Interpretable LLM Control for Sustainable Liquid Cooling in HPC Data Centers

Sahand Ghorbanpour [*1] **Ashwin Ramesh-Babu** [*1] **Avisek Naug** [*1] **Antonio Guillen Perez** [1]
Ricardo Lune Gutierrez [1] **Vineet Gundecha** [1] **Soumyendu Sarkar** [*1]

Abstract

The rise of AI workloads has driven the need for efficient liquid cooling in high-density data centers, yet current systems lack intelligent, interpretable control. We propose a novel framework combining Reinforcement Learning (RL) with Large Language Models (LLMs) to optimize end-to-end liquid cooling, from server cabinets to the cooling towers, while providing natural language explanations for control actions. Our approach includes a hybrid of a multi-agent Reinforcement Learning and a Large Language Model controller. Evaluated on a baseline of Oak Ridge National Lab's Frontier Supercomputer based scalable liquid cooling Modelica model, it improves temperature stability and energy efficiency, offering a scalable and transparent solution for sustainable data center cooling.

1. Introduction

Rising HPC and AI workloads have sharply increased data center energy use, projected to consume a significant % of global electrical energy by 2030. Cooling—especially in liquid-cooled (LC) systems for dense GPU clusters—accounts for a major share (Luo et al., 2024; Ott et al., 2024). LC offers superior thermal efficiency over air cooling and can reduce energy and emissions by up to 63% (Azarifar et al., 2024; Habibi Khalaj & Halgamuge, 2017). Yet, many LC systems still rely on fixed-level or fixed-rule controllers (Chen et al., 2020; Shahi et al., 2022; Lucchese et al., 2020), lacking adaptability to workload and environmental variability.

Advanced setups like ORNL's Frontier use closed-loop LC with Cooling Distribution Units (CDUs), pumps, -cooling towers (CT), and heat exchangers (Wetter et al., 2014; Greenwood, 2020; Kumar et al., 2024), requiring dynamic control across nonlinear, delayed systems. While reinforcement learning (RL) enables adaptive control, its opacity hinders trust in safety-critical environments (Raschka, 2025). This paper proposes and evaluates three cooling controllers:

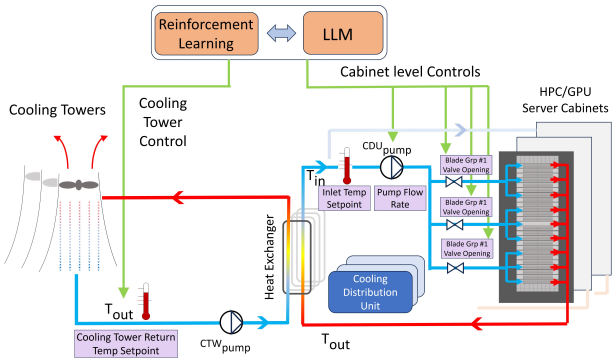

Figure 1: System Overview of end-to-end Control of Liquid Cooled Data Center. The CDU RL agents control the HPC server cabinets. The Cooling Towers are controlled by the CT RL agents.

1. **Pure RL:** Learns end-to-end cooling policies.
2. **Hybrid LLM+RL:** Combines LLMs for planning and high level policies and RL for actuation.
3. **LLM-only:** Generates control actions via prompt-based reasoning.

All are paired with a **universal LLM explanation module** that generates natural language rationales for transparency. Experiments use a simulation environment built on a Frontier-based FMU (Brewer et al., 2024a), evaluated with real workload traces and open-source LLMs (e.g., LLaMA, Qwen). **Key contributions:**

- Comparative study of RL-only, LLM+RL, and LLM-only controllers.
- LLM-based explanation module for interpretable cooling control.
- Evaluation on a realistic platform using performance and interpretability metrics.

2. Related Works

Increasing energy demands and environmental concerns have intensified research into data center cooling. Traditional rule-based methods lack adaptability for modern HPC

[*]Equal contribution [1]Hewlett Packard Labs, Hewlett Packard Enterprise, USA. Correspondence to: Soumyendu Sarkar <soumyendu.sarkar@hpe.com>.

Co-Build Workshop at the 42nd International Conference on Machine Learning, Vancouver, Canada. Copyright 2025 by the author(s).

and AI workloads (Ellsworth Jr, 2012; Fulpagare & Bhargav, 2015), while liquid cooling offers higher efficiency (Westra, 2009; Khalaj & Halgamuge, 2017) but adds control complexity (Patterson et al., 2016).

ML approaches, especially RL, have shown promise in air-cooled systems (Meta Engineering Team, 2024; Zhan et al., 2025) and are now being extended to liquid cooling (Brewer et al., 2024b; Sarkar et al., 2025; Li et al., 2024). Industrial adoption is growing rapidly, with direct-to-chip cooling projected to reach $11.89B by 2034 (Data Center Frontier, 2024; GlobeNewswire, 2025).

Yet, RL's lack of interpretability hinders deployment in safety-critical settings (DataRoot Labs, 2025; AAAI, 2025). LLMs offer a solution by explaining black-box decisions in natural language (Raschka, 2025). This work combines multi-agent RL with LLM-based explanation to enable transparent, trustworthy liquid cooling control.

3. Simulation Environment

We evaluate all controllers using a high-fidelity simulator based on ORNL's Frontier liquid-cooled data center. The environment models detailed interactions between cabinet-side CDUs and cooling towers (CTs), enabling closed-loop RL and LLM control.

3.1. System Overview

Figure 1 illustrates the system: server blades are cooled by CDUs, which circulate coolant via heat exchangers and pumps. Heat is transferred through a shared hot-water loop to CTs, with performance influenced by weather (e.g., wet-bulb temperature).

3.2. Modeling Framework

The system is implemented in Modelica and exported as a Functional Mock-up Unit (FMU), forming a nonlinear control environment. A Gym-compatible Python interface enables ML training. As shown in Figure 2, agents control CDU flow rates, valve positions, and CT outlet temperature. Components are defined via a hierarchical JSON schema and instantiated using AutoCSM (Greenwood et al., 2024).

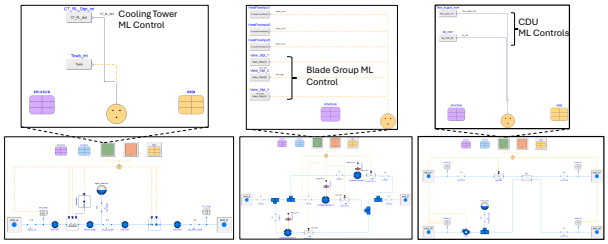

Figure 2: Modelica model augmentations: RL-actuated replacements for rule-based control at CDUs and CTs.

3.3. Control Interfaces and MDPs

The simulator defines two MDPs: (1) a **Cooling Tower MDP** for adjusting CT outlet temperature to reduce energy use, and (2) a **Blade Group MDP** for managing coolant flow and setpoints at the cabinet level. Both share a transition model but are trained independently due to weak thermal coupling. Centralized training was unstable, so we use independent training with shared inference. Inputs like server heating and weather come from public data sets.

3.4. Scalability

The framework scales from small setups to full HPC systems via JSON configuration and FMU presets. It supports multi-agent control, hybrid actions, and interpretability features.

4. Control Strategy Design

4.1. RL-Only Controller

We implement a hierarchical PPO agent comprising a meta-controller (every τ=5 steps) and five cabinet-level agents (every step). Each cabinet observes a 6-D state (return temperatures and power), while the cooling tower observes 4-D (fan power and water temperatures). Cabinet actions:

$$\mathbf{a}^{\text{cab}} = \left[T_{\text{set}},\ \Delta P,\ V_1, V_2, V_3\right],$$

where $V_{1\text{-}3}$ are softmax-normalized valve openings. The tower agent selects discrete deltas for the outlet water temperature. The reward balances thermal accuracy, efficiency, and workload alignment,

$$r_{\text{cab}} = 0.7\, r_{\text{align}} + 0.3\, r_{\text{eff}} + r_{\text{temp}}.$$

Full hyper-parameters are listed in Appendix D.

4.2. Hybrid LLM + RL Controller

As illustrated in Figure 3, the hybrid controller combines a pretrained language model (LLM) with a reinforcement learning (RL) policy. At each timestep, the LLM produces a high-level intent vector $\mathbf{z}_t$, which guides the RL policy in generating refined control actions:

$$\mathbf{a}_t^{\text{RL}} = \pi_\theta(\mathbf{s}_t, \mathbf{z}_t).$$

The final action is a weighted blend of the LLM's proposal $\mathbf{a}_t^{\text{LLM}}$ and the RL output $\mathbf{a}_t^{\text{RL}}$, modulated by a dynamic mixing factor:

$$\mathbf{a}_t = (1 - \alpha(\mathbf{s}_t)) \cdot \mathbf{a}_t^{\text{RL}} + \alpha(\mathbf{s}_t) \cdot \mathbf{a}_t^{\text{LLM}},$$
$$\alpha(\mathbf{s}_t) = w_{\text{temp}} \cdot \alpha_{\text{temp}} + w_{\text{energy}} \cdot \alpha_{\text{energy}}.$$

This design separates roles: the LLM provides high-level guidance via $\mathbf{z}_t$, while the RL policy handles low-level

actuation. The mixing function $\alpha(\mathbf{s}_t)$ adapts based on the current thermal state and energy profile, ensuring the system leans on the RL policy in uncertain or volatile regimes.

Although the LLM can correct or steer policy behavior with reasoning-informed adjustments, its influence is bounded by conservative gating strategies that prioritize stability and policy safety. This architecture reflects practical deployment needs—enabling flexible control while retaining the structure of a trained RL policy.

4.3. LLM-Only Controller

The LLM-only controller removes the RL component entirely and directly generates control actions using a language model fine-tuned via imitation learning on expert RL trajectories. At inference time, it receives structured prompts encoding the current state, actuator limits, and contextual information, and outputs both structured action commands and natural-language rationales.

Unlike the hybrid controller, this architecture enables fully end-to-end decision-making: the LLM interprets observations and issues control decisions without blending constraints or conflicting signals. As a result, the LLM-only controller exhibits greater consistency and coherence—particularly in reasoning-intensive or symbolically aligned tasks—though it needs explicit safety guarantees as guard rail directives unlike RL.

Imitation Learning. A dataset $\mathcal{D} = \{(x_i, y_i)\}$ pairs system observations with expert rationales and actions. The LLM is trained to minimize:

$$\mathcal{L}_{\text{IL}}(\theta) = \frac{1}{N} \sum_{i,t} -\log p_\theta(y_{i,t} \mid x_i, y_{i,<t})$$

Inference. Prompts include system state, actuator limits, and qualitative labels (e.g., "above target"). The LLM outputs a reasoning trace followed by structured control actions.

Actions. Outputs include cabinet actions $\mathbf{a}_t^{\text{cab}} = [\tau_t, f_t, \alpha_t^{(1)}, \alpha_t^{(2)}, \alpha_t^{(3)}]$ and CT delta $\mathbf{a}_t^{\text{tower}} \in \{-0.20, ..., 0.20\}$, parsed inline or structurally.

Design Insights. Key improvements include encoding thermal deviation direction, training on temperature-specific prompts, and validating outputs for safety and plausibility.

4.4. Control Setup

Observations included blade-group return temperatures and heating inputs, as well as cooling tower metrics such as power consumption and water return temperature. Actions comprised setpoints for CDU supply temperature and coolant flow rate, valve openings for blade-group branches, and the cooling tower's output water temperature setpoint.

4.5. LLM-Based Interpretability

To ensure transparency across all controllers, we employ a post hoc LLM-based module that generates natural language rationales from observation–action pairs. This explanation module is model-agnostic and operates independently of training and control execution. At each step during inference, it receives the system state $\mathbf{s}_t$ and action $\mathbf{a}_t$, and formats them into a structured prompt:

`Observation:` [state vector]
`Action Taken:` [action vector]
Explain why this action is appropriate.

Few-shot examples—typically drawn from expert or confident trajectories—guide the LLM to generate grounded, context-aware explanations. Outputs link control actions to thermal conditions, identifying phenomena such as heat imbalance, undercooling risk, or workload alignment.

In the RL-only controller, this module provides retrospective explanations of the agent's black-box behavior, enabling human-in-the-loop validation without modifying the underlying policy. For the hybrid controller, the module has access to both the RL and LLM components of the blended action, allowing for more expressive and interpretable rationales. While the LLM does not introspect RL internals, it helps externalize control behavior in terms aligned with physical constraints and operational goals.

The hybrid design leverages RL for risk-averse fine control but tempers the LLM's influence through a conservative mixing factor $\alpha(\mathbf{s}_t)$. By contrast, the pure LLM variant enjoys full autonomy—yielding higher interpretability and consistency in reasoning-dominant scenarios, albeit without the fallback safety net of an RL policy.

5. Experimental Setup

We evaluate three controller architectures: (1) RL-only, (2) hybrid RL+LLM, and (3) LLM-only. Experiments are run on a Modelica-based simulation of ORNL's Frontier system, compiled as a Functional Mock-up Unit (FMU) and interfaced with Python via Gymnasium.

The simulator models five compute cabinets and one cooling tower, with configurable thermal and fluid dynamics. Controllers act at 1-second intervals based on observed temperatures, power, and environmental signals. All architectures are tested under identical workloads, ambient conditions, and actuator constraints.

To ensure fairness, both the hybrid and LLM-only controllers use the same pretrained LLM, with fine-tuning performed on expert trajectories from the RL-only policy. We also evaluate base (unfine-tuned) LLMs to isolate the effect of imitation learning.

A shared reward function balances thermal accuracy, energy

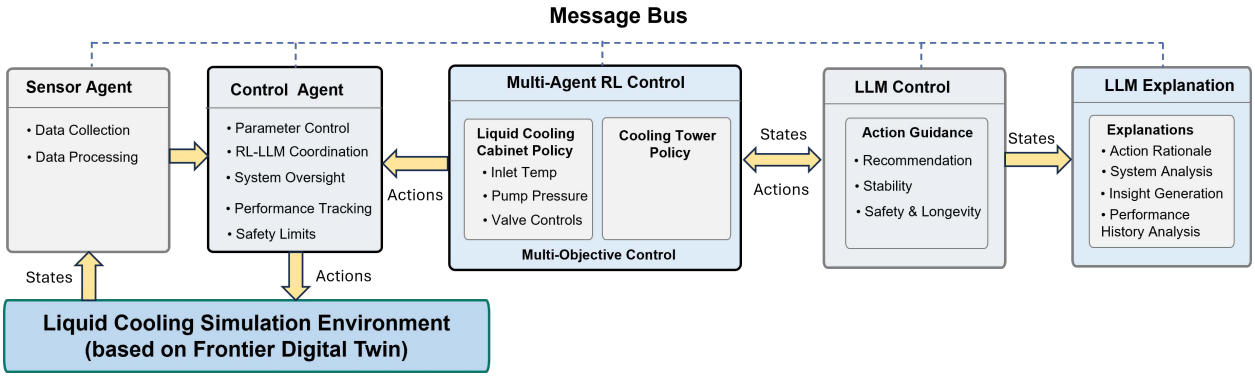

Figure 3: Hybrid controller: LLM proposes high-level intents; RL refines and blends them, with explanations generated on demand.

Question: Given the observation (temp1: 334, temp2: 347, temp3: 345 and power 1: 45426, power2: 20307, power3: 45328), predict the full actions which will be CDU temperature setpoint, the flowrate and the distribution of coolant for the 3 branches. Provide a detailed explanation on why the model suggested the corresponding values for the actions. Given the information in the context, try to first categorize the observations and use that to generate reasoning for the action prediction.

Response: 39.44, 38.74, 0.18, 0.46, 0.37

Reasoning: The system receives temperatures of 309.39, 310.1, and 314.4 for branches 1 to 3, which fall into the Medium, Medium, and High categories respectively. Corresponding power inputs are 15056.03, 15997.03, and 45338.3, classified as Low, Low, and Medium High. Given this thermal and electrical state, the system sets the CDU supply temperature setpoint to 23.69, which is categorized as Low. This decision likely reflects an adaptive cooling strategy where higher temperatures and powers trigger increased flow rate or adjusted valve openings. The resulting flow rate is 27.09, and the coolant is distributed with fractions of 0.36, 0.31, and 0.33 to branches 1 through 3 respectively

Figure 4: LLM explanation for cabinet control

Question: Given the observation (4372.40, 4373, 305, 299, 756), The first two dimensions corresponds to the power consumptions, , third is the temperature of the coolant leaving the tower, 4th representing the current tower temperature and the 5th corresponding to the outside air wetbulb temperature. Considering these variables, suggest the right control for the tower temperature. Strictly choose one from [-0.20, -0.15, -0.10, -0.05, 0, 0.05, 0.10, 0.15, 0.20] actions. Please generate a detailed response on why you chose this action given the observation.

Response: -0.05

Reasoning: Low power usage suggests the system is lightly loaded. Warm coolant leaving the tower indicates poor heat rejection. High outside air wetbulb temp may reduce cooling efficiency. System temperature is very high and needs urgent reduction. A small decrease of 0.05 is suggested to maintain thermal balance.

Figure 5: LLM explanation for cooling tower control

use, and workload alignment, with fixed coefficients across all controllers. We did not optimize the reward weights in this study.

Tests with meta-controller intervals of 1s and 10s showed no significant degradation in thermal response, though longer intervals may require further analysis. Additional details are provided in Appendix B.

5.1. Simulation Environment

We use the `SmallFrontierModel`, simulating five compute blocks with CDUs and a four-cell cooling tower (two observable). Controllers adjust coolant temperature, flow, pressure, valve positions, and tower outlet delta. Constraints include 293.15–313.15 K coolant temps, 25–38 kPa pressure, and 0.1–10 m³/s flow. All controllers use a unified reward balancing thermal stability and energy efficiency, tested under identical workloads and weather traces.

5.2. Controller Architectures

RL Controller. A hierarchical PPO-based setup uses a high-level policy (every 5 steps) to control CT and cabinet targets. Low-level cabinet agents output continuous control actions. Policies share a feedforward architecture and are trained for 2M steps with 8 parallel workers.

Hybrid Controller. Combines RL control with LLM-guided adjustments. The LLM receives structured prompts and its suggestions are mixed with RL actions via a dynamic coefficient $\alpha(\mathbf{s}_t)$, based on thermal and energy conditions. Outputs are safety-checked before execution.

LLM-Only Controller. Trained via imitation learning on RL trajectories, the LLM receives system state prompts and outputs JSON-formatted actions with rationales. Fine-tuned versions of Qwen3 8B and LLaMA 3.1 8B are compared to base models. Fallback mechanisms ensure robustness.

5.3. Evaluation Metrics

Detailed definitions of all metrics are available in Appendix F.

6. Results and Discussion

We compare three controllers—RL-only, hybrid RL+LLM, and LLM-only—using a Frontier-based simulation. The LLM-only controller consistently outperforms others across temperature stability, energy efficiency, and composite reward (Fig. 6, Fig. 7), maintaining tighter thermal control and reduced energy use due to structured reasoning and expert policy imitation.

Cooling tower action distributions reveal distinct patterns: RL favors conservative mid-range actions; hybrid spreads across the range due to policy mixing; and LLM-only prefers low to mid-range, energy-efficient actions, guided

Table 1. Performance comparison of RL, Hybrid, and LLM (LLaMA FT) controllers across all evaluation metrics.

Metric	RL Only	RL+LLM Hybrid	LLM Only (LLaMA FT)
Temperature Stability	0.6326 ± 0.0000	0.6519 ± 0.0007	**0.7473 ± 0.0052**
Energy Efficiency	0.6082 ± 0.0000	0.6025 ± 0.0033	**0.6821 ± 0.0038**
Weighted Reward	0.6253 ± 0.0000	0.6371 ± 0.0012	**0.7278 ± 0.0035**
Average Temperature (°C)	26.37 ± 0.00	26.56 ± 0.01	**27.57 ± 0.05**
Avg. Temperature Deviation (K)	3.677 ± 0.000	3.484 ± 0.007	**2.533 ± 0.052**
Max Temperature Deviation (K)	16.935 ± 0.000	**16.623 ± 0.054**	16.790 ± 0.127
Time out of Target Range (%)	0.3 ± 0.0	0.3 ± 0.0	**17.4 ± 3.9**
Cooling Power (W)	128568.1 ± 0.0	130411.7 ± 1068.3	**104295.7 ± 1237.9**
Power Usage Effectiveness	1.260 ± 0.000	**1.263 ± 0.002**	1.208 ± 0.002

Table 2. LLM Controller Comparison: Base vs. Fine-Tuned variants of LLaMA and Qwen. Best performance is shown in bold.

Metric	LLaMA Base	LLaMA FT	Qwen Base	Qwen FT
Cooling Power (W)	106901.4 ± 381.8	104295.7 ± 1237.9	**101091.6 ± 1673.9**	105590.0 ± 269.6
Power Usage Effectiveness	1.213 ± 0.001	1.208 ± 0.002	**1.202 ± 0.003**	1.210 ± 0.001
Temperature Stability	0.7176 ± 0.0014	**0.7473 ± 0.0052**	0.7234 ± 0.0067	0.7206 ± 0.0009
Energy Efficiency	0.6742 ± 0.0012	0.6821 ± 0.0038	**0.6919 ± 0.0051**	0.6782 ± 0.0008
Weighted Reward	0.7046 ± 0.0010	**0.7278 ± 0.0035**	0.7139 ± 0.0054	0.7079 ± 0.0008
Average Temperature (°C)	27.27 ± 0.01	**27.57 ± 0.05**	27.33 ± 0.07	27.30 ± 0.01
Avg. Temperature Deviation (K)	2.830 ± 0.014	**2.533 ± 0.052**	2.773 ± 0.067	2.800 ± 0.009
Max Temperature Deviation (K)	16.792 ± 0.126	16.790 ± 0.127	16.830 ± 0.144	**16.735 ± 0.106**
Time out of Target Range (%)	1.4 ± 0.7	**17.4 ± 3.9**	4.4 ± 3.4	0.7 ± 0.2

Figure 6: Comparison of controller performance on primary evaluation metrics.

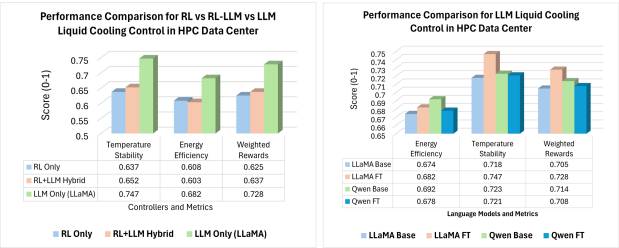

Figure 7: Comparison of controller performance on primary evaluation metrics.

by reasoning-based policies (Fig. 8).

Temperature profiles further highlight the LLM-only controller's stability, with average cabinet temps at 27.57°C, minimal fluctuation (avg. deviation 2.53K), and no thermal spikes. It also significantly outperforms others in time spent within ±2K of the target.

Comparing base vs. fine-tuned LLMs (Fig. 6), fine-tuned LLaMA shows marked gains in thermal control and overall reward. Qwen's base model excels in energy efficiency but sees inconsistent results post-tuning, suggesting architecture-dependent fine-tuning efficacy.

The LLM-only controller excels where RL and hybrid falter. RL struggles with delayed dynamics and sparse rewards; hybrid underutilizes LLM input due to fixed blending and parallel control logic. In contrast, the LLM-only approach leverages pretraining, structured prompts, and symbolic reasoning to produce generalizable, stable, and energy-aware control—learning heuristics like "increase cooling with rising power" without explicit supervision.

Crucially, the LLM also generates intermediate natural language rationales (Figs. 4, 5) that enhance transparency and decision quality. Reasoning-augmented prompts improve performance, especially under variable workloads, enabling the LLM to act as both a controller and an interpretable agent.

6.1. Base vs. Fine-Tuned LLMs

In our experiments at the time of publishing this paper, we got different outcomes between fine-tuned vs base Llama and Qwen models. For Qwen, unlike the prevailing intuition the *base* LLM model which operates in a true few-shot setting—receiving a large number of relevant demonstrations at inference time, outshone the *fine-tuned* LLM model which encodes its policy primarily within adapter weights from RL traces and receives less per-example guidance. This leads to some intriguing insights as follows.
Few-shot prompting allows in-context adaptation, enabling the base model to handle outliers—like a hot cabinet in branch 2—unlike fine-tuned models that follow a fixed policy with limited flexibility. It also enhances robustness to distribution shifts by re-anchoring the model to local conditions under extreme workloads, whereas fine-tuned models remain tied to their training data. Prompting retains generalist priors, avoiding the overspecialization that fine-tuning on narrow data can cause. It also supports better cross-condition generalization, as base models guided by structured prompts often outperform fine-tuned models that overfit to specific RL patterns.

These observations suggest that the base model's few-shot learning capability is a powerful—and sometimes preferable—mechanism for rapid adaptation. However, to close the remaining gap, we are extending fine-tuning with additional RL-generated trajectories to cover outliers and exploring training schemes that encourage the LLM to generalize beyond pure imitation of the expert policy, with a healthy balance of few shot training and chain of thought.

7. Conclusion

We introduced a modular and interpretable control framework for sustainable liquid cooling in data centers by leveraging large language models (LLMs) alongside reinforcement learning (RL). Our study compared three controller paradigms—RL-only, hybrid RL+LLM, and LLM-only—on a high-fidelity simulator modeled after the ORNL Frontier system. Through extensive experiments, we showed that LLMs, when trained via imitation learning and guided by structured prompting, can surpass RL methods in key metrics such as temperature stability and energy efficiency. Beyond performance, our architecture provides interpretability through chain-of-thought reasoning, enabling natural language auditing. These findings support the promise of language-driven control policies that unify optimization and explainability—an essential direction for scalable and trustworthy data center operations.

References

AAAI. Aaai-25 tutorial: Reinforcement learning with large

language models. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, February 2025.

Azarifar, M., Arik, M., and Chang, J.-Y. Liquid cooling of data centers: A necessity facing challenges. *Applied Thermal Engineering*, pp. 123112, 2024.

Brewer, W., Maiterth, M., Kumar, V., Wojda, R., Bouknight, S., Hines, J., Shin, W., Greenwood, S., Grant, D., Williams, W., and Wang, F. A digital twin framework for liquid-cooled supercomputers as demonstrated at exascale. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–18. IEEE, November 2024a. doi: 10.1109/sc41406.2024.00029. URL http://dx.doi.org/10.1109/SC41406.2024.00029.

Brewer, W., Maiterth, M., Kumar, V., Wojda, R., Bouknight, S., Hines, J., Shin, W., Greenwood, S., Grant, D., Williams, W., et al. A digital twin framework for liquid-cooled supercomputers as demonstrated at exascale. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–18. IEEE, 2024b.

Chen, H., Han, Y., Tang, G., and Zhang, X. A dynamic control system for server processor direct liquid cooling. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 10(5):786–794, 2020.

Data Center Frontier. Turn up the volume: Data center liquid immersion cooling advancements fill 2024. *Data Center Frontier*, August 2024.

DataRoot Labs. The state of reinforcement learning in 2025. *DataRoot Labs Blog*, January 2025.

Ellsworth Jr, M. J. New ashrae thermal guidelines for air and liquid cooling. In *2012 SC Companion: High Performance Computing, Networking, Storage and Analysis (SCC)*, volume 1, pp. 942–961. IEEE Computer Society, 2012.

Fulpagare, Y. and Bhargav, A. Advances in data center thermal management. *Renewable and Sustainable Energy Reviews*, 43:981–996, 2015.

GlobeNewswire. Direct-to-chip liquid cooling market set for 5x growth, reaching $11.89 billion by 2034. *GlobeNewswire*, April 2025.

Greenwood, S. Transform–a vision for modern advanced reactor system-level modeling and simulation using modelica. Technical report, Oak Ridge National Laboratory (ORNL), Oak Ridge, TN (United States), 2020.

Greenwood, S., Kumar, V., and Brewer, W. Thermo-fluid modeling framework for supercomputer digital twins: Part 2, automated cooling models. In *America Modelica Conference*, pp. 210–219. Modelica Association, 2024.

Habibi Khalaj, A. and Halgamuge, S. K. A Review on efficient thermal management of air- and liquid-cooled data centers: From chip to the cooling system. *Appl. Energy*, 205:1165–1188, November 2017. ISSN 0306-2619. doi: 10.1016/j.apenergy.2017.08.037.

Khalaj, A. H. and Halgamuge, S. K. A review on efficient thermal management of air-and liquid-cooled data centers: From chip to the cooling system. *Applied energy*, 205: 1165–1188, 2017.

Kumar, V., Greenwood, S., Brewer, W., Williams, W., Grant, D., and Parkison, N. Thermo-fluid modeling framework for supercomputer digital twins: Part 1, fluid modeling framework for supercomputer digital twins: Part 1, demonstration at exascale. In *America Modelica Conference*, pp. 199–207. Modelica Association, 2024.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Li, Q., Liu, Q., Ran, Y., Sun, T., Chen, S., and Luo, J. Deepcchp: Intelligent comprehensive optimization of energy consumption and carbon emission for data center cchp systems. In *2024 9th Asia Conference on Power and Electrical Engineering (ACPEE)*, pp. 1451–1459. IEEE, April 2024. doi: 10.1109/ACPEE60788.2024.10532357.

Lucchese, R., Varagnolo, D., and Johansson, A. Controlled direct liquid cooling of data servers. *IEEE Transactions on Control Systems Technology*, 29(6):2325–2338, 2020.

Luo, W., Fan, R., Li, Z., Du, D., Wang, Q., and Chu, X. Benchmarking and dissecting the nvidia hopper gpu architecture. In *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 656–667. IEEE, 2024.

Meta Engineering Team. Simulator-based reinforcement learning for data center cooling optimization. *Engineering at Meta*, September 2024.

Moritz, P., Nishihara, R., Wang, S., Tumanov, A., Liaw, R., Liang, E., Elibol, M., Yang, Z., Paul, W., Jordan, M. I., and Stoica, I. Ray: A distributed framework for emerging ai applications. In *Proceedings of the 13th USENIX Symposium on Operating Systems Principles*, pp. 561–577. USENIX Association, 2018.

Ott, B., Wenzel, P., and Radgen, P. Analysis of cooling technologies in the data center sector on the basis of patent applications. energies 2024, 17, 3615, 2024.

Patterson, M. K., Krishnan, S., and Walters, J. M. On energy efficiency of liquid cooled hpc datacenters. In *2016 15th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, pp. 685–693. IEEE, 2016.

Raschka, S. The state of reinforcement learning for llm reasoning. *Sebastian Raschka's Blog*, April 2025.

Sarkar, S. et al. Hierarchical multi-agent framework for carbon-efficient liquid-cooled data center clusters. *arXiv preprint arXiv:2502.08337*, February 2025. URL https://arxiv.org/abs/2502.08337. Version 1 (2025-02-12).

Shahi, P., Deshmukh, A. P., Hurnekar, H. Y., Saini, S., Bansode, P., Kasukurthy, R., and Agonafer, D. Design, development, and characterization of a flow control device for dynamic cooling of liquid-cooled servers. *Journal of Electronic Packaging*, 144(4):041008, 2022.

Westra, L. Liquid cooling in data centers. *ASHRAE Transactions*, 115:231, 2009.

Wetter, M., Zuo, W., Nouidui, T. S., and Pang, X. Modelica buildings library. *Journal of Building Performance Simulation*, 7(4):253–270, 2014.

Zhan, X. et al. Data center cooling system optimization using offline reinforcement learning. *arXiv preprint arXiv:2501.15085*, February 2025. URL https://arxiv.org/abs/2501.15085. Version 2 (2025-02-14).

A. Controller Design

System Inputs and Outputs

Compute Cabinet
Inputs (Observations):

- **Temp_1–3 (K):** Return temperatures from blade-groups 1, 2, and 3. (A blade-group is a collection of servers cooled by a single coolant flow channel.)
- **Power_1–3 (kW):** Heat generated by server workloads in blade-groups 1, 2, and 3.

Outputs (Actions):

- **CDU_Supply_Temp_Setpoint (°C, z-scored):** Coolant supply temperature setpoint delivered to all three branches.
- **Pump_Flow_Rate (m^3/h, z-scored):** Coolant flow rate pumped through the cabinet.
- **Valve_Opening_1–3 (0–1):** Fractional opening of the three branch valves, controlling coolant distribution to each blade-group.

Cooling Tower
Inputs (Observations):

- Power consumption of cell 1 in cooling tower 1.
- Water return temperature from the cooling tower.

Output (Action):

- Cooling tower water leaving temperature setpoint.

A.1. Nature of Hybrid and LLM controllers

The hybrid controller combines a pretrained language model (LLM) with a reinforcement learning (RL) policy to produce refined control actions. At each timestep, both the LLM and the RL policy independently generate candidate actions based on the current system state and recent trends. These outputs are then blended using a dynamic mixing factor, $\alpha(st)\alpha(\mathbf{s}_t)\alpha(st)$, computed from features such as temperature deviation and energy usage. This design allows the LLM to adjust or guide the RL output, offering reasoning-informed refinements while retaining the structure and safety of the trained policy. However, because the two modules operate in parallel rather than hierarchically, the LLM's influence is often constrained—particularly in volatile regimes—due to conservative mixing strategies that tend to favor the RL output.

By contrast, the LLM-only controller omits the RL component entirely and directly produces control actions using a language model fine-tuned via imitation learning on expert RL trajectories. This controller receives structured prompts encoding the system state, actuator limits, and relevant context, and returns both a natural language rationale and structured action commands. Its architecture enables fully end-to-end decision-making, where the LLM is solely responsible for interpreting observations and generating control actions, unencumbered by blending constraints or conflicting signals. As a result, the LLM-only controller exhibits greater consistency and coherence in scenarios where reasoning and symbolic alignment are critical to stable thermal management.

B. Experimental Setup

Training and Evaluation Setup. In the hybrid controller, the LLM is used in a purely prompted fashion—no fine-tuning is applied. At each timestep, it receives a structured prompt encoding the current system state and recent trends, and returns a suggested action, which is then blended with the RL policy output using the dynamic mixing factor $\alpha(\mathbf{s}_t)$. This design enables real-time augmentation of the RL policy while preserving its safety properties and learned structure.

In contrast, the LLM-only controller is trained via supervised fine-tuning (SFT) using imitation learning on expert trajectories generated from the same pretrained RL policy used in our RL-only baseline. These trajectories consist of observation-action pairs and, optionally, natural language rationales. This shared expert ensures consistency across baselines and provides high-quality supervision for SFT.

To ensure fair evaluation, we include both base and fine-tuned versions of the LLM in our experiments. Interestingly, the base (unfine-tuned) LLM already performs competitively on several metrics, indicating that performance gains are not solely due to supervised adaptation. Note also that the hybrid controller uses the same base LLM at inference time, reflecting realistic deployment conditions where lightweight integration of pretrained models is often preferred.

C. LLM Intrepretability Examples

C.1. Llama3.1 Detailed Explanation

C.1.1. Detailed Explanation (Llama-3.1-8B-Instruct)

Detailed Explanation

EXECUTIVE SUMMARY:
The current system is focused on balancing temperature stability and energy efficiency. The cabinet valves are distributed to match the power distribution across blade groups, while the cooling tower is set to very low cooling to avoid energy wastage. However, there is room for improvement in aligning temperature setpoints with goals and considering trade-offs between stability and efficiency.
TEMPERATURE STABILITY:
• **Overall Assessment:** The overall temperature stability is rated as moderate (score: 0.6341). While the system maintains reasonable temperature levels, there are deviations from desired goals, especially in cabinets 1 and 5.
• **Cabinet Temperature Control:** The valve positions are distributed to match the power distribution across blade groups, which helps maintain consistent temperature levels across all cabinets. However, this comes at the cost of increased energy usage compared to other configurations.
• **Cooling Tower Impact:** The cooling tower action is set to very low cooling to avoid excessive energy consumption. While this reduces the overall impact on temperature stability, it also limits the system's ability to compensate for large deviations from desired goals.
ACTION RATIONALE:
• **Temperature Setpoints:** The temperature setpoints do not fully align with the desired goal temperatures for each cabinet. This results in deviations from target values, especially in cabinets 1 and 5.
• **Valve Positions:** The valve positions are distributed to match the power distribution across blade groups, which helps maintain consistent temperature levels across all cabinets. However, this results in higher energy usage compared to other configurations.
• **Cooling Tower:** The cooling tower action is set to very low cooling to avoid excessive energy consumption. While this reduces the overall impact on temperature stability, it also limits the system's ability to compensate for large deviations from desired goals.
ENERGY EFFICIENCY:
• **Assessment:** The current actions have a moderate energy efficiency score of 0.6166. While the valve positions are distributed to match the power distribution across blade groups, this results in higher energy usage compared to other configurations. Additionally, the cooling tower action is set to very low cooling which reduces energy wastage but also limits the system's ability to compensate for large deviations from desired goals.
• **Trade Offs:** There is a trade-off between temperature stability and energy efficiency in the current actions. The valve positions are distributed to match the power distribution across blade groups, which helps maintain consistent temperature levels across all cabinets but comes at the cost of increased energy usage. The cooling tower action is set to very low cooling to avoid excessive energy consumption, which reduces the overall impact on temperature stability but also limits the system's ability to compensate for large deviations from desired goals.
RECOMMENDATIONS:
One potential improvement could be to adjust the temperature setpoints to better align with desired goals, especially in cabinets 1 and 5. This would help reduce deviations from target values and improve overall temperature stability. Another option could be to consider alternative valve distribution configurations that balance temperature stability and energy efficiency. By optimizing both factors simultaneously, the system can achieve better results overall.
Inference Time: 4.77 seconds

C.1.2. Very Detailed Explanation (Llama 3.1-8B-Instruct)

This explanation is significantly longer (approximately 1,500 words) and includes detailed scientific analysis across multiple categories. For complete response, refer to Appendix. A representative excerpt:

Very Detailed Explanation

EXECUTIVE SUMMARY:
The cooling system is currently maintaining a stable operating temperature across all five CDUs. This is achieved through a combination of active cooling provided by the cooling towers and passive thermal regulation via the cabinet valves. The cooling towers are operated at a very low cooling level, resulting in a slight increase in temperature compared to ideal operating conditions. The cabinet valves are adjusted to distribute cooling fluid equitably among the CDUs, balancing temperature differences while minimizing energy consumption. Overall, the system is operating within acceptable parameters and maintaining a stable thermal environment for the computational hardware.
SYSTEM STATE ANALYSIS:
• **Temperature Overview:**
The temperature distribution across the five CDUs is relatively uniform, with each cabinet maintaining a temperature within a narrow range. The highest temperatures are observed in CDU-5, which has experienced a higher workload and is currently operating at a temperature of 26.38 Celsius. The lowest temperatures...
• **Workload Distribution:**
The workload distribution across the five CDUs is relatively even, with each cabinet processing an approximately equal share of the total computational workload. The CDUs are configured to handle the workload in a balanced manner, ensuring that each...
• **Historical Context:**
The current system state is characterized by stable temperatures and power consumption levels, indicating that the system has been operating in a consistent and reliable manner over the past several months. The cooling system has maintained...
• **System Goals:**
The current system goals for the cooling system are to maintain a stable operating temperature across all five CDUs, while minimizing energy consumption and maximizing computational performance. The system is currently operating within...
CABINET CONTROL ANALYSIS:
• **Temperature Setpoints:**

- **Description:** The temperature setpoints for the five CDUs are configured to maintain a uniform operating temperature across all cabinets. The setpoints are optimized to balance performance and energy efficiency, taking into account...
- **Justification:** The temperature setpoints for the five CDUs are justified by the specific thermal requirements of the computational hardware. The setpoints are configured to maintain a temperature range that is optimal for...
- **Impact Assessment:** The impact of the current temperature setpoints on system stability and performance is positive. By maintaining a uniform temperature distribution across all five CDUs, the system is able to operate...

• **Valve Distributions:**

- **Description:** The valve distributions for the five CDUs are adjusted to balance cooling fluid flow and minimize energy consumption. The valve positions are optimized to distribute cooling fluid equitably among the cabinets...
- **Power Alignment:** The valve distributions for the five CDUs are aligned with the power consumption distribution across the cabinets. The valve positions are adjusted to ensure that each cabinet receives an equitable...
- **Hydraulic Considerations:** The hydraulic considerations for the valve distributions include ensuring that the cooling fluid is distributed equitably among the cabinets, while also minimizing energy consumption...

COOLING TOWER ANALYSIS:
• **Action Details:** The current cooling tower action is 'very low cooling', indicating that the cooling towers are currently operating at a low cooling level to ensure that the system remains stable and efficient. This action is justified...
• **Thermal Dynamics:** The thermal dynamics of the cooling towers are characterized by a low cooling level, resulting in a slight increase in temperature compared to ideal operating conditions. This is justified by the specific thermal...
• **Weather Interactions:** The current weather conditions are not directly influencing the cooling tower action. The cooling towers are adjusted to maintain a uniform temperature distribution across all five CDUs, ensuring that...
ENERGY EFFICIENCY ANALYSIS:
• **Component Efficiency:**

- **Cooling Tower:** The cooling tower efficiency is relatively low, resulting in a slight increase in temperature compared to ideal operating conditions. This is justified by the specific thermal requirements...
- **Pumps And Valves:** The pump and valve efficiency is relatively high, resulting in minimal energy consumption. The valve positions are adjusted to ensure that each cabinet receives an equitable share...
- **Secondary Cooling:** The secondary cooling loop efficiency is relatively high, resulting in minimal energy consumption. The valve positions are adjusted to ensure that each cabinet receives an equitable...

• **Optimization Strategy:** The current optimization strategy involves balancing energy efficiency and system stability. The temperature setpoints are configured to maintain a uniform operating temperature across all cabinets...
• **Performance Metrics:** The performance metrics for the cooling system include temperature stability, energy efficiency, and workload distribution. The temperature stability is relatively high, with minimal fluctuations...
THERMODYNAMIC IMPLICATIONS:
• **Heat Transfer:** The heat transfer dynamics within the cooling system are characterized by the distribution of cooling fluid through the cabinet valves, ensuring that each cabinet receives an equitable share of the cooling fluid...
• **Thermal Gradients:** The thermal gradients across the five CDUs are minimal, with each cabinet maintaining a uniform temperature distribution. The valve positions are adjusted to minimize thermal gradients, ensuring that...
• **Fluid Dynamics:** The fluid dynamics within the cooling system are characterized by the distribution of cooling fluid through the cabinet valves, ensuring that each cabinet receives an equitable share of the cooling fluid...
RECOMMENDATIONS:
• **Short Term:** The short-term recommendations for the cooling system include monitoring temperature and workload distributions, adjusting valve positions as needed, and optimizing the Cooing tower action...
• **Long Term:** The long-term recommendations for the cooling system include ongoing monitoring of temperature and workload distributions, regular adjustments to valve positions as needed, and continued optimization...
• **Trade Off Analysis:** The trade-off analysis for the cooling system involves balancing system stability and energy efficiency. The current configuration and operating conditions of the cooling system are optimized to maintain...
SCIENTIFIC INSIGHTS:
The scientific insights from the current system behavior include the optimal temperature setpoints for the five CDUs, the appropriate valve positions to ensure equitable cooling fluid distribution, and the ideal cooling tower action to maintain stability and efficiency. These insights are derived from a thorough analysis of the system's performance metrics, including temperature stability, energy efficiency, and workload distribution.
Inference Time: 15.25 seconds

D. RL Controller Details

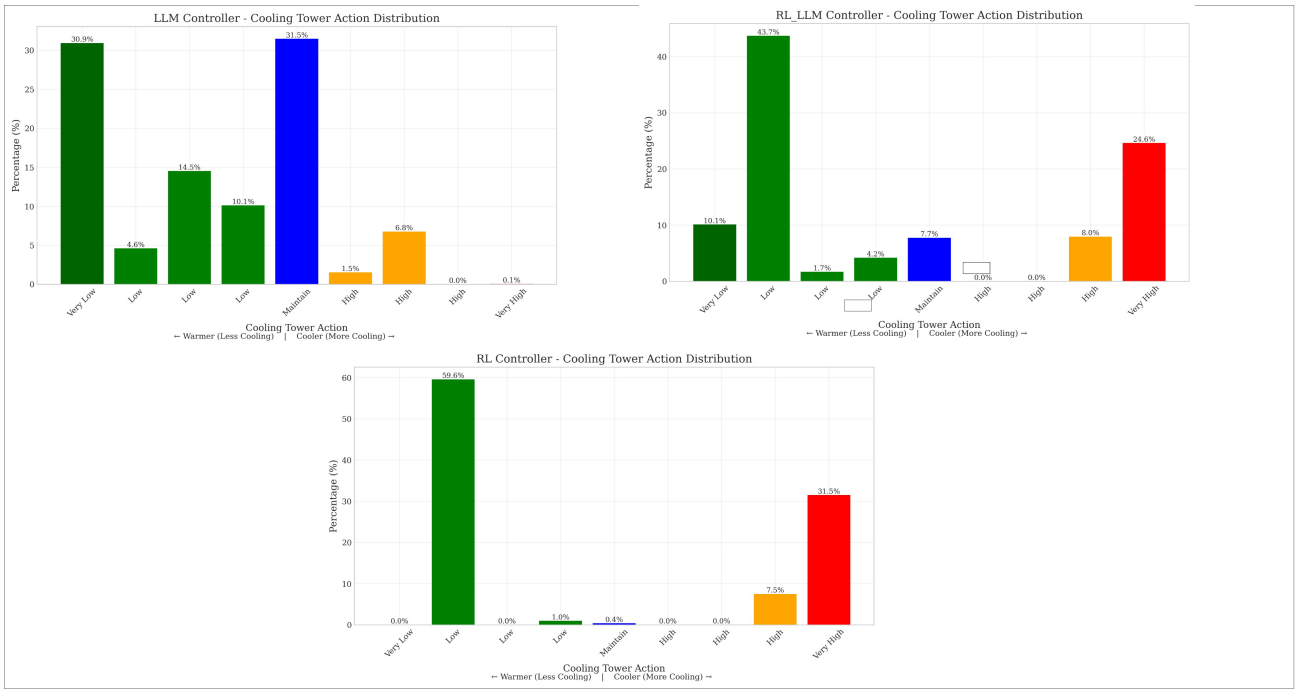

Figure 8: Action choice distribution across different controllers.

D.1. Addtional Details for implementation

Environment Formulation Observations are collected per cabinet ($\mathbb{R}^6$), including boundary temperatures and blade power levels, and from the cooling tower ($\mathbb{R}^4$), including fan powers, water supply temperature, and wet bulb temperature. Cabinet actions are 5-dimensional: secondary supply setpoint, pressure differential, and three valve positions (normalized via softmax). Cooling tower actions are discrete with 9 levels representing temperature offsets.

Architecture The controller uses a two-tier hierarchical structure. The meta-controller, updating every 5 steps, observes the global state and issues cabinet goals and cooling tower actions. Each cabinet controller operates at every timestep using local observations and a scalar temperature goal. Policies are implemented as feedforward neural networks: two-layer MLPs for cabinets and a shared multi-head network for the meta-controller.

Training Methodology We use PPO with the following hyperparameters: learning rates of 3×10^{-4} (cabinets) and 1×10^{-4} (meta), entropy coefficients 0.05 and 0.02 respectively, and discount factors $\gamma = 0.9$ (cabinet), $\gamma = 0.95$ (meta). Training is distributed across 8 workers with 2 environments per worker, using a batch size of 8,000 and minibatch size of 1,024, over 2 million timesteps.

Reward Function Design For cabinet controllers:

$$r_{alignment} = 6.0 - \sum_{i=1}^{3} |a_{valve_i} - o_{power_i}| \tag{1}$$

$$r_{efficiency} = 1.0 - \frac{\sum_{i=1}^{3} a_{valve_i}}{3} \tag{2}$$

$$r_{temp} = -1.5 \cdot \frac{\sum_{i=1}^{3} |T_i - T_{goal}|}{20.0} \tag{3}$$

$$r_{cabinet} = 0.7 \cdot r_{alignment} + 0.3 \cdot r_{efficiency} + r_{temp} \tag{4}$$

For the cooling tower:

$$r_{efficiency} = 1.0 - \frac{P_{fan_1} + P_{fan_2} + 1}{2} \tag{5}$$

$$r_{temp_dev} = -0.4 \cdot \frac{1}{5} \sum_{j=1}^{5} \frac{\sum_{i=1}^{3} |T_{j,i} - T_{j,goal}|}{20.0} \tag{6}$$

$$r_{cooling} = 0.6 \cdot r_{efficiency} + r_{temp_dev} \tag{7}$$

The meta-controller reward combines cabinet and cooling components:

$$r_{meta} = \frac{1}{5} \sum_{j=1}^{5} r_{cabinet_j} + r_{cooling} \tag{8}$$

Challenges and Limitations The RL controller faces several challenges: - *Long thermal delays* complicate credit assignment, limiting PPO's effectiveness even with $\gamma = 0.9/0.95$. - *Coupled dynamics* and *nonlinear responses* produce unstable gradients. - *Discrete cooling tower actions* create reward discontinuities. - *Information bottlenecks*: Cabinet agents receive only scalar goals and cannot coordinate laterally. - *Limited training budget (2M steps)* and conservative exploration reduce learning stability. - *Physics-naïve modeling*: The controller must learn principles LLMs inherit via pretraining, including thermal trends and constraint adherence.

These factors, taken together, contribute to the performance gap observed between the RL controller and LLM-based alternatives in our experiments.

D.2. RL Controller Training Details

The following table summarizes the training configuration used for the hierarchical reinforcement learning controller:

E. Hybrid Controller Details

Special Case Handling. The controller adapts mixing behavior in special regimes. For example, it increases LLM influence under high energy usage with stable temperature, and reduces it when temperatures deviate significantly, relying more on RL's recovery capabilities.

Table 1: RL Controller Training Parameters

Parameter	Value
Learning rate	3×10^{-4} (cabinet), 1×10^{-4} (meta)
Discount factor (γ)	0.9 (cabinet), 0.95 (meta)
GAE parameter (λ)	0.95
Entropy coefficient	0.05 (cabinet), 0.02 (meta)
Batch size	8000
Minibatch size	1024
Training epochs	10
Clip parameter	0.2
Workers	8
Environments per worker	2

Validation and Safety. The hybrid controller applies rule-based validation on LLM-proposed actions. If temperature is below target, cooling tower reductions are blocked; if above target, heating actions are rejected. Cabinet setpoints are also checked for consistency with temperature direction.

Error Handling. In cases where LLM queries fail or return invalid responses, the controller performs structured retries, falls back to last successful actions, or defaults to context-aware safe control templates.

Implementation Notes. Additional safeguards include temperature trend tracking, historical context buffers, and softmax enforcement on valve vectors to ensure valid actuation.

Theoretical Rationale. The hybrid approach overcomes key limitations of both base methods. RL alone lacks explainability and flexibility post-training, while LLMs may hallucinate or drift. Together, they form a cooperative agent with improved adaptability, safety, and control stability.

F. Detailed Evaluation Metric Definitions

This appendix provides the full definitions for all evaluation metrics used in our experiments.

Each controller is evaluated over 10 episodes, each consisting of 1,000 timesteps. To ensure experimental consistency, we use fixed random seeds and identical environment traces across all runs. The experiments are executed on a compute node with 40 CPU cores and 2 GPUs. Metrics are recorded at every timestep and include aspects of thermal behavior, energy consumption, and control smoothness. The reference temperature is fixed at 303.15 K.

We assess performance using a combination of primary and secondary metrics. The primary metrics include *Temperature Stability*, *Energy Efficiency*, and a *Weighted Reward* that emphasizes thermal safety. Secondary thermal metrics capture the average and maximum temperature deviation, as well as the percentage of time spent within the target range (±2 K). Energy-related metrics include cooling power usage and Power Usage Effectiveness (PUE). Control smoothness is quantified using the average control delta and recovery time following thermal disturbances. For LLM-based controllers, we also log average inference latency and the proportion of final actions influenced by the LLM.

F.1. Primary Performance Metrics

Temperature Stability. Measures how closely cabinet temperatures remain near the 30°C target:

$$\text{Temp_Stability} = \max(0, 1 - \frac{\text{avg_deviation_K}}{10.0}) \tag{9}$$

Energy Efficiency. Measures how efficiently the system minimizes cooling power:

$$\text{Energy_Efficiency} = \max(0, 1 - \frac{\text{cooling_power_W}}{\text{nominal_power_W}}) \tag{10}$$

Weighted Reward. A composite metric balancing stability and efficiency:

$$\text{Weighted_Reward} = 0.7 \times \text{Temp_Stability} + 0.3 \times \text{Energy_Efficiency} \tag{11}$$

F.2. Detailed Temperature Metrics

- **Average Temperature**: Mean cabinet temperature (°C)
- **Average Temperature Deviation**: Mean absolute deviation from the target (K)
- **Maximum Temperature Deviation**: Max deviation from the target (K)
- **Time in Target Range**: Percentage of steps where temperature remains within ±2 K of the target

F.3. Energy Performance Metrics

- **Cooling Power**: Average cooling tower energy usage (W)
- **PUE**: Ratio of total facility power to IT power
- **Energy Cost**: Operational cost (USD/hour)
- **Carbon Emissions**: CO_2 emissions per hour (kg)

F.4. Control Stability Metrics

- **Control Changes**: Average magnitude of change in control values across steps
- **Recovery Time**: Steps required to return to target range after deviation
- **Action Distribution**: Histogram of cooling tower action selections

F.5. LLM-Specific Metrics

- **Inference Time**: Average LLM inference duration (sec)
- **LLM Influence**: Proportion of steps where the LLM modified base actions significantly

G. LLM Controller Implementation Details

G.1. Post-Processing and Validation.

Model outputs are validated for syntactic correctness and physical consistency. Parsing is enforced using a predefined JSON schema, and safety checks correct contradictory decisions, such as applying excessive cooling when temperatures are already below target. Fallbacks reuse the most recent valid action when parsing or generation fails. The LLM-based controllers were trained using parameter-efficient fine-tuning (LoRA) on two open-source base models: Qwen 3 (8B) and LLaMA 3.1 (8B). Training was conducted via imitation learning using expert trajectories collected from the RL controller. These trajectories consisted of serialized state observations and corresponding expert actions, formatted as input-output pairs for language modeling.

Model training was implemented using the HuggingFace `transformers` and `peft` libraries. All training was performed in 16-bit precision using mixed-precision optimization and the paged AdamW optimizer. Prompts were structured to include system state variables, recent trends, and physical constraints. The expected output format was a natural language reasoning trace followed by a structured JSON object encoding cabinet and tower control actions.

Training Configuration

- Learning rate: 3×10^{-4}
- Batch size: 4
- Gradient accumulation: 8 steps
- Number of epochs: 50
- Precision: bfloat16
- Optimizer: Paged AdamW

Qwen 3 (8B) LoRA Settings

- LoRA rank (r): 8
- LoRA alpha (α): 32
- Dropout: 0.05
- Target modules: `q_proj`, `k_proj`, `v_proj`, `o_proj`

LLaMA 3.1 (8B) LoRA Settings

- LoRA rank (r): 8
- LoRA alpha (α): 16
- Dropout: 0.1
- Target modules: `q_proj`, `k_proj`, `v_proj`, `o_proj`

H. LLM Inference Optimization

H.1. Experimental Setup

We implemented a high-performance inference system for real-time cooling control using large language models (LLMs). Our experimental framework deployed four model variants: base Llama-3.1-8B, fine-tuned Llama-3.1-8B, base Qwen-7B, and fine-tuned Qwen-7B. All experiments were conducted on a server equipped with four NVIDIA H100 GPUs, yielding a total runtime of approximately four hours for the complete benchmark suite.

H.2. Inference Infrastructure

The inference system was deployed using the vLLM framework (Kwon et al., 2023) with the following configuration:

```
docker run --runtime nvidia --gpus all \
  -v /home/****/ft-llama3dot1-8b:/model \
  -p 0.0.0.0:8000:8000 \
  --ipc=host \
  vllm/vllm-openai:latest \
  -model /model \
  --tensor-parallel-size 4 \
  --max-num-batched-tokens 32768 \
  --max-num-seqs 50 \
  --gpu-memory-utilization 0.9 \
```

This configuration leverages several critical optimizations for efficient real-time LLM inference:

- **Tensor Parallelism:** Model weights are distributed across four H100 GPUs (`tensor-parallel-size 4`), enabling larger models to fit in GPU memory while reducing per-token generation latency.
- **Efficient Token Batching:** The system processes up to 32,768 tokens in a single batch (`max-num-batched-tokens 32768`), optimizing GPU utilization during concurrent requests.
- **Parallel Request Handling:** Up to 50 concurrent sequence requests are supported (`max-num-seqs 50`), essential for our multi-agent evaluation framework.
- **Memory Optimization:** GPU memory utilization is set to 90% (`gpu-memory-utilization 0.9`), balancing resource maximization and prevention of out-of-memory errors.

H.3. Parallelization Strategy

Our implementation employs a multi-level parallelization strategy to maximize throughput:

1. **Parallel Episode Execution:** Using Ray (Moritz et al., 2018), we execute multiple simulation episodes simultaneously, each with its own controller instance.
2. **Tensor Parallelism:** Model weights are sharded across four GPUs, enabling faster matrix multiplications and reducing per-token latency.
3. **Continuous Batching:** vLLM's PagedAttention (Kwon et al., 2023) dynamically allocates key-value (KV) cache blocks, enabling efficient generation for varying sequence lengths.
4. **Prefill Optimization:** KV cache management reduces redundant computation by caching key-value pairs from previous forward passes.
5. **Parallel Requests:** Multiple controller instances query the model concurrently, maximizing GPU utilization through interleaved request processing.

H.4. Inference Performance Analysis

We evaluated inference performance across the fine-tuned model variants over 10 episodes, each consisting of 1,000 timesteps, totaling 10,000 inference calls per model. Table 2 summarizes the inference time statistics. Key observations

Table 2: Inference Performance Comparison

Model	Mean (s)	Min (s)	Max (s)	Total (s)
Fine-tuned Llama-3.1-8B	6.89	6.70	27.68	57,131.87
Fine-tuned Qwen-7B	9.21	8.74	38.46	92,088.90

from the analysis include:

- The fine-tuned Llama-3.1-8B model exhibited significantly faster inference times compared to fine-tuned Qwen-7B (6.89 s vs. 9.21 s), suggesting architectural differences impacting computational efficiency.
- Maximum inference times were substantially higher than means for both models, likely due to cache misses, garbage collection, or resource contention during parallel execution.
- Fine-tuned Llama-3.1-8B achieved a lower cumulative inference time of approximately 57,131.87 seconds compared to 92,088.90 seconds for fine-tuned Qwen-7B, reflecting better efficiency in per-request latency and parallel processing.

I. Experimental Results

I.1. Performance Comparison Across Control Approaches

Our experimental evaluation yields several significant findings about the efficacy of different control approaches for data center cooling optimization. Table 3 summarizes the quantitative results across different model configurations. All differences between controllers are statistically significant ($p < 0.001$) based on ANOVA with post-hoc t-tests.

Table 3: Performance comparison of control approaches across different model configurations

Controller	Temperature Stability	Energy Efficiency	Weighted Reward
Fine-tuned Llama			
RL	0.633 ± 0.000	0.608 ± 0.000	0.625 ± 0.000
RL+LLM	0.652 ± 0.001	0.603 ± 0.003	0.637 ± 0.001
LLM	$\mathbf{0.747 \pm 0.005}$	$\mathbf{0.682 \pm 0.004}$	$\mathbf{0.728 \pm 0.003}$
Base Llama			
RL	0.633 ± 0.000	0.608 ± 0.000	0.625 ± 0.000
RL+LLM	0.651 ± 0.001	0.603 ± 0.003	0.637 ± 0.001
LLM	$\mathbf{0.718 \pm 0.001}$	$\mathbf{0.674 \pm 0.001}$	$\mathbf{0.705 \pm 0.001}$
Fine-tuned Qwen			
RL	0.633 ± 0.000	0.608 ± 0.000	0.625 ± 0.000
RL+LLM	0.642 ± 0.000	0.602 ± 0.002	0.630 ± 0.001
LLM	$\mathbf{0.721 \pm 0.001}$	$\mathbf{0.678 \pm 0.001}$	$\mathbf{0.708 \pm 0.001}$
Base Qwen			
RL	0.633 ± 0.000	0.608 ± 0.000	0.625 ± 0.000
RL+LLM	0.662 ± 0.001	0.603 ± 0.003	0.644 ± 0.001
LLM	$\mathbf{0.723 \pm 0.007}$	$\mathbf{0.692 \pm 0.005}$	$\mathbf{0.714 \pm 0.005}$

I.1.1. Overall Performance Trends

The most striking result is the consistent superior performance of the pure LLM controller across all metrics and model configurations. The LLM approach outperforms both the RL-only and hybrid RL+LLM controllers by substantial margins:

- **Temperature stability**: LLM controllers achieve 10-18% higher stability scores compared to RL controllers, indicating more precise temperature management.
- **Energy efficiency**: LLM controllers demonstrate 12-14% better energy efficiency than RL controllers, suggesting more optimal cooling resource allocation.
- **Weighted reward**: LLM controllers exhibit 13-16% higher composite performance, confirming their superior balance of temperature control and energy conservation.

The fine-tuned Llama-based LLM controller achieved the highest overall performance, with a weighted reward score of 0.728 ± 0.003, representing a 16.5% improvement over the pure RL approach and a 14.3% improvement over the hybrid RL+LLM approach with the same model.

I.1.2. Hybrid Controller Performance

Contrary to our expectations, the hybrid RL+LLM approach showed only modest improvements over the pure RL controller:

- Across all model configurations, the hybrid approach improved temperature stability by 1.4-4.6% over pure RL.
- The hybrid approach consistently showed slightly reduced energy efficiency compared to pure RL (approximately 1% worse), suggesting suboptimal integration of the two control paradigms.
- The weighted reward improved by only 0.8-3.0% compared to pure RL, indicating that the sophisticated mixing strategy failed to effectively leverage the superior capabilities of the LLM component.

This underperformance of the hybrid approach is particularly notable given our implementation of a context-aware mixing strategy that dynamically adjusted the influence of each controller based on system state.

I.1.3. Model Comparison Across Architectures

Our results reveal interesting patterns across different LLM architectures and training regimes:

- **Fine-tuned vs. Base Models**: Fine-tuned models generally outperformed their base counterparts, with the fine-tuned Llama model achieving the highest overall performance. The performance improvements from fine-tuning ranged from 2-4% across metrics.
- **Llama vs. Qwen**: While both model families performed well, Llama-based controllers showed slight advantages in temperature stability (1-3% better), while Qwen-based controllers demonstrated marginally better energy efficiency (1-2% advantage).

I.1.4. Analysis of LLM Success Factors

The surprisingly strong performance of LLM controllers can be attributed to several factors:

- **Domain Knowledge Integration**: LLMs implicitly contain physical and engineering principles relevant to thermal management, effectively providing a rich prior for control decisions that RL must learn from scratch.
- **Contextual Reasoning**: LLMs excel at integrating multiple information streams (temperature readings, power distribution, historical trends) into coherent reasoning about system state.
- **Multi-objective Balancing**: Our prompt structure explicitly encourages consideration of both temperature stability and energy efficiency, enabling more balanced control decisions.
- **Action Consistency**: The logical validation mechanisms in our LLM controller ensure physically consistent actions (e.g., not cooling when heating is needed), preventing counterproductive control choices.

I.1.5. Understanding RL Limitations

The relative underperformance of RL can be attributed to several domain-specific challenges:

- **Delayed System Dynamics**: Thermal systems exhibit significant delays between actions and observable effects, creating a difficult credit assignment problem for RL.
- **Coupled Physical Subsystems**: The interdependencies between cooling towers, heat exchangers, and cabinets create complex state transitions that are challenging for RL to model efficiently.
- **Hierarchical Control Structure Limitations**: Our two-level hierarchy with limited information flow between levels constrains effective coordination among cabinet controllers.
- **Reward Function Design Challenges**: Balancing multiple competing objectives (temperature stability, energy efficiency, control alignment) creates a complex reward landscape that complicates policy learning.

In summary, our experimental results demonstrate that LLM-based controllers can significantly outperform both traditional RL approaches and hybrid RL+LLM strategies for data center cooling optimization. The LLM's ability to integrate domain knowledge, reason contextually, and maintain action consistency contributes to its superior performance, suggesting significant potential for LLM-based control in complex physical systems.

J. Execution details

J.1. Implementation and Execution Details for hybrid controller

The hybrid controller is further supported by several implementation mechanisms designed to ensure robustness, safety, and responsiveness across a wide range of operating conditions. These mechanisms include dynamic action mixing strategies, component-specific blending rules, context-aware behavior switching, and fallbacks for interpretability and control reliability.

The action mixing coefficient $\alpha(\mathbf{s}_t)$, which determines the relative weighting between the RL-generated action and the LLM-suggested alternative, is computed as a convex combination of two factors: thermal stability and energy usage. Specifically, $\alpha(\mathbf{s}_t)$ is defined as $w_{\text{temp}} \cdot \alpha_{\text{temp}}(\mathbf{s}_t) + w_{\text{energy}} \cdot \alpha_{\text{energy}}(\mathbf{s}_t)$, where the individual terms measure proximity to the target temperature and the normalized energy consumption, respectively. The weights w_{temp} and w_{energy} are state-dependent and shift according to the severity of the temperature deviation ΔT. When $|\Delta T|$ is large, thermal regulation is prioritized with a heavier weight on the temperature component. Conversely, when temperatures are within a narrow margin of the target, energy efficiency considerations are emphasized.

The action mixing strategy is further refined at the component level. Control parameters such as the coolant temperature setpoint use the full state-dependent mixing coefficient, while others are mixed with fixed weights. For example, pressure-related control actions use a constant $\alpha = 0.3$, giving more authority to the RL policy, which has demonstrated stronger performance in this area. Valve coefficients, on the other hand, use a mixing coefficient that is modulated by energy usage, increasing LLM influence under stable operating conditions to promote workload-aligned flow distribution. Cooling tower actions are also influenced more heavily by the LLM during energy-saving opportunities, particularly when ambient conditions permit reduced fan usage.

To enhance control robustness, the hybrid system incorporates multiple context-aware behaviors. When energy usage is high but temperatures remain stable, the system shifts toward LLM-guided actions to reduce cooling energy. In contrast, during periods of significant thermal deviation, the controller relies more heavily on the RL agent to perform corrective actions. The system also prioritizes LLM proposals when they indicate opportunities for safe energy savings without compromising thermal limits.

All control outputs are validated prior to actuation. This validation includes checks for consistency between action direction and system state—for instance, ensuring that cooling tower actions do not intensify cooling when the cabinet temperature is already below target. Valve positions are normalized using softmax to preserve flow conservation, and temperature setpoints are constrained to adhere to safe operational thresholds and expected thermal trends.

To handle generation errors from the LLM module, the system includes a robust fallback protocol. If an LLM query fails to return a valid action, the controller retries up to three times, optionally modifying the prompt context to encourage more grounded outputs. If all retries fail, the most recent successful action is reused. If no history is available, a rule-based default policy is used that considers current temperature trends to determine a plausible action.

Finally, the hybrid controller incorporates several runtime enhancements to improve generation quality and stability. A temperature annealing schedule is applied to the LLM sampling process, progressively lowering the decoding temperature over time to encourage more deterministic outputs. The controller maintains rolling histories of recent states, actions, and performance metrics, which are used both for prompt construction and for estimating temperature trends that guide LLM intent selection. Together, these implementation features enable the hybrid architecture to operate reliably and adaptively under realistic, high-dimensional control conditions.

J.2. Cooling Tower Control Patterns

Analysis of cooling tower action distributions revealed distinct control patterns:

- Base Llama-3.1 predominantly utilized moderate warming actions (CT action 2: 4,425 occurrences), with strong warming as the second preference (CT action 0: 2,998 occurrences).
- Fine-tuned Llama-3.1 showed balanced control, distributing actions between strong warming (CT action 0: 3,092 occurrences) and maintaining current conditions (CT action 4: 3,148 occurrences).
- Base Qwen heavily favored maintaining current conditions (CT action 4: 6,984 occurrences).
- Fine-tuned Qwen exhibited a high preference for moderate warming (CT action 2: 8,748 occurrences).

These patterns suggest that model architecture and fine-tuning significantly influence decision-making strategies, potentially explaining variations in control performance.

J.3. Implications for Real-time Control

The inference time results demonstrate the viability of LLMs for real-time data center cooling control. These models are suitable for minute-by-minute control decisions. For sub-second response times, further optimizations are necessary, such as:

- Model distillation to create smaller, faster models approximating larger model behavior.
- Specialized hardware accelerators designed for inference.
- Hybrid approaches combining fast-response controllers (e.g., PID) with LLM supervisors.