When Generative AI Meets the Edge:

Learning from Real-world Interactions

RMIT UNIVERSITY Ready for what's next

When Generative AI Meets the Edge:
Learning from Real-world Interactions

Keynote Speaker:
A/Prof. Wei Peng
Principal Research Fellow in AI

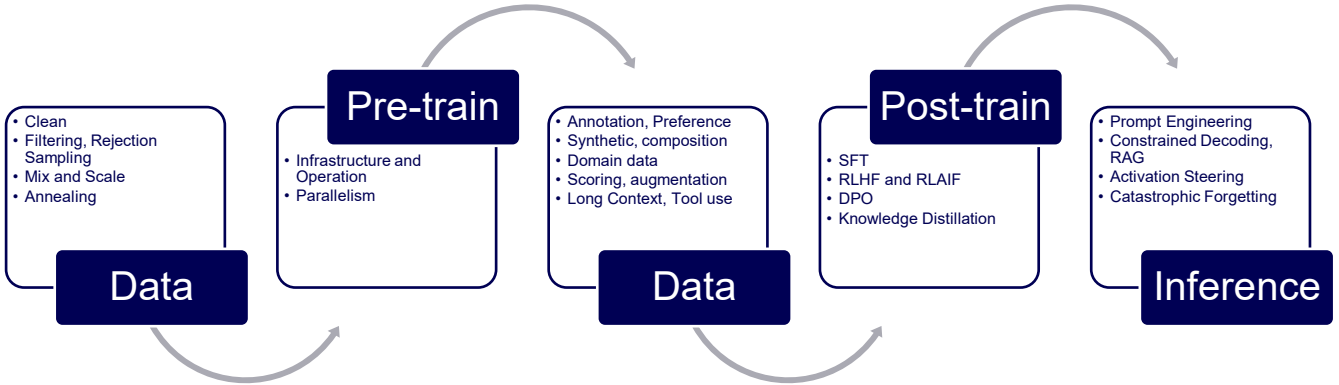
School of Engineering
RMIT University

Overview

- Learning Pathway
- Towards more Generalizable LLM-based Agency: Agent Trajectory Learning
- A Case Study of Reasoning on Edge Device (Nvidia Jetson Orin Nano Super)
- Current Works

Bifurcated Pathway to AGI (Data-driven)

1. Human knowledge build (Cybernetics, KBS, to Data and Computing Power, LLM/scale law/transient learning on features for tasks)



try to scale model in development to host all human knowledge and capability by feeding mega data …

Model	Tokens Processed	GPU Hours	GPU Type	Assumed Cost ($2/hr)
DeepSeek V3 (671B)	~14.8 T	2.788 M	H800	~$5.576 M
Llama 3.1 405B	~15.6 T	~30.84 M	H100	~$61.68 M

https://github.com/deepseek-ai/DeepSeek-V3 https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/MODEL_CARD.md

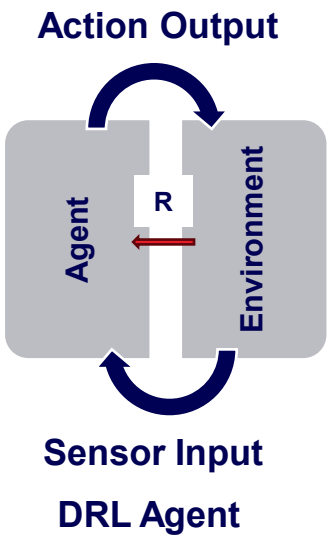
Bifurcated Pathway to AGI

2. Continual learning from interactions with physical world

Rich Sutton's new path for AI : "… RL in AI, we don't have methods to learn continuously except for the linear case …" https://www.youtube.com/watch?v=NvfK1TkXmOQ

Fei-Fei Li "Language is fundamentally a purely generated signal… you don't go out in nature and there's words written in the sky for you…" https://x.com/a16z/status/1989078942645645435

Yann LeCun "We won't reach AGI by scaling up LLMs …" https://www.youtube.com/watch?v=4__gg83s_Do



Learn from Interactions Challenges:

- Generalization: beyond the env. trained
- Inefficiency: learn from limited examples
- Catastrophic Forgetting: learn without forgetting priori knowledge
- Reward Misalignment: multi-objective, human values
- Lack World Model and Abstraction: reason on concept

Learning from Interactions (Embodied Intelligence)

- Rodney Brooks' Intelligence without Representation (Brooks, 1991): no traditional representation, intelligence from sensor motor interaction with the environment, behavior-based model

Take Aways: Learn from situated sensor motor coordination to generate complex behaviors

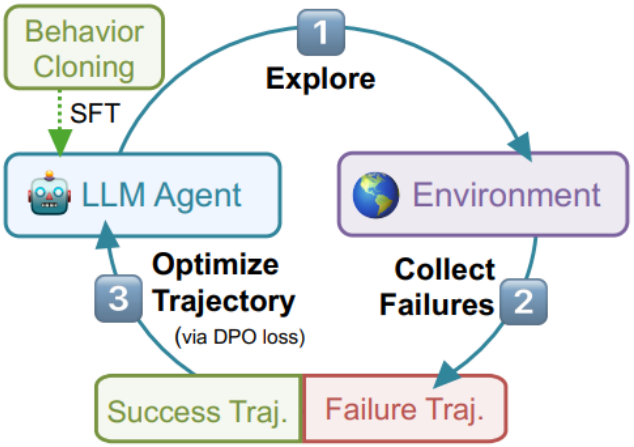
Challenges: Limited memory, planning, reasoning capability, simplistic world model

Key Insight: ground disembodied intelligence (i.e., LLM) in interactions to develop concepts (levels of abstractions) in self-organized manner

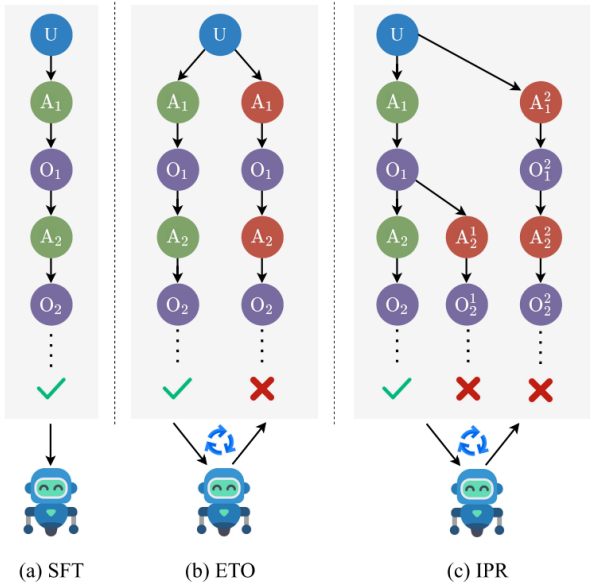
Trajectory Learning

➢ **Agent Learning from Interaction:**

Watch Every Step! LLM Agent Learning via Iterative Step-Level Process Refinement, In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024),* pages 1556-1572, Association for Computational Linguistics (citation 56)



ETO (Song et al., 2024)



➢ **Agents start to learn from Interactions and explorations: from SFT on trajectories to ETO (SAMOYED)**

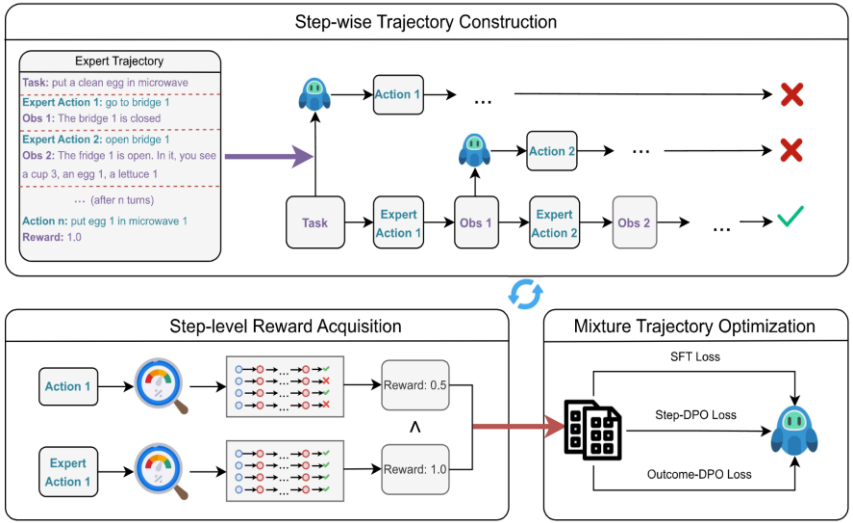
Treat an entire trajectory as single entity during training and prioritize the final reward of a trajectory over the process, thus overlooking exploitable information throughout interaction process.

We need to consider step level optimization

Trajectory Learning (cont.)

- **Agent Learning from Interaction:**

Watch Every Step! LLM Agent Learning via Iterative Step-Level Process Refinement, In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024)*, pages 1556-1572, Association for Computational Linguistics (citation 56)



Design Step-level Process Refinement: Step-level Reward Acquisition and Iterative Agent Optimization.

Monte Carlo method to estimate rewards via sampling N trajectories to construct step award.

$$\{e^{(i)}|i=1,...,N\} = MC^{\pi_s}(e_{t-1};N),$$

$$r_s(s_t,a_t) = \begin{cases} \frac{1}{N}\sum_{i=1}^{N} r_o(u,e^{(i)}), & \text{for } t<n \\ r_o(u,e_n), & \text{for } t=n \end{cases}$$

$$\mathcal{L} = \mathcal{L}_{\text{o-DPO}} + \mathcal{L}_{\text{s-DPO}} + \mathcal{L}_{\text{SFT}}$$

- Outcome-DPO Loss

$$\mathcal{L}_{\text{o-DPO}} = -\mathbb{E}_{(u,e_n^w,e_m^l)\sim\mathcal{D}_t}\left[\log\sigma(\beta\log\frac{\pi_\theta(e_n^w|u)}{\pi_{ref}(e_n^w|u)} - \beta\log\frac{\pi_\theta(e_m^l|u)}{\pi_{ref}(e_m^l|u)})\right],$$

- Step-DPO Loss

$$\mathcal{L}_{\text{s-DPO}} = -\mathbb{E}_{(e_{t-1},e_{t:n}^w,e_{t:m}^l)\sim\mathcal{D}_s}\left[\log\sigma(\beta\log\frac{\pi_\theta(e_{t:n}^w|e_{t-1})}{\pi_{ref}(e_{t:n}^w|e_{t-1})} - \beta\log\frac{\pi_\theta(e_{t:m}^l|e_{t-1})}{\pi_{ref}(e_{t:m}^l|e_{t-1})})\right],$$

- Supervised Loss

$$\mathcal{L}_{\text{SFT}} = -\mathbb{E}_{(u,e_n^w,e_m^l)\sim\mathcal{D}_t}\left[\log\pi_\theta(e_n^w|u)\right],$$

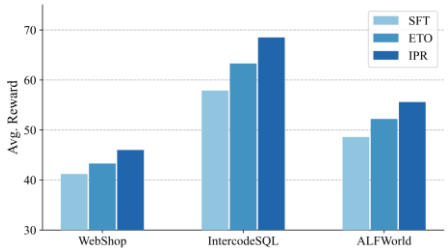
Trajectory Learning (cont.)

- **Agent Learning from Interaction:**

Watch Every Step! LLM Agent Learning via Iterative Step-Level Process Refinement, In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024),* pages 1556-1572, Association for Computational Linguistics (citation 56)

Paradigm	Models	WebShop	InterCodeSQL	ALFWorld		Average
				Seen	Unseen	
Prompt-based	GPT-4 (Achiam et al., 2023)	63.2	38.5	42.9	38.1	45.7
	GPT-3.5-Turbo (Ouyang et al., 2022)	62.4	37.8	7.9	10.5	29.7
	Llama-2-7B (Touvron et al., 2023)	17.9	4.0	0.0	0.0	5.5
Outcome Refinement	Llama-2-7B + SFT (Chen et al., 2023)	60.2	54.9	60.0	67.2	60.6
	Llama-2-7B + PPO (Schulman et al., 2017)	64.2	52.4	22.1	29.1	42.0
	Llama-2-7B + RFT (Yuan et al., 2023)	63.6	56.3	62.9	66.4	62.3
	Llama-2-7B + ETO (Song et al., 2024)	67.4	57.2	68.6	72.4	66.4
Process Refinement	Llama-2-7B + Step-PPO	64.0	60.2	65.7	69.4	64.8
	Llama-2-7B + IPR (ours)	**71.3**	**61.3**	**70.3**	**74.7**	**69.4**

Training Scheme	WebShop	InterCodeSQL	ALFWorld
w/o o-DPO	70.2	59.3	72.4
w/o s-DPO	66.4	58.0	70.2
w/o SFT	61.8	31.7	64.9
Iteration=1	63.6	56.6	68.7
Iteration=2	63.7	58.2	70.2
Iteration=3	68.2	59.2	**74.7**
Iteration=4	**71.3**	**61.3**	73.5
Iteration=5	68.1	57.9	71.4



Conclusion:

- Agent learns from interaction via trajectory with step awards
- Learning from failure actions
- Automated process reward acquisition
- Step level process supervision via mixture trajectory optimization
- Enhanced performance on three benchmarks
- Generalizable on unseen hold out

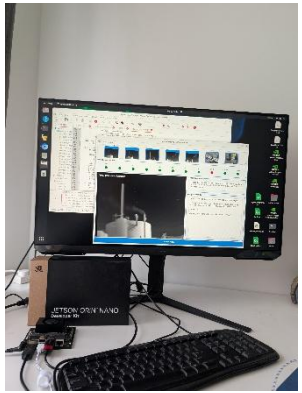
Limitation:

- Overfitting with limited data (need to leverage AgentBank data)
- MC method constrained by sample size
- Consider GPT 4 to label process supervision data
- 5.6 hours on 8 X A100

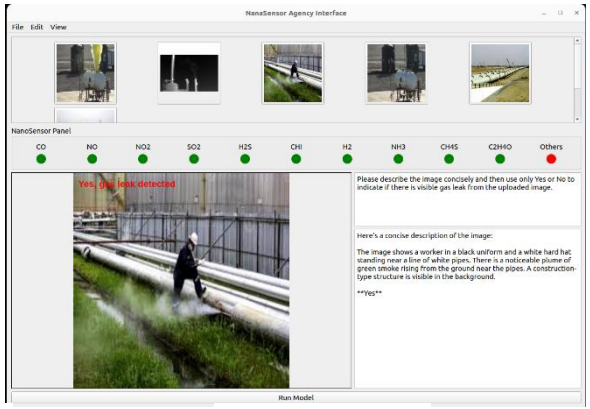
Case Study: Agent Reasons on a General Task on Edge Devices

Learning from Interactions Step 1: Pushing LLMs/VLMs to Edge Devices

Applying VLMs reasoning capability in real-time, on-device hazard detection



NVidia Jetson Orin Nano
(8GB, 5GB available RAM)



Gemma3:4b

Challenges：

- LLMs/VLMs not designed for continuous, low latency perception on constrained edge hardware (compute and memory)
- Heavy vision transformers, slower prefill, low decode throughput
- Painful Optimisation: careful configs (batch, context windows, flash attention) due to resource boundary
- Bottlenecks: memory, model architecture, inconsistent GPU utilisation, fragile memory and system behaviors

VLMs on Cloud vs Edge

Category	Edge (Jetson Orin Nano Super)	Cloud (A100/H100)
Compute Power (TOPS/TFLOPS)	~67 TOPs (INT 8)	~312 TFLOPS (FP 16), ~1979 TFLOPS (FP 16)
Memory	8 GB for small/quantised models (2-4B).	40-80 GB High bandwidth memory for Massive compute: large, accurate models (7-70B)
Latency	Very low, real-time, some network	Higher (network round trip + upload time).
Privacy	Data stays on device: strong privacy	Images/videos on cloud: compliance issues
Thermals & Power	Can throttle under load, must manage heat (7-25w)	Stable power/cooling, consistent performance (400-700W)
Model Quality	Lower accuracy, weaker reasoning/OCR	High accuracy, strong reasoning on vision
Scalability & Updates	Hard to update many devices, each device is different	Centralised updates, easy scaling

Case Study: Agent Reasons on a General Task on Edge Devices (cont.)

Initial Exploration: 3-Runs on resource analytics default on Ollama (4096 context length, temperature 0.7)

640 X 640 JEPG

Give a concise description of the picture in two sentences, is there a visible gas leak?

Prompt

Gemma3:4b

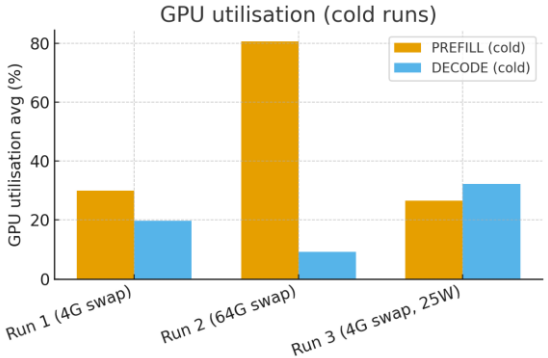
01 **Run 1** 4G Swap, 15W

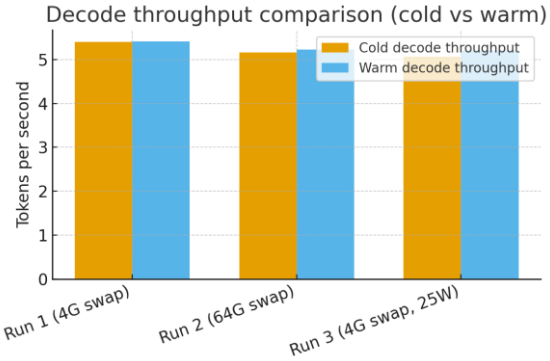
02 **Run 2** 64G Swap, 15W

03 **Run 3** 4G Swap, 25W

VLMs on Jetson Orin Nano (Ollama)

GPUs not well-utilised

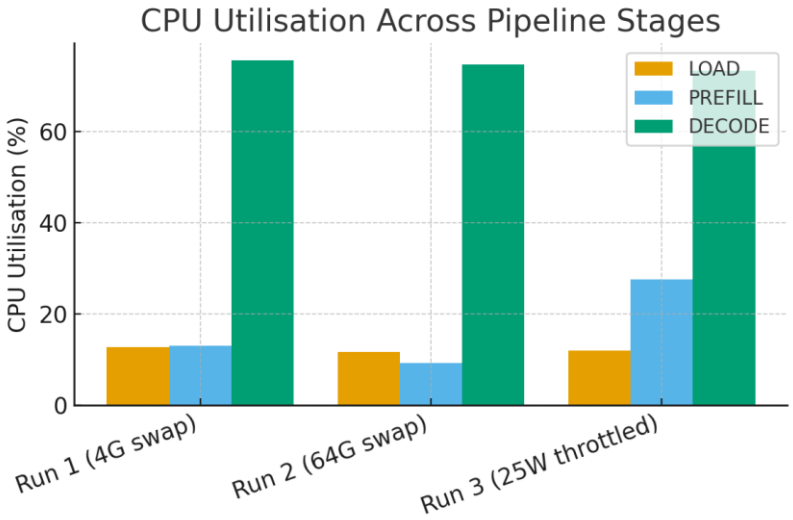


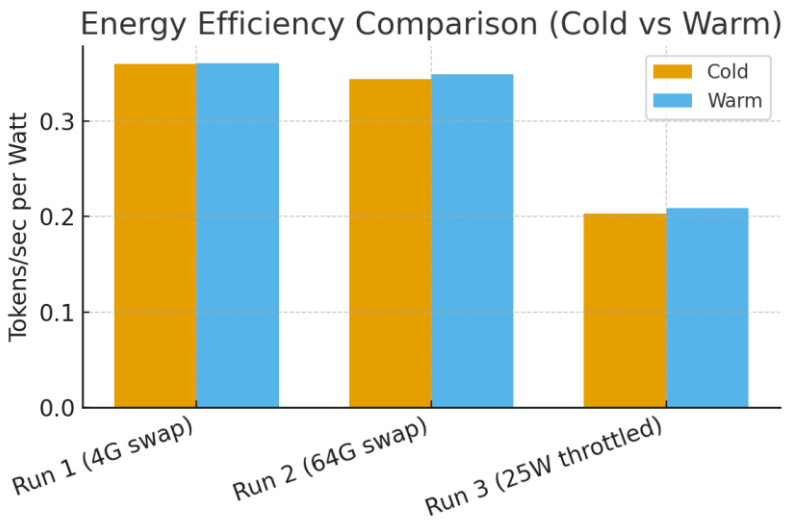


Run	GPULayers (final)	Model weights on GPU	KV cache on GPU	Compute graph on GPU	Total memory (CPU+GPU)
1	11 / 35	605 MB	86 MB	1.2 GB	5.0 GB
2	12 / 35	655.7 MB	92 MB	1.2 GB	5.0 GB
3	13 / 35	713.4 MB	98 MB	1.2 GB	5.0 GB

* Note: Jetson Orin Nano uses one shared one 8 GB memory pool, “on GPU” means GPU owns the allocation

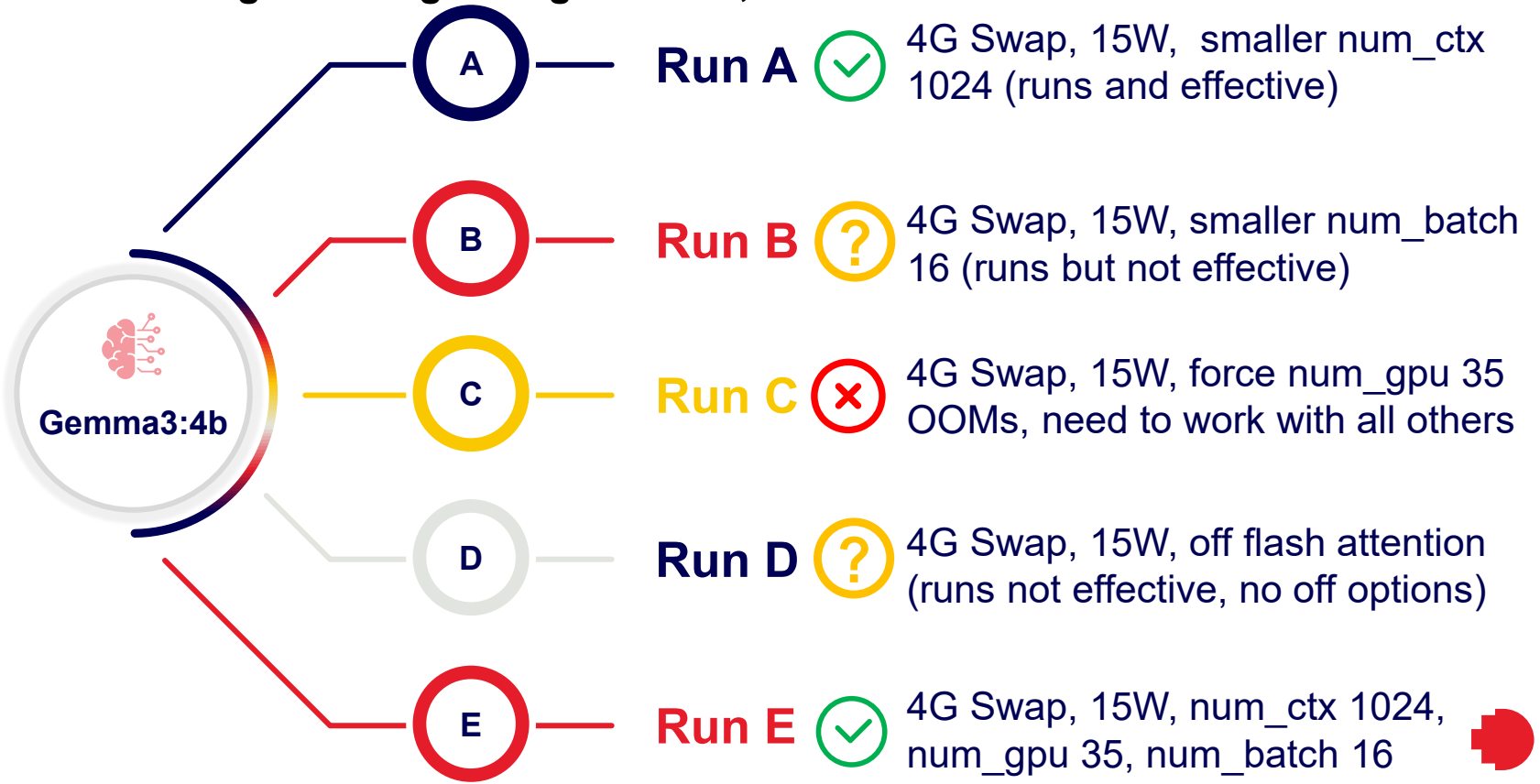
CPUs take the Load



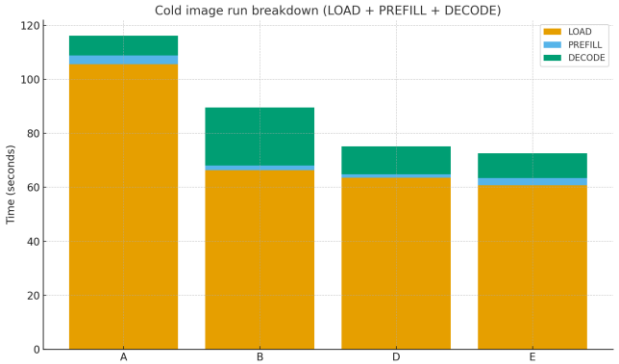


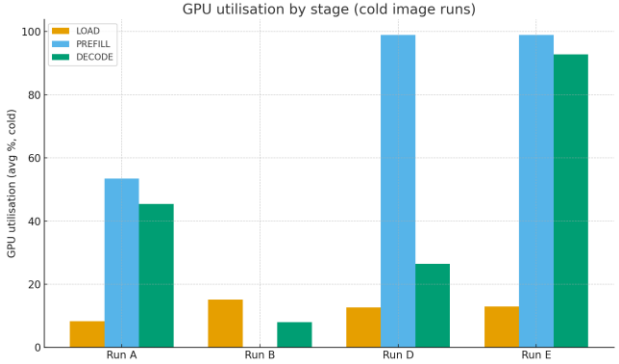
Boost GPU Utilisation

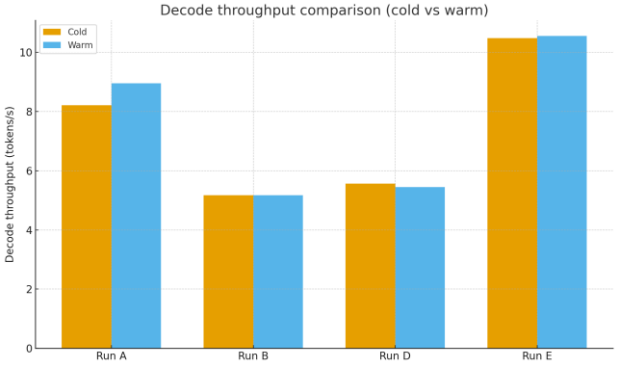
Temperature too high causing wrong answers, use 0.1



GPU Utilisation Enhanced (Run A and E)

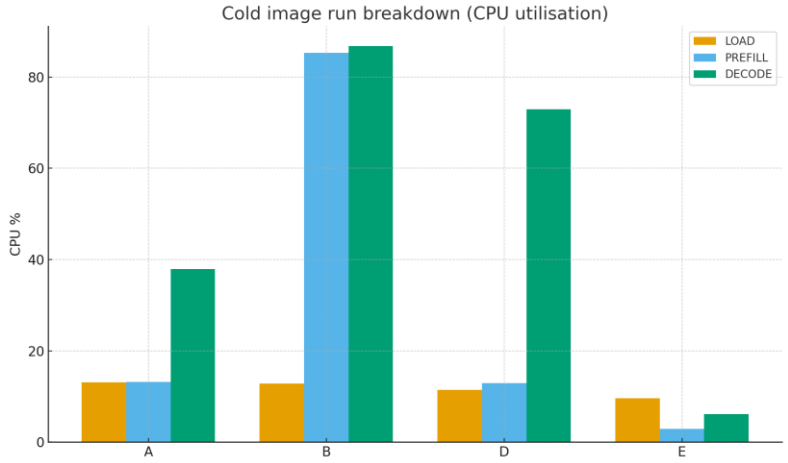


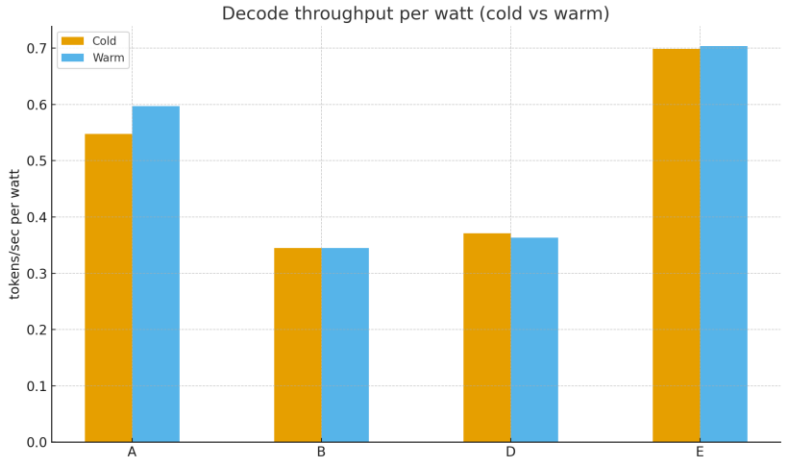




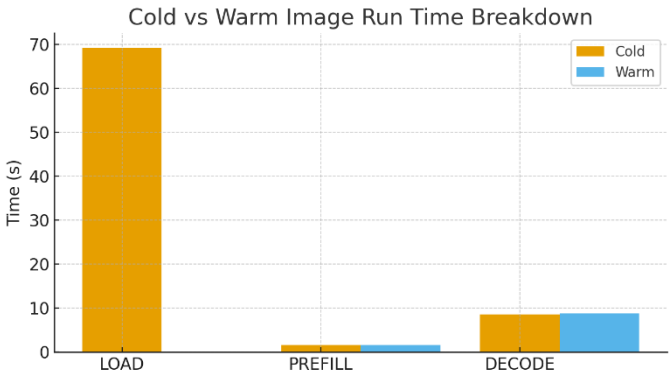
Run	GPULayers	Model weights (GPU)	KV cache (GPU)	Compute graph (GPU)	Total memory (CPU+GPU)
A	34 / 35	1.8 GB	214.0 MB	1.2 GB	5.0 GB
B	5 / 35	286.8 MB	28.0 MB	1.2 GB	5.0 GB
D	13 / 35	713.4 MB	82.0 MB	1.2 GB	5.0 GB
E	35 / 35	3.1 GB	185.0 MB	1.1 GB	5.0 GB

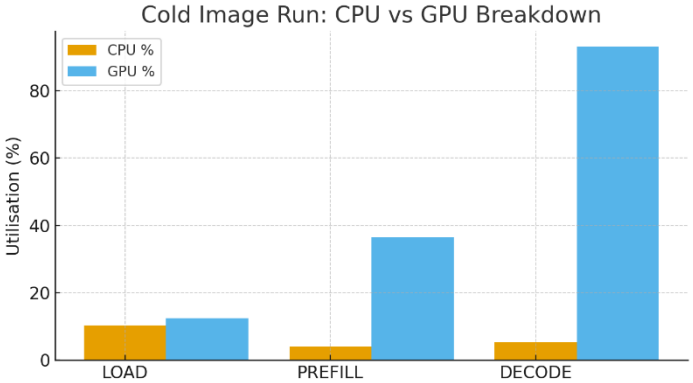
CPUs Reduced with Decode Throughput Increased (Run A and E）

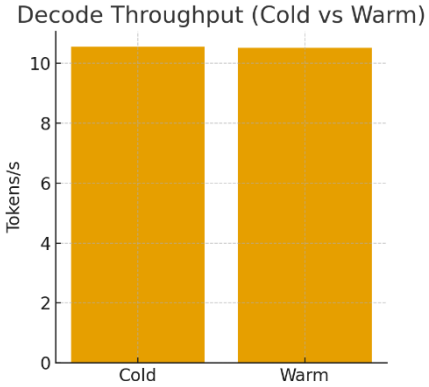




Update：512 context window, batch 16







GPULayers	Model weights (GPU)	Model weights (CPU)	KV cache (GPU)	Compute graph (GPU)	Compute graph (CPU)	Total memory (CPU+GPU)
35 / 35	3.1 GB	525.0 MB	185.0 MB	1.1 GB	2.5 MB	5.0 GB

Run	Total (s)	Load (s)	Prefill (s)	Decode (s)	Tokens	Decode throughput
Cold image	79.34	69.2	1.6	8.54	90	10.54
Warm image	10.34	–	1.6	8.74	92	10.52

Findings

- Load time is the **dominant latency** cost: Cold starts take 60-105 s, while warm PREFILL and DECODE are real-time (1-3 s PREFILL, 8-11 tokens/s decode).
- **Memory is the primary bottleneck on Jetson Orin Nano (8 GB):** Model weights + compute graph + KV cache + activation buffers frequently exceed available RAM, triggering allocator failures.
- Two clear GPU utilisation profiles appear across runs: CPU-heavy runs (B, D): high CPU usage (70-95%), very low GPU usage (0-10%); GPU-heavy runs (A, E): GPU PREFILL and DECODE near saturation (90-99%).
- **High refill batch size (default 512) is the main cause of CPU-heavy runs** Large activation memory causes CUDA OOM → Ollama offloads layers to CPU so PREFILL becomes CPU-bound

Findings

- **Large context** increases KV cache size; Larger num_ctx increases persistent VRAM use but is not the main reason for OOM. Large context + large batch together significantly increase VRAM pressure. Combined KV + activation memory exceeds VRAM triggers allocator backoff and CPU fallback.
- **Backoff mode reduces GPU residency**
  When CUDA OOM occurs, Ollama progressively drops GPULayers (e.g., 35 -> 34 -> 25 -> 15 -> 13), placing more layers and KV on CPU.

Current Works

- VLMs on edge dealing with snapshot image reasoning (Done)
- Design of memory-augmented agent learns from trajectory (Done)
- Multi-modal streaming fusion (in progress)
- Memory management and optimisation for edge devices (To Do)
- Deploy memory-augmented agent to edge to handle stream reasoning (To Do)
- Mobilise agent with ROS for real-world embodied general tasks (To Do)
- Safety, robustness, real-world evaluation (To Do)

Major Literature

- Rodney A. Brooks. 1991. Intelligence without Representation. *Artificial Intelligence,* 47:139- 159.
- Richard Sutton, 2024, New Path for AI: Approximately Correct Podcast, URL: https://www.youtube.com/watch?v=NvfK1TkXmOQ
- Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. ArXiv preprint, abs/2308.03688, 2023. URL https://arxiv.org/abs/2308.03688.
- Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. 2024. AgentTuning: Enabling Generalized Agent Abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3053–3077, Bangkok, Thailand. Association for Computational Linguistics.
- Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and Error: Exploration-Based Trajectory Optimization of LLM Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7584–7600, Bangkok, Thailand. Association for Computational Linguistics.
- Yifan Song, Weimin Xiong, Xiutian Zhao, Dawei Zhu, Wenhao Wu, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. 2024. AgentBank: Towards Generalized LLM Agents via Fine-Tuning on 50000+ Interaction Trajectories. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2124–2141, Miami, Florida, USA. Association for Computational Linguistics.
- Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. 2024. Watch Every Step! LLM Agent Learning via Iterative Step-level Process Refinement. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1556–1572, Miami, Florida, USA. Association for Computational Linguistics.

Thanks

Questions & discussion welcome.

Acknowledgement of Country

RMIT University acknowledges the people of the Woi wurrung and Boon wurrung language groups of the eastern Kulin Nation on whose unceded lands we conduct the business of the University.

RMIT University respectfully acknowledges their Ancestors and Elders, past and present.

RMIT also acknowledges the Traditional Custodians and their Ancestors of the lands and waters across Australia where we conduct our business.

Artwork ‘Sentient’ by Hollie Johnson
Hollie is a Gunaikurnai and Monero Ngarigo woman from Gippsland who graduated from RMIT with a BA in Photography in 2016.