SAMSUNG

Optimizing RAG Inference efficiency using CXL Memory Expander

Raghu Vamsi Krishna Talanki
Associate Director
Samsung Semiconductor India Research

FMS

Contributors

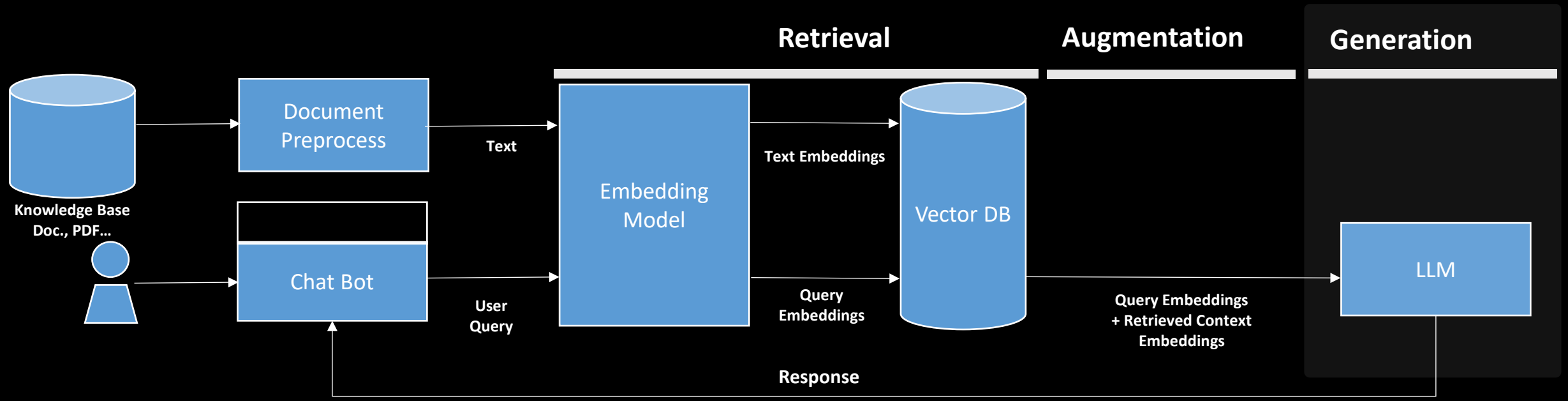
- Shubham S. Deshmukh – Staff Engineer, SSIR
- Raghu Vamsi Krishna Talanki – Associate Director, SSIR
- Keerthi Kiran J – Director, SSIR
- Rajeev Verma – Senior Director, SSIR
- Vishnu Charan T – Director, SSIR
- Jehoon Park – Engineer, Samsung, Korea
- Junhyeok Im – Staff Engineer, Samsung, Korea
- JinIn So – Senior Director, Samsung, Korea

Agenda

- Challenges in RAG Acceleration
- LLM Inference Acceleration using CXL
- Optimizing RAG with CXL
- Optimization Scenarios and results

Retrieval Augmented Generation (RAG)

- Retrieval: Embedding and Context Retrieval from data source and Vector DB
- Augmentation : Combines User Query with Context retrieved from Vector DB
- Generation: Response generation with query and context from LLM

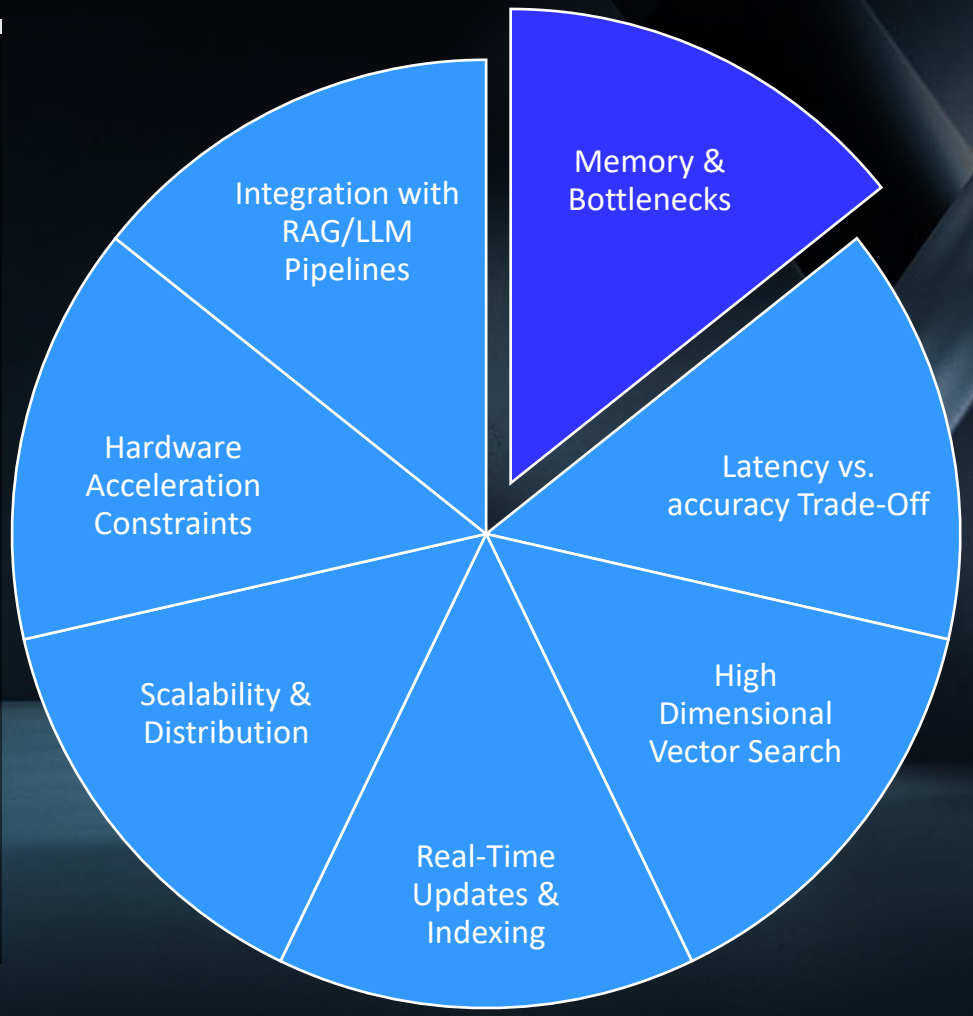


Challenges in RAG Acceleration

- Large Scale Vector DBs (Millions to Billions of Vectors) consume huge memory
- Search in high-dimensional space (e.g, 768 to 2048 dimensions for embeddings) requires more memory to store subsequent data.

Offloading is needed to make the main memory free by optimal placement of data (hot & cold)

Fast Access Time



SAMSUNG

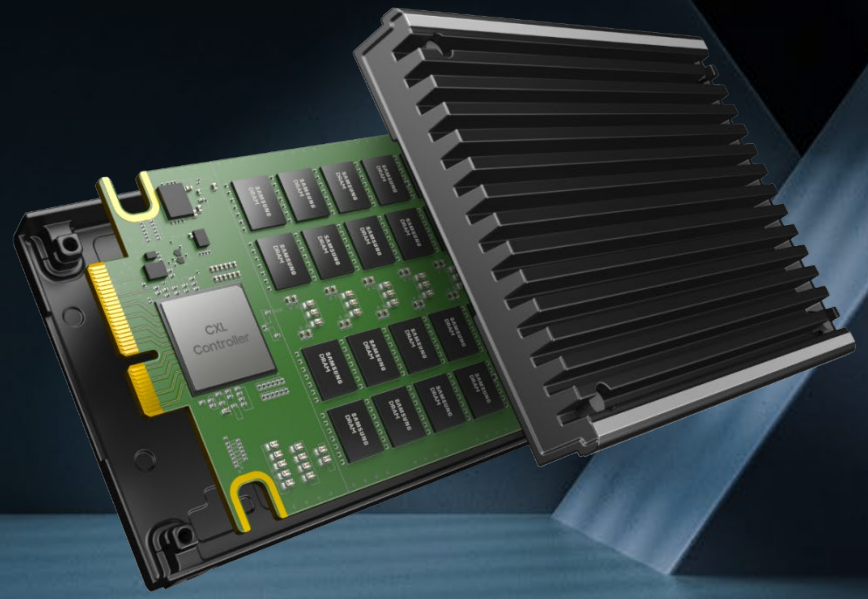
CXL Memory Expander

Key Features

- Seamless Memory Expansion without Architectural changes
- Reduce DDR memory Pressure
- Memory Pooling and sharing
- Enhance AI/ML model Scalability
- Minimize Latency Between CPU and Device

CXL for RAG

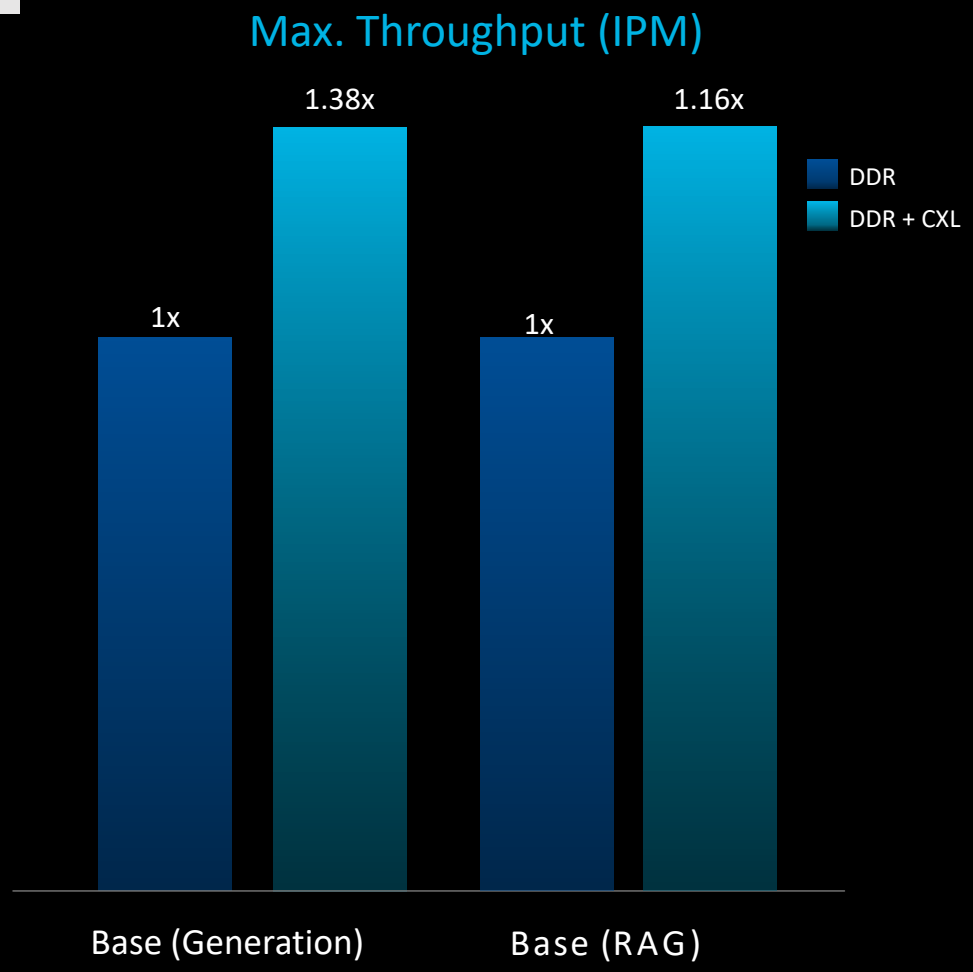
- Memory extension to Vector DB and LLM for Retrieval and Generation Acceleration



FMS

Problem Definition

- Default memory allocation method (page level interleaving)
  - Capacity
  - Latency
  - Bandwidth
- No importance to data based on
  - Access (hot or cold)
  - Memory footprint



Proposed Solution: Optimal Data and Load Distribution using Static Profiling

SAMSUNG

System Profiling

- CPU & Memory Utilization
- DRAM & CXL Bandwidth

Application Profiling

- Access & Memory Level
  - Function Level
  - Weight Level
  - Parameter Level

Algorithm

Input: Meminfo (BT, Inst), totMem, totBW, totCPU, memThres, BWThres, max Sat

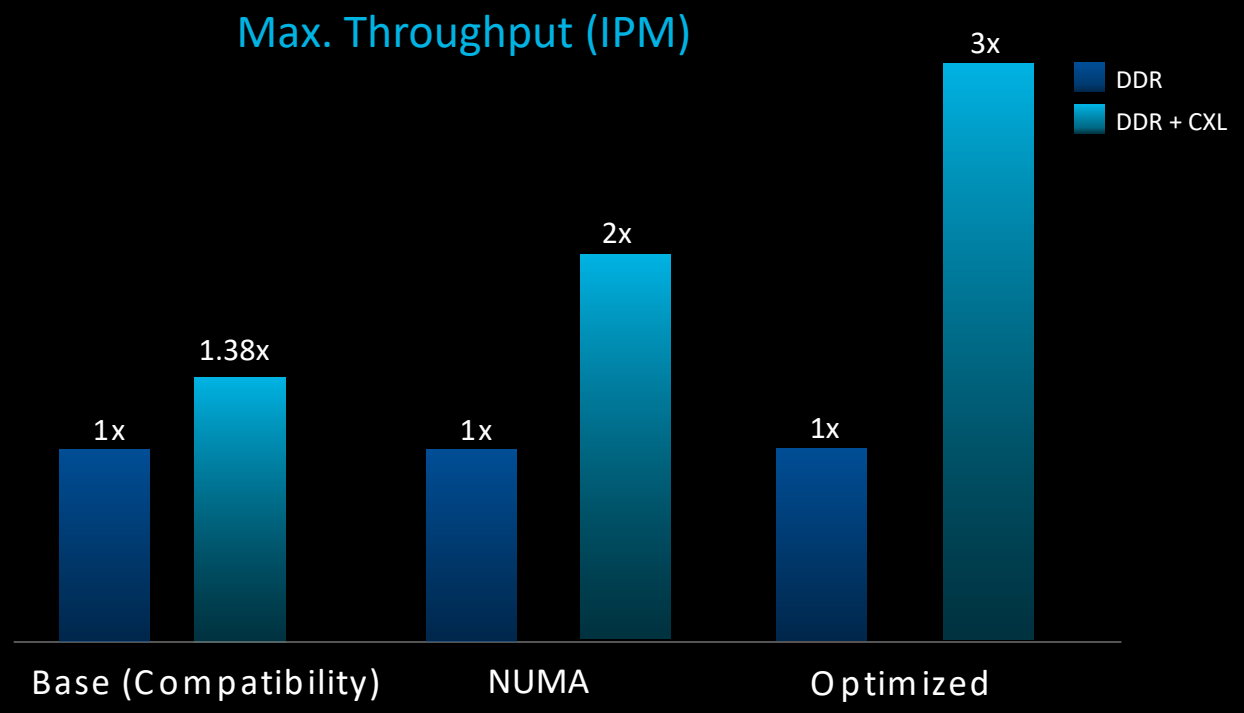
Output: OptimBT, OptimInst

1. Find Optimal batch size using binary search
    1. For every batch size find optimal instance sizeusing binary search
        1. Check whether saturation of CPU, DDR and CXL is reached
        2. If not change instance size
    2. Change batch size

FMS

Inference Optimization using CMM-D (CXL)

Scenario		Batch Size	No. of Parallel Instances	Cores per instance
Compatibility	DDR (OOM)	764	1	32
	DDR+CXL	764	1	32
NUMA	DDR	525	4	8
	DDR+CXL	675	6	10
Optimized	DDR	6	9	8
	DDR+CXL	24	16	6



Model:

IMDB reviews
LLM: GPT-2 Large, 774M FP32 params,
GPT2-Medium, 117M parameters

Environment:

- CPU: Intel Sapphire Rapids (96C)
- DDR: 128 GB CXL: 128G
- OS: Ubuntu 22.04, Kernel: 6.0.0-rc6-smdk

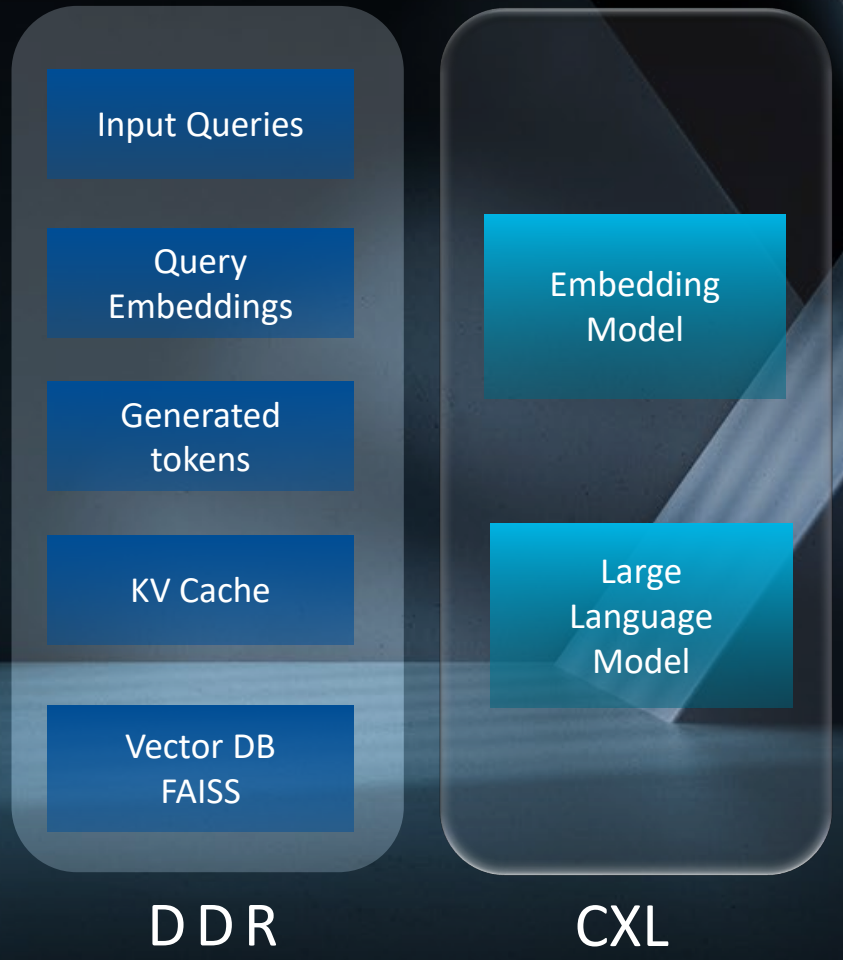
S. S. Deshmukh *et al*., "Optimizing Transformer Based Inference Efficiency Using CMM-D," *2024 IEEE 21st India Council International Conference (INDICON)*, Kharagpur, India, 2024, pp. 1-8, doi: 10.1109/INDICON63790.2024.10958315.

Optimizing RAG with CXL

SAMSUNG

- Objective: Offload cold embeddings to CXL memory and Keep hot embeddings in DRAM.

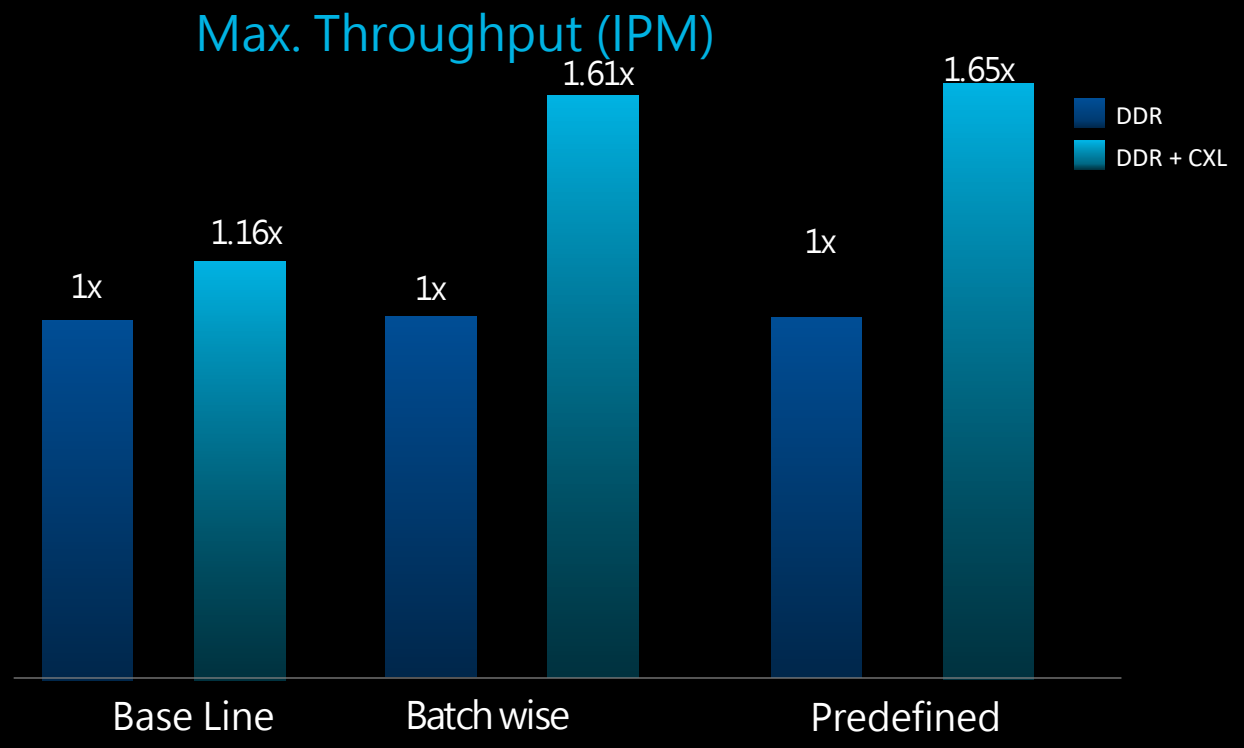
1. **Batch-wise:** Vector DB which gets updated based on the inference output
    - **Optimized Updated index**
    - **Scalability**
    - **Improved Throughput and efficiency**
2. **Pre-defined: Vector DB with pre-computed indexing**
    - **Pre-computed Indexing**
    - **Batch Query Processing**
    - **Enhanced Performance**



Optimizing RAG Inference using CMM-D (CXL)

SAMSUNG

Scenario		Batch size	No. of Parallel Instances	Sequence Length	Cores per instance
Base Line	DDR	1550	1	48	144
	DDR+CXL	1800	1	48	144
Batch wise	DDR	32	9	128	16
	DDR+CXL	54	12	128	12
Pre Defined	DDR	36	6	128	24
	DDR+CXL	48	8	128	18



Model:

multi-qa-mpnet-base-dot-v1,
LLM: GPT-2 Large, 774M FP32 params,
VectorDB index - FAISS flat
Data: sources: github and huggingface

Environment:

- CPU: Intel Granite Rapids (144C)
- DDR: 128 GB CXL: 128G
- OS: Ubuntu 22.04, Kernel: 6.0.0-rc6-smdk

FMS

Conclusion & Future Work

- CXL Bridges DDR Memory Bottlenecks
- Scalable inference via Hybrid Memory architecture
- Optimize Memory and CPU utilization
- Improve Inference efficiency
- Dynamic placement of data based on CHMU

SAMSUNG

Q&A

FMS

SAMSUNG

Thank you

FMS