Mixed-Precision Processing-in-Memory Architecture for Edge Device LLM Inference

Shunchen Shi*†, Xueqi Li*†, Ninghui Sun*†
*State Key Lab of Processors, Institute of Computing Technology, CAS
†University of Chinese Academy of Sciences
Email: {shishunchen22z, lixueqi, snh}@ict.ac.cn

***Abstract*—Large language model (LLM) inference on edge devices faces significant challenges due to limited resources and memory bottlenecks. Mixed-precision quantization can effectively reduce resource demands and computational overhead, but existing processing-in-memory (PIM) solutions often lack support for efficient mixed-precision execution. To address these bottlenecks, we propose MixPIM, a hardware-software co-designed system that integrates a decoder-wise mixed-precision quantization engine with a bank-level parallel multi-precision DRAM PIM accelerator. MixPIM dynamically allocates bit-widths across decoder blocks based on importance scores while efficiently executing general matrix-vector multiplication (GEMV) operations under multiple precision modes within memory, eliminating off-chip data movement. Experimental results on LLaMA models show that MixPIM improves energy efficiency by 4.1× and 4.5× compared to state-of-the-art GPU and heterogeneous PIM systems.**

I. Introduction

Large language models (LLMs) have achieved great success in various natural language tasks [1]–[5]. With the development of embodied intelligence, IoT and other scenarios, there is growing interest in LLM inference on the edge devices [6]. Local deployment of LLMs mitigates the bandwidth and energy costs associated with cloud computing [7]; it reduces the dependency of network conditions, enhancing privacy protection and personalization with local user context [6]. However, the edge device LLM inference has resources constraints and is mainly bottlenecked by memory-bound GEMV operations. Quantization can compress the model to reducing computing and memory requirements [8], which is suitable for edge device inference [9]. Although traditional accelerators such as GPUs have rich computing resources, they are inefficient in single-batch inference and have high area and power consumption overhead. PIM can effectively accelerate GEMV operations [10]–[13]. However, we observe that the current PIM solution cannot support the high-performance execution for mixed precision quantization inference.

II. Motivation and Challenges

For on-device LLM inference, both attention and FFN is mainly GEMV operations. The model weights are read from memory, used once and replaced. This inefficient reuse leads to high memory bandwidth demands and limits computing unit utilization, making conventional platforms inefficient. As shown in Fig. 1, our experiments on LLaMA2-7B with an NVIDIA A100 GPU (batch size 1) reveal that GEMV operations dominate over 85% of kernel execution time for FP16 models and over 70% even after 8/4 bit quantization, showing low operational intensity and memory bottlenecks.

Quantization is a solution to reduce LLM memory and computation overhead. However, existing methods mainly take uniform precision quantization, which applies identical bit-widths across all decoder blocks, ignoring functional difference and leading to suboptimal trade-offs [14]–[16]. To address these limitations, we are motivated to propose a decoder-wise mixed-precision strategy that quantize the model based on decoder block sensitivity, preserving critical computations while compressing less sensitive blocks.

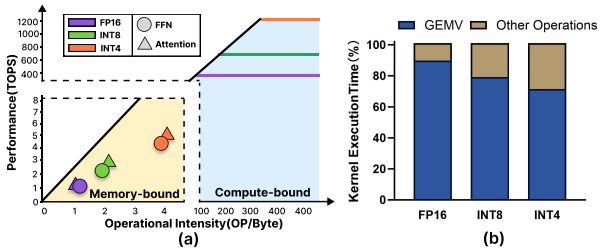

Fig. 1. (a) Roofline model analysis. (b) Kernel execution time breakdown.

III. Design

Figure 2 illustrates the overall architecture of the proposed MixPIM system, which accelerates LLM inference on edge platforms through hardware-software co-design.

At the software level, MixPIM introduces a decoder-aware mixed-precision quantization engine. Starting from a original LLM model, it iteratively analyzes and assigns different bit-widths to decoder blocks based on our defined importance score that derived from the sparsity of input/output activation matrices. Quantization engine can also takes grouping strategy to take several decoders as a group and increase the efficiency of the process. Blocks with lower scores are quantized to lower precision, while more sensitive blocks retain higher precision, thereby reducing model size and computation cost while maintaining accuracy.

Each DRAM bank within HBM3 is equipped with a GEMV execution unit capable of configurable precision. Under host control, these GEMV units can be configured and executed with different precisions to match the quantization bit-widths of different decoder blocks. This design eliminates the off-chip data movement, thereby minimizing memory traffic. Each

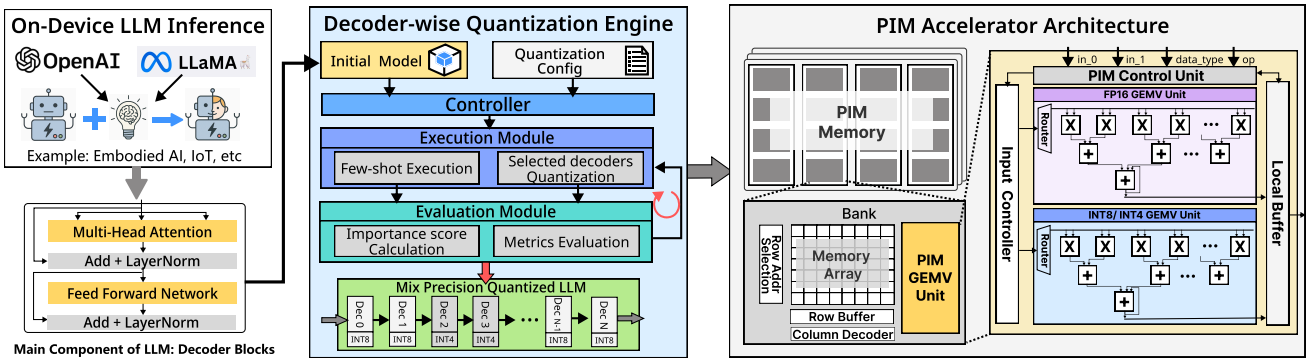

Fig. 2. Overview of the MixPIM system for edge device LLM inference. The design contains a decoder-wise quantization engine and a multi-precision PIM accelerator.

PIM GEMV unit can switch precision modes and processes input vectors in bank-level parallelism, retaining intermediate results in local buffers. The results are eventually retrieved and reduced by the host. We takes the similar data layout strategy proposed in AttAcc [11], but we extend it to support all GEMV operations throughout the inference process.

In the end-to-end execution workflow, the LLM model is first quantized offline using the mixed-precision engine, generating a mix-precision quantized model and a bit-width configuration. Based on this configuration, model weights are laid out onto the PIM memory. During inference, PIM units directly execute GEMV operations of varying precision inside the DRAM arrays, buffering partial results locally before writing them back to main memory. Non-GEMV operations (such as softmax and residual addition) are performed on the host CPU. This process iterates until the sequence generation is completed, enabling efficient offloading of memory-bound GEMV operations to PIM and establishing a memory-centric inference path optimized for edge deployment of LLMs.

IV. Evaluation

Mixed-precision Quantization We evaluate the effectiveness of MixPIM decoder-wise mixed-precision quantization algorithm by comparing it against SmoothQuant [14] 8-bit (W8A8) and 4-bit (W4A4) variant, across benchmarks including WikiText [17], LAMBADA [18], ARC-Easy [19], PIQA [20], and BoolQ [21]. MixPIM achieves higher model compression ratios compared to the existing 8-bit baseline, with 52.1% and 47.9% compression on LLaMA-2 7B and LLaMA-3 8B, respectively, representing improvements of 8.3% and 10.1% over W8A8. In terms of accuracy and perplexity, MixPIM introduces slight degradation: 1.4 and 3.0 perplexity increases and 1.9% and 2.9% accuracy drops, respectively. Moreover, while W4A4 achieves higher compression rates (72.1% and 65.2%), it suffers from severe perplexity and accuracy degradation, rendering it impractical for deployment. Overall, MixPIM achieves a better trade-off between quantization and inference quality.

End-to-end Performance Comparison We evaluate end-to-end token generation throughput on LLaMA2-7B [22] and LLaMA3-8B [23] in single-batch inference. MixPIM achieves 45.4 and 43.5 tokens/s, enabled by efficient multi-precision PIM execution. Compared to an single A100 GPU with the same capacity AttAcc [11] system, MixPIM reaches 86% throughput. The high performance of AttAcc + GPU is due to two reasons: GPU provides massive compute resources for FC layers and AttAcc PIM efficiently processes the attention layer. Compared with A100 GPU, MixPIM achieves 1.1 × higher throughput on average, and outperforms the Intel Xeon Gold 6354 CPU by up to 9.3×.

Performance per Watt Comparison We further evaluate energy efficiency of edge device LLM inference by measuring throughput-per-watt (tokens/s/W) across platforms. As shown in Fig. 3, MixPIM achieves 0.70 tokens/s/W on average, outperforming NVIDIA A100 GPU by 4.1×. Compared with the AttAcc+GPU systems, MixPIM delivers 4.5× higher energy efficiency. These results highlight the efficiency benefit of MixPIM for on-device LLM inference, where optimizing energy efficiency is often prioritized over absolute throughput.

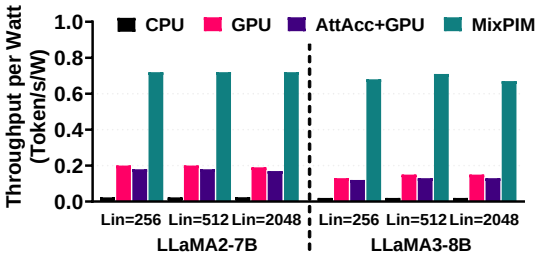

Fig. 3. Energy efficiency comparison (tokens/s/W) of MixPIM against CPU, GPU, and GPU+AttAcc baselines for single-batch inference on LLaMA2-7B and LLaMA3-8B

V. Conclusion

We propose MixPIM to accelerate the edge device LLM inference. MixPIM is a codesign of a decoder-wise mixed-precision quantization engine and a multi-precision PIM accelerator to address the memory bottlenecks in on-device LLM inference. MixPIM reaches 4.1× and 4.5× higher energy efficiency over GPU and state-of-the-art PIM baselines, demonstrating the effectiveness for energy-efficient LLM inference on edge devices.

VI. Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62202454 and 62488101, the National Key R&D Program of China under Grant 2022YFB4500403 and Beijing Natural Science Foundation under Grant QY23181.

References

[1] OpenAI, "Chatgpt." https://openai.com/index/chatgpt, 2023. Accessed: 2025-04-07.

[2] W. Zhu, H. Liu, Q. Dong, J. Xu, S. Huang, L. Kong, J. Chen, and L. Li, "Multilingual machine translation with large language models: Empirical results and analysis," in *Findings of the Association for Computational Linguistics: NAACL 2024* (K. Duh, H. Gomez, and S. Bethard, eds.), (Mexico City, Mexico), pp. 2765–2781, Association for Computational Linguistics, June 2024.

[3] Microsoft, "Copilot." https://github.com/features/copilot, 2023. Accessed: 2025-04-07.

[4] C.-W. Chiang, Z. Lu, Z. Li, and M. Yin, "Enhancing ai-assisted group decision making through llm-powered devil's advocate," in *Proceedings of the 29th International Conference on Intelligent User Interfaces*, IUI '24, (New York, NY, USA), p. 103–119, Association for Computing Machinery, 2024.

[5] OpenAI, "Sora: Creating video from text." https://openai.com/sora, 2024. Accessed: 2025-04-07.

[6] J. Xu, Z. Li, W. Chen, Q. Wang, X. Gao, Q. Cai, and Z. Ling, "On-device language models: A comprehensive review," 2024.

[7] S. Luccioni, Y. Jernite, and E. Strubell, "Power hungry processing: Watts driving the cost of ai deployment?," in *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, p. 85–99, ACM, June 2024.

[8] S. Li, X. Ning, L. Wang, T. Liu, X. Shi, S. Yan, G. Dai, H. Yang, and Y. Wang, "Evaluating quantized large language models," 2024.

[9] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14, 2014.

[10] M. He, C. Song, I. Kim, C. Jeong, S. Kim, I. Park, M. Thottethodi, and T. N. Vijaykumar, "Newton: A dram-maker's accelerator-in-memory (aim) architecture for machine learning," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 372–385, 2020.

[11] J. Choi, J. Park, K. Kyung, N. S. Kim, and J. H. Ahn, "Unleashing the potential of pim: Accelerating large batched inference of transformer-based generative models," *IEEE Computer Architecture Letters*, vol. 22, no. 2, pp. 113–116, 2023.

[12] S.-S. Park, K. Kim, J. So, J. Jung, J. Lee, K. Woo, N. Kim, Y. Lee, H. Kim, Y. Kwon, J. Kim, J. Lee, Y. Cho, Y. Tai, J. Cho, H. Song, J. H. Ahn, and N. S. Kim, "An lpddr-based cxl-pnm platform for tco-efficient inference of transformer-based large language models," in *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 970–982, 2024.

[13] J. Gómez-Luna, I. E. Hajj, I. Fernandez, C. Giannoula, G. F. Oliveira, and O. Mutlu, "Benchmarking a new paradigm: Experimental analysis and characterization of a real processing-in-memory system," *IEEE Access*, vol. 10, pp. 52565–52608, 2022.

[14] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "Smoothquant: accurate and efficient post-training quantization for large language models," in *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, JMLR.org, 2023.

[15] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han, "Awq: Activation-aware weight quantization for llm compression and acceleration," in *Proceedings of Machine Learning and Systems (MLSys)*, 2024.

[16] Y. Zhao, C.-Y. Lin, K. Zhu, Z. Ye, L. Chen, S. Zheng, L. Ceze, A. Krishnamurthy, T. Chen, and B. Kasikci, "Atom: Low-bit quantization for efficient and accurate llm serving," in *Proceedings of Machine Learning and Systems* (P. Gibbons, G. Pekhimenko, and C. D. Sa, eds.), vol. 6, pp. 196–209, 2024.

[17] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," 2016.

[18] D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández, "The LAMBADA dataset: Word prediction requiring a broad discourse context," June 2016.

[19] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try ARC, the AI2 reasoning challenge," 2018.

[20] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi, "PIQA: Reasoning about physical commonsense in natural language," 2019.

[21] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "BoolQ: Exploring the surprising difficulty of natural yes/no questions," 2019.

[22] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, *et al.*, "Llama 2: Open foundation and fine-tuned chat models," 2023.

[23] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, *et al.*, "The llama 3 herd of models," 2024.