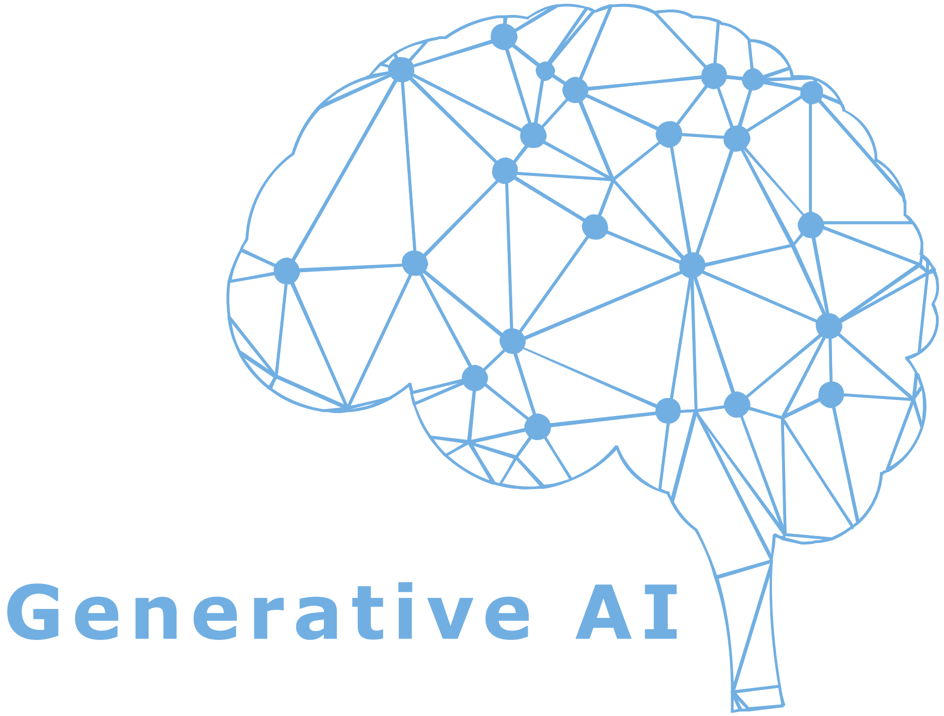
Computational Storage Drive for LLM

Sebastien Jean, Phison Electronics, CTO

Adoption Constraints for On-Premise Generative AI



Fine-Tuning

- Rapid growth in model size
- Insufficient memory capacity
- Difficulty in scaling
- High machine costs slows adoption

Inference

- Insufficient memory for tokens
- Limited context for chat and prompt
- Slow responses hurt user experience

Making on-premise affordable

WHAT?

1. Make **Edge AI affordable** to a larger market → your hardware, your control
2. SMB: Offer edge ***training + inference*** to businesses as workstation or small server
3. Education: Support **AI access** for **university** professors, classes, labs and students
   (ie: wait time reduced from weeks → hours)

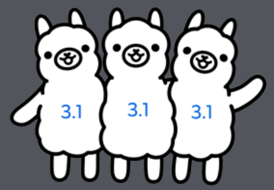
HOW?

a. Decouple: DRAM, Compute and Model Size

b. Scale each item independently to match need and budget

c. aiDaptiv+ beyond RAM limits → Fine-Tune, Model Streaming, KV Cache & Context Window

Reducing Post-training RAM with Flash offload



Llama-3.1 70B Training

1.4TB memory required

Data Data Data Data Data Data Data Data
Data Data Data Data Data Data Data Data
Data Data Data Data Data Data Data Data
Data Data Data Data Data Data

Data cut into slices

Traditional Architecture

Limited by GPU Memory

30 GPU to train Llama-2 (70B)

(Requires 8 Workstations and Network Infrastructure)

Node 1
CPU
HBM Data HBM Data
HBM Data HBM Data

Node 2
CPU
HBM Data HBM Data
HBM Data HBM Data

Node 3
CPU
HBM Data HBM Data
HBM Data HBM Data

• • •

Node 6
CPU
HBM Data HBM Data
HBM Data HBM Data

Node 7
CPU
HBM Data HBM Data
HBM Data HBM Data

Node 8
CPU
HBM Data HBM Data
HBM Idle HBM Idle

Note: Assumes 48GB / GPU

aiDAPTIV+ Architecture

Flexible Model Size

4 GPU to train Llama-2 (70B)

(Requires 1 Workstation)

Phison aiDAPTIV+ AI
1.4 kW

CPU

HBM Data HBM Data HBM Data HBM Data

"Extend GPU Memory"

aiDAPTIV Cache M.2 SSD
2x 2TB w/ 100 DWPD

Data Data Data Data Data Data Data Data
Data Data Data Data Data Data Data Data
Data Data Data Data Data Data Data Data

PENDING SLICES

Phison aiDAPTIV+ solution

AI Application

PyTorch
AI Framework

aiDAPTIVLink
Middleware Library

GPU

aiDAPTIVCache
Specialized SSD

aiDAPTIVLink

Middleware

Coordinates the swapping between HBM/DRAM and Flash Memory

aiDAPTIVCache

AI-Series SSD Family

Seamless Integration with VRAM/DRAM

E28 – AI Optimized Gen5x4 SSD, 6nm w/ DSP

aiDAPTIV+ Value Propositions

1. Move low complexity update task off the GPU and eliminate extra PCIe hop to CPU
2. Improve pipeline performance by freeing up GPU sooner
3. 40% improvement over other GPU+CPU and GPU+CPU+NAND solutions

Key Technologies

1. Integrate advanced math engine DSP directly into E28 controller
2. Enhanced LDPC engine to support greater throughput
3. Move to 6nm to reduce power requirements

Fine-tune DSP Concept

Basic training flow Total # of time ticks: 30 ← Linear processing on the GPU

	Batch 1	Batch 2	Batch 3
GPU Time	F F F B B B U U U U	F F F B B B U U U U	F F F B B B U U U U

With aiDAPTIVCache 2.0 DPS Total # of Ticks: 18 **Post Training Improved by 40% !**

	Batch 1	Batch 2	Batch 3
GPU Time	F F F B B B	F F F B B B	F F F B B B
DSP Time		U U U U	U U U U

← Parallel processing on E28 DSP, releases GPU 12 ticks earlier

F Forward Propagation B Backward Propagation U Update

Performance Results w/ Model Streaming (tokens/sec)

Basic DSP Fine-tune (+86% ~ +146%)

- Error reduction is applied on every gradient

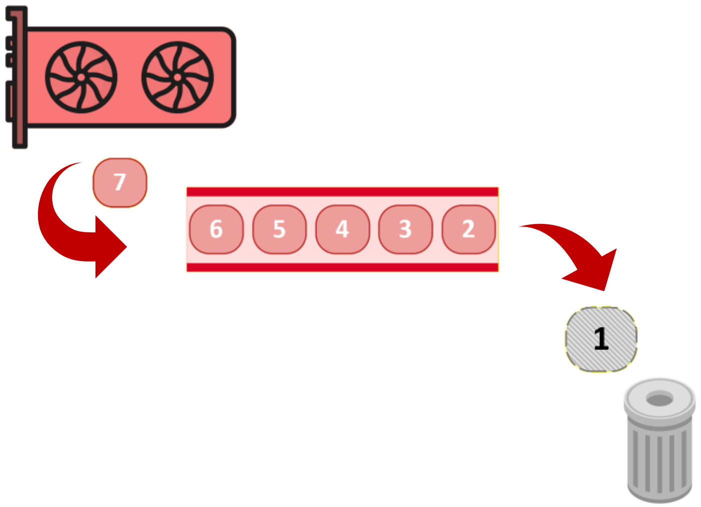
Model	Llama2-7B	Llama2-13B	Llama-70B	CodeLlama-70B	Falcon-180B
aiDAPTIV	4,338	2,717	388	403	108
aiDAPTIV w/ DSP	9,634	5,062	954	949	217
Improvement	122%	86%	146%	135%	102%

Efficient DSP Fine-tune (+50% ~ +87%)

- Error reduction is updated every 4 gradients

Model	Llama2-7B	Llama2-13B	Llama-70B	CodeLlama-70B	Falcon-180B
aiDAPTIV	5,750	3,393	519	524	138
aiDAPTIV w/ DSP	9,727	5,098	971	964	247
Improvement	69%	50%	87%	84%	78%

Faster inference after pre-fill, bigger context window

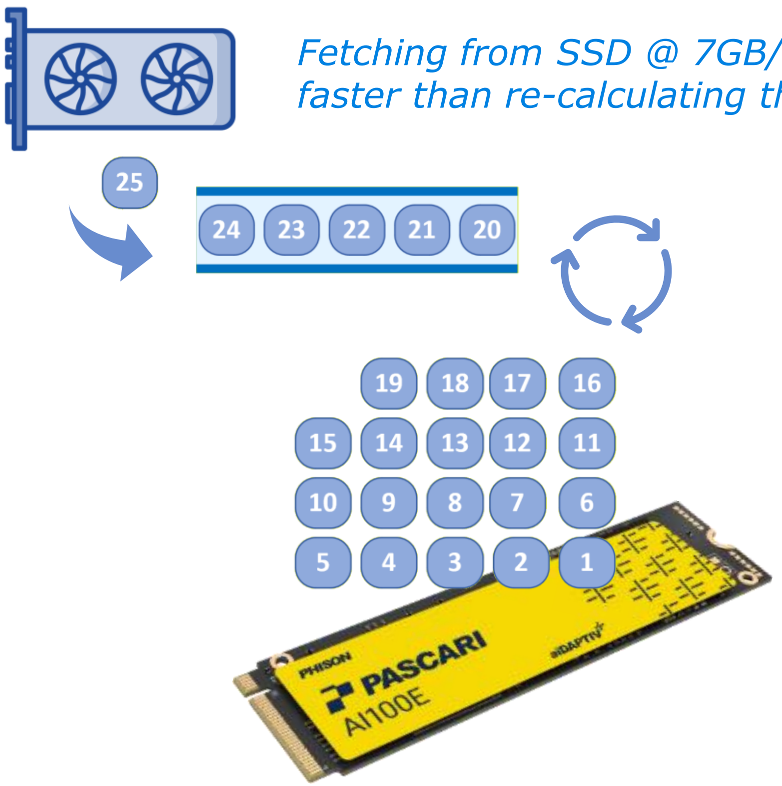


When the KV Cache runs out of room, the old entry is evicted.

If it is needed later, it must be re-calculated, which is surprisingly slow.

With aiDaptiv$^+$, the evicted entries are moved to the SSD.

Fetching from SSD @ 7GB/s is still faster than re-calculating the value.



Performance Results

Test Configuration

- GPU: H200 w/ 141 GB HMB
- Model: Qwen 2.5 32B
- Input/Output: 4000 / 200 token
- Parallel user: 220
- Vllm version: 0.8.3 (v0 engine)

Observations

1. Orange blocks mean reuse performance is better than recalculating
2. Hit Rate is the ratio of KV Cache entry reuse vs recalculating
3. Data Points
   - Memory: 32GB (model) + 109 GB (KV Cache) ← Full
   - SSD BW: Gen4 x 4 = 7 GB/s
   - PCIe DDR BW: Gen4 x 16 = 28 GB/s

Even though DDR BW over PCIe is > SSD BW, this is not the bottleneck.

Benefits

- Support longer context and more user per GPU
- Archive summarize and recall past conversation

Hit Rate	100%	75%	50%	25%
Recompute w/o cache (baseline)	**Output: 464 tok/s** TTFT Mean: 2,0134 ms TPOT Mean: 137 ms			
aiDAPTIV+ SSD	**624 tok/s** TTFT 344 ms TPOT 101 ms	**605 tok/s** TTFT 611 ms TPOT 108 ms	**532 tok/s** TTFT 1,152 ms TPOT 121 ms	**462 tok/s** TTFT 2,185 ms TPOT 138 ms
aiDAPTIV+ DRAM	**627 tok/s** TTFT 344 ms TPOT 101 ms	**608 tok/s** TTFT 601 ms TPOT 108 ms	**531 tok/s** TTFT 1,141 ms TPOT 121 ms	**468 tok/s** TTFT 2,151 ms TPOT 136 ms
LMcache SSD	**342 tok/s** TTFT 4,155 ms TPOT 195 ms	**261 tok/s** TTFT 5,713 ms TPOT 262 ms	**263 tok/s** TTFT 5,628 ms TPOT 258 ms	**273 tok/s** TTFT 5,909 ms TPOT 246 ms
LMcache DRAM	**408 tok/s** TTFT 2172 ms TPOT 164 ms	**380 tok/s** TTFT 3,483 ms TPOT 175 ms	**365 tok/s** TTFT 3,884 ms TPOT 179 ms	**364 tok/s** TTFT 3,717 ms TPOT 182 ms
GPU HMB (optimal)	**781 tok/s** TTFT 146 ms TPOT 71 ms	**668 tok/s** TTFT 302 ms TPOT 89 ms	**615 tok/s** TTFT 849 ms TPOT 105 ms	**514 tok/s** TTFT 1,736 ms TPOT 123 ms

Thank you

aiDAPTIV+

- Faster offload fine-tune
- Inference larger models
- Bigger and faster KV Cache
- Accessible & Data Privacy

Learn More