EnergyLens: LLM Inference Energy Framework for Energy-Driven Model Implementation

Zhiye Song Kyungmi Lee Eun Kyung Lee* Xin Zhang*
Tamar Eilam* Anantha P. Chandrakasan
Massachusetts Institute of Technology, Cambridge, MA 02139 *IBM Research, Yorktown Heights, NY 10598
{zhiye,kyungmi}@mit.edu,{eunkyung.lee,xzhang,eilamt}@us.ibm.com

***Abstract*—We present EnergyLens, an end-to-end energy framework for large language model (LLM) inference optimization at early stages of model implementation. As LLMs scale, predicting and reducing their energy footprint has become critical for sustainability and datacenter operations, yet no existing tool provides accurate multi-GPU energy prediction before production-level code is available. EnergyLens addresses this gap with an intuitive einsum-based interface that captures LLM specifications including fusion, parallelism, and compute-communication overlap, combined with an empirically-driven overlap-aware communication energy model for multi-GPU settings. We validate EnergyLens on Llama3 8B and 70B across tensor-parallel configurations, achieving mean absolute percentage errors (MAPEs) of 12.18% and 11.36% for multi-GPU prefill and decode energy, and 12.97% across SM allocations for Megatron-style overlap. Our energy-driven exploration reveals up to $1.47\times$ and $52.9\times$ energy variations across configurations in prefill and decode efficiency. EnergyLens correctly predicts non-intuitive optimal configurations across overlap and non-overlap configurations.**

I. Introduction

The arrival of generative AI, with LLMs at its core, revolutionized machine learning. As LLMs scale to hundreds of billions of parameters and an ever-increasing user base, their growing compute demands translate to substantial energy costs and carbon footprints [7], [10], [23], [24]. This makes understanding and optimizing energy consumption critical for both sustainability and datacenter operations.

Meeting this challenge requires energy models that are useful during model implementation, not just after deployment. Existing approaches each cover part of the problem but fall short for early-stage LLM optimization. Direct measurement is accurate, but requires GPU access and existing implementations, making broad design-space exploration expensive and slow. Coarse holistic methods, including thermal design power (TDP)-based estimation [14], [20], are easy to apply but miss the strong variation in power across LLM inference regimes; in our Llama3-70B measurements on A100 GPUs, they can overestimate decode energy by up to 60%. LLMCO2 [8] predicts power but does not support fusion natively or include energy attribution. Kernel-level predictors like EnergAIzer [13] provide compute kernel attribution, but they do not capture the multi-GPU effects that increasingly dominate practical LLM serving, including parallelism and compute-communication overlap.

This limitation is increasingly important because production LLMs are commonly deployed across multiple GPUs to satisfy memory and latency constraints [4], [22]. As multi-GPU parallelism and emerging paradigms such as disaggregated inference [26] place greater demand on the network, accurate communication energy modeling becomes essential. Without it, significant energy optimization opportunities are missed: for Llama3-70B at TP8, communication accounts for 23% of total energy, and compute-communication overlap [1], [11] complicates latency-energy trade-offs in ways that existing tools cannot capture. The resulting design space spans fusion strategies, tensor parallelism degrees, overlap SM allocations, and batch sizes, each interacting with sequence lengths and latency targets. This combinatorial explosion makes exhaustive implementation and profiling impractical.

EnergyLens addresses this gap by enabling rapid exploration of the energy-latency design space without requiring existing code or GPU access. Developers specify LLM architectures with fusion, parallelism, and overlap settings in a few lines of code using our simple interface. The framework parses these specifications and outputs detailed energy breakdowns and energy-latency trade-offs by extending kernel-level prediction to multi-GPU settings with empirically-driven and overlap-aware communication modeling. For example, comparing two fusion settings for Llama3-70B prefill across 7 batch sizes and 32 input lengths would require profiling on an 8-GPU server for 12 hours, plus substantial engineering time. Using EnergyLens, generating results and visualizations takes only 15 minutes.

Our main contributions are as follows:

- We introduce EnergyLens, an end-to-end framework where model optimization engineers specify LLM architectures and distributed settings in a few lines using an intuitive einsum-based interface. EnergyLens enables energy-aware optimization before implementation, without requiring functional code or GPU access.
- We achieve accurate multi-GPU energy prediction through empirically-driven communication and overlap modeling, achieving 12.18% and 11.36% MAPEs for prefill and decode energy on Llama3-70B across tensor-parallel configurations. We achieve accurate overlap modeling, with an energy MAPE of 12.97% across SM allocations for Megatron-style overlap.
- We demonstrate EnergyLens's capability to do energy-

driven design space exploration. We reveal that compute-communication overlap is not always beneficial. We further show that prefill and decode have opposing optimal configurations, motivating disaggregated serving.

II. Background

A. Kernel-Level Performance Modeling

Several kernel performance models [13], [16], [18] offer faster alternatives to cycle-accurate GPU simulation [12]. We adopt EnergAIzer [13] as our primary compute kernel backend because it provides both latency and power estimates; our framework's modular design also supports alternative latency backends such as NeuSight [16] and Li et al. [18].

B. LLM Performance Modeling

Existing LLM performance modeling spans both training [17], [19], [25] and inference [3], [6], [16], but these tools share two key limitations: they focus on latency and throughput while neglecting energy, and most require execution traces from existing implementations, precluding early-stage design exploration. An energy-focused work, LLMCO2 [8], does not produce the energy breakdowns needed to facilitate optimization opportunities. Moreover, the graph neural network assumes the unfused version of the standard transformer (e.g. Llama2). Applying operation fusion alters this topology, requiring profiling of the fused implementation and retraining the model, neither of which is addressed in the paper. LLMServingSim v2.0 [5] adds a coarse three-state power model to its performance-focused model, but uses a single power number for active GPUs, missing the diverse power profiles such as between prefill and decode settings. EnergyLens addresses all these gaps: energy-centric modeling, breakdown visibility, and execution-trace-free input.

C. Compute-Communication Overlap

Recent work overlaps tensor-parallel collectives with GEMMs to hide communication latency in distributed training; this is also increasingly explored in the prefill phase of distributed inference [1], [11], [15]. Predictive modeling of its energy and latency impact, however, remains unexplored in inference modelling [5], [8], [16]. Lee et al [15] characterize overlap empirically but does not predict performance. Hong et al [11] optimize overlap strategies using a latency model that assumes a fixed SM allocation for communication and linear scaling of GEMM tiles. This latency model deteriorates with varying SM allocations or under smaller workloads.

III. EnergyLens Framework

Figure 1 illustrates the EnergyLens framework. The *EnergyLens Interface* (Section III-A) provides an einsum-based specification language for describing LLM architectures, fusion strategies, and parallelism settings. A *Lightweight Interpreter* (Section III-B) parses these specifications to extract GEMM dimensions, arithmetic intensity, and required communication kernels. The resulting kernel information feeds into an *Overlap-Aware Communication Model* (Section III-C),

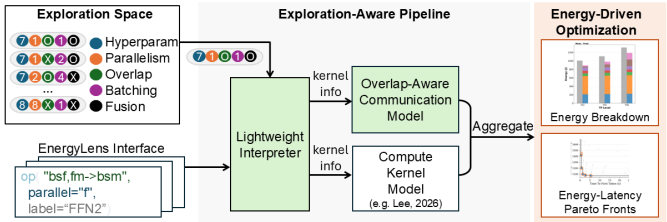


Fig. 1: EnergyLens framework enables energy-latency optimization in the high-dimensional exploration space at early stages of model implementation.

which captures multi-GPU communication energy through empirical profiling, and a *Compute Kernel Model* that leverages EnergAIzer [13] for single-GPU kernel latency and power estimation, with support for alternative latency backends such as NeuSight and Li's model [16], [18]. The kernel level results go through an aggregation stage that accounts for compute-communication overlap (Section III-D) to produce end-to-end energy breakdowns and energy-latency Pareto fronts.

A. EnergyLens Interface

EnergyLens exposes an einsum-based interface for specifying LLM inference operators, with added lightweight annotations for parallelism and overlap. We choose einsum notation because it is widely used in frameworks such as PyTorch for expressing tensor operations [2], enabling intuitive compact descriptions that are easily extended to both single-GPU and distributed implementations.

Listing 1 shows a complete fused Llama3 specification in a few lines of code, with symbols defined in Table I. For example, the fused QKV projection is written as `bsm,miKh->bsiKh`, and tensor parallelism is specified by annotating the sharded dimension (e.g., `parallel="K"`). The same specification covers both prefill and decode: s denotes sequence length in prefill and s=1 in decode. Model dimensions are supplied at runtime, so one specification can represent models at different scales.

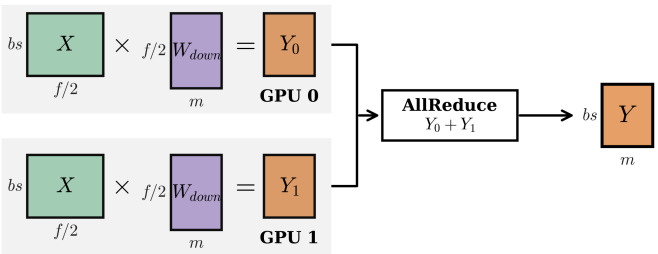
Listing 1 Full specification of a fused Llama3 implementation.

```
1 llm_model.EQS = [
2     op("bsm,miKh->bsiKh", parallel="K",
          label="QKV Projection"),
3     op("attention", parallel="H",
          label="Attention"),
4     op("bsHh,Hhm->bsm", parallel="H",
          label="Output Projection"),
5     op("bsm,mF->bsF", parallel="F",
          label="Gate & Up Projection"),
6     op("bsf,fm->bsm", parallel="f",
          label="Down Projection")
7 ]
```

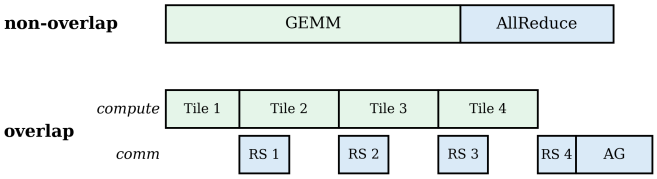
Alternative fusion strategies are expressed by modifying only the symbolic operator descriptions. For instance, Listing 1 fuses Q, K, and V projections into one operator (line 2) and combines the gate and up projections (line 5). EnergyLens

TABLE I: LLM symbols used in Listing 1

Symbol	Description
b	Batch size
s	Input length in prefill; 1 in decode
h	Head dimension
H	Number of query heads
K	Number of key/value heads
m	d_{model}, i.e. embeddings or hidden dimension
f	FFN dimension
i	$2 + H/K$, used in the fused QKV projection



(a) Down projection with TP2.



(b) Timeline (RS: ReduceScatter; AG: AllGather).

Fig. 2: EnergyLens enables intuitive specification of the parallelism and overlap settings.

also supports distributed execution by annotating the partitioned dimension, e.g., `parallel="s"` for prefill sequence parallelism or `parallel="m"` for tensor parallelism. This compact interface enables rapid exploration of the energy-latency optimization space across input sequence length (ISL), output sequence length (OSL), batch size, model size, and parallelism.

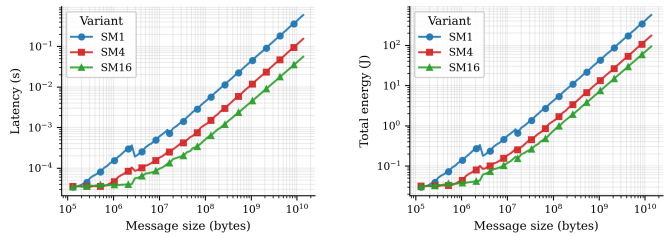
B. Lightweight Interpreter

The lightweight interpreter maps each einsum specification to kernel-level features used for energy and latency prediction. For compute operators, it extracts the corresponding general matrix multiplication (GEMM) dimensions in the standard form $A[M, K] \times B[K, N] = C[M, N]$. The interpreter also computes arithmetic intensity to characterize whether an operator is compute- or memory-bound.

For distributed configurations, the interpreter derives the required communication kernels from the einsum reduction and the annotated sharded dimension. If the parallelized dimension is reduced, the operator requires a collective communication step (e.g., AllReduce), whose message size is determined from the output tensor shape and data type.

C. Communication Energy Modeling

We propose an empirically-driven and overlap-aware communication energy model. Unlike prior work that assumes fixed bandwidth utilization and only estimates latency [16], we profile latency and energy consumption across various communication kernels and transfer sizes and use interpolation during prediction. As the example shown in Figure 3, overhead dominates at small workload sizes, resulting in higher energy cost per bit. This behavior is missed by bandwidth-and-queuing-based models used in prior works [16], [18]. Figure 3 further shows that the number of SMs dedicated to communication impacts both latency and energy, creating a trade-off between SM allocations for compute and communication during overlap (Section III-D).



(a) Latency (b) Energy

Fig. 3: The behavior of 8-GPU ReduceScatter differs based on the number of SMs dedicated to communication.

D. Compute-Communication Overlap

Distributed inference frameworks increasingly overlap tensor-parallel collectives with GEMMs to hide communication latency [1]. EnergyLens models such overlap with three additions to the einsum specification. `overlap_stage` sets the number of tiles into which compute and communication are split; `overlap_SM` sets the number of SMs dedicated to communication; and `overlap` specifies the dimension to tile. For example, a down projection with four-stage overlap and 16 SMs reserved for communication is written as `op("bsf,fm->bsm", parallel="f", overlap_stage=4, overlap_SM=16, overlap="s", label="Down Projection")`.

EnergyLens models an example GEMM-AllReduce overlapped layer as three phases (Figure 2b): (i) the first non-overlapped compute tile using all SMs, (ii) overlapped stages whose duration is the maximum of the SM-restricted GEMM tile and the SM-restricted NCCL tile, and (iii) a final exposed communication tile using all SMs. Tile latencies are obtained by invoking the kernel-level model on the smaller tiled GEMM and obtaining NCCL latency and energy at constrained SM counts (as shown in Figure 3). The total latency is then calculated as

$$t_{total} = t_{GEMM} + t_{RS} + \max\left(t_{GEMM_{overlap}}, t_{RS_{overlap}}\right) \times (stage - 1) \quad (1)$$

, where t_{GEMM} and t_{RS} represent the per-tile latency. The latency of an overlapped segment is larger than a non-overlapped segment due to less SMs being available. The impact of SM allocation on an example communication kernel is shown in Figure 3a. Prior overlap latency model assumes that a quarter-sized tile of GEMM would take a quarter of the latency [11], which is not accurate at smaller workloads. EnergyLens queries the compute kernel model for the quarter-sized tile. This is important at small batch-sequence products to accurately determine whether multi-stage overlap is profitable.

For overlapped stages we attribute power using the GEMM estimate, since the majority of SMs and memory bandwidth serve compute. For the exposed communication tile we use the same overlapped-stage power rather than steady-state NCCL power; we find the latter underestimates energy, potentially because GPU DVFS cannot adjust voltage during such short kernel durations.

E. Energy Trade-offs

We introduce energy-centric metrics analogous to traditional latency metrics for LLM inference: energy to first token (ETFT) measures the energy consumption during prefill (similar to time to first token (TTFT)), while energy per output token (EPOT) measures decode phase energy normalized by OSL (analogous to time per output token (TPOT)). Both ETFT and EPOT are normalized per request. EnergyLens's interface enables users to directly explore these workload-level energy metrics, along with traditional latency and throughput metrics, from a few lines of model descriptions.

IV. Evaluation

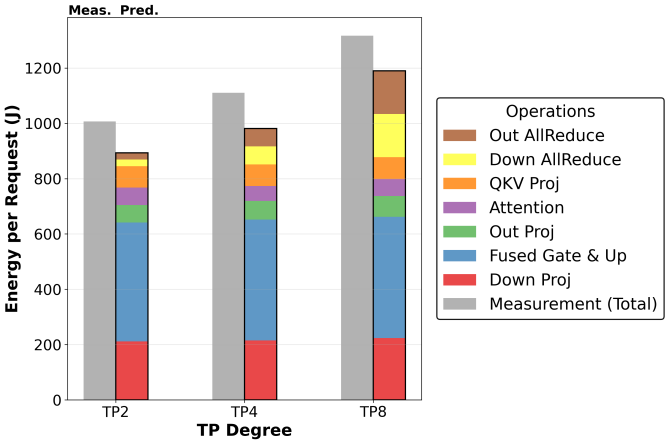
A. Evaluation Methods

We evaluate on Llama3-8B and Llama3-70B in bf16 precision [9]. Ground-truth measurements use TensorRT-LLM v0.14 [21], whose fusion strategy is described in Listing 1, on a server with 8 A100-SXM4-80GB GPUs. To capture steady-state behavior, we apply a 20 s warmup before measurement. Each workload is then characterized for at least one full iteration and at least 40 s. We measure end-to-end latency directly and GPU energy via NVIDIA Management Library (NVML). We use Torch Profiler [2] to validate the predicted latency breakdown and confirm the actual batch sizes used by TensorRT-LLM, which may differ from the requested values at long contexts.

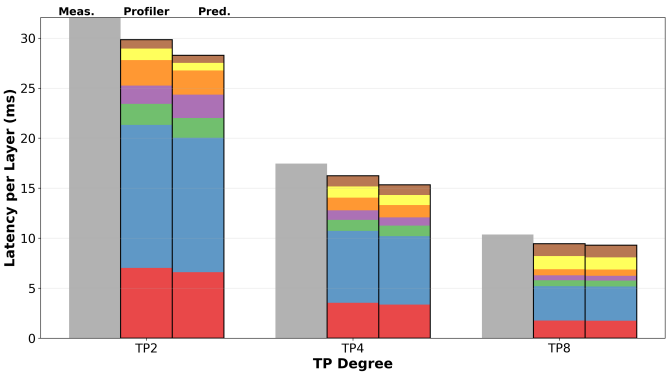
For compute-communication overlap experiments (Section IV-D), we additionally use Megatron-LM v0.15.3 [22] as ground truth. Profiler traces reveal that tensor-parallel all-reduce is decomposed as a reduce-scatter overlapped with GEMM followed by an exposed all-gather.

B. Single-GPU Evaluation

We first validate EnergyLens on single-GPU Llama3-8B inference, sweeping ISL up to 8192 tokens and batch sizes up to 64. Prefill ETFT predictions achieve a MAPE of 11.31%. For the decode phase, which was overlooked by prior works



(a) Prefill energy breakdown.



(b) Prefill latency breakdown with the same legend as (a).

Fig. 4: (a) Energy and (b) latency breakdown for Llama3-70B prefill phase at B2, ISL=4096 across tensor parallelism configurations (TP2, TP4, TP8), demonstrating increasing communication overhead as TP degree increases.

[13], [16], the energy consumption per token is orders of magnitude higher than in the prefill phase. EnergyLens achieves a decode MAPE of 12.87% and correctly predicts that energy per request decreases significantly with larger batch sizes, in contrast with prefill behavior.

C. Multi-GPU Evaluation

For prefill, most operators are compute-bound at practical ISLs, so further increasing arithmetic intensity through batching yields less energy efficiency gains than lower tensor parallelism. Figure 4a shows that compute energy remains nearly constant across tensor parallelism (TP) configurations, while inter-GPU collectives grow with TP. Torch profiler execution traces verify that there is no overlap between compute and communication kernels. At TP8, AllReduce accounts for 23% of total energy. EnergyLens achieves an energy MAPE of 12.18%, with ISLs up to 8192, batch sizes up to 64, and TP up to 8. Figure 4b further shows that the predicted latency breakdown closely matches the measured execution trace, validating the kernel-level decomposition used for energy attribution.

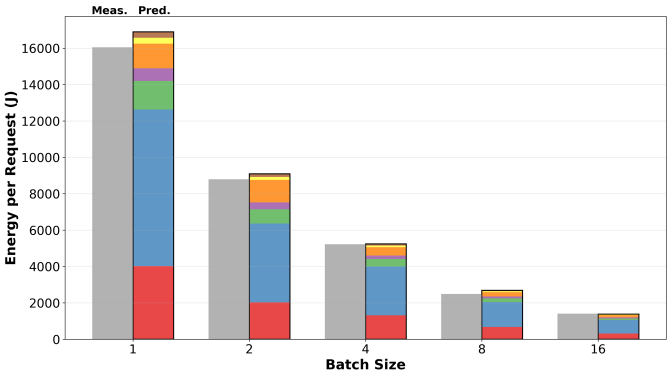


Fig. 5: Llama3-70B decode energy breakdown at TP4 across batch sizes with the same legend as Figure 4a. Larger batches substantially reduce decode energy per token.

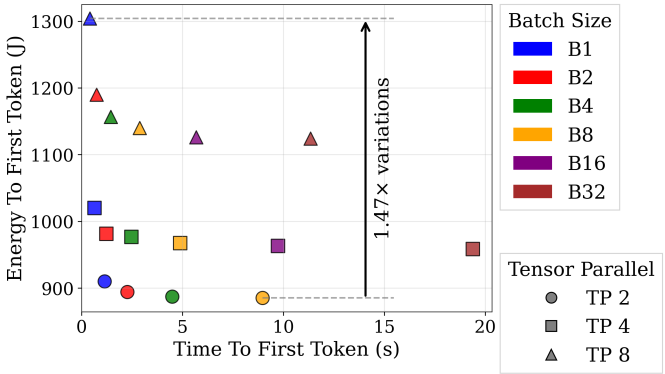


Fig. 6: Energy-latency trade-off for Llama3-70B prefill phase at ISL=4096, showing that TP2 configurations dominate the Pareto frontier due to reduced communication overhead. Combinations that exceed GPU memory are omitted.

Decode exhibits a different trade-off. As shown in Figure 5, energy varies more strongly with batch size than with TP: larger batches substantially reduce decode energy. However, batching is constrained by GPU memory, since increasing batch size reduces the maximum sequence length that can fit in DRAM. This trade-off arises because decode runs with s=1, producing skewed GEMMs with low arithmetic intensity. For example, the down projection `op("bsf,fm->bsm", parallel="f")` maps to a matmul with $M = b$ and $N = m$. At small batch sizes, these kernels are memory-bound and underutilize the GPU, while scaling up batch sizes increases arithmetic intensity and reduces energy per output token. Across TP2, TP4, and TP8, batch sizes 1–16, and OSL up to 7680, EnergyLens achieves a decode energy MAPE of 11.36%, with EPOT MAPE lower at 7.78%.

Decode latency is harder to predict accurately with a MAPE of 27.93%, due to low-arithmetic-intensity kernels and highly skewed matrix shapes. We observe similar limitations across alternative compute backends [16], [18], indicating that this issue is not backend-specific. Nevertheless, EnergyLens still captures the key trend relevant for design exploration: decode latency changes only modestly with batch size, while energy improves substantially.

D. Compute-Communication Overlap

We validate overlap modeling on Llama3-70B. We sweep batch sizes 1–64, TP degrees 2–8, ISL up to 8192 and overlap configurations including no overlap and 4-stage overlap with 1, 4, and 16 SMs dedicated to communication. Across this sweep, EnergyLens predicts energy with a MAPE of 12.97% and latency with a MAPE of 11.18%. More importantly, EnergyLens helps to predict the relative trade-offs across SM allocations: dedicating too many SMs to communication wastes compute resources, while allocating too little makes communication the bottleneck of the overlapped stages. The ability to include overlap as a dimension in energy-latency trade-off will be exploited in Section V-B.

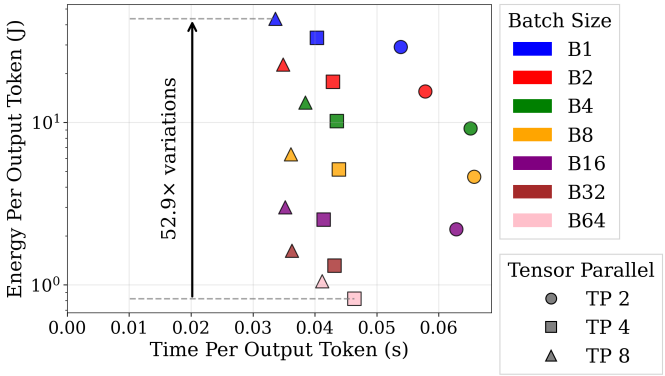


Fig. 7: Energy-latency trade-off for Llama3-70B decode phase (ISL=512, OSL=1024) showing that larger batch sizes achieve order-of-magnitude improvement in EPOT with minimal latency increase. Combinations that exceed GPU memory are omitted.

V. Energy-Centric Exploration

The energy prediction errors presented in Section IV are substantially smaller than the energy variations across configurations reported below. We leverage this accuracy to derive the following energy-driven insights.

A. Disaggregated Serving

Key Insight 1: *At equal TTFT, smaller batch size and lower tensor parallelism are more energy efficient.* Two approaches can reduce TTFT: smaller batch sizes decrease compute per batch, while higher TP distributes the same compute across more GPUs. Figure 6 reveals that most Pareto-optimal configurations use TP2, due to reduced communication overhead (Figure 4a); the lower bound on TP degree comes from GPU memory requirements. At ISL 4096 (Figure 6), TP2 with batch size 4 is both faster and consumes 24% less energy than TP8 with batch size 16. This is confirmed by measurements showing 30% energy savings. Across all ISLs for 70B prefill,

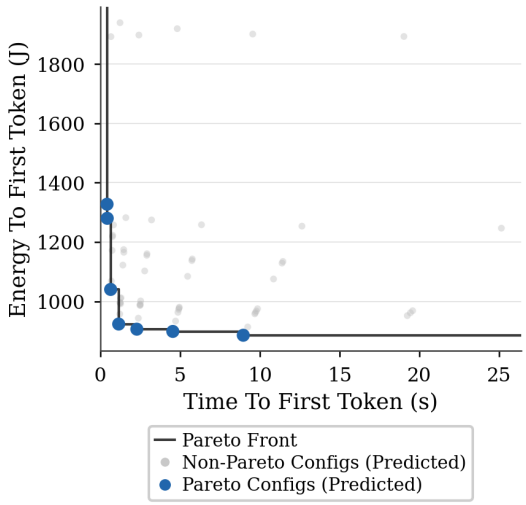
B4 TP2 over B16 TP8 yields an average 20.87% energy reduction with lower latency.

***Key Insight 2**: Increasing batch size has significant energy savings during decode, at minimal latency cost.* Figure 7 plots EPOT against TPOT: larger batch sizes dramatically reduce EPOT with only modest latency increase. At batch size 1, skewed matrix shapes ($M = b = 1$) severely underutilize GPUs; increasing batch size shifts GEMM kernels toward compute-bound regimes, improving efficiency. Across all TPs and OSL up to 7680, batch size 16 achieves 92.63% EPOT reduction versus batch size 1. The memory requirement of larger batch size can be mitigated with higher degrees of parallelism. These contrasting objectives, lower batch sizes for prefill latency versus larger batch sizes for decode efficiency, strongly motivate disaggregated serving [26], where prefill and decode are independently optimized.

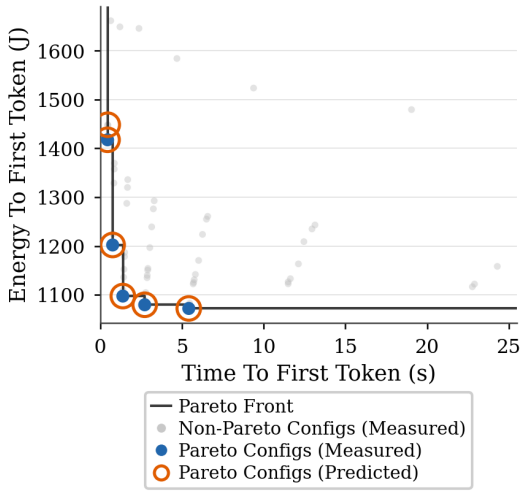
B. Megatron-Style Overlap

Given EnergyLens's ability to accurately model Megatron-style compute-communication overlap, we demonstrate the ability to find the optimal configurations without model deployment. At an example ISL of 4096, we sweep 55 configurations spanning different overlap SM allocations, batch sizes, and tensor parallelism degrees. Figure 8a shows that EnergyLens identifies a Pareto frontier, distinguishing energy-latency-optimal overlap settings from suboptimal ones.

Notably, the Pareto-optimal configurations here are predominantly non-overlapped TP2 settings, with overlap only profitable at the most latency-aggressive point (TP8). This reveals that overlap is not universally beneficial, reducing tensor parallelism and avoiding overlap overhead (smaller GEMM tiles, non-optimal SM allocation) often yield better energy efficiency, while overlap only becomes worthwhile when tight latency targets demand higher TP degrees. Figure 8b validates these predictions with energy and latency measurements, confirming that they fall on the measured Pareto frontier. This demonstrates that EnergyLens can reliably identify these non-obvious trade-offs without requiring implementation or GPU access.



(a) EnergyLens predicted optimal configurations.



(b) The predicted Pareto-optimal points fall on the Pareto Front of the measured configurations.

Fig. 8: Llama3-70B (ISL=4096), showing 55 settings for Megatron-style compute-communication overlap, batch size, and tensor parallelism.

VI. Conclusion

We present EnergyLens, an end-to-end framework that predicts LLM inference energy through a high-level interface without existing implementation or GPU access, and provides multi-GPU support with empirically-driven communication and overlap energy models. Evaluation on Llama3-70B achieves mean absolute percentage errors (MAPEs) of 12.18% and 11.36% for multi-GPU prefill and decode energy, and 12.97% for overlap configurations. Energy-driven exploration reveals up to 20.87% and 92.63% energy reduction by decoupling prefill and decode configurations. The exploration also shows that compute-communication overlap is not universally beneficial: reducing TP degree often outperforms overlap except at the most latency-aggressive operating points. These insights showcase how EnergyLens enables energy-aware design decisions before implementation.

Acknowledgements

This work was supported by the MIT-IBM Watson AI Lab and by the Natural Sciences and Engineering Research Council of Canada (NSERC) Postgraduate Scholarship program.

References

[1] "NVIDIA NeMo Framework User Guide: Communication Overlap." [Online]. Available: https://docs.nvidia.com/nemo-framework/user-guide/24.09/nemotoolkit/features/optimizations/communication_overlap.html

[2] "Torch Profiler," 2025. [Online]. Available: https://docs.pytorch.org/docs/stable/profiler.html

[3] A. Agrawal, N. Kedia, J. Mohan, A. Panwar, N. Kwatra, B. S. Gulavani, R. Ramjee, and A. Tumanov, "Vidur: A Large-Scale Simulation Framework for LLM Inference," in *MLSys*, 2024.

[4] R. Y. Aminabadi, S. Rajbhandari, A. A. Awan, C. Li, D. Li, E. Zheng, O. Ruwase, S. Smith, M. Zhang, J. Rasley, and Y. He, "DeepSpeed-Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale," *International Conference for High Performance Computing, Networking, Storage and Analysis, SC*, 2022.

[5] J. Cho, H. Choi, G. Heo, and J. Park, "LLMServingSim 2.0: A Unified Simulator for Heterogeneous and Disaggregated LLM Serving Infrastructure," Mar. 2026, arXiv:2602.23036 [cs]. [Online]. Available: http://arxiv.org/abs/2602.23036

[6] J. Cho, M. Kim, H. Choi, G. Heo, and J. Park, "LLMServingSim: A HW/SW Co-Simulation Infrastructure for LLM Inference Serving at Scale," in *2024 IEEE International Symposium on Workload Characterization (IISWC)*, Sep. 2024, pp. 15–29, arXiv:2408.05499 [cs]. [Online]. Available: http://arxiv.org/abs/2408.05499

[7] A. Faiz, S. Kaneda, R. Wang, R. Osi, P. Sharma, F. Chen, and L. Jiang, "LLMCarbon: Modeling the end-to-end Carbon Footprint of Large Language Models," pp. 1–15, 2023, arXiv: 2309.14393. [Online]. Available: http://arxiv.org/abs/2309.14393

[8] Z. Fu, F. Chen, S. Zhou, H. Li, and L. Jiang, "LLMCO2: Advancing Accurate Carbon Footprint Prediction for LLM Inferences," *ACM SIGENERGY Energy Informatics Review*, vol. 5, no. 2, 2025.

[9] Grattafiori, Aaron et al, "The Llama 3 Herd of Models," 2024.

[10] P. Henderson, J. Hu, J. Romoff, E. Brunskill, D. Jurafsky, and J. Pineau, "Towards the systematic reporting of the energy and carbon footprints of machine learning," *Journal of Machine Learning Research*, vol. 21, 2020, arXiv: 2002.05651.

[11] K. Hong, X. Li, M. Liu, Q. Mao, T. Wu, Z. Huang, L. Chen, Z. Wang, Y. Zhang, Z. Zhu, G. Dai, and Y. Wang, "Efficient and Adaptable Overlapping for Computation and Communication via Signaling and Reordering," Oct. 2025, arXiv:2504.19519 [cs]. [Online]. Available: http://arxiv.org/abs/2504.19519

[12] V. Kandiah, S. Peverelle, M. Khairy, J. Pan, A. Manjunath, T. G. Rogers, T. M. Aamodt, and N. Hardavellas, "AccelWattch: A power modeling framework for modern GPUs," *Proceedings of the Annual International Symposium on Microarchitecture, MICRO*, pp. 738–753, 2021, iSBN: 9781450385572.

[13] Kyungmi Lee, Zhiye Song, Eun Kyung Lee, Xin Zhang, Tamar Eilam, and Anantha P. Chandrakasan, "EnergAIzer: Fast and Accurate GPU Power Estimation Framework for AI Workloads." ISPASS, 2026.

[14] A. Lacoste and T. Dandres, "Quantifying the Carbon Emissions of Machine Learning," 2019, arXiv: 1910.09700v2.

[15] S. Lee, J. Oh, J. Kim, S. Go, J. Park, and D. Mahajan, "Characterizing Compute-Communication Overlap in GPU-Accelerated Distributed Deep Learning: Performance and Power Implications," Jul. 2025, arXiv:2507.03114 [cs]. [Online]. Available: http://arxiv.org/abs/2507.03114

[16] S. Lee, A. Phanishayee, and D. Mahajan, "Forecasting GPU Performance for Deep Learning Training and Inference," in *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, Mar. 2025, pp. 493–508, arXiv:2407.13853 [cs]. [Online]. Available: http://arxiv.org/abs/2407.13853

[17] Y. Li, Y. Bao, G. Wang, X. Mei, P. Vaid, A. Ghosh, A. Jog, D. Bunandar, A. Joshi, and Y. Sun, "TrioSim: A Lightweight Simulator for Large-Scale DNN Workloads on Multi-GPU Systems," in *Proceedings of the 52nd Annual International Symposium on Computer Architecture*. Tokyo Japan: ACM, Jun. 2025, pp. 1524–1538. [Online]. Available: https://dl.acm.org/doi/10.1145/3695053.3731082

[18] Y. Li, Y. Sun, and A. Jog, "Path Forward Beyond Simulators: Fast and Accurate GPU Execution Time Prediction for DNN Workloads," in *56th Annual IEEE/ACM International Symposium on Microarchitecture*. Toronto ON Canada: ACM, Oct. 2023, pp. 380–394. [Online]. Available: https://dl.acm.org/doi/10.1145/3613424.3614277

[19] M. Liang, H. T. Kassa, W. Fu, B. Coutinho, L. Feng, and C. Delimitrou, "Lumos: Efficient Performance Modeling and Estimation for Large-scale LLM Training," in *MLSys Conference*, 2025.

[20] A. S. Luccioni, S. Viguier, and A.-L. Ligozat, "Estimating the carbon footprint of BLOOM, a 176B parameter language model," *Journal of Machine Learning Research*, vol. 24, no. 1, Jan. 2023, number of pages: 15 tex.articleno: 253 tex.issue_date: January 2023.

[21] NVIDIA, "TensorRT-LLM." [Online]. Available: https://github.com/NVIDIA/TensorRT-LLM

[22] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism," 2019, arXiv: 1909.08053. [Online]. Available: http://arxiv.org/abs/1909.08053

[23] E. Strubell, A. Ganesh, A. McCallum, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Modern Deep Learning Research," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 09, pp. 13 693–13 696, 2020, arXiv: 1906.02243v1. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7123

[24] A. Tschand, A. T. R. Rajan, S. Idgunji, A. Ghosh, J. Holleman, C. Kiraly, P. Ambalkar, R. Borkar, R. Chukka, T. Cockrell, O. Curtis, G. Fursin, M. Hodak, H. Kassa, A. Lokhmotov, D. Miskovic, Y. Pan, M. P. Manmathan, L. Raymond, T. S. John, A. Suresh, R. Taubitz, S. Zhan, S. Wasson, D. Kanter, and V. J. Reddi, "MLPerf Power: Benchmarking the Energy Efficiency of Machine Learning Systems from Microwatts to Megawatts for Sustainable AI," pp. 1–14, 2024, arXiv: 2410.12032. [Online]. Available: http://arxiv.org/abs/2410.12032

[25] W. Won, T. Heo, S. Rashidi, S. Sridharan, S. Srinivasan, and T. Krishna, "ASTRA-sim2.0: Modeling Hierarchical Networks and Disaggregated Systems for Large-model Training at Scale," *Proceedings - 2023 IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS 2023*, pp. 283–294, 2023, arXiv: 2303.14006 ISBN: 9798350397390.

[26] Y. Zhong, S. Liu, J. Chen, J. Hu, Y. Zhu, X. Liu, X. Jin, and H. Zhang, "DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving," in *18th USENIX Symposium on Operating Systems Design and Implementation*. arXiv, Jun. 2024, arXiv:2401.09670 [cs]. [Online]. Available: http://arxiv.org/abs/2401.09670