with extremely low inference latency and high computational accuracy. Furthermore, PiERN's substantial improvements in inference speed and energy efficiency contribute to the advancement of "green AI", achieving a 1-2 order of magnitude increase in energy efficiency compared to the mainstream multi-agent systems, thereby greatly reducing computational resource consumption and carbon emissions. Through natural language interface, PiERN democratizes AI for Science technologies, further fostering cross-disciplinary innovations.

However, the potential negative impacts and ethical concerns of PiERN must also be acknowledged. First, excessive reliance on PiERN's "high-precision" outputs, without consideration of the applicability boundaries of the underlying expert models may introduce risks in sensitive domains such as aerospace safety or medical device design. Second, while PiERN's efficient physical simulation capabilities are beneficial for scientific research, they may also be misused to accelerate illicit research in sensitive areas (e.g., fluid dynamics simulations for controlled substances). To mitigate these risks, we recommend implementing cross-validation mechanisms in safety-critical applications. Overall, aside from the ethical considerations mentioned, no other significant societal negative impacts have been identified.

REFERENCES

Nawaf Alampara, Mara Schilling-Wilhelmi, Martiño Ríos-García, Indrajeet Mandal, Pranav Khetarpal, Hargun Singh Grover, N. M. Anoop Krishnan, and Kevin Maik Jablonka. Probing the limitations of multimodal language models for chemistry and materials research, 2025. URL https://arxiv.org/abs/2411.16955.

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

Jiajun Bao, Nicolas Boullé, Toni JB Liu, Raphaël Sarfati, and Christopher J Earls. Text-trained llms can zero-shot extrapolate pde dynamics. *arXiv preprint arXiv:2509.06322*, 2025a.

Jiajun Bao, Nicolas Boullé, Toni J. B. Liu, Raphaël Sarfati, and Christopher J. Earls. Text-trained llms can zero-shot extrapolate pde dynamics, revealing a three-stage in-context learning mechanism, 2025b. URL https://arxiv.org/abs/2509.06322.

Biswajit Biswal, Subhasish Deb, Subir Datta, Taha Selim Ustun, and Umit Cali. Review on smart grid load forecasting for smart energy management using machine learning and deep learning techniques. *Energy Reports*, 12:3654–3670, 2024. ISSN 2352-4847. doi: https://doi.org/10.1016/j.egyr.2024.09.056. URL https://www.sciencedirect.com/science/article/pii/S2352484724006346.

Junyi Chen, Longteng Guo, Jia Sun, Shuai Shao, Zehuan Yuan, Liang Lin, and Dongyu Zhang. Eve: Efficient vision-language pre-training with masked prediction and modality-aware moe. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 1110–1119, 2024a.

Weize Chen, Jiarui Yuan, Chen Qian, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Optima: Optimizing effectiveness and efficiency for llm-based multi-agent system. *arXiv preprint arXiv:2410.08115*, 2024b.

Chengyuan Deng, Shihang Feng, Hanchen Wang, Xitong Zhang, Peng Jin, Yinan Feng, Qili Zeng, Yinpeng Chen, and Youzuo Lin. OpenFWI: Large-scale multi-structural benchmark datasets for seismic full waveform inversion. *arXiv preprint arXiv:2111.02926*, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020. URL https://api.semanticscholar.org/CorpusID:225039882.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Wei Gao, Yuheng Zhao, Tianyuan Wu, Shaopan Xiong, Weixun Wang, Dakai An, Lunxi Cao, Dilxat Muhtar, Zichen Liu, Haizhou Zhao, Ju Huang, Siran Yang, Yongbin Li, Wenbo Su, Jiamang Wang, Lin Qu, Bo Zheng, and Wei Wang. Rollart: Scaling agentic rl training via disaggregated infrastructure, 2025. URL https://arxiv.org/abs/2512.22560.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Mahmudul Hasan, Zannatul Mifta, Sumaiya Janefar Papiya, Paromita Roy, Pronay Dey, Nafisa Atia Salsabil, Nahid-Ur-Rahman Chowdhury, and Omar Farrok. A state-of-the-art comparative review of load forecasting methods: Characteristics, perspectives, and applications. *Energy Conversion and Management: X*, 26:100922, 2025. ISSN 2590-1745. doi: https://doi.org/10.1016/j.ecmx.2025.100922. URL https://www.sciencedirect.com/science/article/pii/S2590174525000546.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*, 2021.

Philipp Hennig, Michael A. Osborne, and Mark Girolami. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2179):20150142, 2015.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, January 2025. ISSN 1558-2868. doi: 10.1145/3703155. URL http://dx.doi.org/10.1145/3703155.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Xi Jiang, Jian Li, Hanqiu Deng, Yong Liu, Bin-Bin Gao, Yifeng Zhou, Jialin Li, Chengjie Wang, and Feng Zheng. Mmad: A comprehensive benchmark for multimodal large language models in industrial anomaly detection, 2025a. URL https://arxiv.org/abs/2410.09453.

Zhuoxuan Jiang, Haoyuan Peng, Shanshan Feng, Fan Li, and Dongsheng Li. Llms can find mathematical reasoning mistakes by pedagogical chain-of-thought, 2025b. URL https://arxiv.org/abs/2405.06705.

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

Marc C. Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 63(3):425–464, 01 2002. ISSN 1369-7412. doi: 10.1111/1467-9868.00294. URL https://doi.org/10.1111/1467-9868.00294.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. Understanding catastrophic forgetting in language models via implicit inference, 2024. URL https://arxiv.org/abs/2309.10105.

Hongyu Li, Liang Ding, Meng Fang, and Dacheng Tao. Revisiting catastrophic forgetting in large language model tuning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4297–4308, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.249. URL https://aclanthology.org/2024.findings-emnlp.249/.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models, 2025. URL https://arxiv.org/abs/2501.05366.

Yunxin Li, Shenyuan Jiang, Baotian Hu, Longyue Wang, Wanqi Zhong, Wenhan Luo, Lin Ma, and Min Zhang. Uni-moe: Scaling unified multimodal llms with mixture of experts, 2024b. URL https://arxiv.org/abs/2405.11273.

Sean McLeish, Arpit Bansal, Alex Stein, Neel Jain, John Kirchenbauer, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, Jonas Geiping, Avi Schwarzschild, et al. Transformers can do arithmetic with the right embeddings. *Advances in Neural Information Processing Systems*, 37:108012–108041, 2024.

OpenAI. Hello gpt-4o, 2024. URL https://openai.com/index/hello-gpt-4o/.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: large language model connected with massive apis. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385.

Can Qin, Shu Zhang, Ning Yu, Yihao Feng, Xinyi Yang, Yingbo Zhou, Huan Wang, Juan Carlos Niebles, Caiming Xiong, Silvio Savarese, Stefano Ermon, Yun Fu, and Ran Xu. Unicontrol: A unified diffusion model for controllable visual generation in the wild, 2023. URL https://arxiv.org/abs/2305.11147.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. URL https://api.semanticscholar.org/CorpusID:231591445.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessí, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: language models can teach themselves to use tools. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Seyed Mohammad Sharifhosseini, Taher Niknam, Mohammad Hossein Taabodi, Habib Asadi Aghajari, Ehsan Sheybani, Giti Javidi, and Motahareh Pourbehzadi. Investigating intelligent forecasting and optimization in electrical power systems: A comprehensive review of techniques and applications. *Energies*, 17(21), 2024. ISSN 1996-1073. doi: 10.3390/en17215385. URL https://www.mdpi.com/1996-1073/17/21/5385.

Dimitris Spathis and Fahim Kawsar. The first step is the hardest: pitfalls of representing and tokenizing temporal data for large language models. *Journal of the American Medical Informatics Association*, 31(9):2151–2158, 07 2024a. ISSN 1527-974X. doi: 10.1093/jamia/ocae090. URL https://doi.org/10.1093/jamia/ocae090.

Dimitris Spathis and Fahim Kawsar. The first step is the hardest: Pitfalls of representing and tokenizing temporal data for large language models. *Journal of the American Medical Informatics Association*, 31(9):2151–2158, 2024b.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: an extensive benchmark for scientific machine learning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, et al. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning, 2025. URL https://arxiv.org/abs/2507.01006.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018. URL https://api.semanticscholar.org/CorpusID:49670925.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2019 International Conference on Learning Representations (ICLR)*, 2019.

Peng-Yuan Wang, Tian-Shuo Liu, Chenyang Wang, Ziniu Li, Yidi Wang, Shu Yan, Chengxing Jia, Xu-Hui Liu, Xinwei Chen, Jiacheng Xu, et al. A survey on large language models for mathematical reasoning. *ACM Computing Surveys*, 2025.

Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, Jiazheng Xu, Keqin Chen, Bin Xu, Juanzi Li, Yuxiao Dong, Ming Ding, and Jie Tang. Cogvlm: visual expert for pretrained language models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Ting-Ruen Wei, Haowei Liu, Xuyang Wu, and Yi Fang. A survey on feedback-based multi-step reasoning for large language models on mathematics. *arXiv preprint arXiv:2502.14333*, 2025.

Hengkui Wu, Panpan Chi, Yongfeng Zhu, Liujiang Liu, Shuyang Hu, Yuexin Wang, Chen Zhou, Qihao Wang, Yingsi Xin, Bruce Liu, Dahao Liang, Xinglong Jia, and Manqi Ruan. Scaling particle collision data analysis, 2024. URL https://arxiv.org/abs/2412.00129.

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S Yu. Multimodal large language models: A survey. In *2023 IEEE International Conference on Big Data (BigData)*, pp. 2247–2256. IEEE, 2023a.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 3(4), 2023b.

Haotong Yang, Yi Hu, Shijia Kang, Zhouchen Lin, and Muhan Zhang. Number cookbook: Number understanding of language models and how to improve it, 2025a. URL https://arxiv.org/abs/2411.03766.

Liu Yang, Siting Liu, and Stanley J. Osher. Fine-tune language models as multi-modal differential equation solvers. *Neural Networks*, 188:107455, 2025b. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2025.107455. URL https://www.sciencedirect.com/science/article/pii/S089360802500334X.

Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. Mm-llms: Recent advances in multimodal large language models, 2024a. URL https://arxiv.org/abs/2401.13601.

Hanchen Zhang, Xiao Liu, Bowen Lv, Xueqiao Sun, Bohao Jing, Iat Long Iong, Zhenyu Hou, Zehan Qi, Hanyu Lai, Yifan Xu, Rui Lu, Hongning Wang, Jie Tang, and Yuxiao Dong. Agentrl: Scaling agentic reinforcement learning with a multi-turn, multi-task framework, 2025. URL https://arxiv.org/abs/2510.04206.

Xiang Zhang, Juntai Cao, and Chenyu You. Counting ability of large language models and impact of tokenization. *arXiv preprint arXiv:2410.19730*, 2024b.

Yiyuan Zhang, Kaixiong Gong, Kaipeng Zhang, Hongsheng Li, Yu Qiao, Wanli Ouyang, and Xiangyu Yue. Meta-transformer: A unified framework for multimodal learning, 2023. URL https://arxiv.org/abs/2307.10802.

A An End-to-End PiERN Running Example

A.1 PDE Task Descriptions

The 1d diffusion-reaction task involves solving a 1d PDE that models the diffusion and reaction of substances, covering both linear and nonlinear dynamics. The governing equation is

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} + R(u), \tag{6}$$

where $u(x,t)$ represents the concentration of the substance at time t and position x, D is the diffusion coefficient, and $R(u)$ is the reaction term, which represents the chemical reaction rate. The 1d diffusion-sorption task builds on 1d diffusion-reaction task by incorporating adsorption phenomena, further increasing its complexity. The governing equation is

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - k_a u(1-u), \tag{7}$$

where $u(x,t)$ is the concentration of the adsorbed substance, D is the diffusion coefficient, k_a is the adsorption constant, and $(1-u)$ represents the availability of adsorption sites. The Burgers equation is a widely used nonlinear PDE for modeling shock waves, turbulence, and other phenomena. The governing equation is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}, \tag{8}$$

where $u(x,t)$ is the velocity field, and ν is the fluid viscosity.

The Navier-Stokes task models one-dimensional compressible fluid motion. The governing equations are (Takamoto et al., 2022)

$$\partial_t \rho + \nabla\cdot(\rho\mathbf{v}) = 0, \tag{9a}$$

$$\rho(\partial_t \mathbf{v} + \mathbf{v}\cdot\nabla\mathbf{v}) = -\nabla p + \eta\nabla^2\mathbf{v} + (\zeta+\eta/3)\nabla(\nabla\cdot\mathbf{v}), \tag{9b}$$

$$\partial_t\left(\epsilon + \frac{\rho|\mathbf{v}|^2}{2}\right) + \nabla\cdot\left[\left(\epsilon + p + \frac{\rho|\mathbf{v}|^2}{2}\right)\mathbf{v} - \mathbf{v}\cdot\sigma'\right] = 0, \tag{9c}$$

where ρ is the mass density, $\mathbf{v}$ is the fluid velocity, p is the gas pressure, $\epsilon = p/(\Gamma-1)$ is the internal energy density with adiabatic index $\Gamma = 5/3$, σ' is the viscous stress tensor, and η and ζ are the shear and bulk viscosities, respectively. The Acoustic Wave task models acoustic-wave propagation in a two-dimensional medium with constant density and spatially varying wave speed. The governing equation is (Deng et al., 2021)

$$\nabla^2 p(\mathbf{r},t) - \frac{1}{c(\mathbf{r})^2}\frac{\partial^2 p(\mathbf{r},t)}{\partial t^2} = s(\mathbf{r},t), \tag{10}$$

where $\mathbf{r} = (x,z)$ denotes spatial position, t is time, $p(\mathbf{r},t)$ is the pressure wavefield, $c(\mathbf{r})$ is the wave speed, and $s(\mathbf{r},t)$ is the seismic source term.

Data Preprocessing. For the PDEBench tasks, the raw data are derived from the official PDEBench. To reduce computational cost, we apply fixed-step spatial downsampling (reducing resolution from 1024 to 64 or 32 grid points) and temporal sliding windows to generate historical-future state pairs (u_t, u_{t+1}). These downsampled numerical pairs serve as the numerical ground truth for our subsequent computation-reasoning tasks data synthesis pipeline.

A.2 Language Templates and Data Synthesis

As shown in Figure 8, to endow PiERN with the ability to efficiently integrate language text with numerical computation and perform token-level routing, we constructed a **language templates and data synthesis pipeline**, which transforms raw numerical pairs into pairs that combine both text and numbers, and we further constructed the data for training the token router.

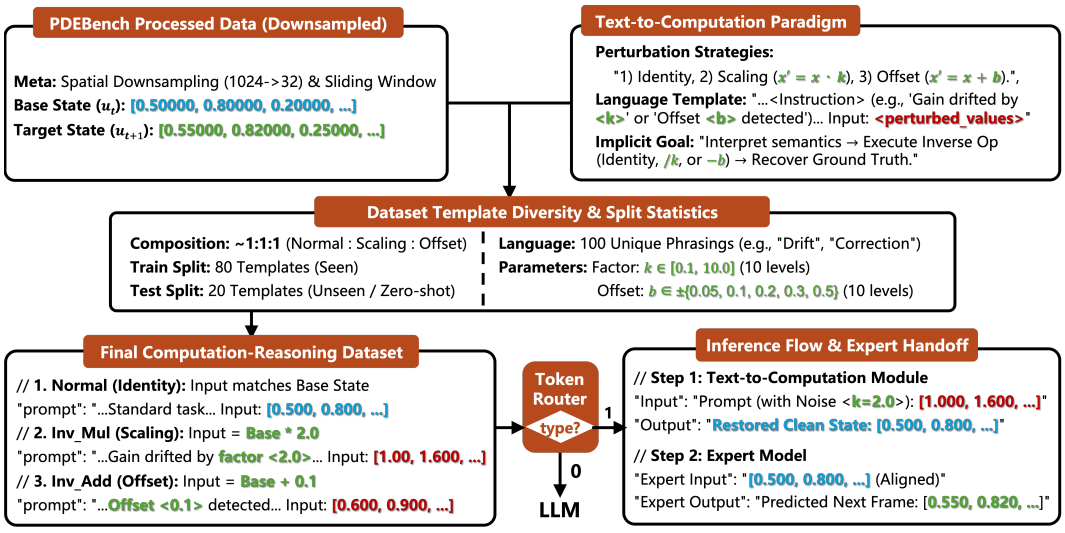

Figure 8: **The language templates and data synthesis pipeline. Top:** Raw data is spatially downsampled and paired, then injected with controlled perturbations (e.g., Scaling $\times k$) via natural language templates. **Middle:** An 80/20 split to evaluate text-to-computation module generalization. **Bottom:** Token router dynamically chose the LLM or experts.

A.2.1 Text-to-Computation Language Patterns

We designed $N = 100$ **unique language patterns** to wrap the numerical data pairs within natural language contexts.

- **Perturbation Strategies:** The dataset is balanced across three types of transformations, which are applied to the numerical pairs through corresponding textual description, and with an approximate ratio of $1 : 1 : 1$:
  - **Identity:** Standard prediction tasks where the model learns to predict values directly from the input.
  - **Scaling:** The input x is multiplied by a constant factor k ($x' = x \cdot k$), and the model must learn to divide by k.
  - **Offset:** The input x is shifted by a constant b ($x' = x + b$), and the model must learn to subtract b.
- **Split Strategy:** We enforce an **80/20 split**, reserving 20 templates exclusively for testing (Zero-shot). This evaluates the PiERN's ability to generalize to unseen test language instruction patterns, rather than memorizing.

A.2.2 PiERN Training Data Instances

To clarify the training process, we present concrete examples of token router and text-to-computation module of PiERN from our dataset. These examples illustrate the specific input-output pairs for training.

Example 1: Training Sample for Text-to-Computation Module

Task Type: Offset (Correction of System Error $b = 0.1$)

Input Prompt (Text + Numbers):
"The following is the history record of the 1d diffusion-reaction state. The data contains a system error of 0.1. Please subtract 0.1 to correct it and use the FNO model to calculate the next frame. Input stream: [0.79920, 0.79920, 0.79920, ...]"

Target Output (Reconstruction Tensor):
`[0.69920, 0.69920, 0.69920, ...]`

Note: The text-to-computation module learns to execute the operation $x_{clean} = x_{input} - 0.1$ as instructed by the semantic cue "subtract 0.1",thereby allowing the potential text semantics to influence the numerical input of the expert, enabling text-numeric multimodal learning..

Example 2: Training Sample for Token Router

The token router is trained to output `Type 1` (Computation Mode) **only** when the model response is fully prepared for expert high-precision computation.

Case A: High-Precision Computation Trigger (Type 1)
Input Context: *"...The data contains a zero-point drift (offset 0.1). Please subtract 0.1 compensation... Data follows: [0.80836, ...]. Okay, the scientific computation prediction result is:"*
Label: 1 (Trigger Expert for High-Precision Computation)

Case B: Ongoing Next Token Prediction Generation (Type 0)
Input Context: *"Okay, the scientific computation prediction"*
Label: 0 (Continue Next Token Prediction)

A.3 An End-to-End PiERN Inference Running Example

During inference, PiERN dynamically passes data between the LLM and Experts. The following workflow illustrates the core mechanism for the 1d diffusion-reaction task described above (Offset $b = 0.1$).

User Prompt: *"The data contains a system error of 0.1. Please subtract 0.1... Input stream: [0.79920, ...]"*

1. **Phase 1: Semantic Parsing & LLM Reasoning Phase**
   The LLM generates/processes the prefix text. The **Token Router** monitors each token (Output `Type 0`) until the instruction is fully formulated.
2. **Phase 2: Token Router Trigger**
   Trigger: The LLM generates the final cue: *"...prediction result is:"*.
   Action: The Router detects this specific suffix and switches output to **Type 1**.
   Transfer: Text generation is halted. The full context (instruction + noisy data `[0.79920...]...prediction result is:`) is passed to the **Text-to-Computation Module**.
3. **Phase 3: Reconstruction & Executing Experts High-Precision Computation**
   Restoration: The module interprets *"subtract 0.1"* and executes the correction:

   $$\hat{x}_{clean} = [0.79920, \dots] - 0.1 = [0.69920, \dots]$$

   Expert Execution: The reconstruction tensor $\hat{x}_{clean}$ is then forwarded to the frozen 1d diffusion-reaction expert.
   Prediction: The expert calculates the next state: $y_{pred} = [0.72550, \dots]$.
4. **Phase 4: Response Output**
   The numerical result y_{pred} is appended to the response.
   Final Output: *"...prediction result is: [0.72550, 0.72550, ...]."*

B Experimental Details for Component Generalization

This appendix provides the experimental settings for the component generalization study in Section 4.3.

Task and template protocol. We evaluate the text-to-computation module (text2comp) and Token Router separately on Diffusion-Sorption, using Identity, Scaling, and Offset instructions. Seen and unseen requests are paired on the same test numerical inputs, isolating the effect of instruction-template changes. Both components are based on Qwen3.5-0.8B and trained independently.

text2comp evaluation. We evaluate 1,000 template pairs with 100 numerical samples per pair, yielding 100,000 requests per split. The metric is mean input reconstruction RMSE.

Router evaluation. The Router is evaluated separately on 1,000 template pairs with 1,000 numerical samples per pair, yielding 1,000,000 requests per split. At predefined token decision points, it predicts whether to continue language generation or invoke the task's expert. Accuracy is measured over these decision points, totaling 22,000,000 across both splits.

C Metrics for PiERN and Multi-Agent Systems

We compare performance along four dimensions, which directly reflect the core differences between PiERN and the multi-agent systems: **(i) Latency**: determines the response speed of the system in interactive scenarios, directly affecting user experience and the feasibility of real-time decision-making tasks; **(ii) Token Usage**: measures the additional overhead in inference caused by long-context understanding and cross-agent communication, directly reflecting the user-side inference cost and the overall economic efficiency of task execution; **(iii) GPU Energy Consumption**: reflects resource utilization and energy efficiency, serving as a key metric for evaluating deployment scalability and sustainability; **(iv) Success Rate**: measures the proportion of correct high-precision results obtained in computation–reasoning tasks, directly reflecting the system's reliability and task completion capability.

Table 4 reports the latency and GPU energy consumption of LangGraph and Programmatic across the five tasks. Figure 9 compares token usage and success rate across MAF, LangGraph, and Programmatic, with PiERN as the reference.

Table 4: Latency and GPU energy consumption of LangGraph and Programmatic across five tasks.

Model (LLM)	Latency (s)					GPU Energy Consumption (kJ)				
	Diffusion Reaction	Diffusion Sorption	Burgers	Navier–Stokes	Acoustic Wave	Diffusion Reaction	Diffusion Sorption	Burgers	Navier–Stokes	Acoustic Wave
LangGraph										
Qwen3.5-0.8B	54.96	41.73	20.42	65.68	108.94	5.511	4.187	2.076	7.240	13.660
Qwen3.5-2B	86.23	43.30	12.81	65.81	101.59	12.658	6.358	1.888	8.787	13.254
Qwen3.5-4B	15.23	72.24	12.35	85.08	135.01	2.354	11.208	1.911	12.889	20.738
Qwen3.5-9B	12.54	70.72	14.14	78.13	123.84	2.581	14.648	2.944	14.552	23.387
Qwen3-14B	10.20	53.45	10.45	62.39	101.57	2.692	14.411	2.737	15.114	25.100
Qwen3.8-27B	26.22	96.73	25.48	133.75	194.89	6.053	21.628	6.030	32.926	46.974
Qwen3.6-35B-A3B	24.25	98.94	23.18	114.58	176.81	3.641	14.978	3.527	18.040	26.415
Programmatic										
Qwen3.5-0.8B	19.99	44.42	39.15	23.81	52.83	2.034	4.487	3.935	2.677	5.506
Qwen3.5-2B	27.39	31.21	10.49	9.79	2.86	4.027	4.568	1.536	1.303	0.409
Qwen3.5-4B	4.68	4.18	7.59	22.28	4.37	0.726	0.668	1.176	3.427	0.736
Qwen3.5-9B	5.55	13.85	19.29	20.22	4.96	1.143	2.881	3.961	3.789	0.997
Qwen3-14B	2.94	3.35	3.20	3.75	3.65	0.768	0.872	0.815	0.934	0.951
Qwen3.8-27B	17.03	19.44	31.35	8.81	8.33	3.793	4.516	6.978	2.373	2.277
Qwen3.6-35B-A3B	22.59	17.34	34.17	13.04	10.38	3.490	2.740	5.258	2.070	1.671

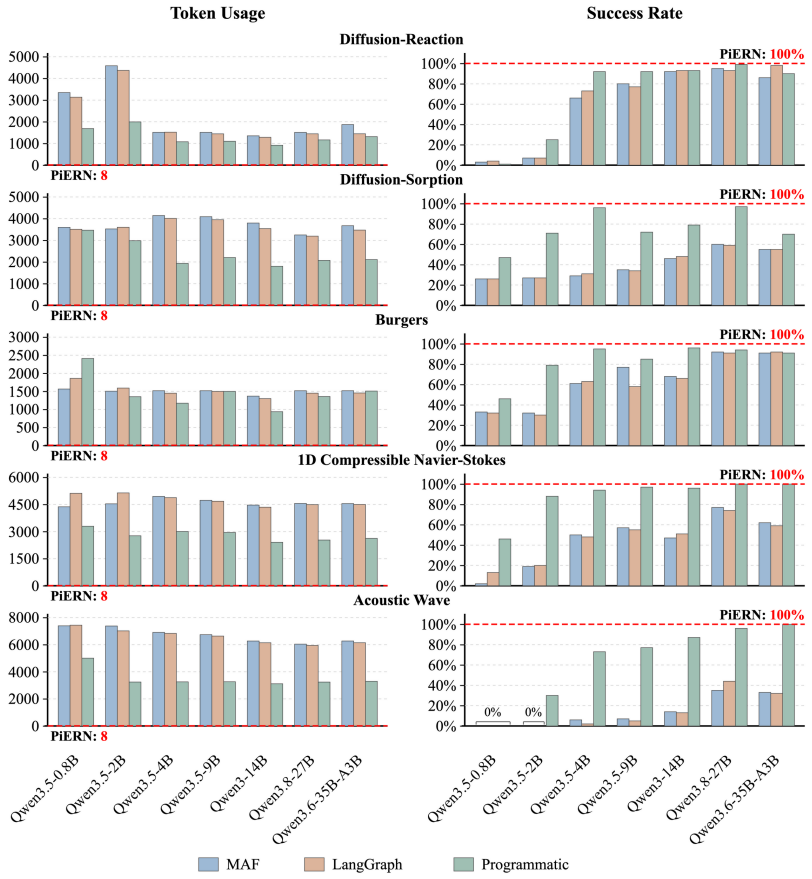

Figure 9: Token usage and success rate across different tasks comparing PiERN with multi-agent systems. **The red dashed line denotes PiERN**.

D General Language Evaluation

Table 5: Performance Comparison of PiERN and Qwen Models on MMLU and GLUE Benchmark

Model	MMLU	MNLI	RTE	SST2	QQP	QNLI	MRPC
PiERN-2.4B	50.14	33.50	53.09	51.43	36.87	49.65	67.83
Qwen3.5-0.8B	50.24	33.25	52.35	51.03	36.81	49.55	68.38
PiERN-3.6B	57.67	46.57	64.87	80.80	78.59	62.08	74.11
Qwen3.5-2B	57.62	46.58	64.26	80.16	79.90	62.02	72.79
PiERN-5.6B	70.23	47.04	73.30	89.00	64.41	76.45	47.54
Qwen3.5-4B	70.12	46.77	73.29	88.88	64.49	76.37	47.30
PiERN-10.6B	70.70	39.98	81.58	89.46	83.56	54.33	75.17
Qwen3.5-9B	70.64	39.61	81.59	89.45	83.42	54.59	75.00
PiERN-15.6B	78.93	67.32	77.64	92.42	82.34	83.75	78.42
Qwen3-14B	78.82	67.02	77.62	92.43	82.29	83.96	78.68
PiERN-28.6B	80.27	64.33	83.20	87.92	73.70	83.37	41.17
Qwen3.8-27B	80.22	64.27	82.67	88.07	73.81	83.49	40.69
PiERN-36.6B	82.76	60.05	76.88	87.50	80.93	80.17	78.93
Qwen3.6-35B-A3B	82.77	60.05	76.17	87.73	81.06	80.27	78.19

E PiERN-BMS Modeling Paradigm

In this section, we provide a detailed description of the PiERN-BMS (Battery Management System). We describe the battery remaining capacity estimation and arbitrage profit prediction tasks, implemented with DeepONet and MLP experts, respectively, along with the data synthesis process used to construct the dataset for PiERN-BMS.

E.1 Battery Remaining Capacity Prediction Task (Non-Linear Task)

The primary objective of this task is to predict the remaining State of Health (SoH) of a battery based on time-series current data. Battery health is a core indicator for measuring battery performance degradation, typically defined as the ratio of the current remaining capacity to the initial capacity of the battery:

$$SoH = \frac{C_{current}}{C_{initial}} \times 100\% \tag{11}$$

The battery degradation is a complex non-linear process affected by multiple coupled physicochemical reactions and mechanisms, such as electrode material aging, electrolyte decomposition, and solid electrolyte interphase growth. These dynamics are usually modelled with PDEs. In the PiERN-BMS framework, this task serves as the benchmark for the non-linear neural network expert.

The input data for this task comprises two main categories of information constituting 13 input feature values. The first component is the time-series current data, consisting of 11 current values collected over a 2-hour period with a sampling interval of 12 minutes. The second component includes the specific time point to be predicted and its corresponding current value. The initial health status of the battery is set to 1 by default.

Regarding the data scale and source, the dataset is generated based on the Pseudo-Two-Dimensional (P2D) model constructed via COMSOL Multiphysics modeling. The repository contains a total of 9,600 samples matching the pattern of "current data - time point - state of health". Specifically, the training dataset contains 7,200 samples, while the test dataset contains 2,400 samples.

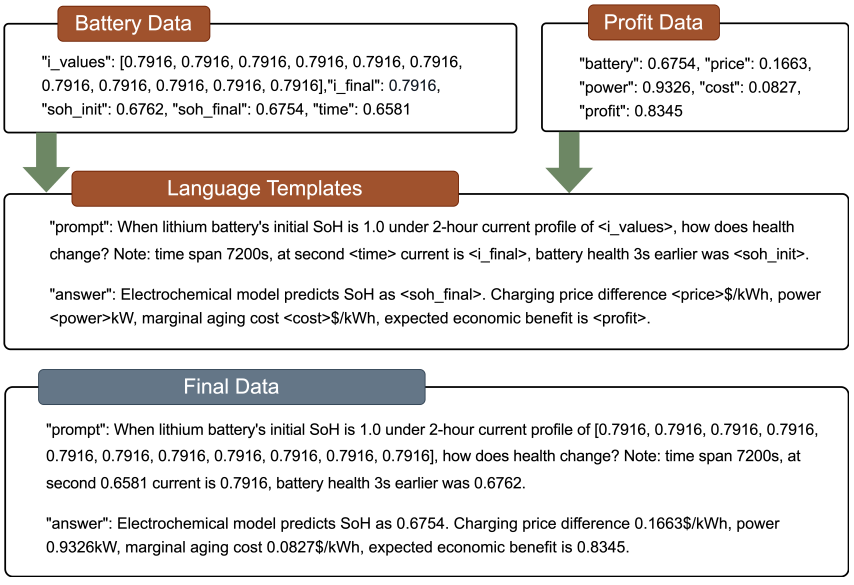

Figure 10: **Pipeline of language-template-based data synthesis for PiERN-BMS**. **Top**: Extraction of raw physical and numerical information, including battery capacity data and profit data. **Middle**: Application of Language templates to wrap structured data into natural language contexts. **Bottom**: The Final data stream, where reasoning tokens and computational tokens are interleaved to form a unified training sequence.

E.2 Battery Arbitrage Profit Calculation Task (Linear Task)

The core of this task is to calculate the battery profit from arbitrage in electricity markets or electricity bill saving. This represents a linear calculation task where the model must perform precise arithmetic based on physical states and market conditions. The calculation involves four key parameters: the battery degradation coefficient α, the price difference between charging and discharging Δp, the battery charge-discharge power P, and the marginal degradation cost c_a. The final profit R is formulated as:

$$R = \Delta p \cdot P - \alpha \cdot c_a \cdot 1200 \tag{12}$$

where 1200 represents a specific time constant or cycle factor.

The dataset for this task consists of 10,000 data entries. In terms of data partitioning, the training data accounts for 90% of the total dataset, with the remaining 10% allocated for testing.

E.3 PiERN-BMS Modeling Paradigm

As shown in Figure 10, to explore the collaborative iterative computation capabilities of the PiERN architecture within multiple experts and to better train the text-to-computation module and token-level router, we have designed multiple language templates specifically for the two tasks described above. These templates are essential for bridging the gap between raw numerical data and natural language reasoning. By combining the templates with the numeric data from the P2D model and market simulations, we enable the text-to-computation module to understand semantic contexts and generate numbers accurately across various scenarios.

The data construction process involves wrapping the 13-feature vector of the non-linear task and the parameters of the linear task into a cohesive narrative. For instance, a data sample starts with a user query embedding the current profile, prompting the router to trigger the non-linear expert for SoH estimation. This is followed by an LLM reasoning phase where the health state is analyzed, leading to a decision phase where market parameters are introduced. Finally, the router triggers the profit-prediction expert to compute the profit. This interleaved format of "text-computation-reasoning-computation" ensures the model learns to coordinate different experts effectively.

E.4 PERFORMANCE OVER LLM FINETUNING

Building upon the constructed PiERN-BMS modeling paradigm and datasets, we further conduct experiments to evaluate the performance of PiERN against traditional LLM finetuning approaches.

Setups. Finetuning is a common solution to bring domain knowledge into LLMs, thereby it can also be used to enhance LLMs with high-precision computation capabilities. We compare the MSE among the finetuned LLMs and PiERN on the test data of both Non-Linear Task and Linear Task. For fair comparison, LLMs are finetuned by the same training data used in the training of expert models. Meanwhile, only one language template is used to generate training data to let LLMs focus on computation. On the selection of LLMs, we used open-source Qwen3.5 models of various sizes (0.8B, 2B, 4B, and 9B) with full-parameter finetuning, while PiERN uses Qwen3.5-0.8B for its text-to-computation modules.

Table 6: Comparison of MSE on battery tasks.

Task	**PiERN (Ours)**	**Qwen3.5 (SFT)**			
		0.8B	**2B**	**4B**	**9B**
Non-Linear Task	$\mathbf{9.095 \times 10^{-5}}$	3.098×10^{-2}	2.818×10^{-3}	9.272×10^{-3}	4.357×10^{-3}
Linear Task	$\mathbf{1.772 \times 10^{-5}}$	1.243×10^{-4}	1.056×10^{-4}	2.567×10^{-5}	2.303×10^{-5}

Results. Table 6 shows the accuracy of PiERN on these two tasks is consistently better than all finetuned LLMs. Our method has the lowest MSE in all cases. The MSE of PiERN on the Non-Linear Task is one to two orders of magnitude lower than that of the best finetuned baseline.

Due to the pre-training data, LLMs are better at text understanding and generation than computational tasks. Therefore, LLMs usually cannot achieve high accuracy in computation tasks even after finetuning. Moreover, because the training of LLMs is an end-to-end, data-driven process, the models have very low interpretability. By comparison, our method integrates a pre-trained expert model (DeepONet for SoH prediction and MLP for profit prediction), which greatly enhances the model interpretability and stability. In summary, our method has demonstrated its advantages over other LLMs even with finetuning in terms of accuracy and interpretability in computation-reasoning tasks.