Learning to Compress: Energy-Aware Large Language Model Serving via Adaptive KV Caching

Yuanbin Cheng, Zhentong Shao, Nanpeng Yu
Department of Electrical and Computer Engineering
University of California, Riverside
Riverside, California 92507 USA
ychen871@ucr.edu, zhentons@ucr.edu, nyu@ece.ucr.edu

Yu Fu, Yue Dong
Department of Computer Science and Engineering
University of California, Riverside
Riverside, California 92507 USA
yfu093@ucr.edu, yued@ucr.edu

***Abstract*—The rapid expansion of Large Language Model (LLM) inference has led to growing power and carbon footprints in data centers, largely driven by the compute and memory intensity of attention operations over long sequences. To address this challenge, we propose a sustainable LLM serving framework that optimizes energy efficiency and inference performance through intelligent KV cache compression. Unlike external scheduling or batching strategies, cache compression acts directly on the model's internal key–value memory and enables fine-grained control over compute and memory usage with minimal latency overhead. We develop a reinforcement learning-based control strategy that dynamically selects optimal compression ratios in response to real-time energy prices and service quality constraints. Evaluations using real-world electricity market data and LLM inference traces show that the proposed approach substantially reduces energy cost while preserving service quality. By embedding electricity market awareness into artificial intelligence (AI) infrastructure design, this work establishes a scalable and practical path toward energy-efficient and grid-friendly LLM deployment. All code and datasets are made publicly available at: https://github.com/cybsbbb/LLM_energy_optimization_rl.**

***Index Terms*—Large Language Models, KV Cache Compression, Energy Efficiency, Power Systems, Reinforcement Learning.**

I. Introduction

The rapid growth of AI, particularly the widespread use of Large Language Models (LLMs), has led to major advances in productivity but also a significant increase in energy demand. Modern LLM services such as ChatGPT rely on continuously operating GPU clusters that consume large amounts of electricity and cooling water [1]: a single ChatGPT query requires about 2.9 Wh, nearly ten times the energy of a conventional web search [2], illustrating the high per-task energy cost of AI inference. As these models are deployed at scale across cloud platforms, AI workloads already account for a rapidly growing share of global data center energy use [3]. If current growth trends continue, data centers supporting AI are projected to reach the gigawatt scale [4], [5], could consume between 4.6% and 9.1% of total U.S. electricity by 2030 [3], highlighting the urgent need for energy-aware algorithms and sustainable deployment strategies.

Most existing efficiency research on LLMs has focused on *static* algorithmic improvements such as pruning, quantization, and knowledge distillation [6]. While these methods reduce computation and memory cost, they remain fixed once deployed and cannot adapt to dynamic grid or workload conditions. In current practice, power management for AI workloads is handled mainly at the hardware level through techniques such as dynamic voltage and frequency scaling [7], which regulate GPU power in real time but operate at coarse granularity and lack visibility into model-level behavior. This hardware dominance leaves an important unexplored opportunity: algorithm-level control that can adjust a model's computational behavior directly and dynamically in response to energy conditions. Such control complements existing hardware-based mechanisms by providing finer-grained, model-aware regulation of energy use.

To address this gap, we propose on-the-fly, energy-aware inference control that dynamically adjusts a model's internal computation during deployment. Among the various components of transformer-based LLMs [8], the key–value (KV) cache accounts for a large portion of GPU computation, memory, and power consumption during inference, as it stores intermediate attention states whose matrix multiplications grow linearly with sequence length. However, not all cached tokens equally affect output quality, and recent studies on KV cache compression [9]–[11] show that even aggressive compression can preserve 95–98% of model quality. Yet these methods use fixed compression ratios and do not account for dynamic power grid or service conditions.

Building on this robustness, our framework introduces dynamic cache compression that adapts in real time to energy price, workload, and latency conditions. During periods of high demand or grid stress, the system can temporarily increase compression to reduce power use, while off-peak conditions allow full precision to be restored. We propose a reinforcement learning–based controller that continuously optimizes this trade-off between energy cost and inference performance, ensuring that any quality reduction is both minimal and economically justified. This adaptive control turns inference efficiency into a tunable variable, aligning LLM operation with energy prices and service constraints.

This work represents the first comprehensive framework for dynamic KV cache compression optimization that explicitly integrates energy price into large-scale LLM serving. The main contributions are summarized as follows:

- A reinforcement learning–based optimization framework that jointly balances service quality and energy efficiency.
- A comprehensive empirical evaluation of real-world electricity market data and large-scale LLM inference traces.
- An in-depth analysis of quality–energy trade-offs under dynamic and uncertain operating conditions.

II. Background

KV caching is a core optimization in transformer-based language models that stores intermediate attention states to avoid redundant computation during auto-regressive generation [8]. In each layer ℓ, the attention module maintains key and value tensors $K_\ell, V_\ell \in \mathbb{R}^{t \times d_h}$ for head dimension d_h, where t is the current sequence length. When producing token $(t+1)$, the query $Q_\ell^{(t+1)} \in \mathbb{R}^{1 \times d_h}$ attends to all previous tokens:

$$\text{Attention}(Q_\ell^{(t+1)}, K_\ell, V_\ell) = \text{softmax}\left(\frac{Q_\ell^{(t+1)} K_\ell^\top}{\sqrt{d_h}}\right) V_\ell.$$

Both memory and computation thus scale linearly with t, yielding total KV storage of $\mathcal{O}(LHd_ht)$ for L layers and H heads. For example, Mistral 7B [12] requires about 7.3 GB of KV memory for a 30,000-token context, which becomes prohibitive in large-scale serving with many concurrent users [13].

Recent KV cache compression methods [9]–[11] alleviate this by retaining only a fraction r of tokens with the highest attention importance, reducing memory and bandwidth cost. For instance, SnapKV [10] keeps about 25–50% of entries and reports up to 50% memory savings while maintaining 95–98% of baseline quality. In these works, the compression ratio r is typically chosen offline based on accuracy and latency trade-offs and then kept fixed during inference.

III. Problem Formulation

A. System Model

We model the LLM serving infrastructure as a multi-server queueing system, where N homogeneous GPU servers handle incoming inference requests under configurable KV cache compression strategies. The state of the system at time t, denoted by $S(t)$, captures both the workload and external environmental factors:

- $n_p(t)$: Number of requests currently being processed;
- $n_q(t)$: Number of pending requests in the queue;
- $\lambda(t)$: Instantaneous request arrival rate;
- $p_e(t)$: Real-time energy price (\$/MWh);
- $h(t)$: Normalized time index (e.g., hour of day or day of week) reflecting daily or weekly energy price patterns.

Each server processes requests sequentially, and the total power draw and latency of the system depend on the selected KV cache compression strategy.

B. Algorithmic Control via KV Cache Compression

We define the action space as $\mathcal{A} = \{a_0, a_1, \ldots, a_5\}$, where each action corresponds to a distinct KV cache compression configuration, following common practice in KV cache compression research [9]–[11]. Specifically, a_0 denotes FullKV (no compression), while a_1–a_5 represent SnapKV-1024, 512, 256, 128, and 64, corresponding to progressively stronger compression levels. Each strategy a_j is associated with empirically characterized attributes: T_j (processing time), E_j (energy consumption), and SQ_j (output quality). Lower compression ratios generally improve inference quality but increase both latency and power use, whereas higher ratios reduce resource demand at the cost of slight performance degradation. We adopt five discrete compression levels to balance control granularity with policy stability, allowing the controller to adaptively select $a_t \in \mathcal{A}$ at each decision step (every 10 seconds) to trade off energy cost and service quality under real-time conditions.

C. Optimization Objective and Reward Design

The goal is to maximize the expected long-term return while ensuring service quality commitments. The cumulative objective function is formulated as:

$$\mathcal{J}(t) = \mathbb{E}\left[\sum_{k=t}^{T} \gamma^{k-t} R(s_k, a_k)\right], \tag{1}$$

where $\gamma \in [0, 1]$ is the discount factor, s_k is the system state at time k, and $R(s, a)$ denotes the immediate reward.

The immediate reward integrates three components, service quality, energy consumption, and latency penalty:

$$R(s, a) = R_{\text{quality}} - R_{\text{energy}} - R_{\text{latency}}. \tag{2}$$

1) Service Quality Reward: The service quality reward quantifies the overall inference quality based on successfully completed requests. Let SQ_i denote the service quality score of the request i. The reward is defined as:

$$R_{\text{quality}} = \sum_{i \in \text{completed}} SQ_i, \tag{3}$$

which encourages the agent to sustain high response fidelity even under variable workloads and system conditions.

2) Energy Cost Penalty: The energy consumption reward penalizes the agent proportionally to the instantaneous energy price and the data center's power usage effectiveness (PUE), aligning decisions with real-time electricity market conditions. It is formulated as

$$R_{\text{energy}} = \text{PUE} \cdot p_e(t) \cdot \sum_{i \in \text{completed}} E_{a_i}, \tag{4}$$

where E_{a_i} denotes the energy consumed for action a_i. This formulation incentivizes energy-efficient scheduling, especially during high price or peak-demand periods.

3) Latency Penalty: To ensure compliance with service-level agreements (SLAs), latency violations are penalized proportionally to the normalized response delay of completed requests. The latency-related penalty is formulated as

$$R_{\text{latency}} = \sum_{i \in \text{completed}} (l_i / l_{\max}), \tag{5}$$

where l_i denotes the actual latency of request i, and $l_{\max}$ represents the maximum acceptable delay.

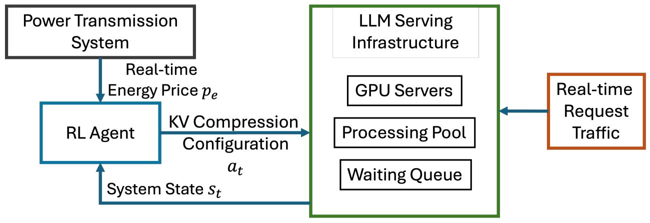
Power Transmission System
Real-time Energy Price p_e
RL Agent
KV Compression Configuration a_t
System State s_t
LLM Serving Infrastructure
GPU Servers
Processing Pool
Waiting Queue
Real-time Request Traffic

Fig. 1. Overview of the RL-based sustainable LLM serving framework.

All reward components are normalized to comparable scales and are treated equally in this work. Their relative weights can be adjusted to reflect specific operational priorities.

D. Operational Constraints

The optimization process is governed by a set of operational and service-level constraints that ensure responsiveness and consistent compliance with user experience requirements.

1) Server Capacity Constraint:

$$n_p(t) \le N, \quad \forall t. \tag{6}$$

2) Queue Length Constraint:

$$n_q(t) \le C_{\text{queue}}, \quad \forall t, \tag{7}$$

with C_{queue} denoting the maximum queue capacity.

3) Maximum Wait Time Constraint:

$$l_i \le l_{\max}, \quad \forall i, \tag{8}$$

where requests exceeding $l_{\max}$ are rejected to preserve overall service quality and guarantee timely responses.

These constraints ensure that the system remains stable and responsive under dynamic workload and price conditions.

IV. Technical Methods

A. System Architecture Overview

Figure 1 illustrates the overall architecture of the proposed framework. The reinforcement learning (RL) agent interacts with the LLM serving infrastructure and the power system in a closed feedback loop. At each decision step, the agent observes the current system state, including request queue length, GPU utilization, and real-time energy price, and determines the optimal KV cache compression configuration. The chosen action directly influences inference latency, energy consumption, and service quality. Through this feedback-driven control, the system continuously adapt to dynamic workload and grid conditions, achieving energy-efficient and reliable LLM serving.

B. Reinforcement Learning Framework

We formulate the KV cache compression control problem as a Markov Decision Process (MDP), defined by the tuple

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle, \tag{9}$$

where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, $\mathcal{P}$ the transition dynamics, $\mathcal{R}$ the reward function, and γ the discount factor. The RL agent learns an adaptive policy $\pi_\theta(a|s)$ that selects appropriate KV cache compression levels in response to real-time system and grid conditions to maximize the long-term expected reward.

1) State Space: The system state $s_t \in \mathcal{S}$ encodes both internal workload statistics and external environmental factors that influence energy cost and service quality:

$$s_t = \left[n_p(t), n_q(t), p_e(t), h(t), \bar{q}(t), \bar{l}(t)\right], \tag{10}$$

where $p_e(t)$ is the normalized energy price, $h(t)$ represents the normalized time index (capturing daily or weekly cycles), $\bar{q}(t)$ is the moving average of the recent inference quality scores and $\bar{l}(t)$ is the moving average of the recent latencies. This compact state representation enables the agent to adapt to both workload intensity and time-varying energy market dynamics.

2) Action Space: The action space corresponds to the set of available KV cache compression strategies introduced in Section III-B: At each decision step t, the RL agent selects an action $a_t \in \mathcal{A}$ that balances the trade-offs among quality, latency, and energy consumption.

3) State Transition Dynamics: The transition function $P(s_{t+1}|s_t, a_t)$ captures the stochastic evolution of the system state driven by:

- Workload dynamics: Request arrivals following empirical temporal patterns;
- Service completions: Processing times dependent on the selected compression level a_t;
- Electricity market variation: Fluctuations in real-time energy prices $p_e(t)$;
- Queue updates: Changes in $n_q(t)$ and $n_p(t)$ due to arrivals, completions, and timeout events.

These transitions are inherently non-stationary, making conventional optimization intractable and motivating the use of RL for adaptive decision-making.

4) Reward Function: The immediate reward $R(s_t, a_t)$ integrates the performance, energy, and latency objectives, as defined in (2). This reward encourages the agent to maximize model quality while minimizing energy cost and latency penalties, promoting energy-aware service policies that adapt to dynamic energy price and workload conditions.

C. Algorithm Selection and Training

To identify the most effective control policy for dynamic KV cache compression, we evaluate several state-of-the-art RL algorithms, each adapted to the discrete action space defined in Section III-B. These algorithms represent complementary design philosophies balancing sample efficiency and stability. All algorithms are trained using historical traces of real-world energy prices and LLM inference workloads from public datasets [14], [15]. Each training episode simulates one week of system operation, capturing stochastic request arrivals and dynamic energy price fluctuations.

1) Proximal Policy Optimization (PPO): PPO [16] is selected as the primary method due to its robustness, scalability, and effectiveness in non-stationary environments. By employing a clipped surrogate objective, PPO stabilizes training by limiting excessive policy updates, leading to more reliable convergence. The policy network uses a shared feature extractor for both actor and critic branches, enabling efficient representation learning. This architecture, coupled with PPO's inherent stability and adaptability, makes it particularly effective for real-time decision-making under dynamic energy prices and fluctuating workloads.

2) Deep Q-Network (DQN): DQN [17] is adopted as a value-based baseline for discrete KV cache compression control. It estimates action value via a Q-function approximator and uses experience replay with a target network to stabilize learning. Its simplicity and computational efficiency make it an effective reference model for discrete decision-making.

3) Advantage Actor-Critic (A2C): A2C [18] serves as a lightweight synchronous baseline that jointly learns a policy (actor) and a value function (critic) using shared network layers.

4) Twin Delayed Deep Deterministic Policy Gradient (TD3): TD3 [19] enhances the deterministic actor-critic framework by using dual critic networks to mitigate overestimation bias and introducing delayed policy updates for improved stability. While primarily designed for continuous control, TD3 serves as a valuable comparative baseline for future extensions toward fine-grained or continuous KV cache compression.

D. Baseline Comparison Methods

To evaluate the effectiveness of the proposed RL–based control framework, we compare it against several static baseline strategies that represent common non-adaptive approaches to KV cache management.

- Always FullKV: All requests are processed without compression. This configuration provides the upper bound for the quality of inference, but results in the highest energy consumption, GPU memory usage, and reject rate.
- Always SnapKV-64: Applies the most aggressive compression level uniformly across all requests, maximizing energy efficiency and minimizing memory footprint. However, this strategy often leads to noticeable degradation in model output quality and response stability.

These static strategies serve as reference points for evaluating the adaptability, energy efficiency, and quality of service achieved by the proposed RL-based approach.

V. Numerical Study

A. Data Collection and Preprocessing

a) Energy Price Data: Real-time electricity market prices are obtained from the California Independent System Operator (CAISO) Default Load Aggregation Point for Southern California Edison, sampled at 5-minute intervals from May 2024 to May 2025, capturing realistic temporal fluctuations in real-world market conditions.

b) LLM Inference Traces: Inference request patterns are obtained from public production workloads in the Microsoft Azure dataset [14], [15], sampled at 10-second resolution. These traces enable realistic simulation of LLM serving demand under dynamically varying arrival rates.

TABLE I
Performance Comparison of Baseline and RL Algorithms

Algorithm	**Succ. Rate**	**Avg. Score**	**Energy** (MWh)	**Energy Cost** ($)	**Cum. Reward**
FullKV	81.4	0.328	49.97	865.08	53027.2
SnapKV-64	97.2	0.187	28.38	439.73	46657.5
RL_PPO	97.0	0.250	40.10	602.21	**55199.9**
RL_A2C	96.3	0.256	41.36	634.23	55123.9
RL_DQN	96.2	0.254	40.96	624.98	55067.2
RL_TD3	92.8	0.260	39.87	581.32	54575.8

c) KV Cache Performance Characterization: We benchmark the Mistral-7B model on NVIDIA A6000 Ada GPUs using the GovReport dataset (200 samples, average input length: 8,734 words, maximum generation length: 512 tokens). SnapKV compression levels of {1024, 512, 256, 128, 64} are evaluated with an observation window size of 8. Model quality is measured using ROUGE scores, while per-sample energy consumption and processing latency are measured following the methodology in [20].

d) Simulation Setup: The simulated data center consists of 200 homogeneous GPU servers with a Power Usage Effectiveness (PUE) of 1.3. The maximum response latency is set to $l_{\max} = 10s$. Requests arrival follow the Azure inference traces, with each request sampled from the GovReport dataset. The RL agent adjusts the KV cache compression level every 10 seconds. For every incoming request, the system retrieves empirical processing time, energy consumption, and quality metrics corresponding to the compression level.

B. Evaluation Metrics

We evaluate all methods using five key metrics that jointly capture service quality, energy efficiency, and overall system performance:

- Success Rate: Ratio of successfully completed requests.
- Average Score: Mean output quality across all inferences.
- Total Energy (MWh): Aggregate energy consumption.
- Total Energy Cost ($): Total energy expenditure calculated using real-time energy prices.
- Episode Reward: Cumulative reward integrating quality, latency, and energy cost objectives.

Together, these metrics provide a holistic assessment of operational efficiency and service quality, enabling direct comparison between the proposed RL framework and baselines.

C. Results and Discussion

Table I summarizes the performance of all baseline and RL-based algorithms across key evaluation metrics. Overall, the RL methods achieve superior balance between service quality and energy efficiency compared to static strategies.

1) Static Baselines: The *FullKV* configuration attains the highest average score (0.328) but at the cost of the largest energy consumption (49.97 MWh) and energy cost ($865). In contrast, *SnapKV-64* minimizes energy usage (28.38 MWh) and cost ($440) but significantly reduces output quality (0.187). These extremes highlight the limitations of fixed compression policies, which cannot adapt to workload or energy price fluctuations.

2) RL-Based Algorithms: Adaptive control policies trained via RL outperform static baselines by dynamically adjusting compression levels according to system state. Among them, *PPO* achieves the best overall trade-off, maintaining a 97.0% success rate and strong output quality (0.250) while reducing total energy by roughly 20% compared with *FullKV*. *A2C* and *DQN* deliver comparable cumulative rewards, validating the generality of both actor–critic and value-based approaches. *TD3*, though designed for continuous control, also demonstrates competitive performance and stable learning dynamics.

3) Key Insight: The results demonstrate that the RL framework enables fine-grained, energy-aware adaptation of KV cache compression, achieving sustainable energy cost reductions while preserving LLM inference quality and reliability.

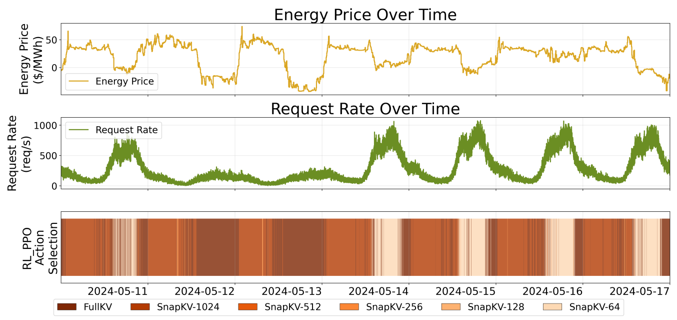

Fig. 2. Temporal evolution of the RL_PPO agent's compression actions.

D. Analysis of KV Cash Compression Dynamics

To further examine how the RL algorithm achieves the balance between quality and energy efficiency, we analyze the temporal evolution of its compression decisions. Figure 2 illustrates the dynamic behavior of the PPO policy over time, aligned with real-world energy prices and request arrival rates.

When energy prices spike or request loads increase, the RL agent proactively selects higher compression levels (e.g., SnapKV-256 or SnapKV-128) to reduce power consumption and latency. Conversely, during off-peak periods with lower prices or lighter workloads, it reverts to milder compression settings (e.g., SnapKV-1024 or FullKV) to preserve inference quality. This adaptive switching demonstrates that the policy's ability to capture temporal correlations between electricity market conditions and workload intensity, making real-time trade-offs between efficiency and quality.

Overall, the action timeline highlights the PPO agent's capability for sustainable, context-aware control, achieving superior responsiveness and energy efficiency compared with static compression strategies.

VI. Conclusion

This paper proposed a reinforcement learning-based framework for sustainable LLM serving through dynamic KV cache compression. By jointly optimizing for service quality and energy cost, the proposed method adaptively adjusts compression levels in real time to balance performance and efficiency. Experiments using real-world energy price and workload traces demonstrate up to 20% reduction in energy consumption without compromising output quality. These findings underscore the potential of energy-aware, intelligent control as a key enabler of sustainable large-scale AI deployment.

References

[1] N. Jegham, M. Abdelatti, L. Elmoubarki, and A. Hendawi, "How hungry is AI? benchmarking energy, water, and carbon footprint of LLM inference," 2025.

[2] A. de Vries, "The growing energy footprint of artificial intelligence," *Joule*, vol. 7, no. 10, pp. 2191–2194, 2023.

[3] Electric Power Research Institute (EPRI), "Powering intelligence: Analyzing artificial intelligence and data center energy consumption," Electric Power Research Institute, Tech. Rep., 2024.

[4] M. Koot and F. Wijnhoven, "Usage impact on data center electricity needs: A system dynamic forecasting model," *Applied Energy*, vol. 291, p. 116798, 2021.

[5] I. Riu, D. Smiley *et al.*, "Load growth is here to stay, but are data centers: strategically managing the challenges and opportunities of load growth," *Energy and Environmental Economics, Inc.*, July 2024.

[6] X. Zhu, J. Li, Y. Liu, C. Ma, and W. Wang, "A survey on model compression for large language models," *Trans. Assoc. Comput. Linguist.*, vol. 12, 2024.

[7] W. Wang, A. Abdolrashidi, N. Yu, and D. Wong, "Frequency regulation service provision in data center with computational flexibility," *Applied Energy*, vol. 251, p. 113304, 2019.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[9] G. Xiao, Y. Tian, B. Chen, S. Han, and M. Lewis, "Efficient streaming language models with attention sinks," in *The Twelfth International Conference on Learning Representations*, 2024.

[10] Y. Li, Y. Huang, B. Yang, B. Venkitesh, A. Locatelli, H. Ye, T. Cai, P. Lewis, and D. Chen, "SnapKV: LLM knows what you are looking for before generation," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[11] Y. Fu, Z. Cai, A. Asi, W. Xiong, Y. Dong, and W. Xiao, "Not all heads matter: A head-level KV cache compression method with integrated retrieval and reasoning," in *The Thirteenth International Conference on Learning Representations*, 2025.

[12] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7b," 2023.

[13] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with pagedattention," in *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023, p. 611–626.

[14] P. Patel, E. Choukse, C. Zhang, A. Shah, Í. Goiri, S. Maleki, and R. Bianchini, "Splitwise: Efficient generative LLM inference using phase splitting," in *51st ACM/IEEE Annual International Symposium on Computer Architecture*, 2024, pp. 118–132.

[15] J. Stojkovic, C. Zhang, Goiri, J. Torrellas, and E. Choukse, "Dynamollm: Designing LLM inference clusters for performance and energy efficiency," in *2025 IEEE International Symposium on High Performance Computer Architecture*, Mar. 2025, p. 1348–1362.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, 2017.

[17] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[18] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33nd International Conference on Machine Learning*, 2016.

[19] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[20] J. You, J.-W. Chung, and M. Chowdhury, "Zeus: Understanding and optimizing GPU energy consumption of DNN training," in *USENIX NSDI*, 2023.