KVSort: Drastically Improving LLM Inference Performance via KV Cache Compression

Baixi Sun[1], Dr. Xiaodong Yu[2]
Dr. Dingwen Tao[3], Dr. Fengguang Song[1]

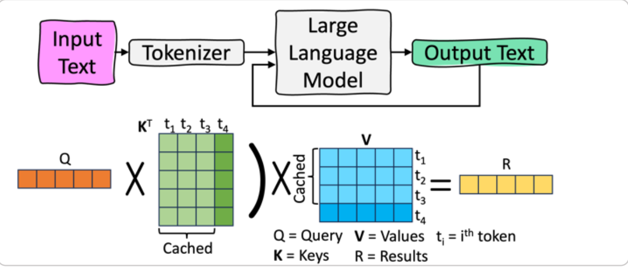
[1]Indiana University Bloomington, [2]Stevens Institute of Technology,
[3]Institute of Computing Technology, Chinese Academy of Sciences

INDIANA UNIVERSITY
BLOOMINGTON

Background & Motivation

Large Language Model (LLM) Inference

- **Prefill:** Tokenize input text, compute key and value vectors.
- **Decode:** Generates output tokens autoregressively.
- **KV Cache** improves LLM inference performance via avoiding repeated computation.



KV Cache Size is Enormous.

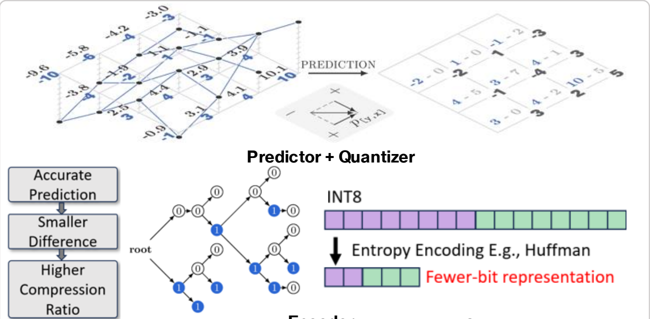
Model Name	Weight Size (GB)	KV Cache Size (GB)
Llama-3-8B	16	69
MPT-30B	60	206
Llama-3-70B	140	344
BLOOM-176B	352	526

Weight size and KV cache size in GB on existing LLMs using bf16 precision, context length 2048 and inference batch size 32.

Problem 1: KV Cache Size Exceeds GPU memory size.
Problem 2: Reading KV Cache from storage is slow.

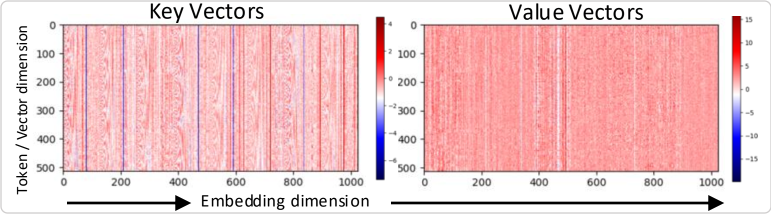
Error-bounded Lossy Compression

- **[Lossless] Predictor**: Use surrounding points for prediction and store difference.
- **[Lossy] Quantizer**: Quantizes a floating value to represent using fewer bits.
- **[Lossless] Encoder**: Further reduces the bits to represent via merging repetitions.



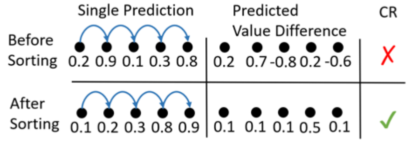
Credits to Jiannan Tian.

KVSort Design

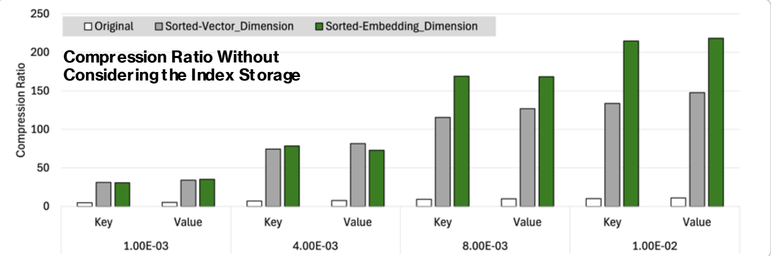


Sorting Drastically Improves Compression Ratio (CR)

- **Observation 1**: There are similar but discrete distributed values along the embedding dimension.
- **Implication 1.1**: Sorting Improves CR.
- **Implication 1.2**: Sorting on the embedding dimension achieves higher CR compared to the Token dimension.

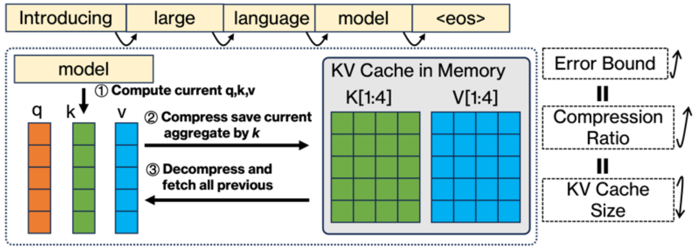


This is because sorting improves the effectiveness of prediction, reducing the prediction – original value difference. As a result, the encoder can use fewer number of bits to represent quantized value.



Challenge of Applying Sorting: Index storage requires additional bits.
For example, an 512x2048 bf16 tensor requires *at least 20 bits* for index after sorting.

Design 1: Sort the embedding dimension only, according to Implication 1.2.
Observation 2: Attention block's keys and values within a model shares the pattern shown above.
Design 2: Sorted index sharing across tokens. Specifically, we sort the first block and record the sorted index and re-order the next *k* tokens.
Design 3: Integrate compression/decompression into inference workflow.



Evaluation

- **Environment:** AMD EPYC 7742 CPU, 40GB NVIDIA A100 GPU, and 256GB RAM.
- **Baseline**: Q-Hitter [1] is the state-of-the-art KV cache compression approach that prunes the less important KV vectors.
- **Model & Dataset:** Llama3-8B-Instruct [2] on GSM8K [3].

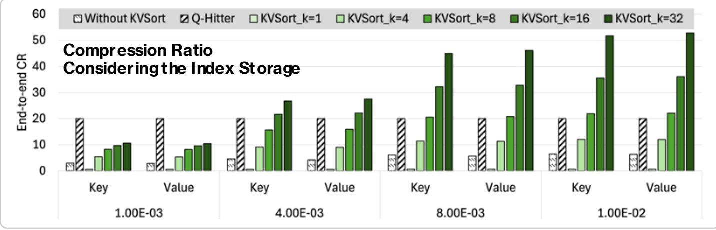
Inference Accuracy

Increasing error-bound up to 1E-2 does not affect inference accuracy.

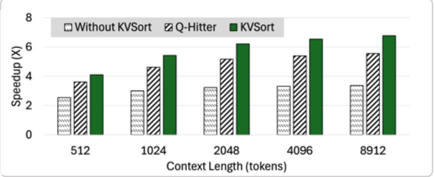
Error Bound	No compression	1.00E-03	4.00E-03	8.00E-03	1.00E-02	3.00E-02	6.25E-02	1.00E-01	2.50E-01	5.00E-01
Accuracy	76.10350076	**77.16895**	**77.01674277**	**75.190259**	**75.723**	70.16743	23.21157	0	0	0

Compression Ratios When Preserving Accuracy

1. Increasing the error bound improves compression ratio.
2. Significant Index storage overhead without index sharing (*k*=1).
3. Up to **52x** CR compared to Q-Hitter 20x CR.



End-to-End Performance Improvement



Using off-load mode
1. Up to **6.8x** speedup.
2. **Scaling** with context length.

Future Work

Apply KVSort to Purne-based KV Cache Compression Approaches: Attention block's keys and values within a model shares the pattern shown above.
Dynamic Index Sharing Mechanism: For error-bounded lossy compressors, the compression ratio can be estimated efficiently via sampling [4]. This enables KVSort to online determine the parameter *k* to share the sorted index. KVSort will increase *k* until the estimated compression ratio significantly drops.
System Design for Long Context Length: Efficiently utilize multi-layer storage (e.g., CPU RAM, GPU HBM, and SSD) to manage KVSort algorithm and develop an end-to-end LLM inference serving system.

Reference

[1] Zhang, Zhenyu, et al. "Q-Hitter: A Better Token Oracle for Efficient LLM Inference via Sparse-Quantized KV Cache." Proceedings of Machine Learning and Systems 6 (2024): 381-394.
[2] AI@Meta, "Llama 3 model card," 2024. [Online].
Available: https://github.com/meta-llama/llama3/blob/main/MODELCARD.md.
[3] Cobbe, Karl, et al. "Training verifiers to solve math word problems." arXiv preprint arXiv:2110.14168 (2021).
[4] Sian Jin, Sheng Di, Jiannan Tian, Suren Byna, Dingwen Tao, and Franck Cappello. "Significantly Improving Prediction-Based Lossy Compression Via Ratio-Quality Modeling." IEEE International Conference on Data Engineering 2022, (Virtual) Kuala Lumpur, Malaysia, May 9–May 12, 2022..