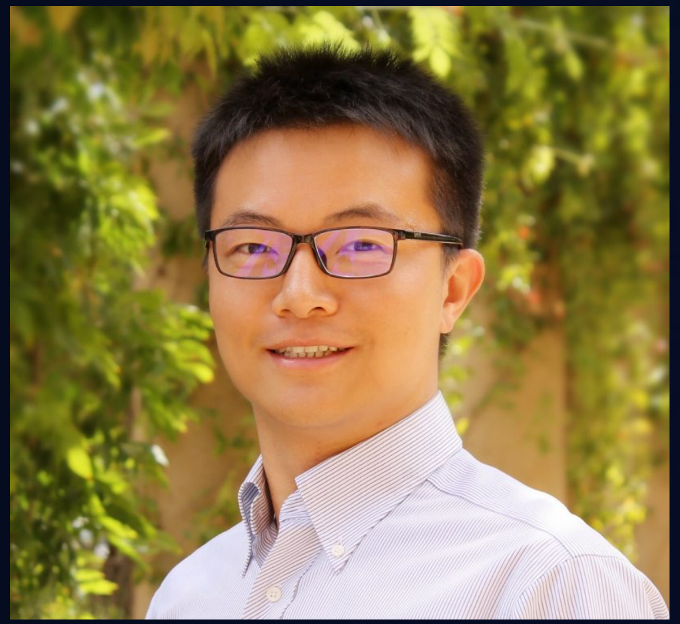
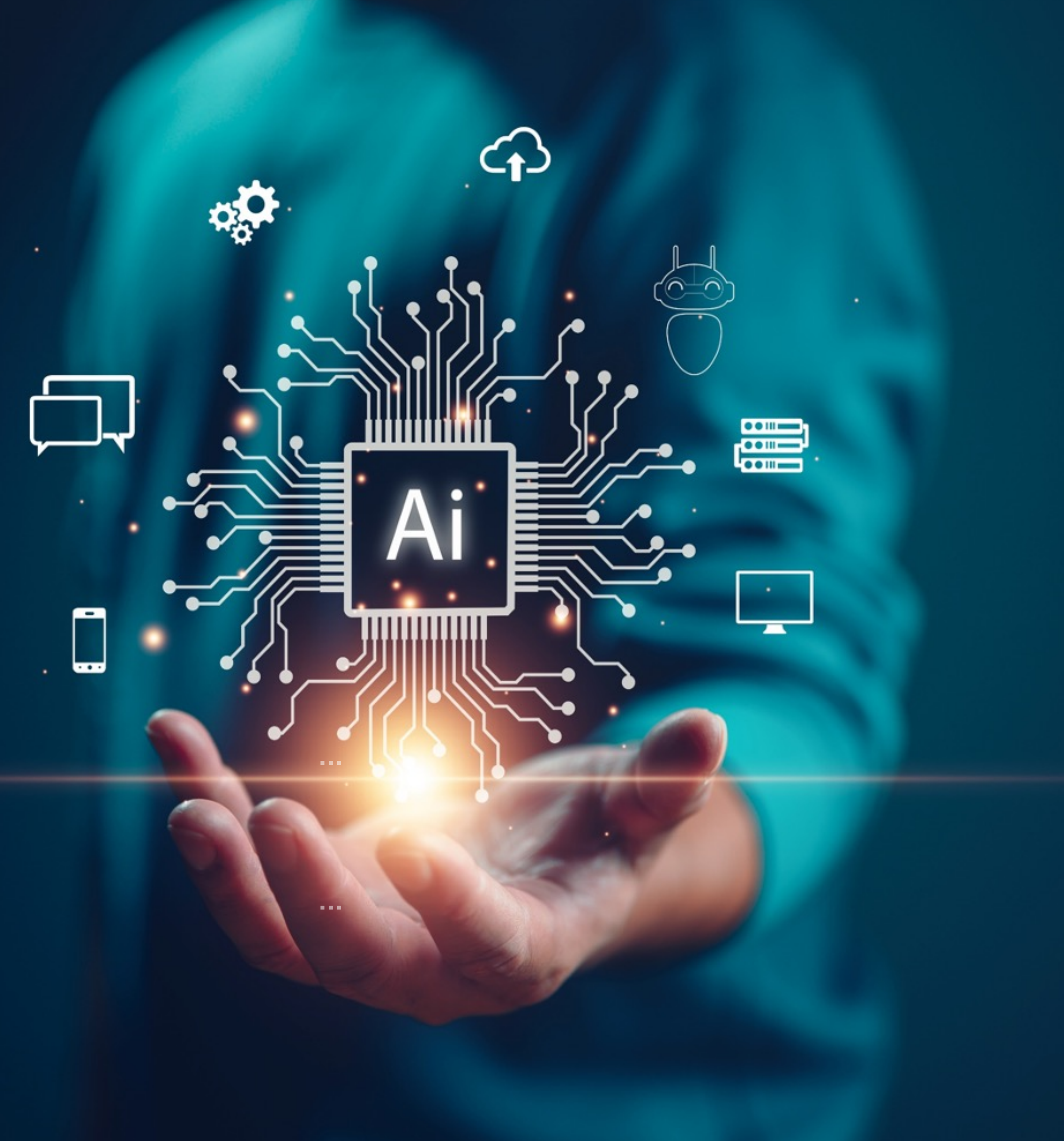
Visual Language Models for Edge AI 2.0



Song Han

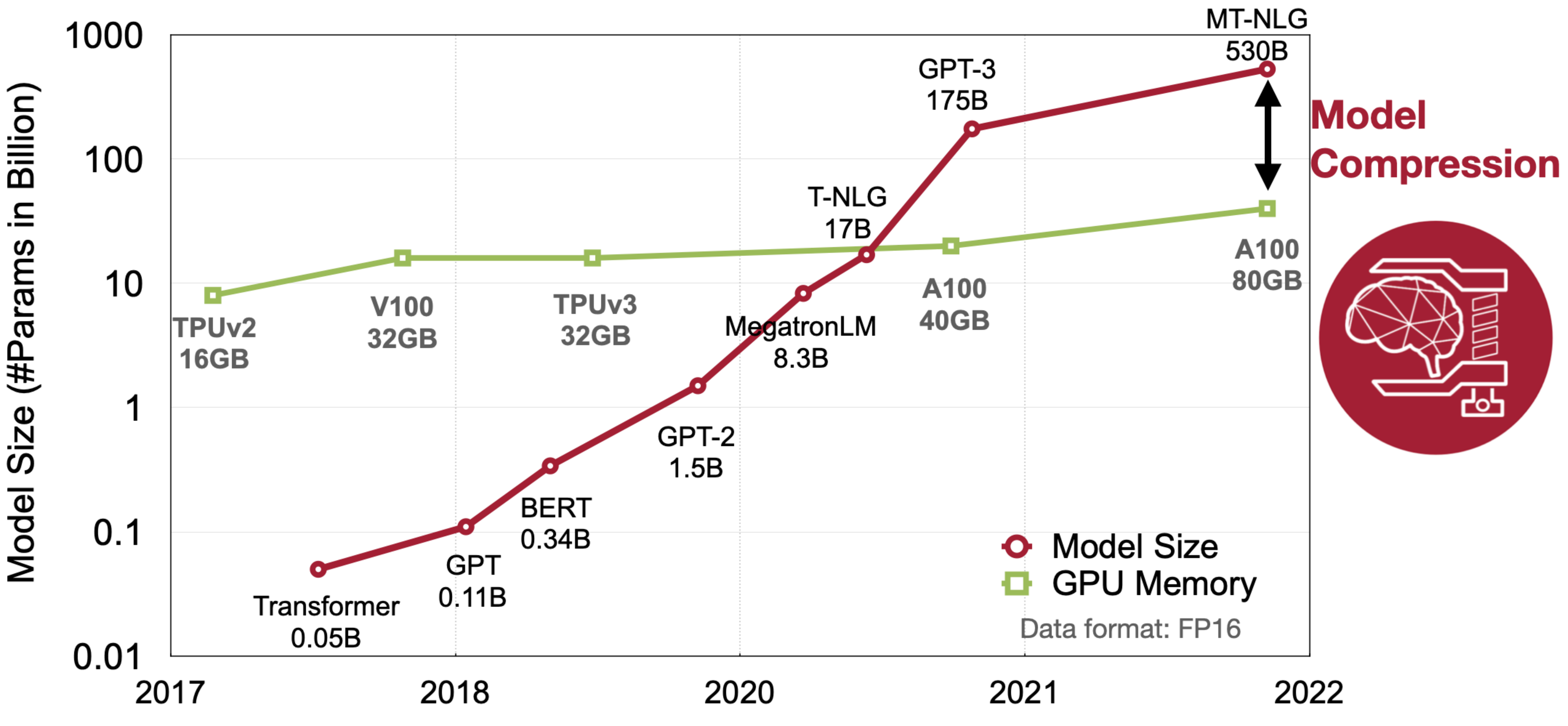
Associate Professor, MIT
Distinguished Scientist, NVIDIA

https://songhan.mit.edu

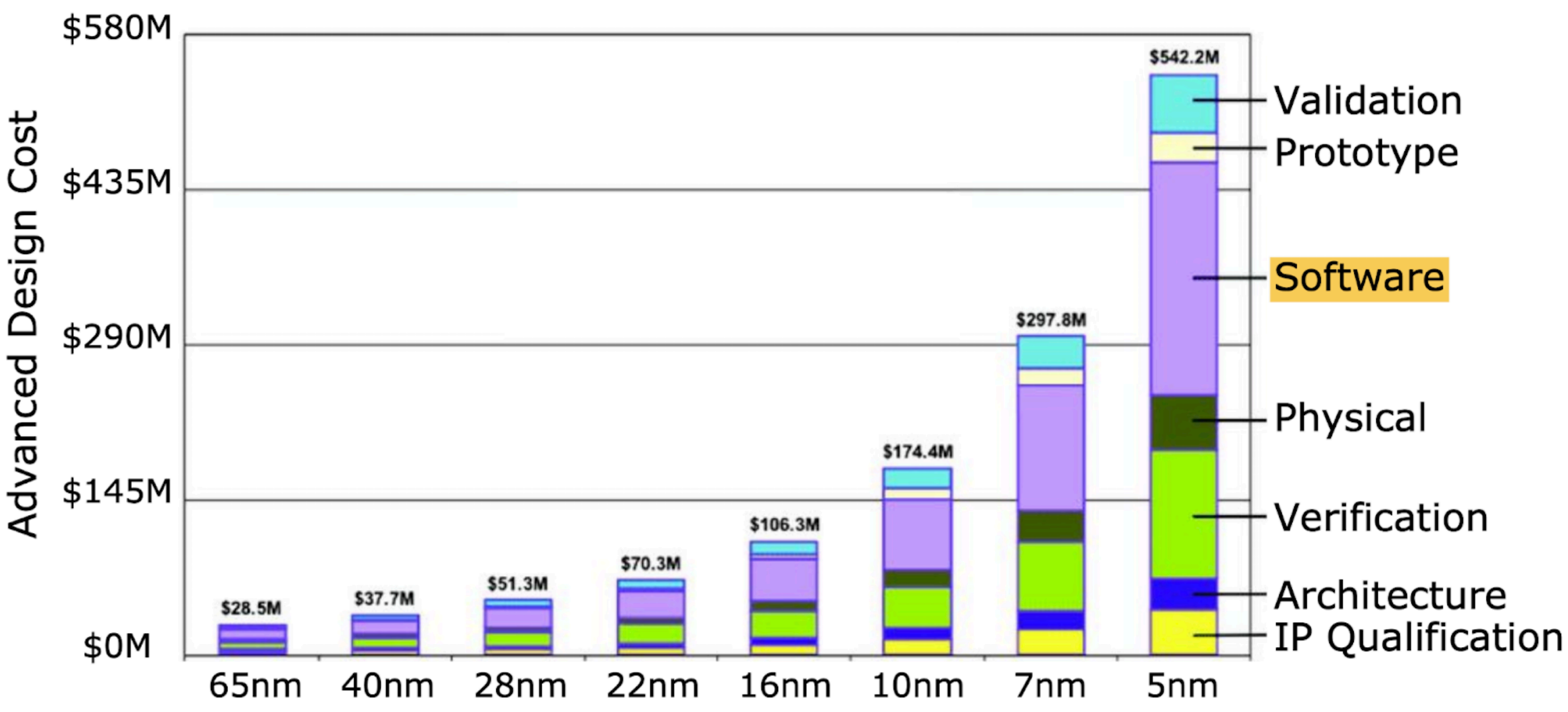


The Need for Efficient AI Computing

Move up the stack, co-design software and hardware



The demand for AI computing is increasing fast

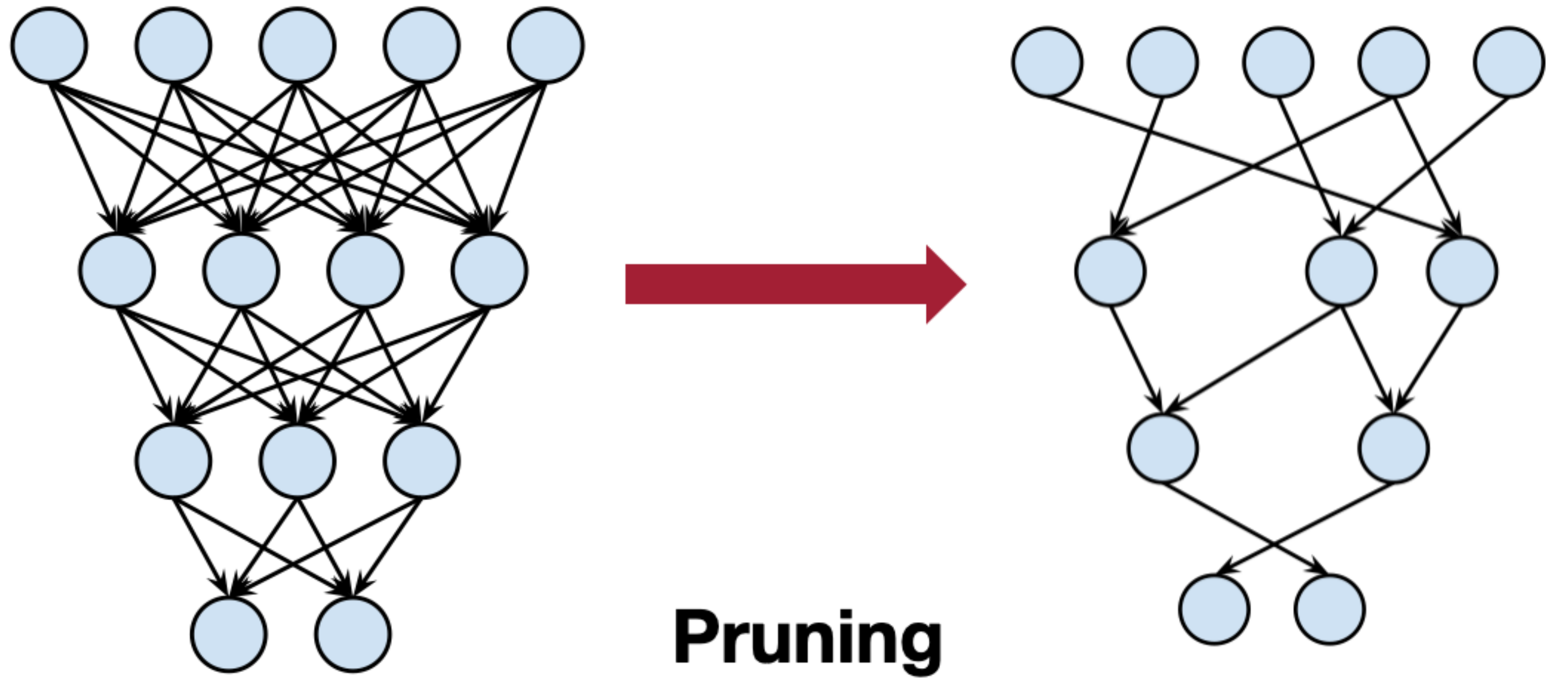


Software is important, the cost is high

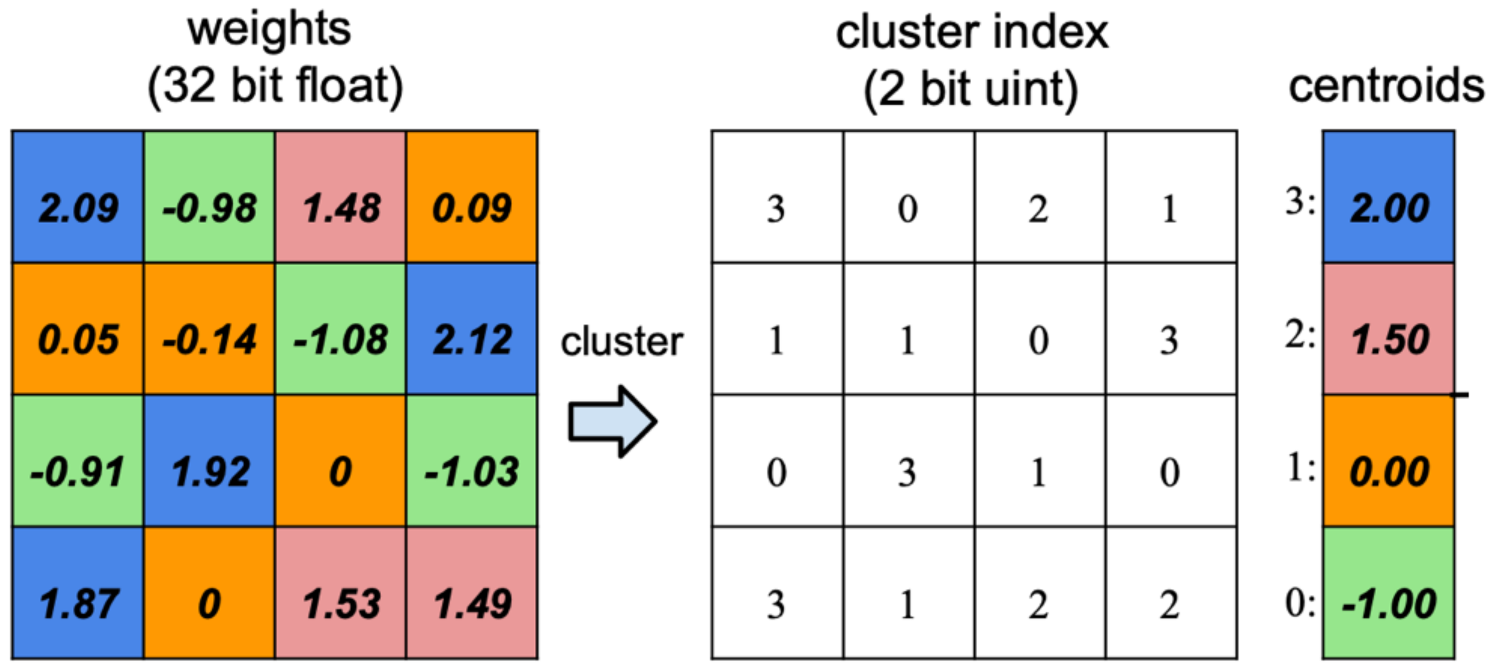
[source]

Previous Work

Deep Compression and EIE



Deep Compression

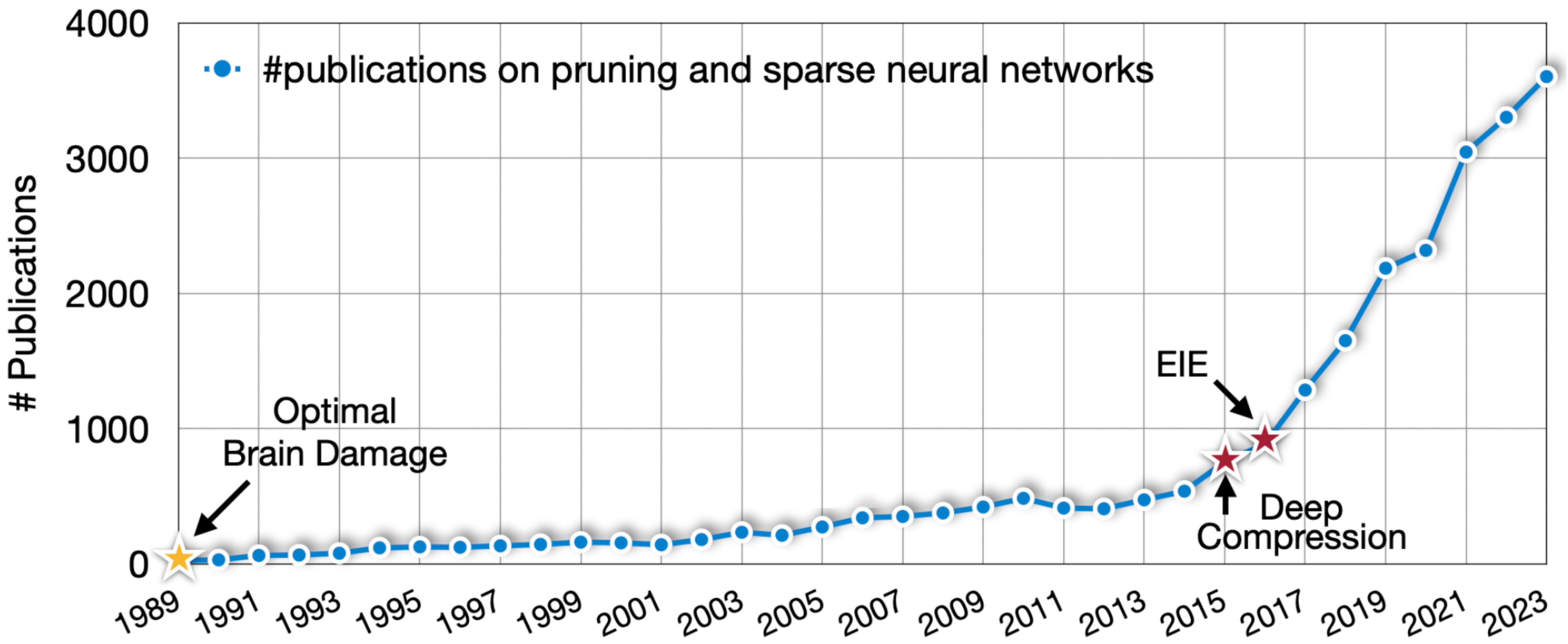


Quantization

Top-5 most cited papers in 50 years of ISCA (1953-2023)

Rank	Citations	Year	Title (★ means it won the ISCA Influential Paper Award)	First Author + HOF Authors	Type	Topic
1	5351	1995	The SPLASH-2 programs: Characterization and methodological considerations	Stephen Woo, *Anoop Gupta*	Tool	Benchmark
2	4214	2017	In-datacenter performance analysis of a Tensor Processing Unit	*Norm Jouppi, David Patterson*	Arch	Machine Learning
3	3834	2000	★ Wattch: A framework for architectural-level power analysis and optimizations	*David Brooks, Margaret Martonosi*	Tool	Power
4	3386	1993	★ Transactional memory: Architectural support for lock-free data structures	Maurice Herlihy	Micro	Parallelism
5	2690	2016	EIE: Efficient inference engine on compressed deep neural network	Song Han, *Bill Dally, Mark Horowitz*	Arch	Machine Learning

Efficient Inference Engine



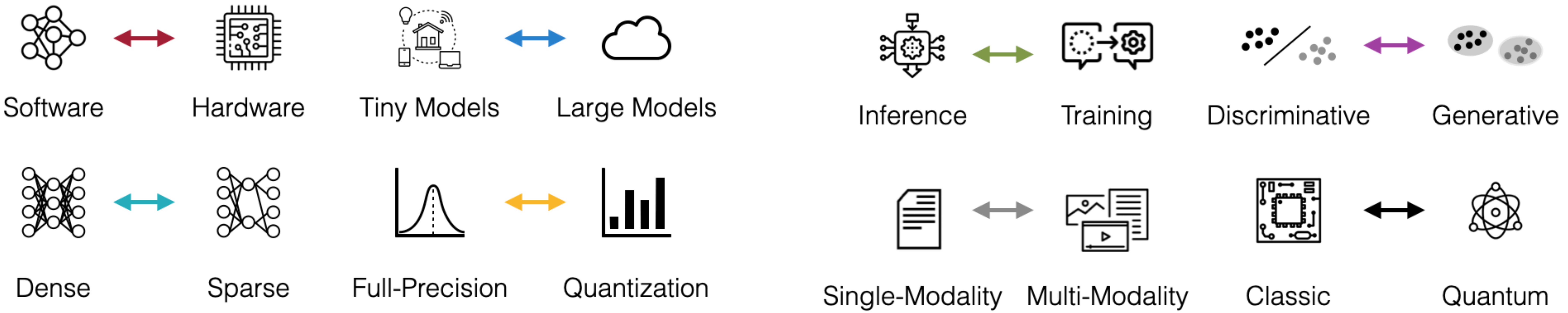
[NIPS'15, ICLR'16, ISCA'16]

EfficientML Project

Bridge the supply and demand of AI computing

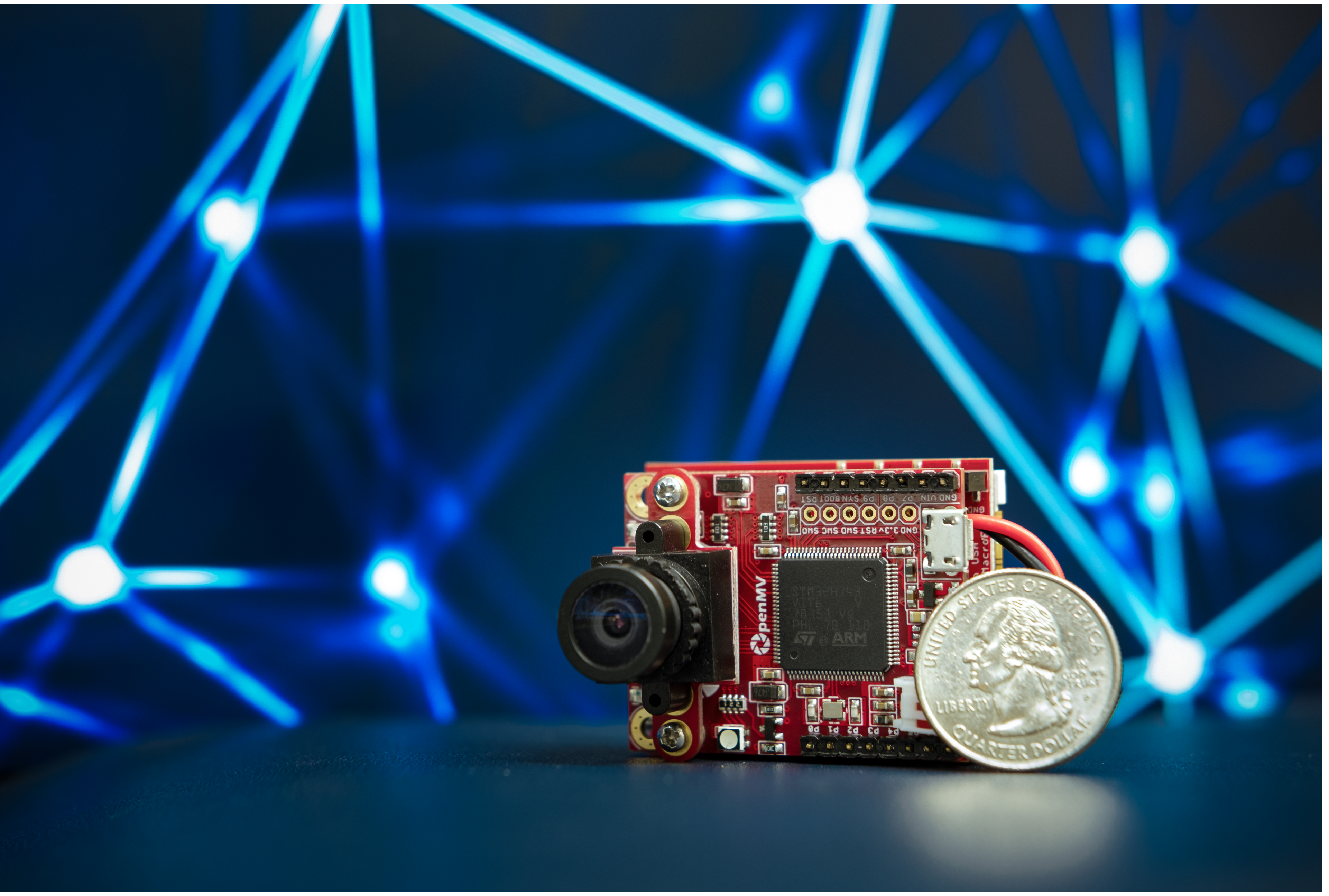
Algorithm and system co-design for accelerated AI computing

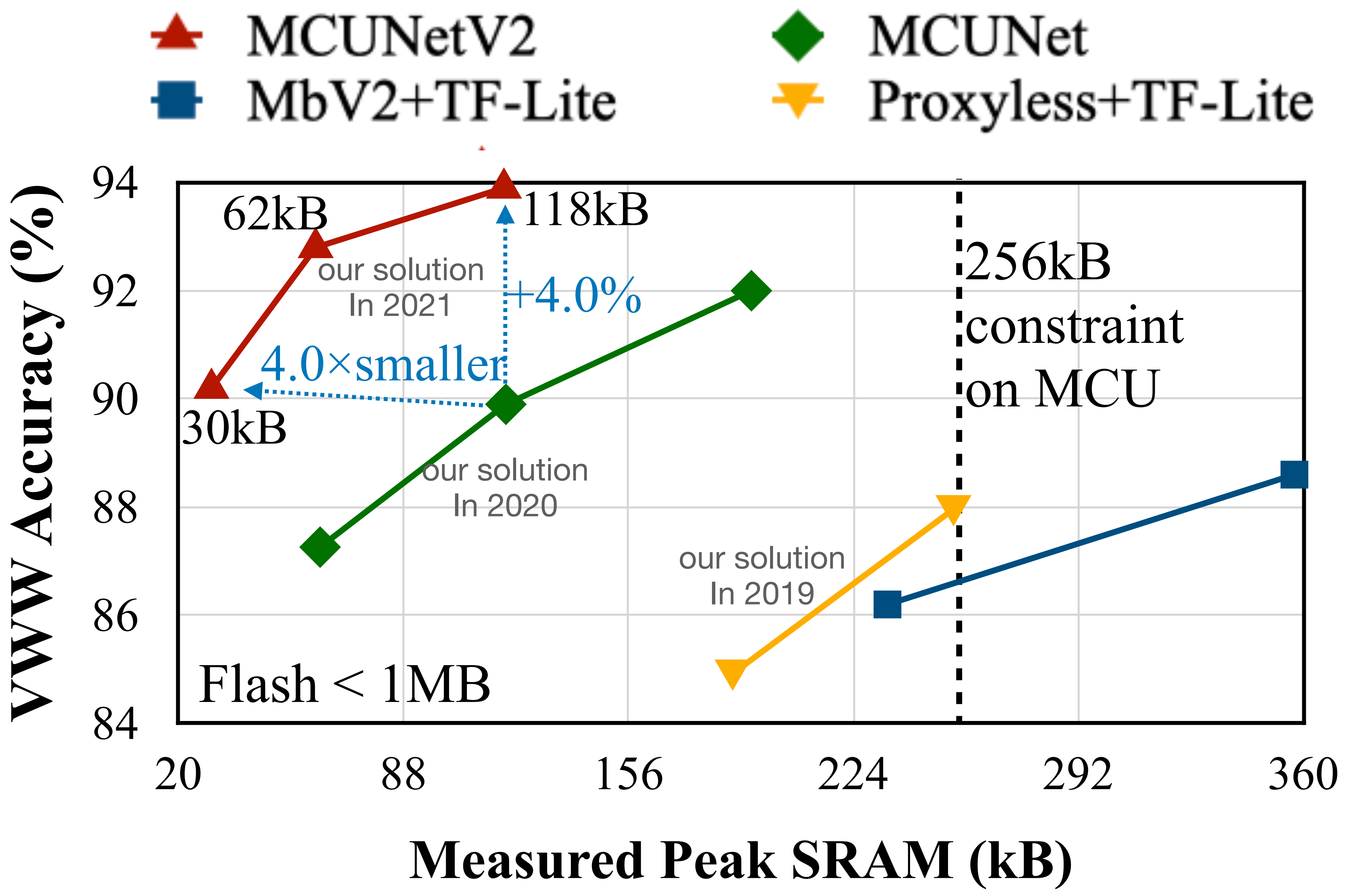
Goal: reduce latency, memory, low power/energy; increase throughput, accuracy, scalability.

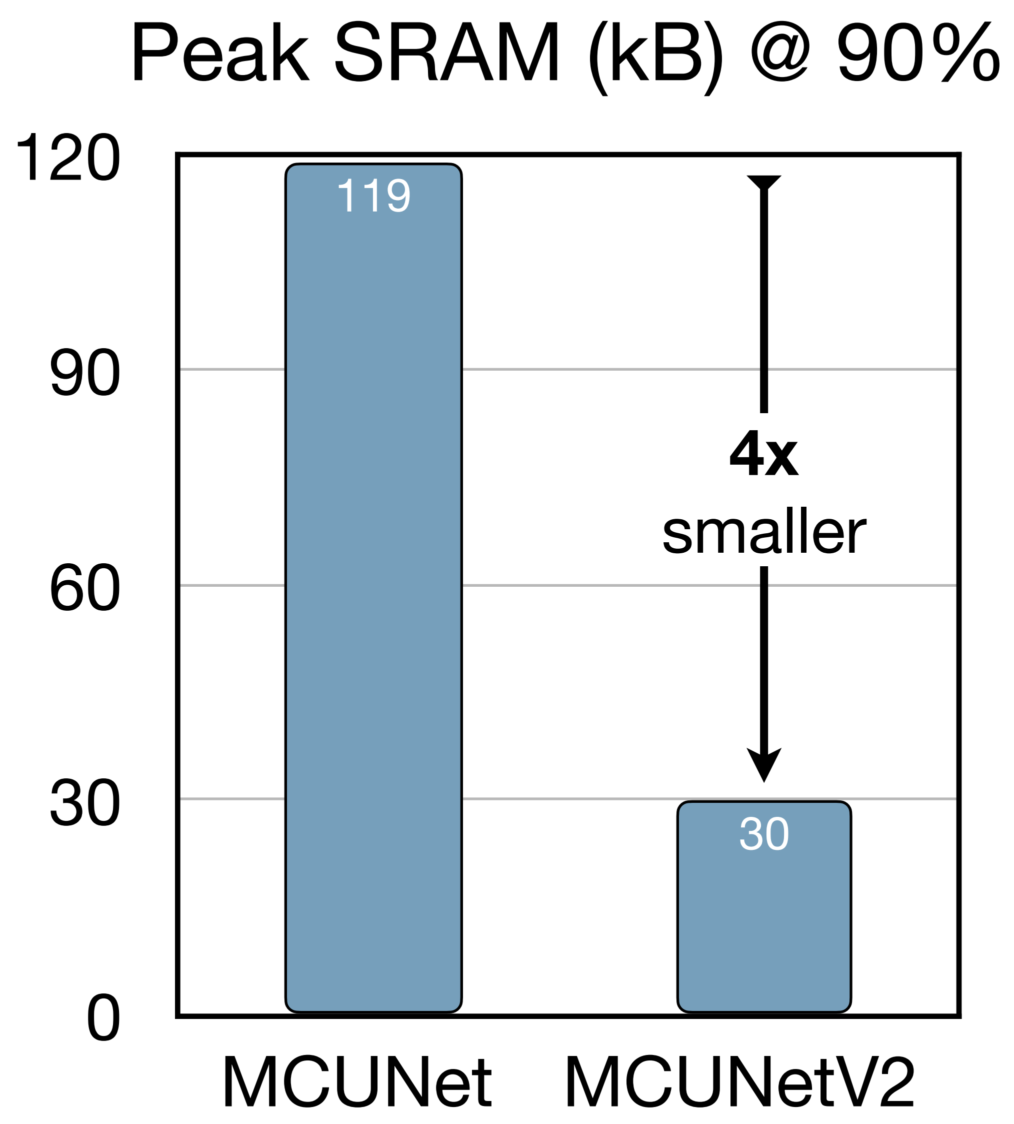


hanlab.mit.edu

Tiny Machine Learning with MCUNet







- TinyML: design light-weighted neural networks and deploy on cheap edge devices that has low power, computing, and memory.
- Billions of IoT devices around the world based on microcontrollers, much cheaper ($1-2), much smaller, everywhere in our lives, but very memory-constraint.
- MCUNet and TinyEngine paves the way for tiny machine learning on edge devices.

[NeurIPS’20/21]

demo link

Demo

MCUNetV2: Memory-Efficient Patch-based Inference for Tiny Deep Learning

Ji Lin, Wei-Ming Chen, Han Cai, Chuang Gan, Song Han

mcunet.mit.edu

demo link

On-Device Training Under 256KB Memory

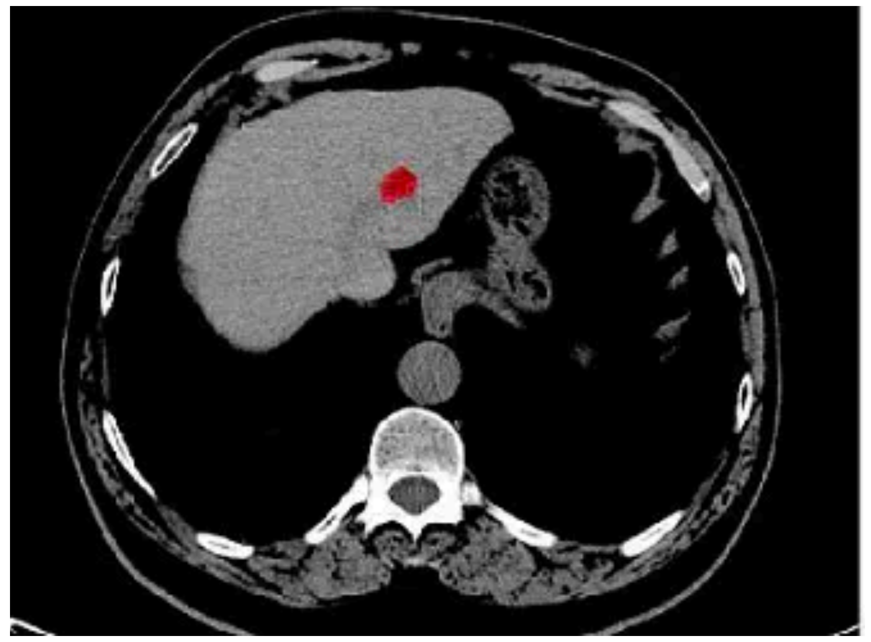
Demo Video

ImageNet Pre-trained MCUNet -> VWW
Running on OpenMV Cam H7 MCU

Edge AI 1.0

Train a specific model for each task

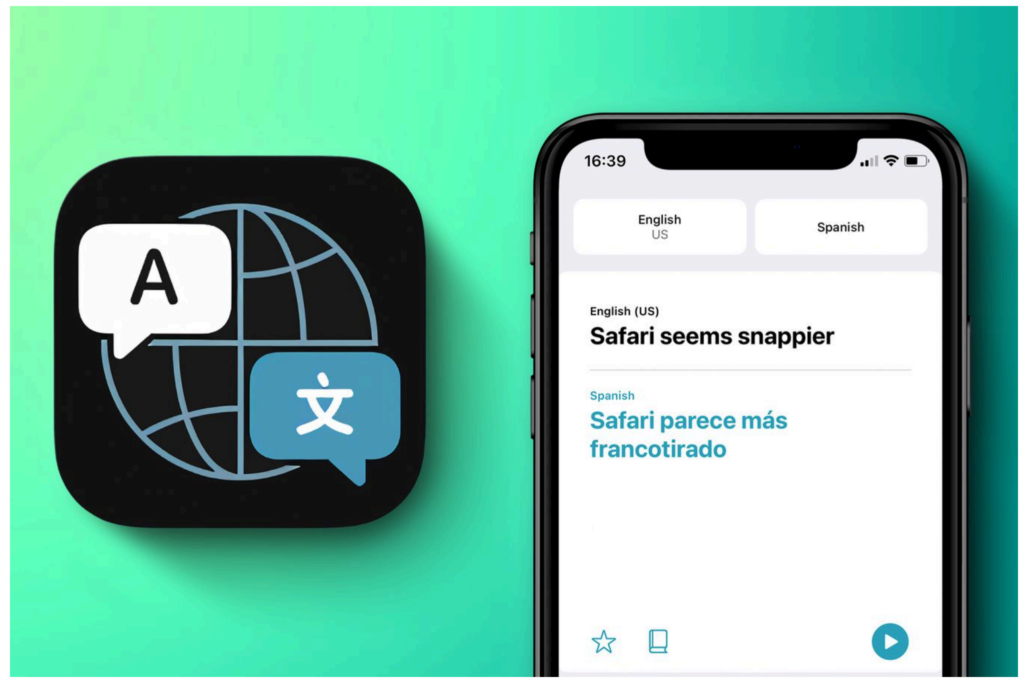
Medical image processing



Autonomous driving

Smart manufacturing

Machine translation

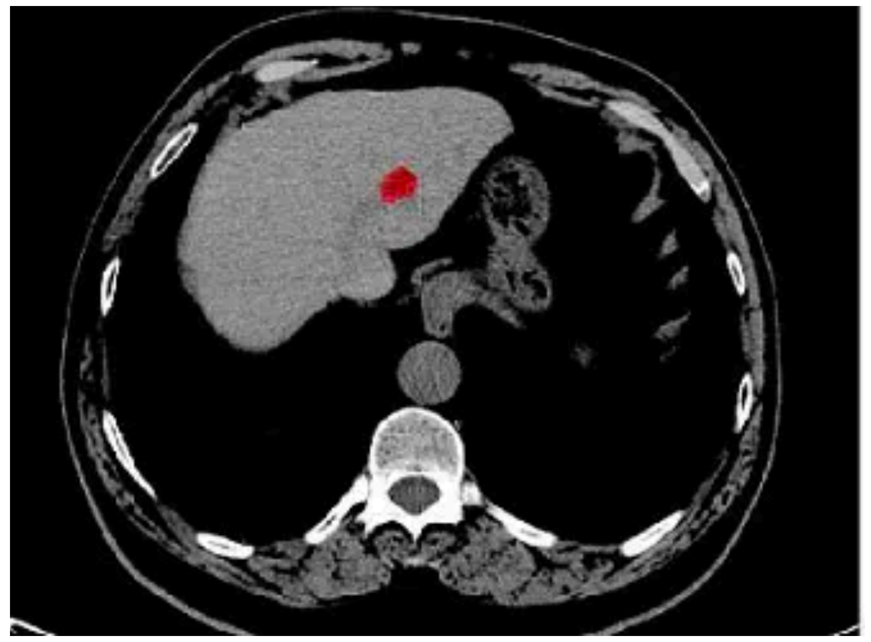


Task-specific models

Edge AI 1.0

Train a specific model for each task

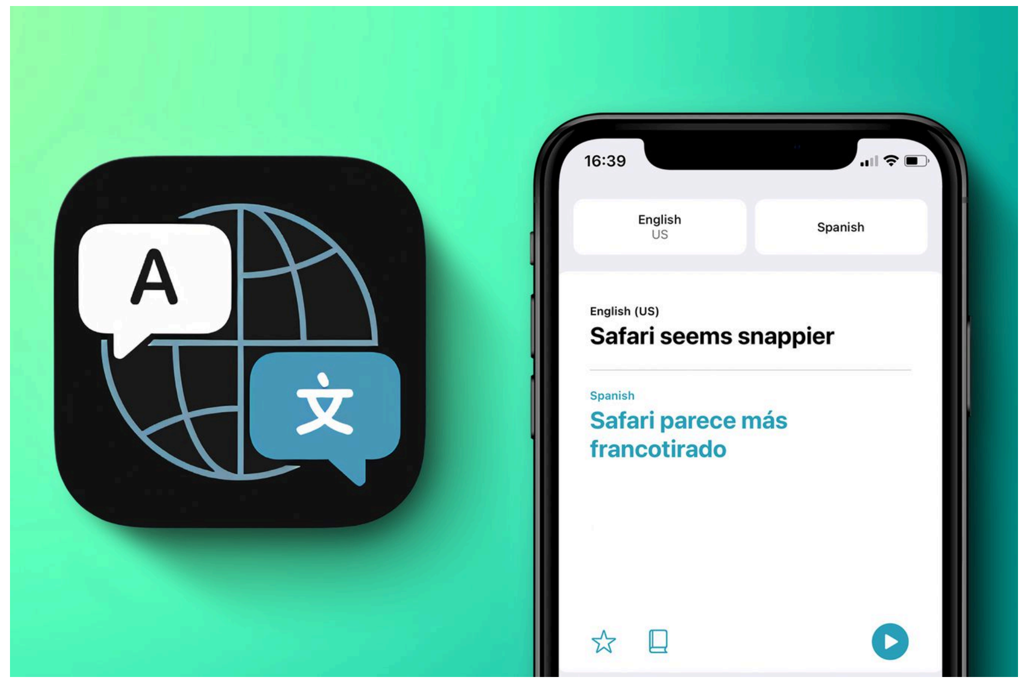
Medical image processing



Autonomous driving

Smart manufacturing

Machine translation

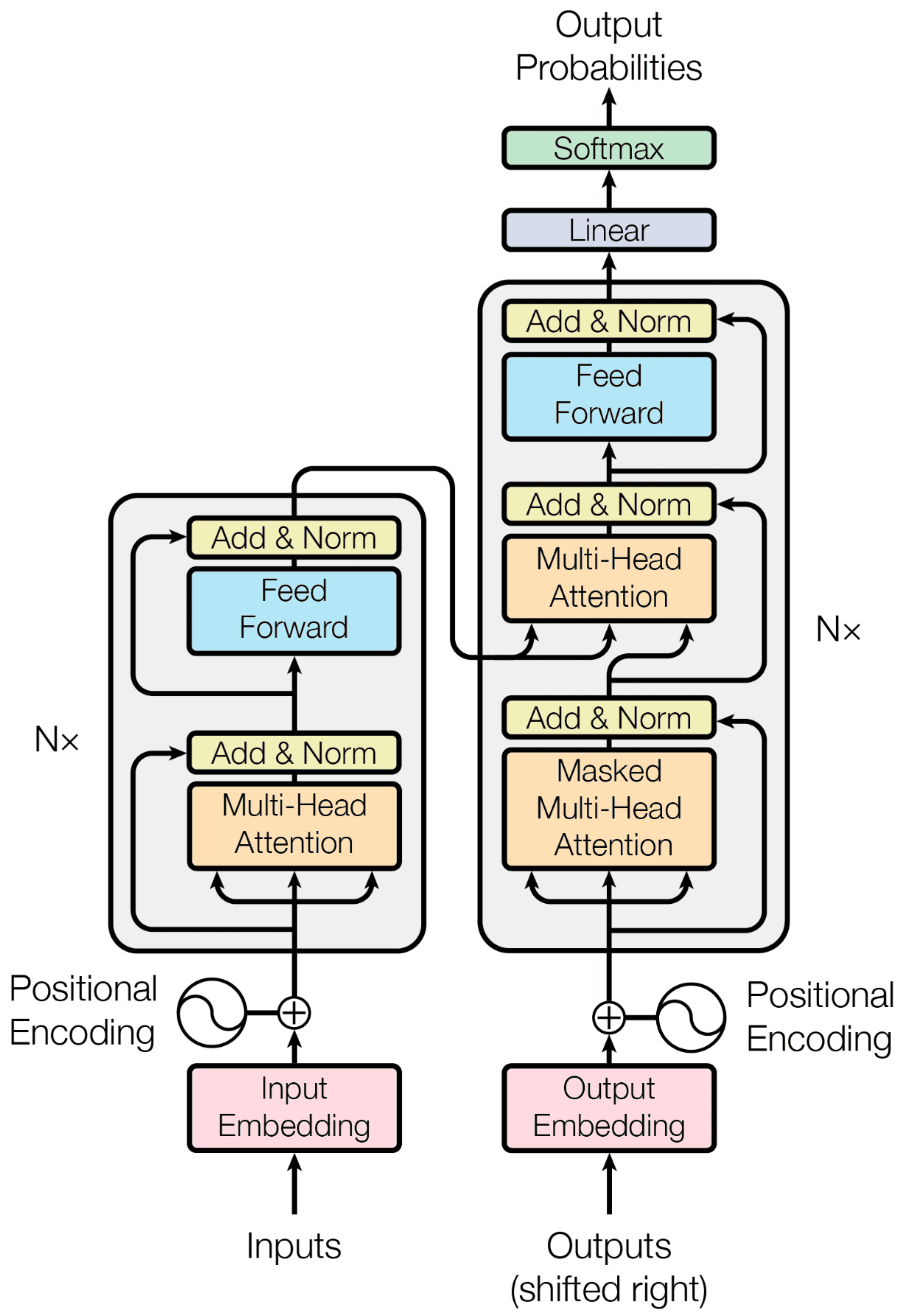


Task-specific models

- Need different model / data to train different tasks
- Lack of 'negative samples' for training
- Limited generalization; failure of corner cases

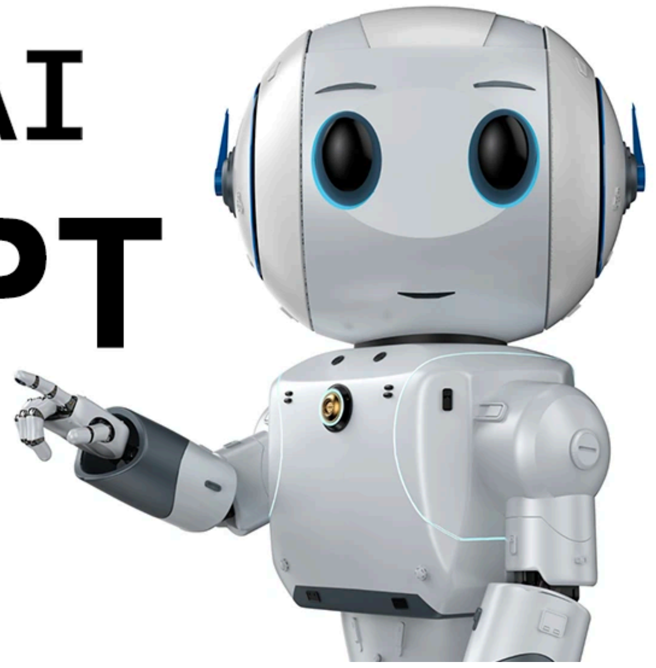
Edge AI 2.0

General models with world knowledge

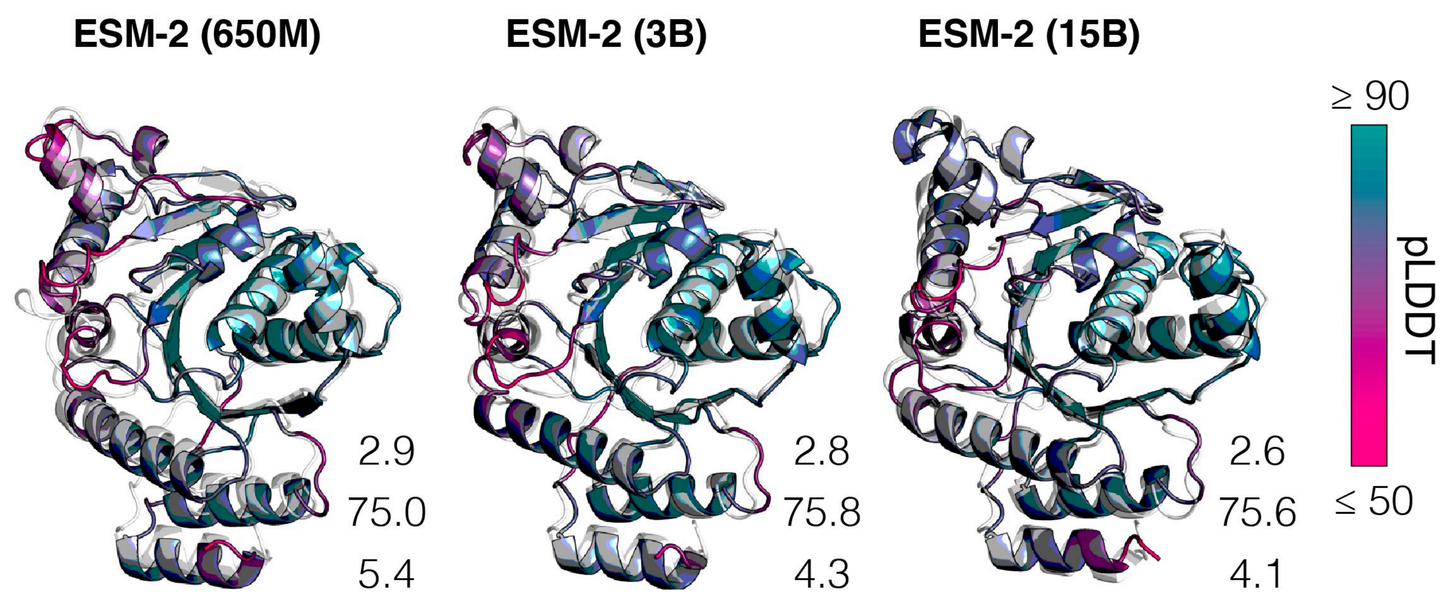


Transformer-based

OpenAI
ChatGPT



ChatBots



Scientific Discovery

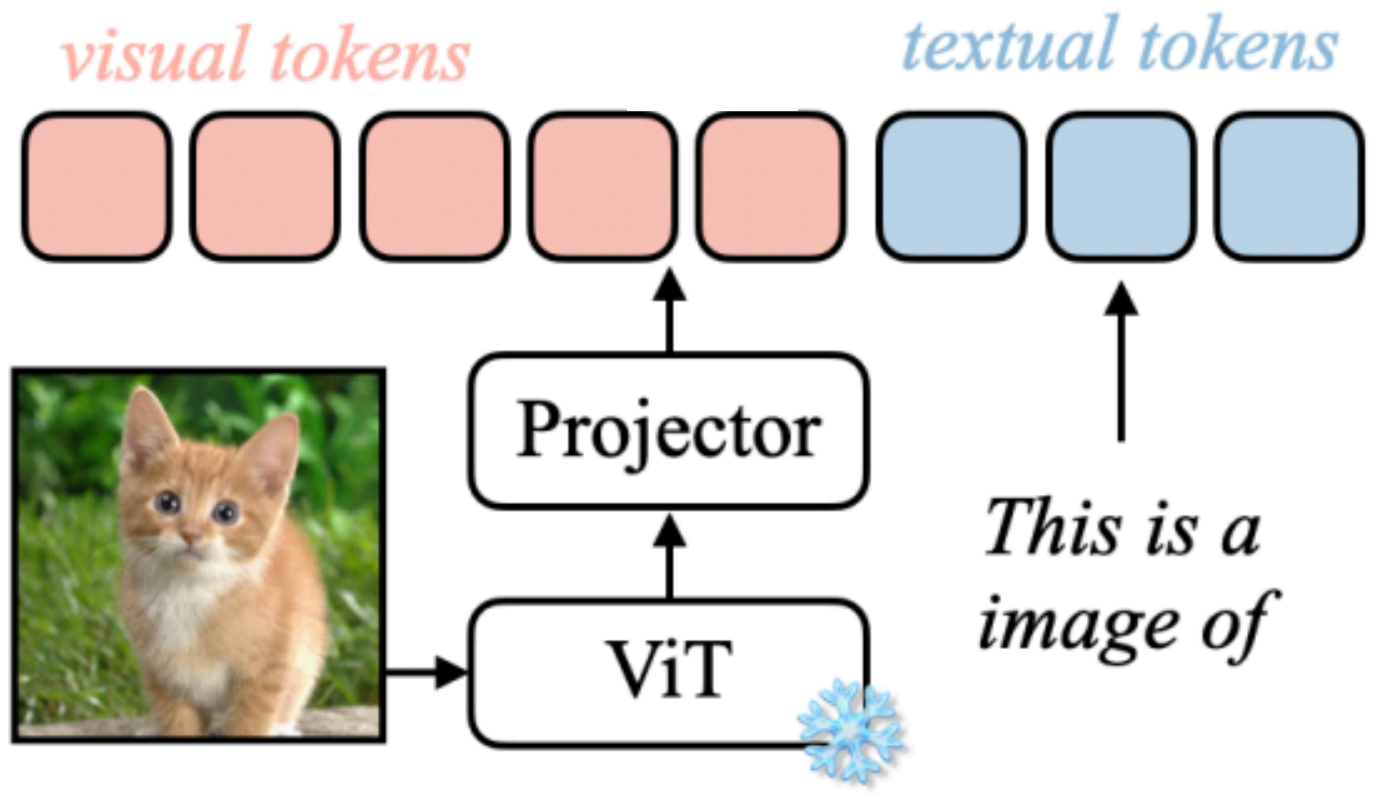
GitHub Copilot

Software Development

Edge AI 2.0

General models with world knowledge

- One model - multiple tasks
- Enhanced by LM's world knowledge
- Advanced reasoning capabilities
- Instruction-following proficiency



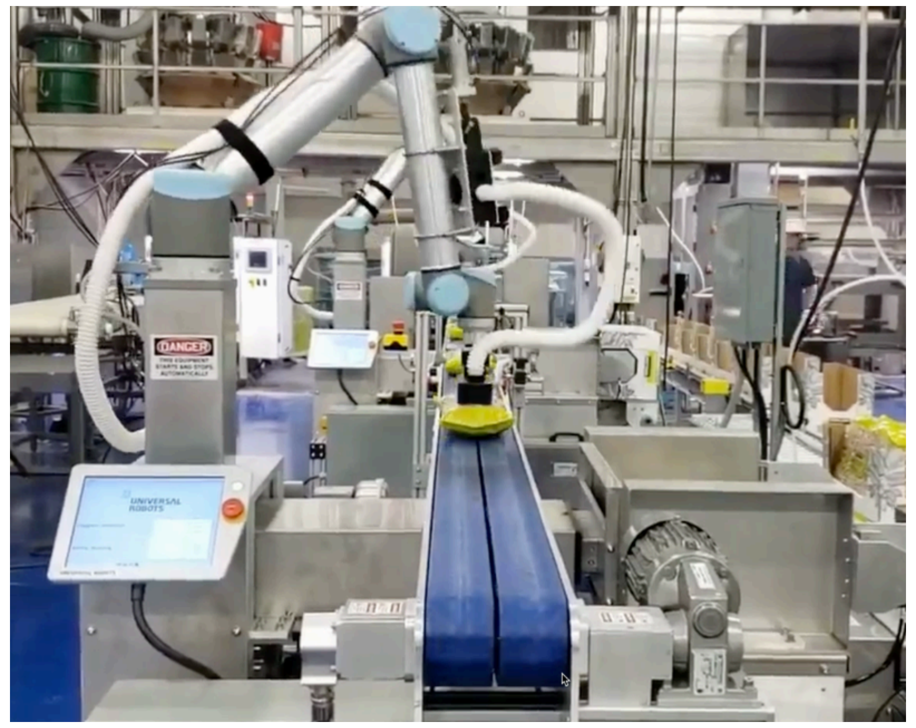
World knowledge embedded on-device

Visual Language Model (VLM)

Landmark recognition

Driving assistant

Patient monitoring

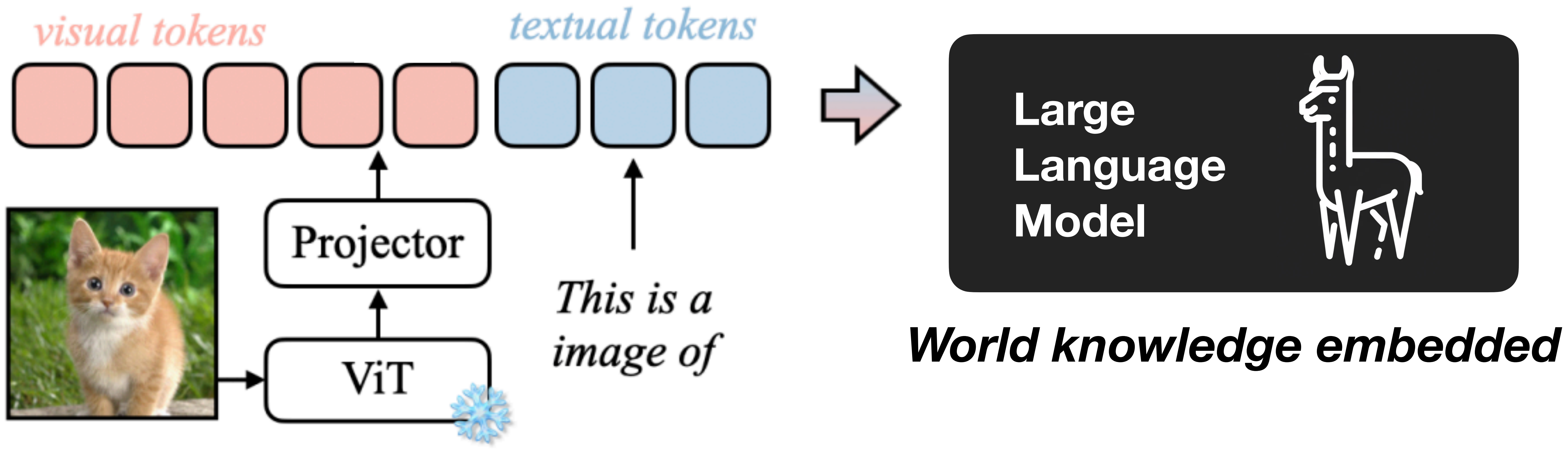


Smart manufacturing

Edge AI 2.0

General models with world knowledge

- One model - multiple tasks
- Enhanced by LM's world knowledge
- Advanced reasoning capabilities
- Instruction-following proficiency



Visual Language Model (VLM)

VLM Pre-training (VILA) → Model Compression (AWQ) → Efficient Deployment (TinyChat)

VILA

VILA —AWQ→ VILA

TinyChat

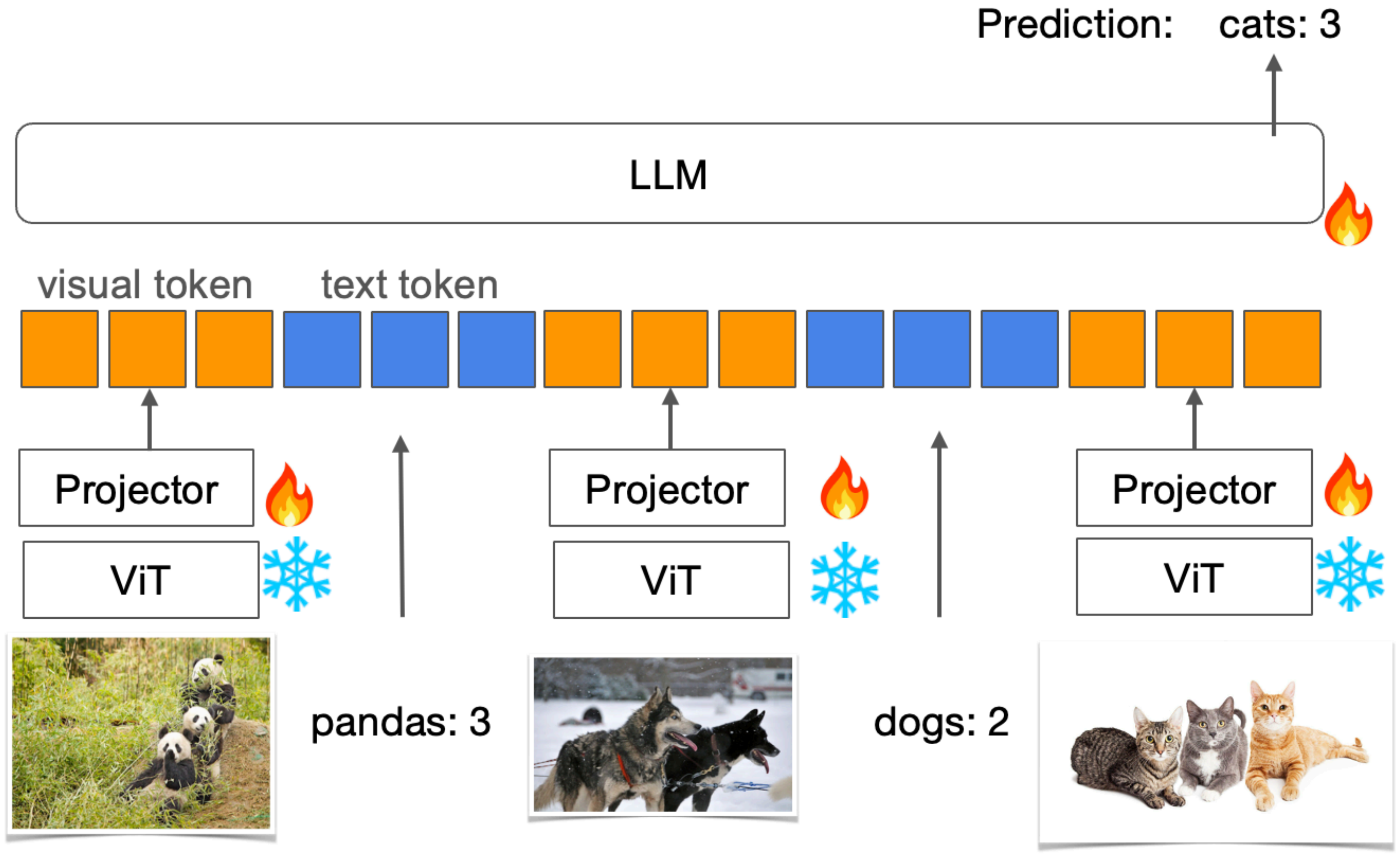
VILA: Visual Language Model on the Edge

ViLA: Visual Language Model

Goal: multi-modal LLM, enhance visual reasoning by language model, enable in-context learning and reasoning across images

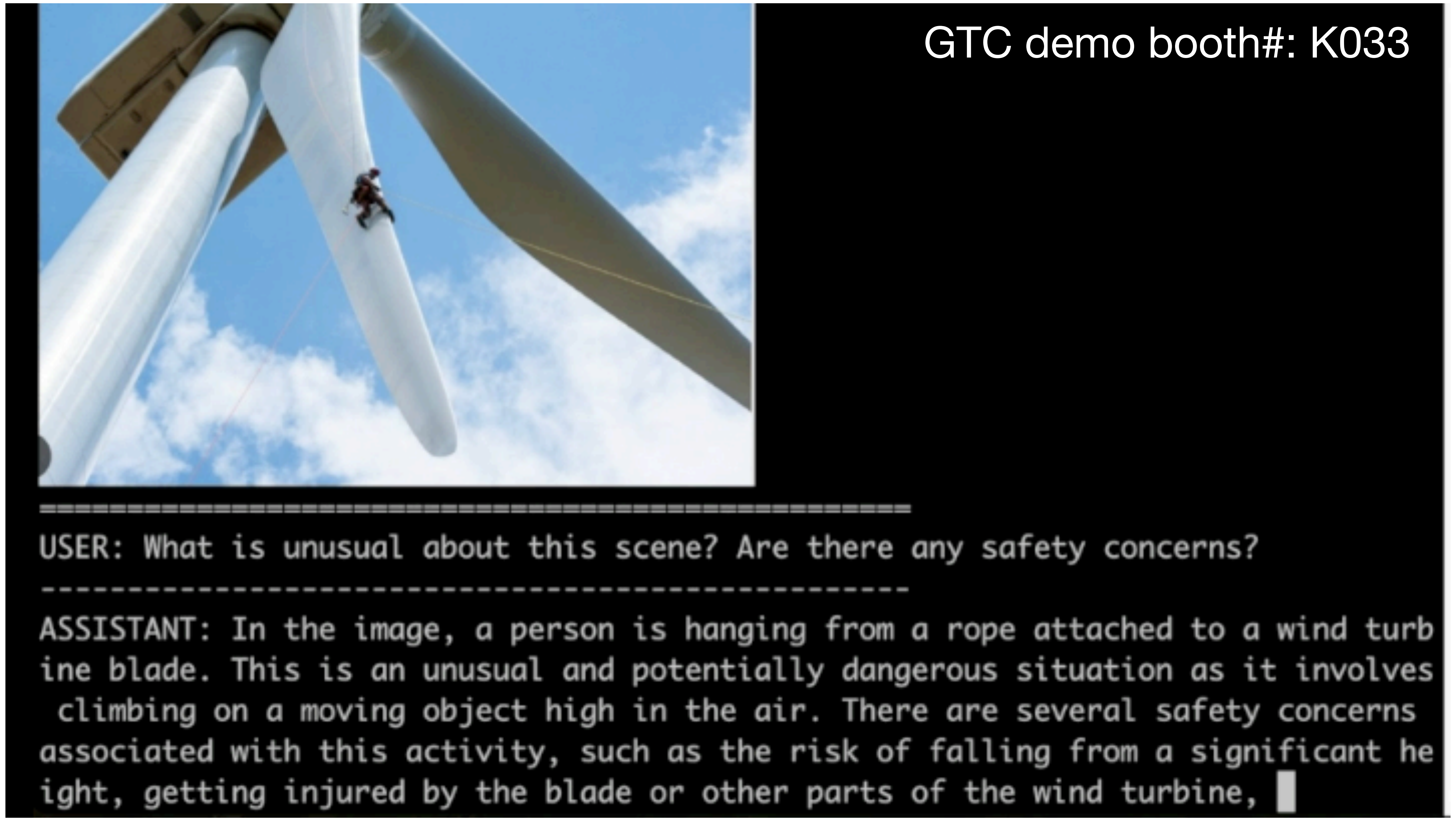
Challenge: learn from visual inputs without destroying the LLM. Computationally heavy to deploy on the edge.

Our Solution: VILA provides efficient recipe from data curation to training to deployment for VLM. Interleaved image-text pretraining enables in-context learning.



Augments the language model with interleaved image-text

VILA deployed on Jetson Orin (click to play)



VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

ViLA: Visual Language Model

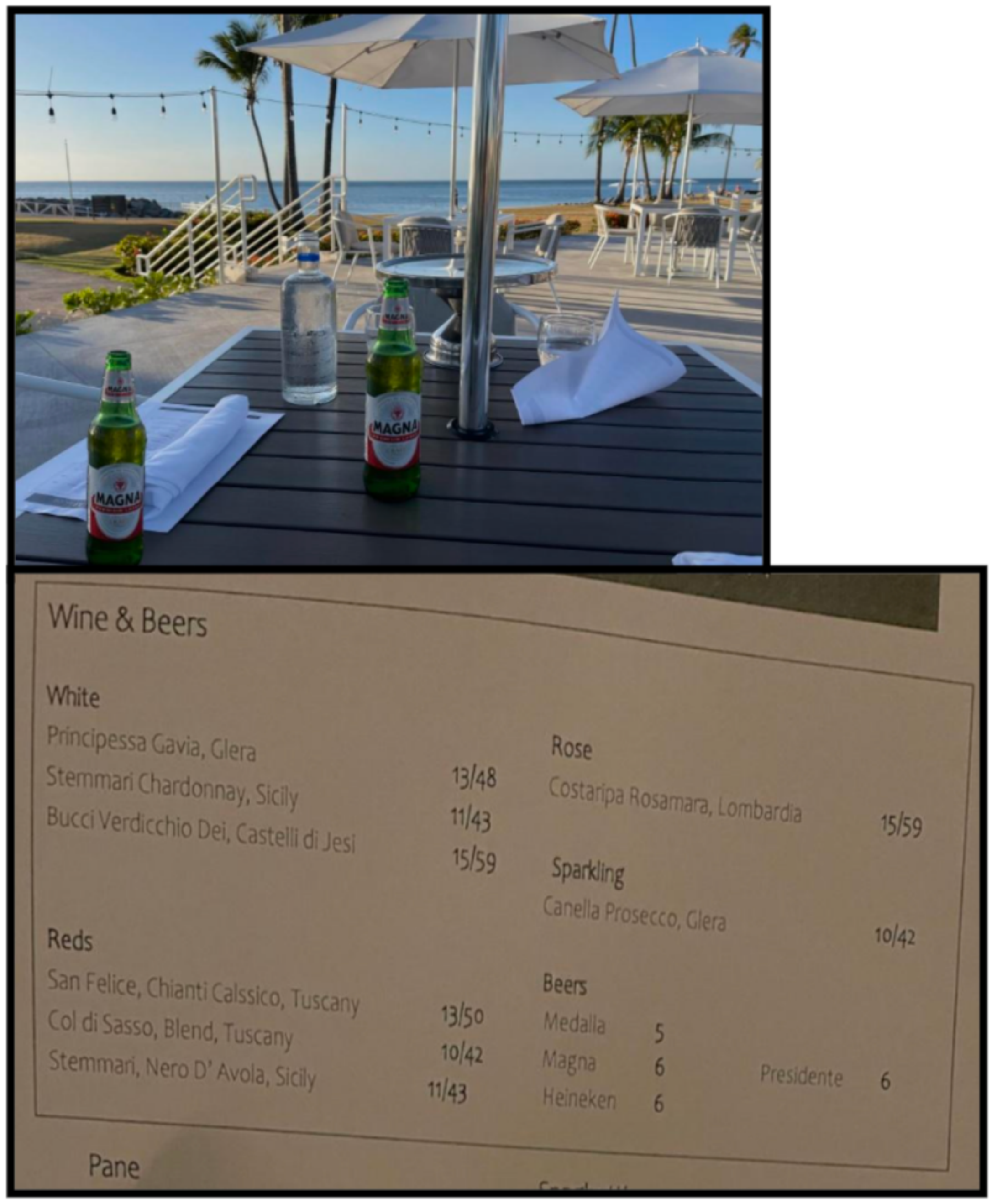
In context learning and multi-image reasoning

The company is famous for its search engine.

The company is famous for the operating system.

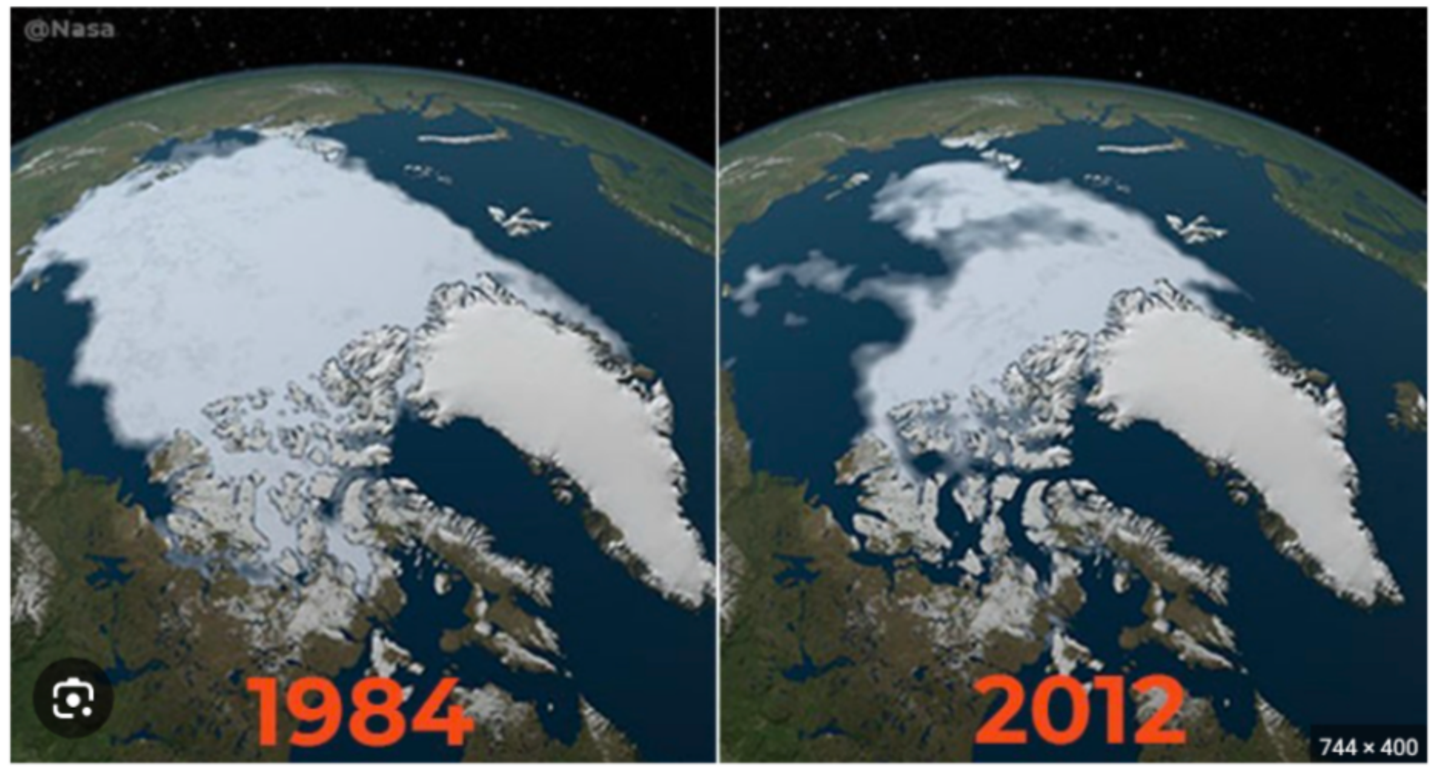
The company is famous for iPhone and Mac.

Pred: The company is famous for its graphics processing units (GPUs)



Prompt: Photo: <img1
much should I pay for a
according to the price o
Answer: According to
on the table is $6.

Prompt: … + ***Think s***
Answer: The beer on t
priced at $6.00. There a
table. Therefore, the to
$6.00 x 2 = $12.00.



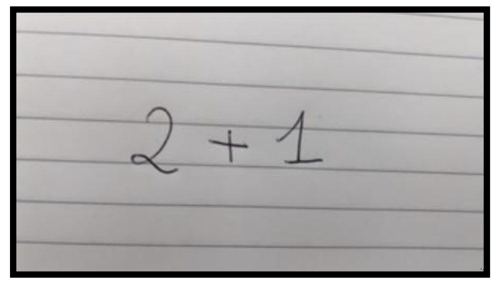
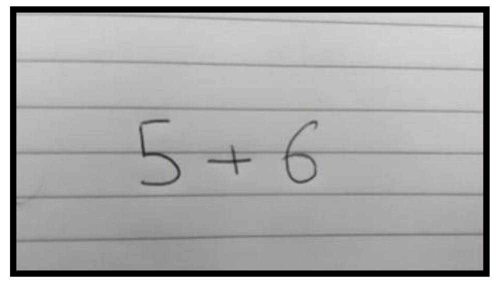
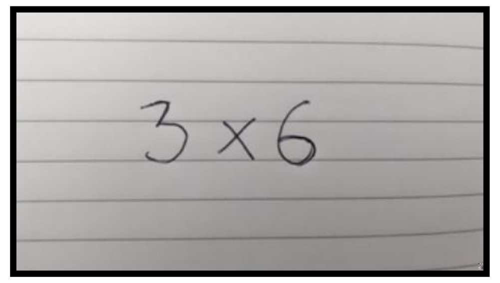
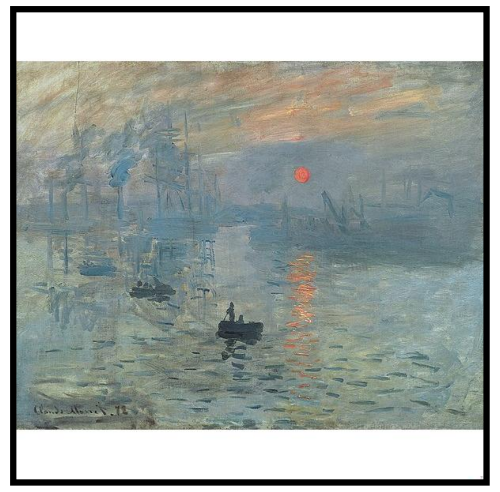
User: What is the implication of temperature based on this image?

VILA: The image shows a comparison of the Arctic ice cover in 1984 and 2012. The implication of this image is that the Arctic ice cover has significantly decreased over the years, which is a result of climate change. The reduction in ice cover can have various implications, such as increased exposure of the Arctic environment to warmer temperatures, which can lead to the melting of sea ice, loss of habitat for polar bears and other Arctic species, and potential disruptions to the global climate system. This image serves as a visual representation of the impact of climate change on the Arctic environment and the need for action to mitigate its effects.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

ViLA: Visual Language Model

More in context learning examples

Input few shot examples + target image					Output
	Underground.		Congress.		**LLaVA-1.5**: Soulemes. **Ours**: Soulomes.
	2+1=3		5+6=11		**LLaVA-1.5**: 3x6=18 **Ours**: 3x6=18
	Les sanglots longs des violons de l’automne blessent mon coeur d’une langueur monotone.		Pour qui sont ces serpents qui sifflent sur vos têtes?		**LLaVA-1.5**: Les serpents *(snake)* qui sifflent sur vos têtes sont-ils vos amis? **Ours**: Pourquoi ces flamands *(flamingo)* roses s'embrassent-ils?
	Romanticism		Surrealism		**LLaVA-1.5**: Surrealism **Ours**: Impressionism

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

ViLA for Autonomous Driving

Zero-shot learning, explainable self-driving

Q: If you are driving, should you honk at the pedestrians crossing the road in this case? Why?

A: In the image, a car is driving down a street with a group of people crossing the road. It is not appropriate to honk at pedestrians crossing the road, as it can be considered impolite and disrespectful. Instead, the driver should wait patiently for the pedestrians to cross safely before proceeding. Honking can cause distress and discomfort to the pedestrians and may also lead to accidents if the pedestrians are not aware of the approaching vehicle. It is essential to prioritize the safety and well-being of pedestrians in such situations.

Q: What color is the traffic light in the image?

A: The traffic light in the image is red.

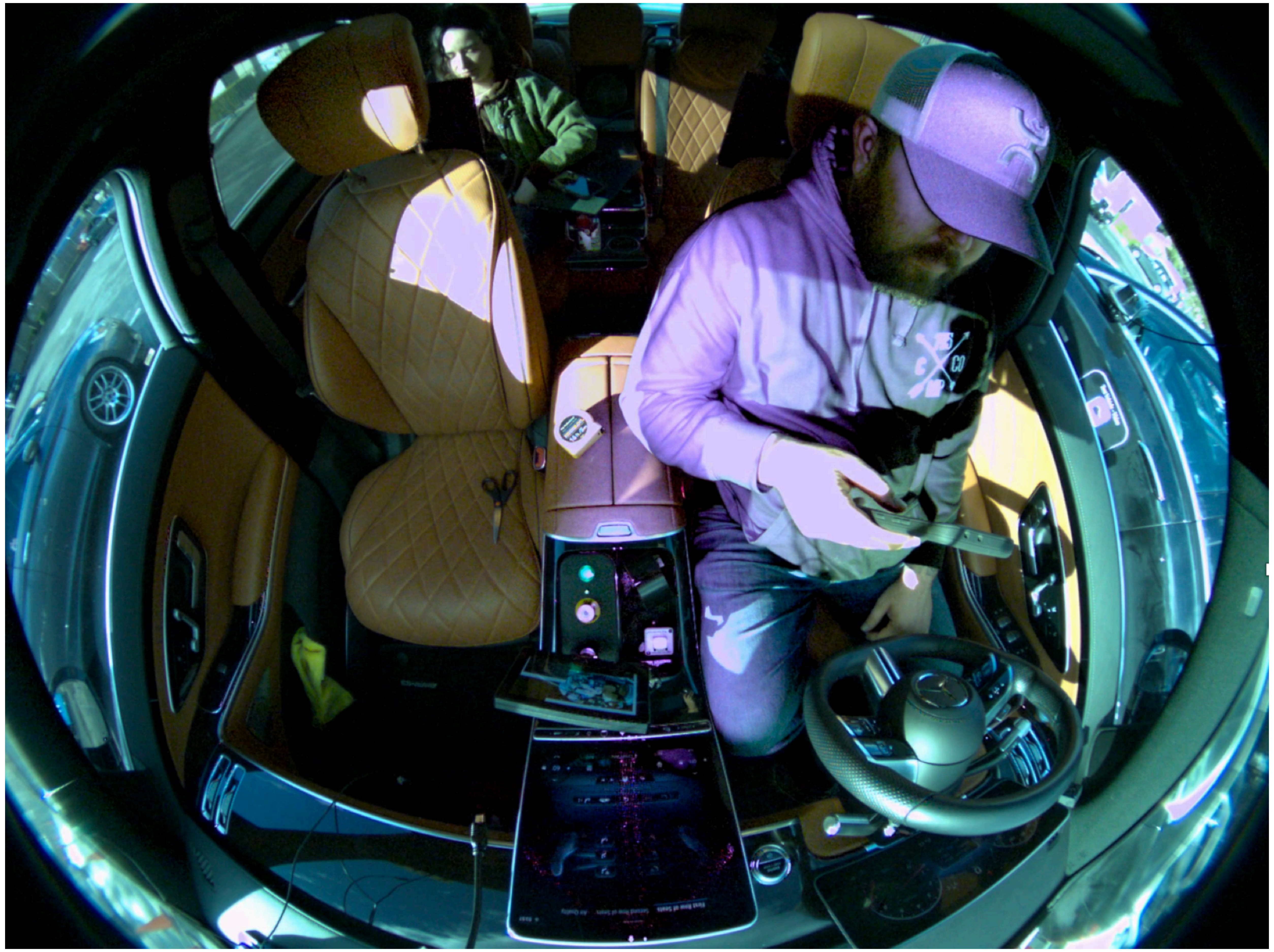
Q: What is the person in the center of the image doing?

A: The person in the center of the image is pushing a stroller with a baby in it.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

VILA for Driver Monitor System

Zero-shot learning



Q: Is the driver distracted?
VILA : Yes, the driver is distracted, as he is looking at his cell phone while driving. This is a dangerous behavior that can lead to accidents and endanger the lives of the driver, passengers, and other road users.

Q: Is the driver on the phone?
VILA : Yes, the driver is on the phone while sitting in the driver's seat.

Q: How many people are in the car?
VILA: There are two people in the car.

Q: Where is the passenger sitting?
VILA: The passenger is sitting in the back seat of the car.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

ViLA for UAV monitoring

Zero-shot learning: UAV captured image explaining

User: Do facilities here look normal or need maintenance?

VILA: The facilities in the image, which are wind turbines, appear to be in good condition and functioning properly. They are situated in a large field with a clear blue sky, indicating that they are well-maintained and not in need of immediate maintenance. The presence of multiple wind turbines in the field suggests that they are part of a larger wind farm, which is likely to be managed and maintained by professionals to ensure optimal performance and efficiency.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

ViLA for Healthcare

Zero-shot learning: patient situation understanding

User: What is the patient status?

VILA: The patient is in a critical condition, as indicated by the presence of a ventilator and the fact that she is hooked up to a lot of wires.

User: Is the patient asleep?

VILA: Yes, the patient is asleep in the hospital bed.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

VILA for Healthcare

Zero-shot learning: training session understanding

User: What is the training session about?

VILA: The training session is about learning how to perform CPR (cardiopulmonary resuscitation) on a mannequin. The group of people, including nurses and possibly other medical professionals, are gathered around a mannequin to practice their skills and learn the proper techniques for administering CPR.

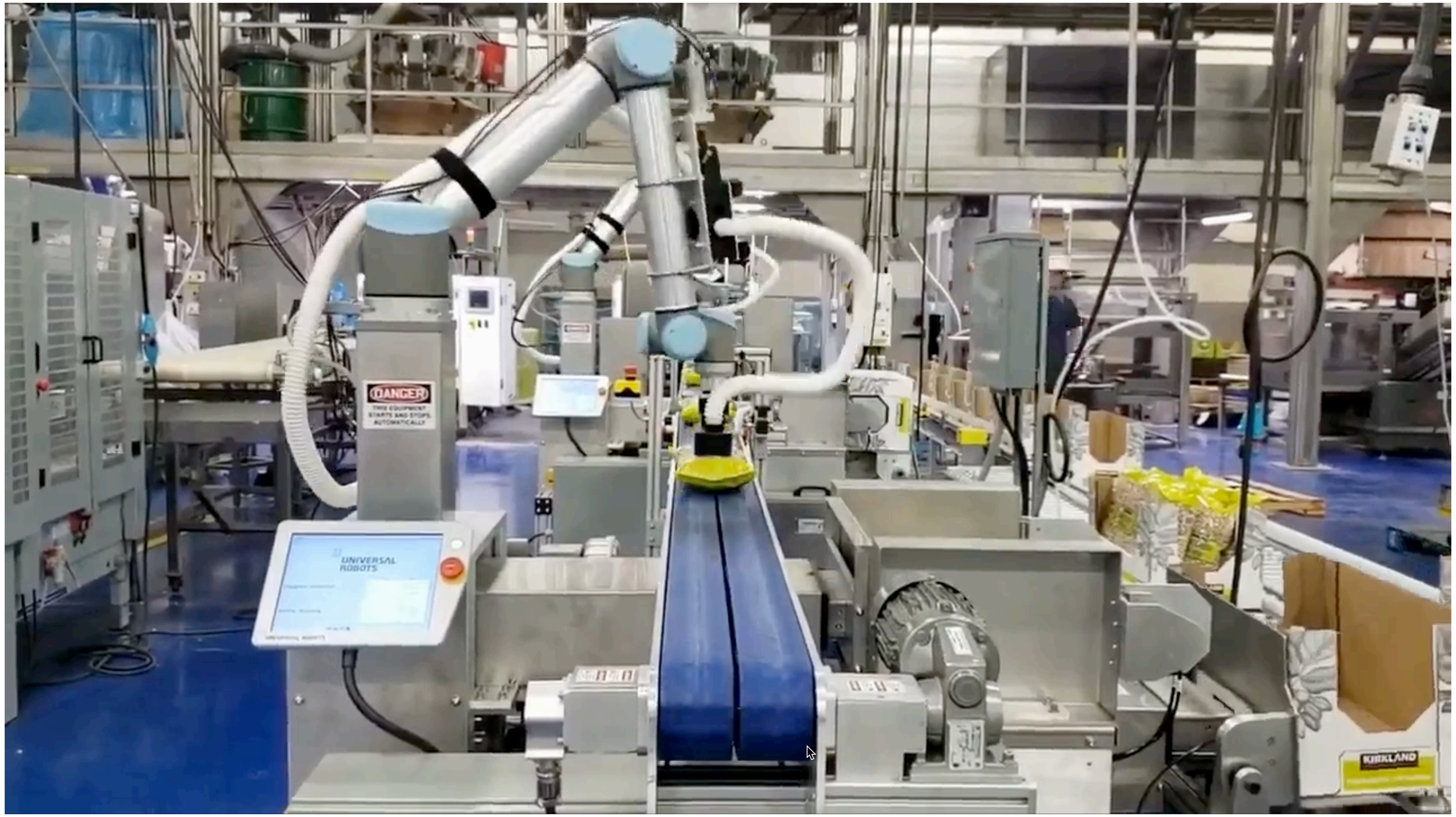
User: Will the practice cause pain to the patient in the image?

VILA: The image shows a group of people, including nurses and a mannequin, practicing CPR on a mannequin. While practicing CPR is essential for healthcare professionals, it is important to note that it is a simulation and not a real patient. The practice does not cause pain to the patient in the image, as it is a mannequin and not a living person.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

VILA for Smart Factory

Zero-shot learning: monitoring factory production lines from video



User : How many chip bags are picked by robot arm:

VILA: Two chip bags are picked by robot arm

User: The 8 input images are sampled at 2 frames per second. How long does it take for the robot arm to pick up one chip bag?

VILA : The robot arm takes 1.5 seconds to pick up the chip bag.

VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

VILA for defect inspection

Zero-shot learning: crack detection

User: Is the building in healthy condition?

VILA: The building appears to be in a state of disrepair, with visible cracks and chips in the paint. This suggests that the building may be old, poorly maintained, or has experienced some form of damage or wear over time. The presence of these structural issues could potentially pose safety risks or negatively impact the building's overall appearance and value.

User: What are the likely service needed for this building?

VILA: The building in the image has a cracked wall and a damaged roof, which indicates that it requires repair and maintenance services. The cracked wall may require repairs to ensure structural integrity and prevent further damage. The damaged roof may need to be replaced or repaired to prevent water leakage and potential damage to the building's interior. Additionally, the building may need to be inspected for any other potential issues that may arise due to the damage.

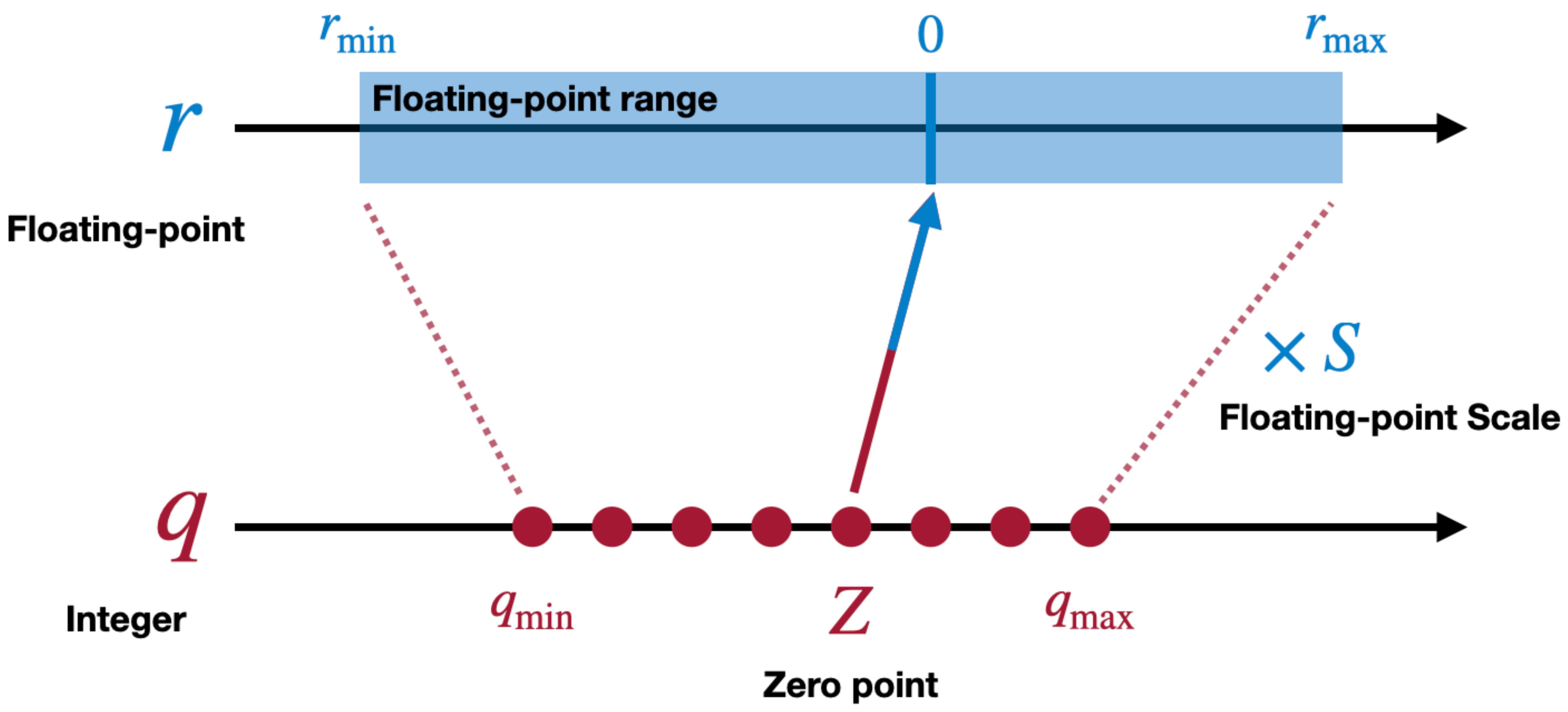
VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

LLM Compression: Lower-bit Quantization

Quantization can reduce deployment costs

Quantization lowers the bit-width and improves efficiency

- Serving a 175B GPT-3 model at least requires:
  - FP16: 350GB memory ➡️ 5 x 80GB A100 GPUs
  - **INT8:** 175GB memory ➡️ **3 x 80GB A100 GPUs**



VILA: On Pre-training for Visual Language Models [Lin et al., CVPR 2024]

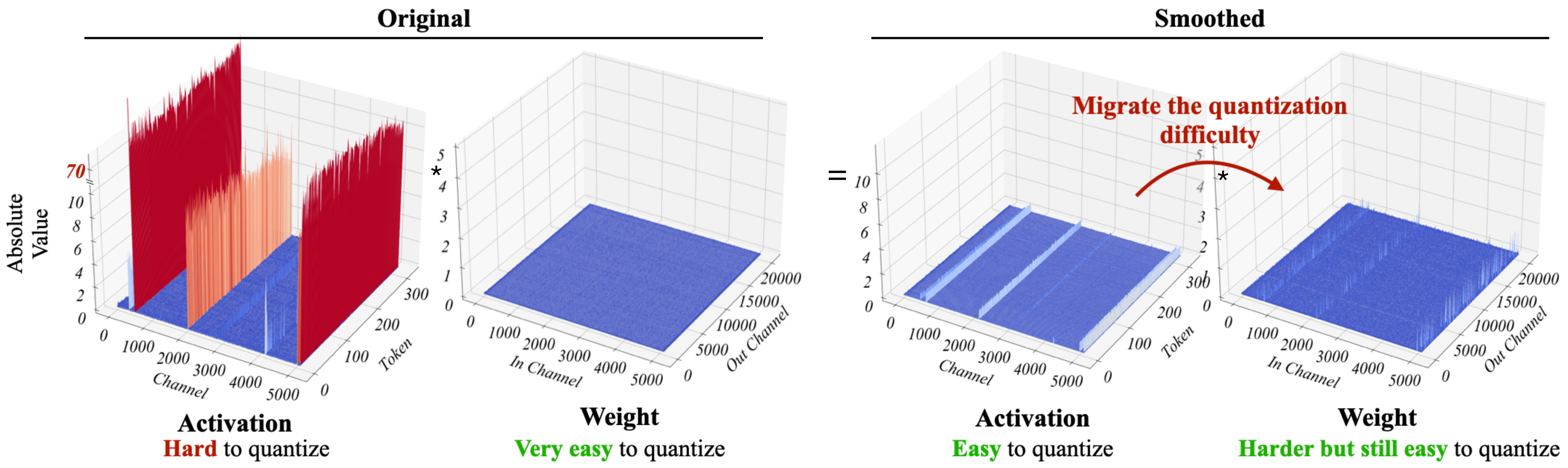
SmoothQuant: W8A8 Quantization for Cloud

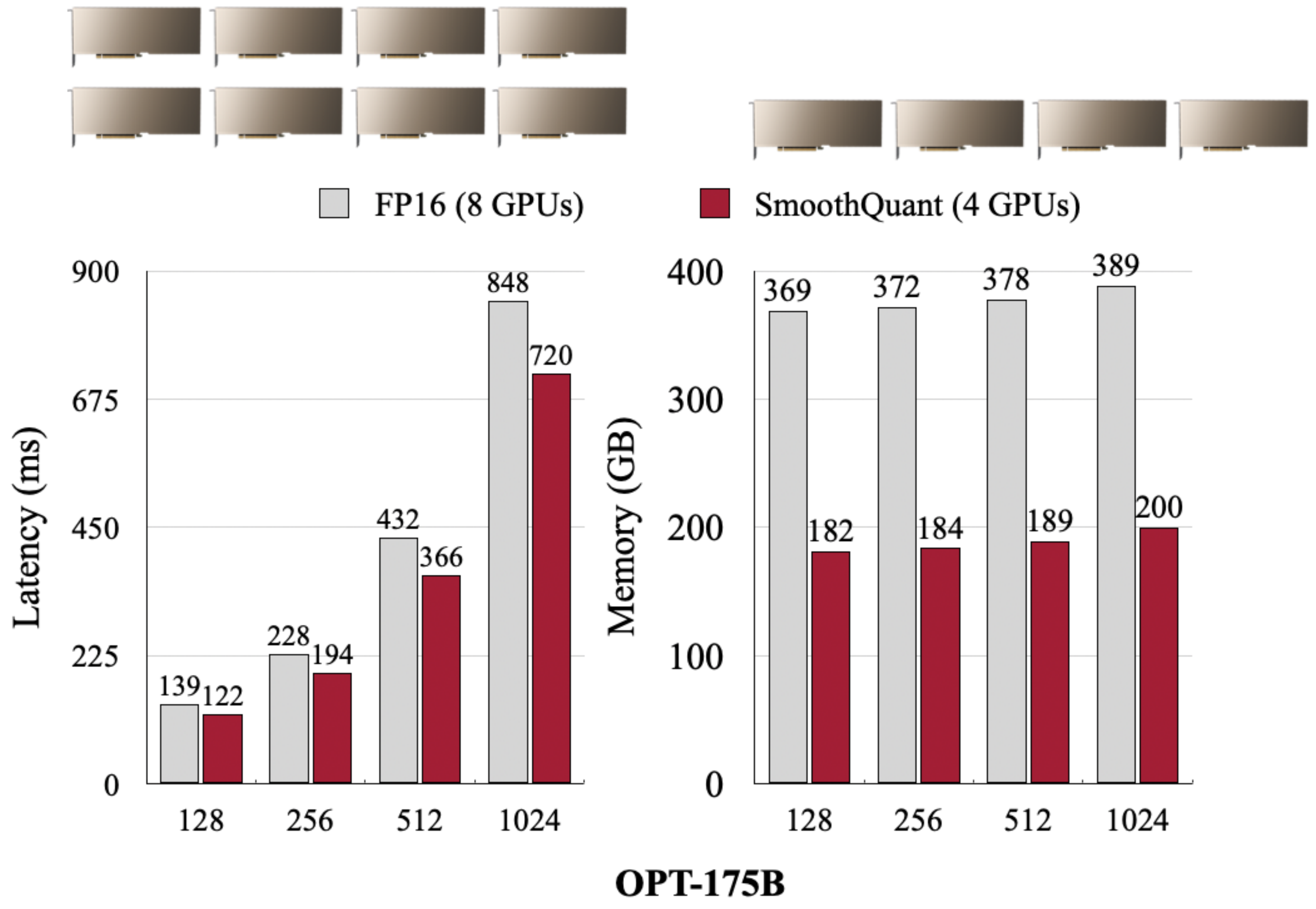
SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models

Goal: Quantize LLM to lower precision, both activation and weight

Challenge: activation channels have many outliers, wasting the dynamic range (many channels became zero)

Our Solution: Smooth the activations: 100*1= 10*10; Equalize the quantization difficult from activation to weights.



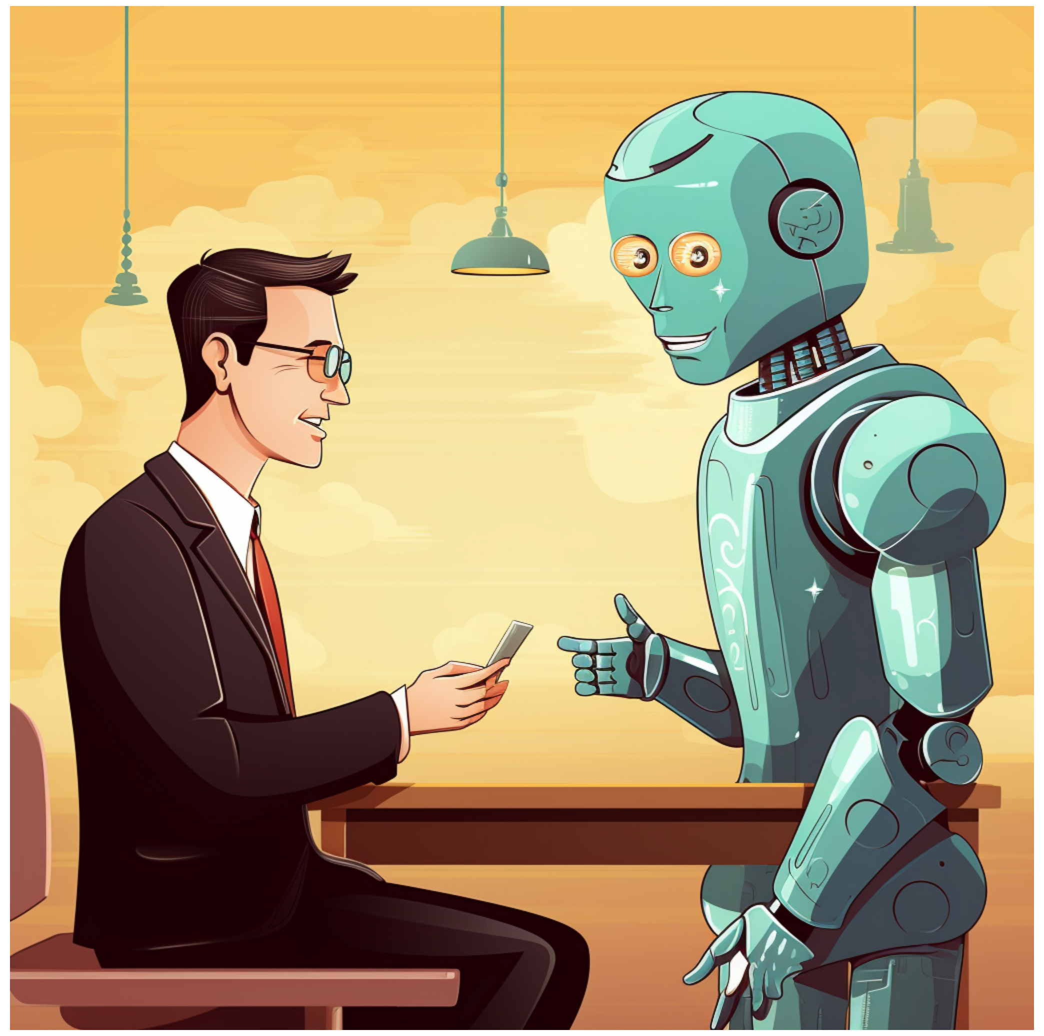


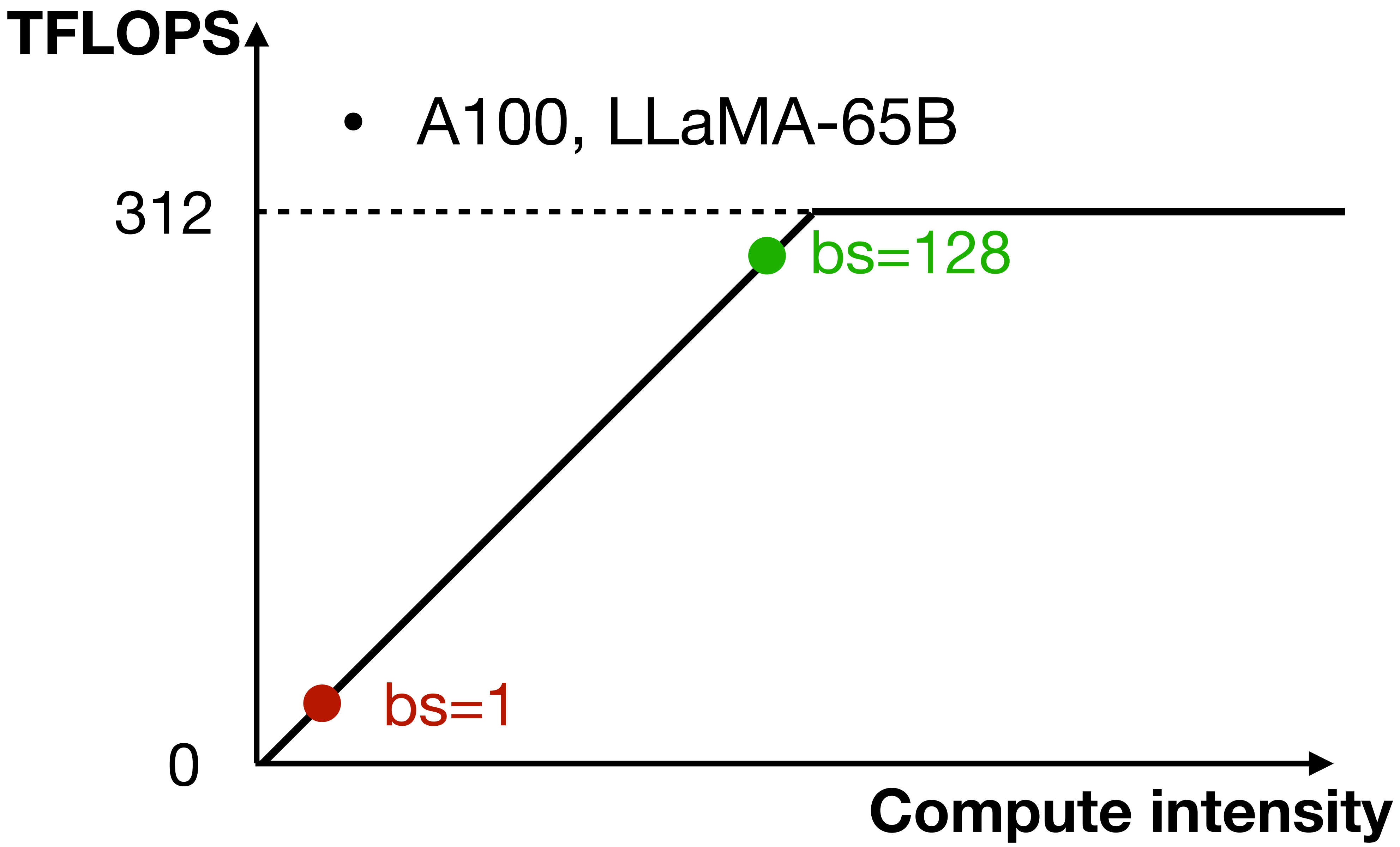
SmoothQuant: Accurate and Efficient Post-Training Quantization for Large Language Models (Xiao *et al.*, ICML 2023)

W4A16 for Single Batch Serving

W8A8 cannot address low computational intensity of decoding

- W8A8 quantization improves arithmetic efficiency and memory efficiency by **2x** compared to FP16. Is it enough?
- But single-query LLM inference (e.g., local) is still highly memory-bounded.
- We need **low-bit weight-only quantization (e.g., W4A16)**

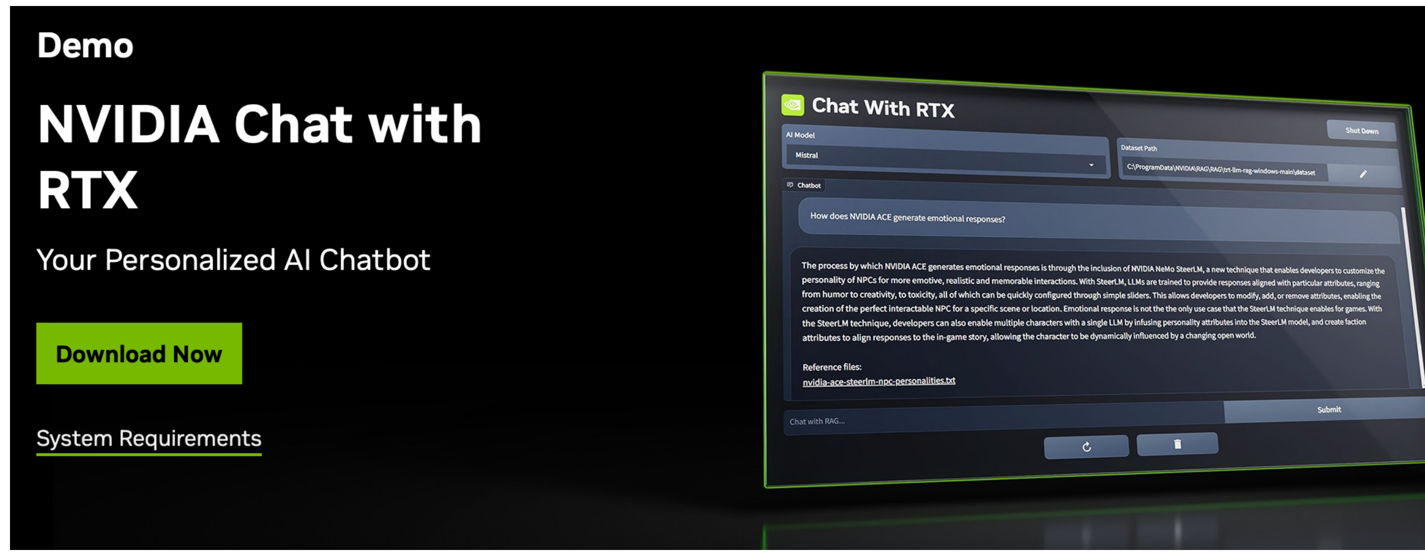




- LLaMA-65B GEMV $[1, 8192] \times [8192, 8192]$
- NVIDIA A100 GPU 80GB: 312TFLOPS (int8), 2000GB/s
- Computational intensity: $\frac{\text{FLOP}}{\text{Byte}} \quad \frac{8192^2}{8192^2} << \frac{312}{2000} \times 10^3$
- Highly memory-bounded (~10^2 gap)!

AWQ for On-Device LLM

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration

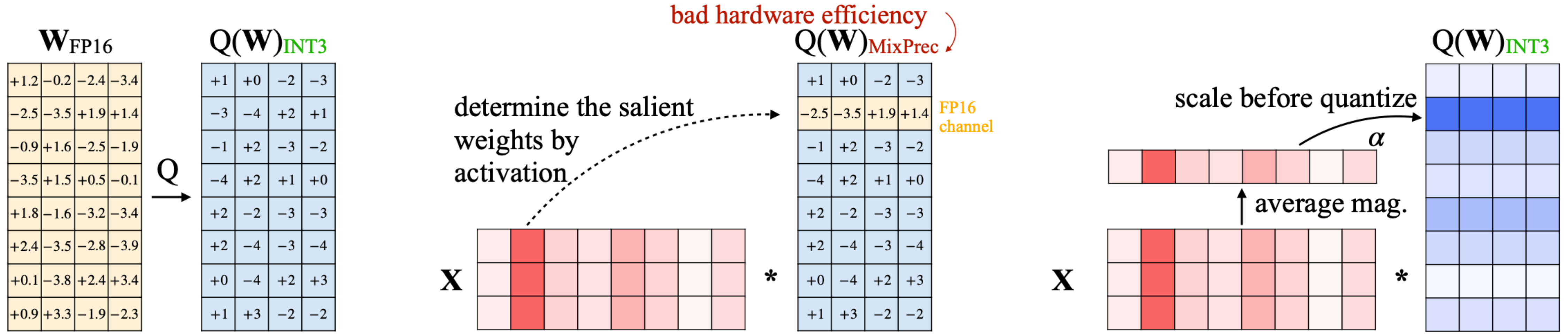


Goal: deploy LLM on the edge: Jetson Orin, AI PC

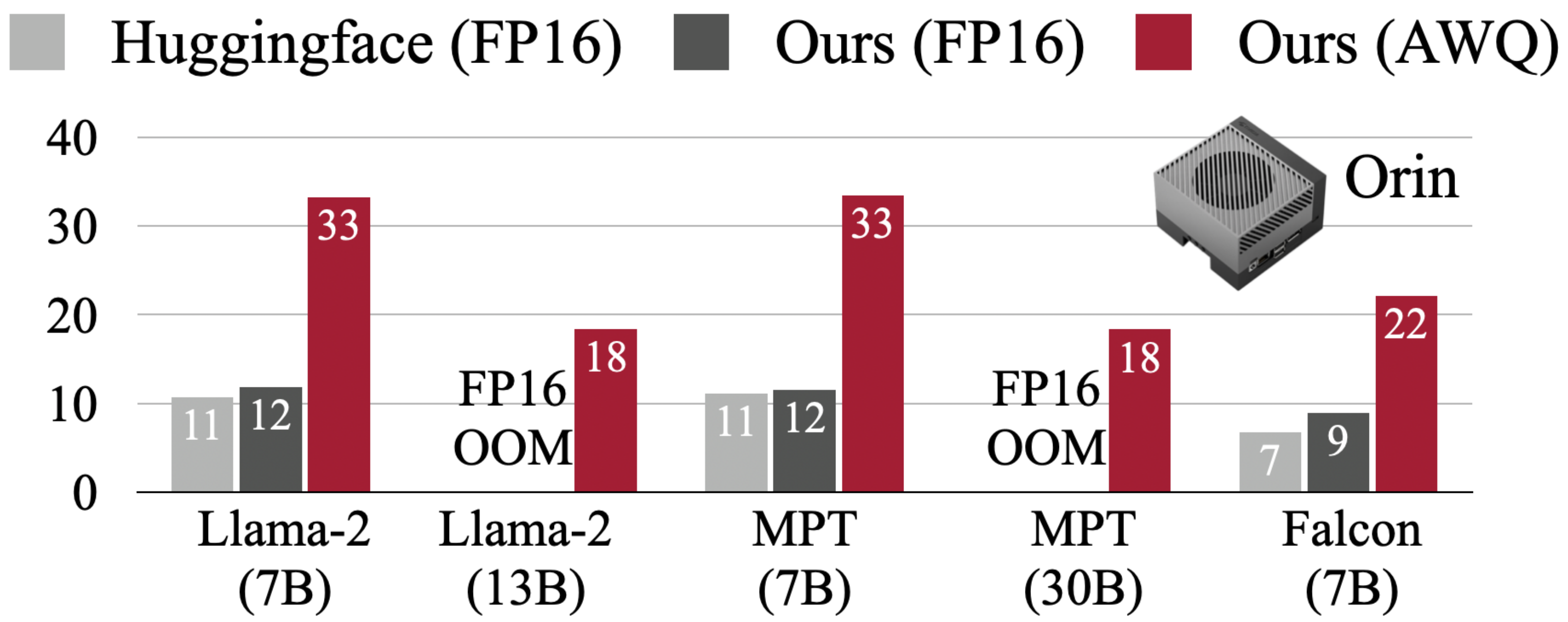
Challenge: weight memory bounded @low batch size; can't fit; idle ALU.

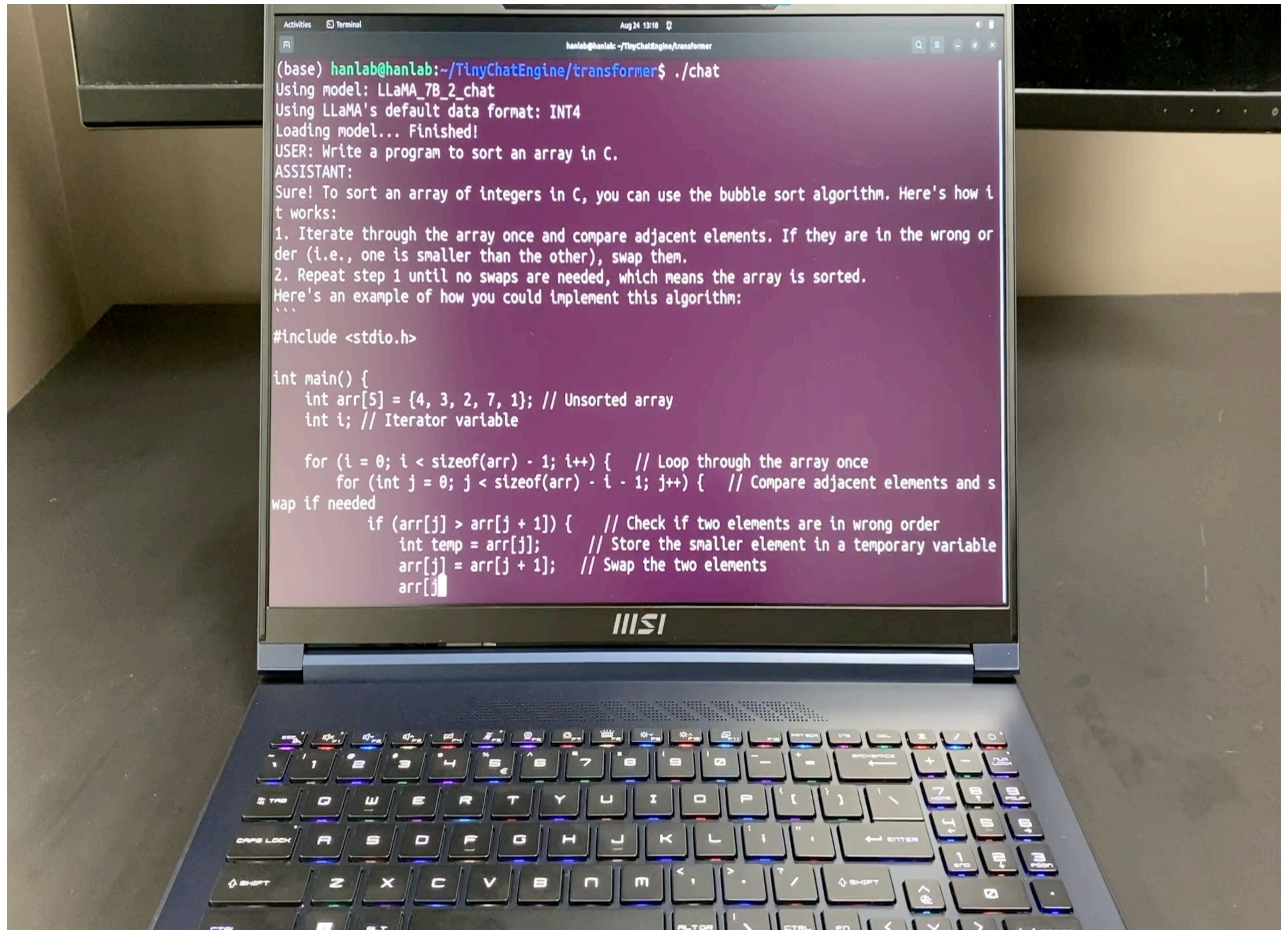
Our Solution: 4bit weights, fp16 activation, fp16 arithmetic.

Activation-awareness: preserve the salient weight channel by scaling according to the activation magnitude.



(a) RTN quantization (**PPL** 43.2) (b) Keep 1% salient weights in FP16 (**PPL** 13.0) (c) Scale the weights before quantization (**PPL** 13.0)





AWQ [Lin et al., MLSys 2024]

AWQ: Activation-aware Weight Quantization

Targeting group-wise low-bit weight-only quantization (W4A16)

$$\mathbf{W}_{\text{FP16}} = \begin{bmatrix} +1.2 & -0.2 & -2.4 & -3.4 \\ -2.5 & -3.5 & +1.9 & +1.4 \\ -0.9 & +1.6 & -2.5 & -1.9 \\ -3.5 & +1.5 & +0.5 & -0.1 \\ +1.8 & -1.6 & -3.2 & -3.4 \\ +2.4 & -3.5 & -2.8 & -3.9 \\ +0.1 & -3.8 & +2.4 & +3.4 \\ +0.9 & +3.3 & -1.9 & -2.3 \end{bmatrix} \xrightarrow{\text{RTN}} Q(\mathbf{W})_{\text{INT3}} = \begin{bmatrix} +1 & +0 & -2 & -3 \\ -3 & -4 & +2 & +1 \\ -1 & +2 & -3 & -2 \\ -4 & +2 & +1 & +0 \\ +2 & -2 & -3 & -3 \\ +2 & -4 & -3 & -4 \\ +0 & -4 & +2 & +3 \\ +1 & +3 & -2 & -2 \end{bmatrix}$$

- Weight-only quantization reduces the memory requirement, and accelerates token generation by alleviating the memory bottleneck.
- Group-wise/block-wise quantization (e.g., 64/128/256) offers a better accuracy-model size trade-off.

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ: Activation-aware Weight Quantization

Targeting group-wise low-bit weight-only quantization (W4A16)

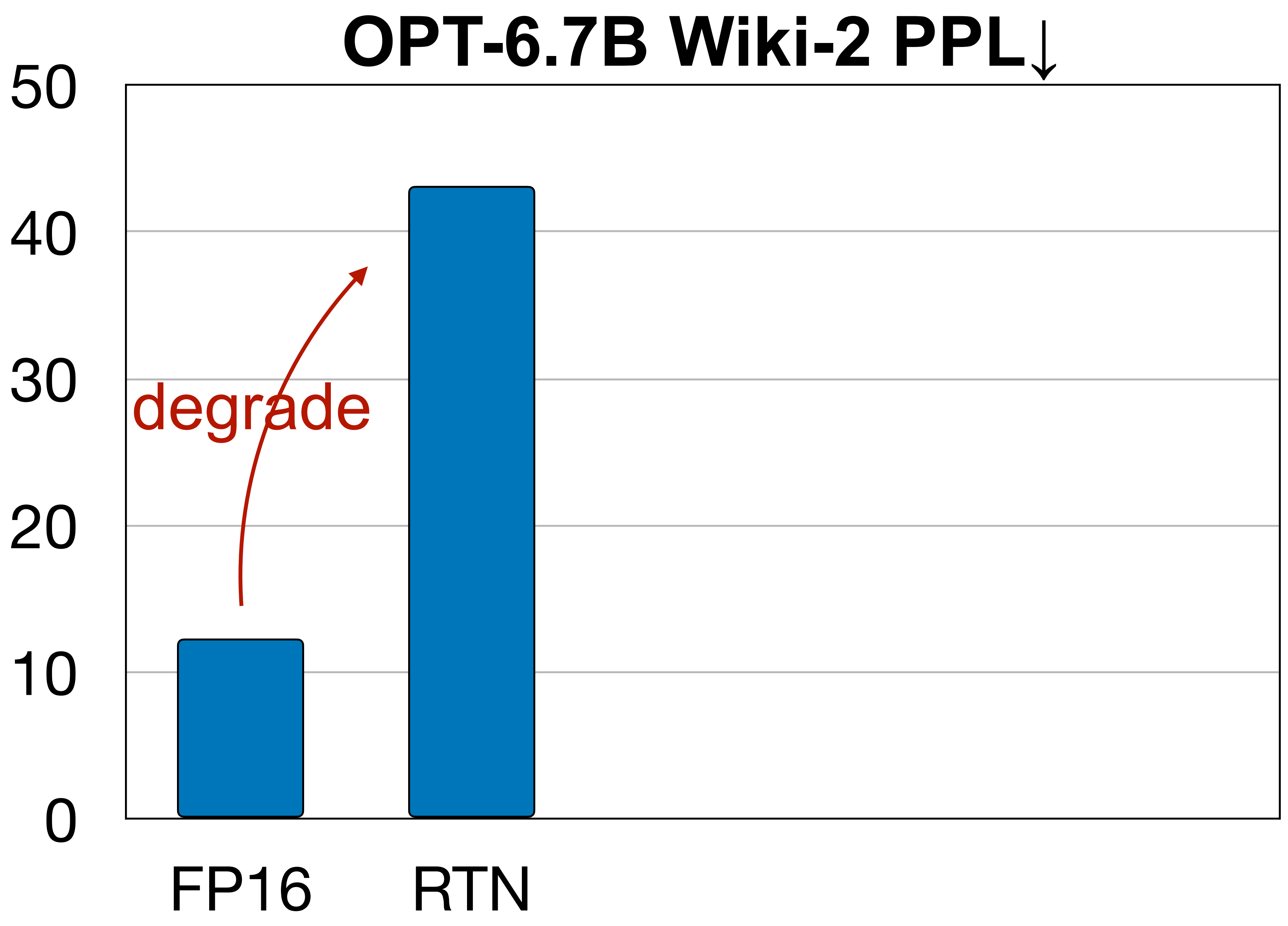
$\mathbf{W}_{FP16}$

+1.2	−0.2	−2.4	−3.4
−2.5	−3.5	+1.9	+1.4
−0.9	+1.6	−2.5	−1.9
−3.5	+1.5	+0.5	−0.1
+1.8	−1.6	−3.2	−3.4
+2.4	−3.5	−2.8	−3.9
+0.1	−3.8	+2.4	+3.4
+0.9	+3.3	−1.9	−2.3

$\xrightarrow{\text{RTN}}$

$Q(\mathbf{W})_{INT3}$

+1	+0	−2	−3
−3	−4	+2	+1
−1	+2	−3	−2
−4	+2	+1	+0
+2	−2	−3	−3
+2	−4	−3	−4
+0	−4	+2	+3
+1	+3	−2	−2



- Weight-only quantization reduces the memory requirement, and accelerates token generation by alleviating the memory bottleneck.
- Group-wise/block-wise quantization (e.g., 64/128/256) offers a better accuracy-model size trade-off.
- But there is still a performance gap with round-to-nearest (RTN) quantization (INT3-g128)

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ: Activation-aware Weight Quantization

Observation: Weights are not equally important; 1% salient weights

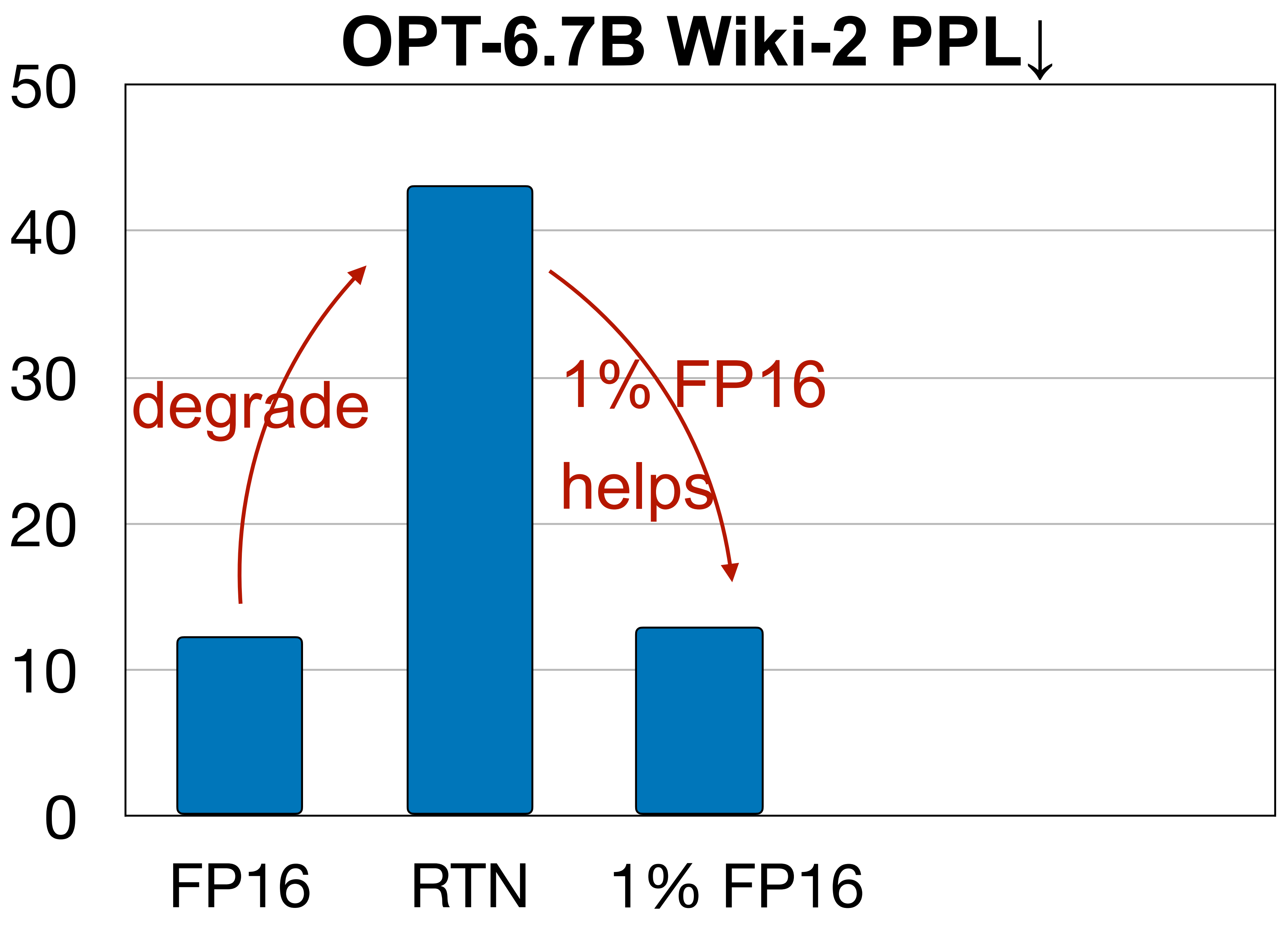
$\mathbf{W}_{\text{FP16}}$

+1.2	−0.2	−2.4	−3.4
−2.5	−3.5	+1.9	+1.4
−0.9	+1.6	−2.5	−1.9
−3.5	+1.5	+0.5	−0.1
+1.8	−1.6	−3.2	−3.4
+2.4	−3.5	−2.8	−3.9
+0.1	−3.8	+2.4	+3.4
+0.9	+3.3	−1.9	−2.3

RTN →

$Q(\mathbf{W})_{\text{MixPrec}}$

+1	+0	−2	−3	
−2.5	−3.5	+1.9	+1.4	FP16 channel
−1	+2	−3	−2	
−4	+2	+1	+0	
+2	−2	−3	−3	
+2	−4	−3	−4	
+0	−4	+2	+3	
+1	+3	−2	−2	



- We find that weights are not equally important, keeping **only 1%** of salient weight channels in FP16 can greatly improve perplexity

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ: Activation-aware Weight Quantization

Observation: Weights are not equally important; 1% salient weights

$\mathbf{W}_{\text{FP16}}$

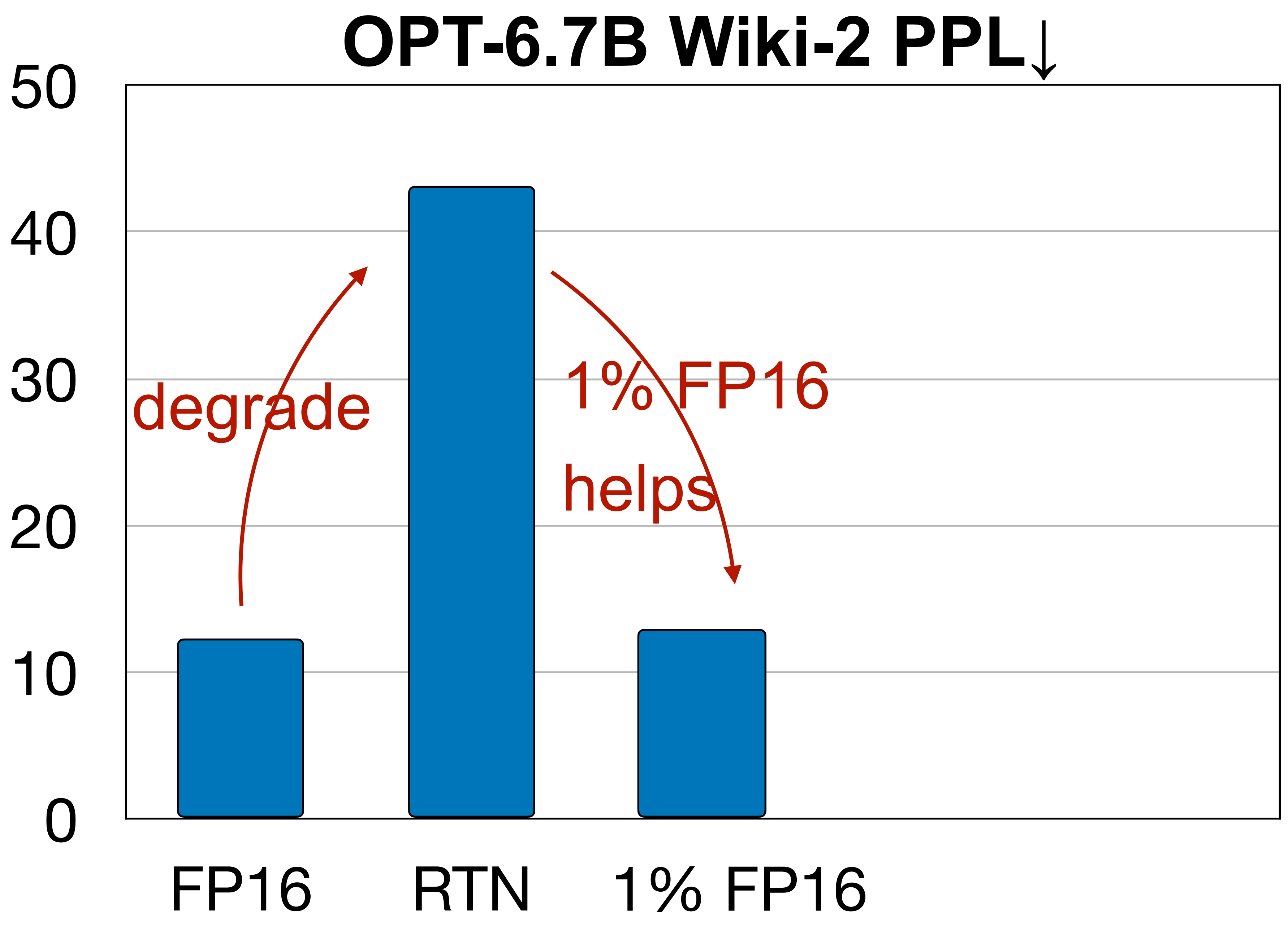
+1.2	−0.2	−2.4	−3.4
−2.5	−3.5	+1.9	+1.4
−0.9	+1.6	−2.5	−1.9
−3.5	+1.5	+0.5	−0.1
+1.8	−1.6	−3.2	−3.4
+2.4	−3.5	−2.8	−3.9
+0.1	−3.8	+2.4	+3.4
+0.9	+3.3	−1.9	−2.3

$\xrightarrow{\text{RTN}}$

$Q(\mathbf{W})_{\text{MixPrec}}$

+1	+0	−2	−3
−2.5	−3.5	+1.9	+1.4
−1	+2	−3	−2
−4	+2	+1	+0
+2	−2	−3	−3
+2	−4	−3	−4
+0	−4	+2	+3
+1	+3	−2	−2

FP16 channel

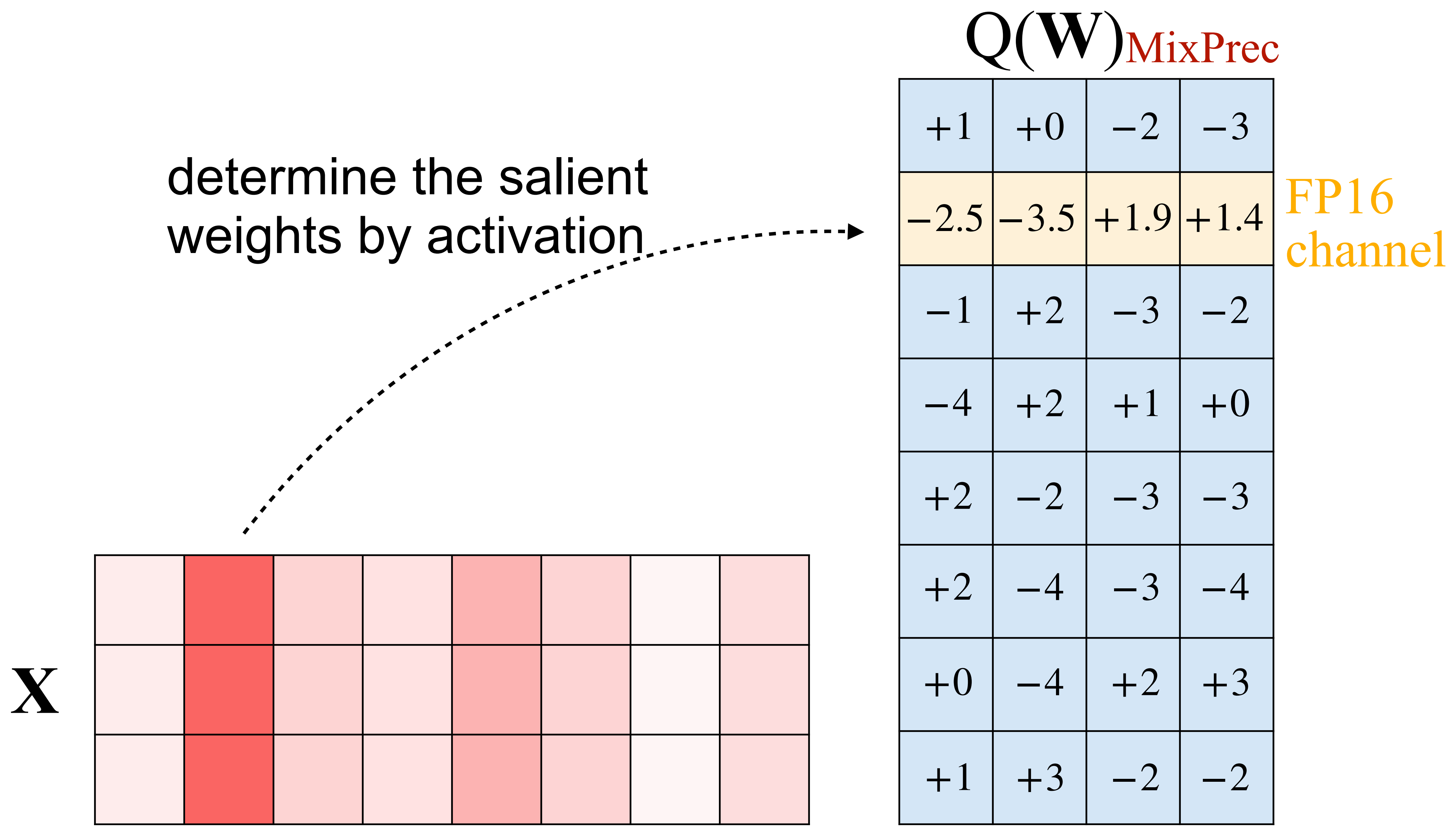


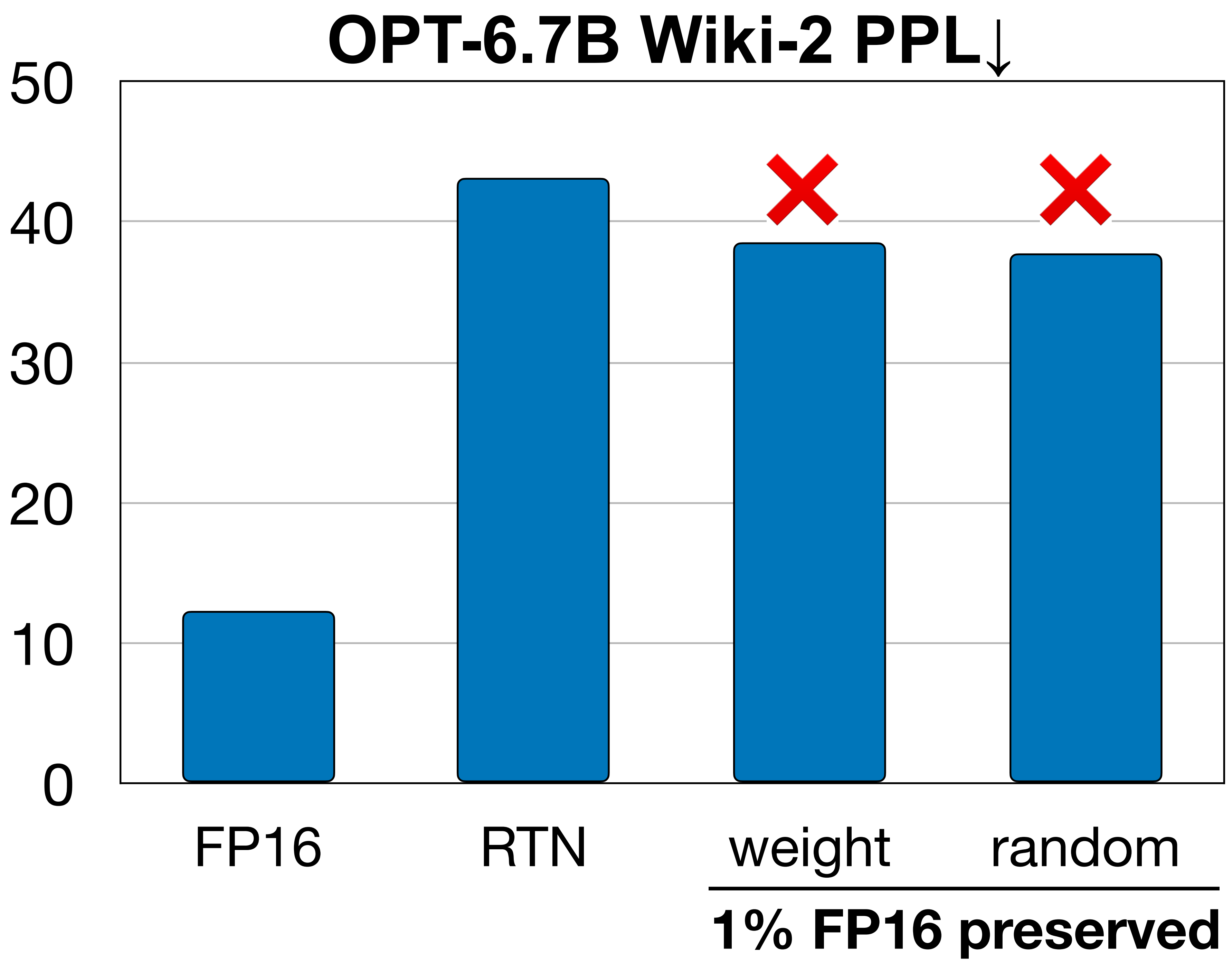
- We find that weights are not equally important, keeping **only 1%** of salient weight channels in FP16 can greatly improve perplexity
- But how do we select salient channels? Should we select based on weight magnitude?

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ for Low-bit Weight-only Quantization

Salient weights are determined by activation distribution, not weight



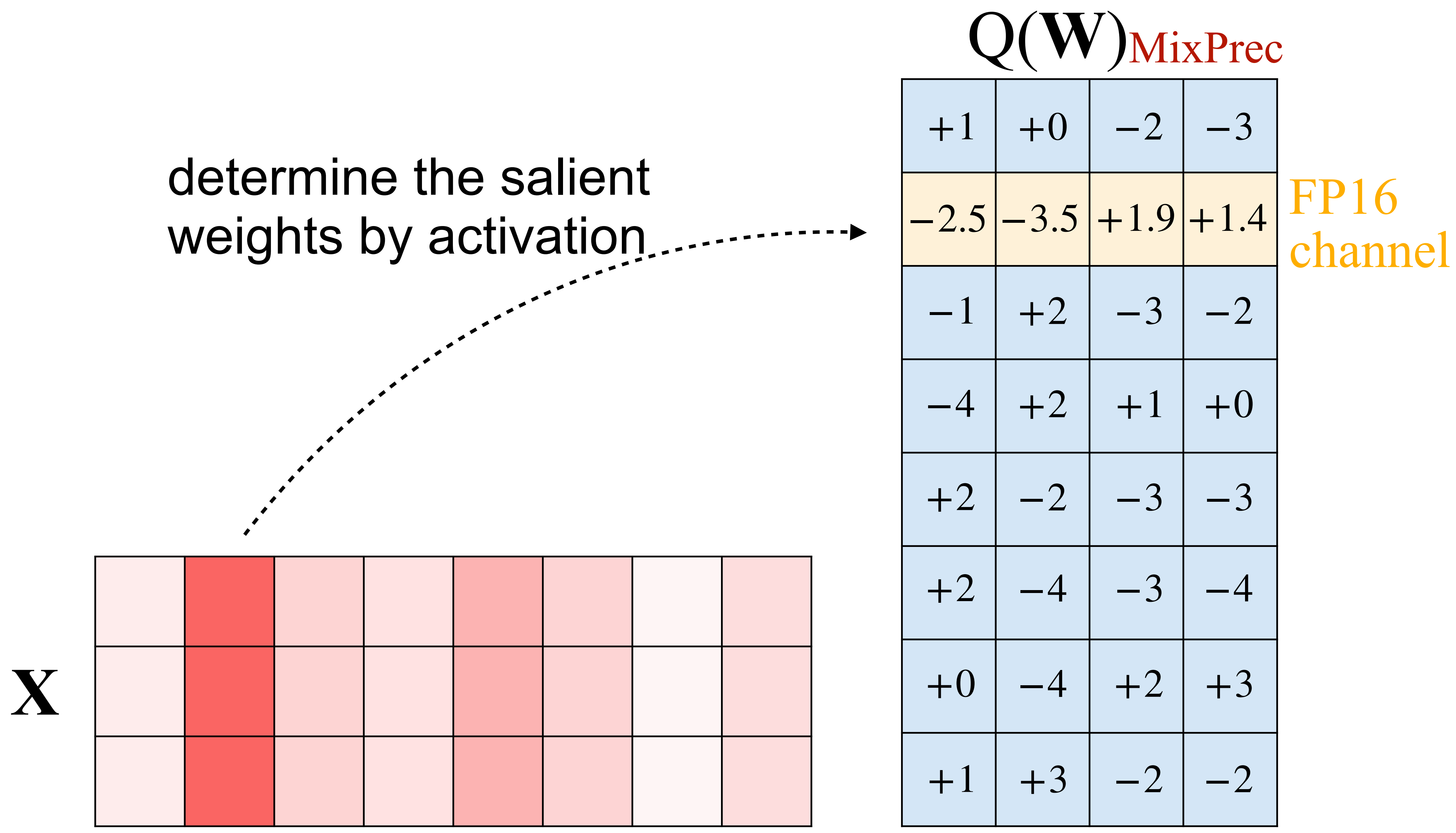


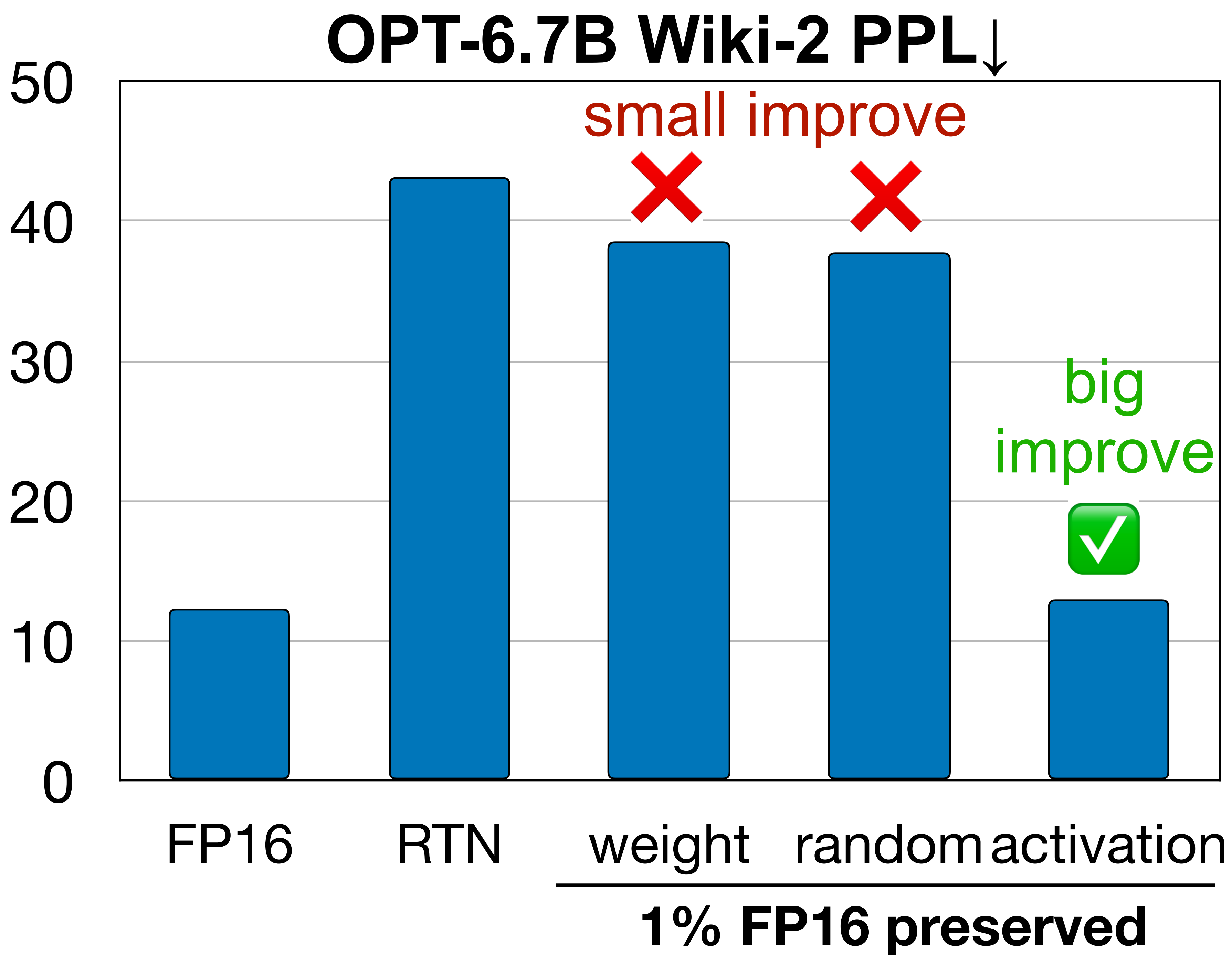
- We find that weights are not equally important, keeping **only 1%** of salient weight channels in FP16 can greatly improve perplexity
- But how do we select salient channels? Should we select based on weight magnitude?
- **This is not the truth!**

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ for Low-bit Weight-only Quantization

Salient weights are determined by activation distribution, not weight





- But how do we select salient channels? Should we select based on weight magnitude?
- No! We should look for **activation distribution, but not weight**! **(Activation has outliers!)**
- **However, 1% FP16 is not hardware-friendly.**

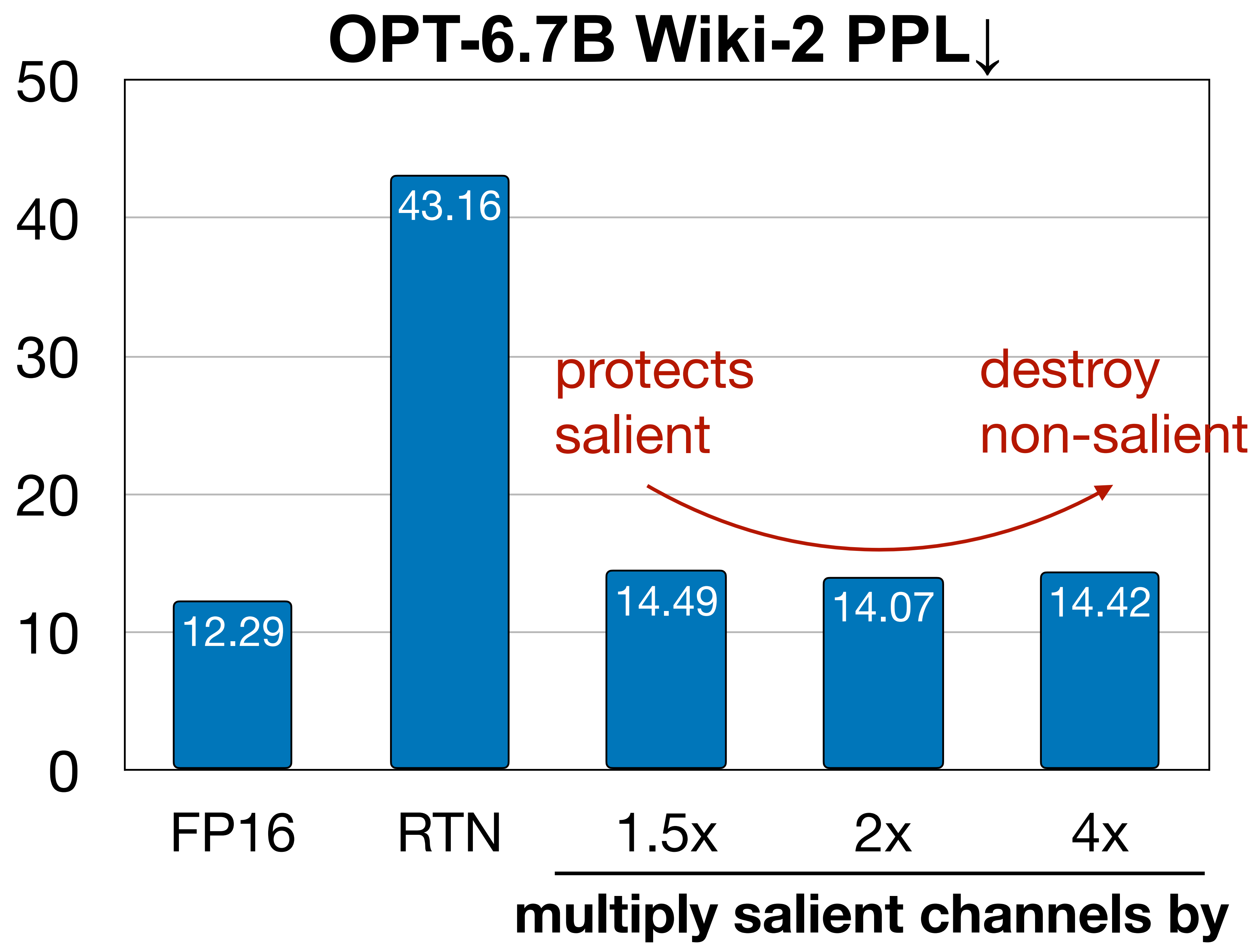
AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ for Low-bit Weight-only Quantization

Protecting salient weights by scaling (no mixed prec.)

fuse to previous op

$$\mathbf{W}\mathbf{X} \longrightarrow Q(\mathbf{W}\cdot\mathbf{s})(\mathbf{s}^{-1}\cdot\mathbf{X})$$



- We need to consider **activation-awareness** for salient channels.
- We solve for best hyper-parameters with a simple grid search.

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ for Low-bit Weight-only Quantization

Protecting salient weights by scaling (no mixed prec.)

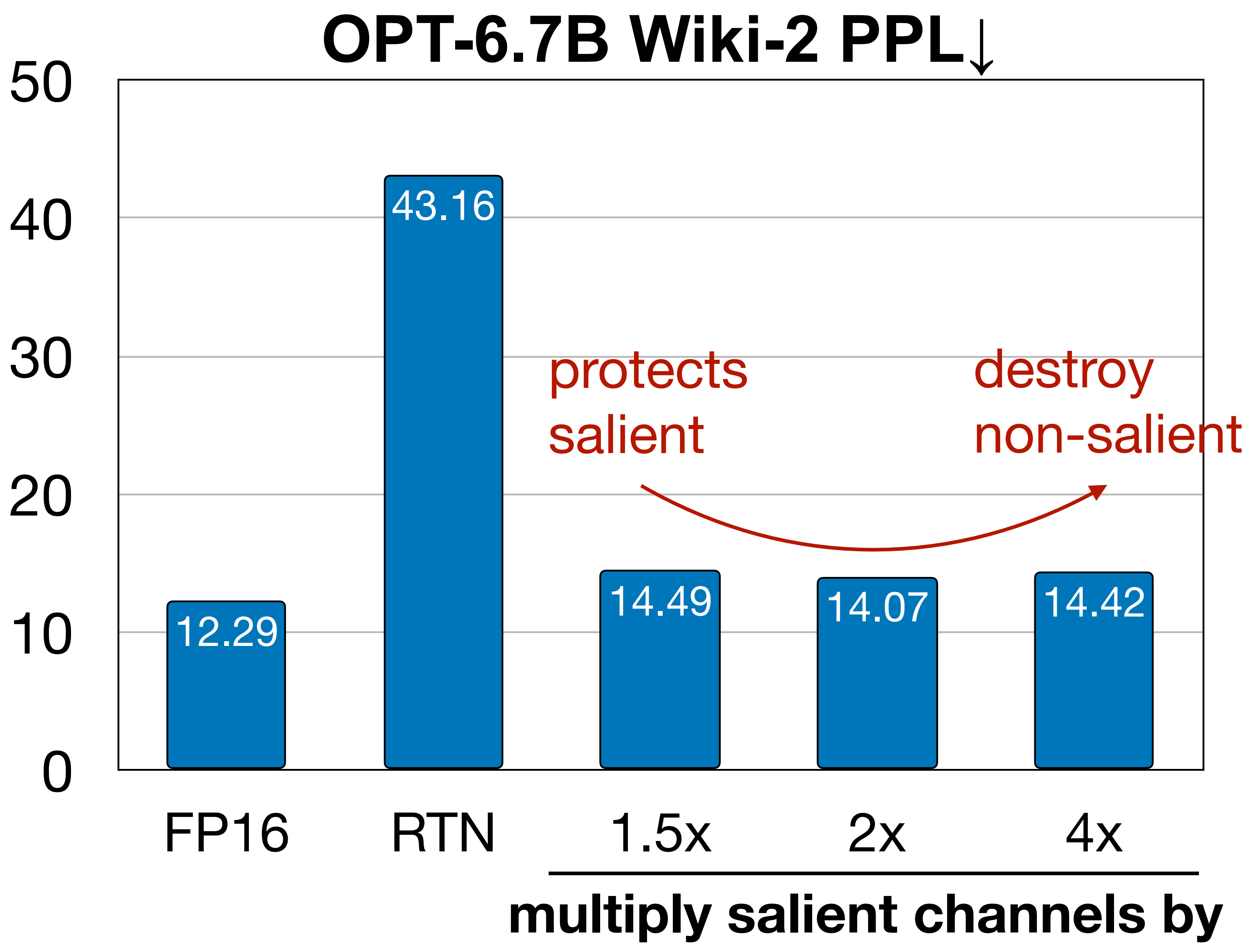
fuse to previous op

$$\mathbf{WX} \longrightarrow Q(\mathbf{W} \cdot \mathbf{s})(\mathbf{s}^{-1} \cdot \mathbf{X})$$

$$\mathcal{L}(\mathbf{s}) = \|Q(\mathbf{W} \cdot \mathbf{s})(\mathbf{s}^{-1} \cdot \mathbf{X}) - \mathbf{WX}\|$$

$$\mathbf{s} = \mathbf{s_X}^{\alpha}, \quad \alpha^* = \arg\min_{\alpha} \mathcal{L}(\mathbf{s_X}^{\alpha})$$

avg. magnitude

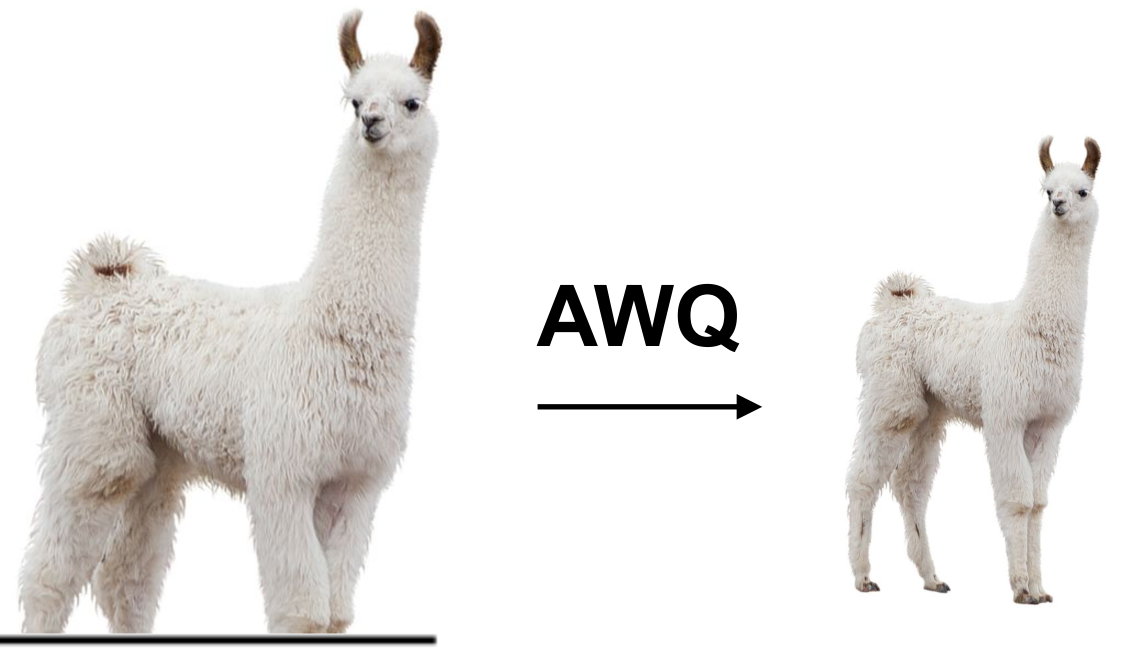


- We need to consider **activation-awareness** for salient channels.
- We solve for best hyper-parameters with a simple grid search.

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ Results

Improving general LLM quantization (LLaMA & OPT)



LLaMA Faimily		MMLU (5-shot) average ↑				Common Sense QA (0-shot) average ↑			
		7B	13B	30B	65B	7B	13B	30B	65B
FP16	-	38.41%	45.21%	56.84%	60.50%	67.30%	70.65%	72.97%	74.49%
INT3 g128	RTN	33.43%	39.20%	50.58%	57.77%	64.55%	68.63%	72.07%	72.58%
	GPTQ	30.53%	40.90%	52.32%	58.04%	59.66%	68.71%	70.77%	73.03%
	AWQ	**35.43%**	**41.84%**	**53.22%**	**58.83%**	**65.53%**	**69.22%**	**72.10%**	**73.39%**

OPT / PPL↓		125M	1.3B	2.7B	6.7B	13B	30B	66B
FP16	-	31.95	16.41	14.32	12.29	11.5	10.67	10.09
INT3 g128	RTN	58.49	206.54	595.28	43.16	45.37	28.84	423.39
	GPTQ	41.93	**18.53**	15.79	13.13	**12.01**	**11.00**	11.48
	AWQ	**41.10**	**18.53**	**15.62**	**12.99**	12.03	11.03	**10.46**
INT4 g128	RTN	35.51	17.70	15.12	13.02	11.89	11.00	10.44
	GPTQ	34.23	16.92	14.69	12.51	**11.60**	**10.74**	10.24
	AWQ	**33.96**	**16.85**	**14.61**	**12.44**	**11.60**	10.75	**10.16**

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ Results

Quantization of multi-modal LMs (OpenFlamingo, captioning)

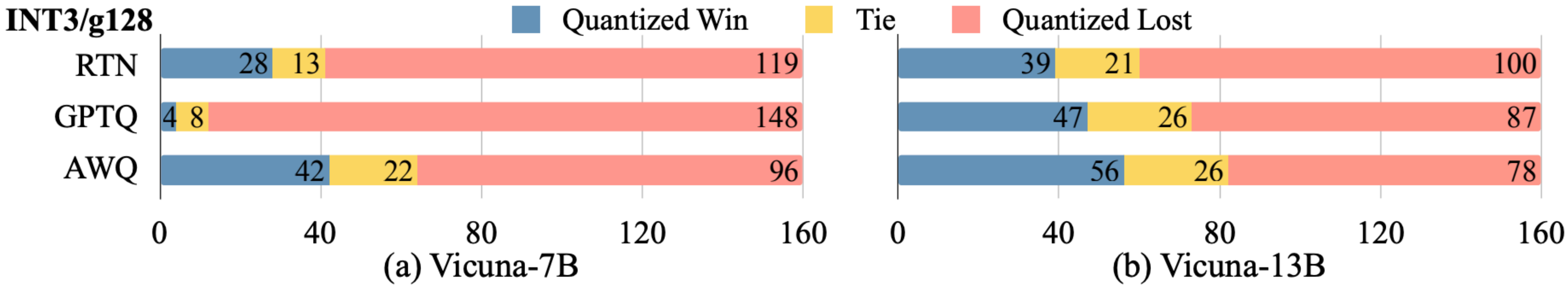
COCO (CIDEr ↑)		0-shot	4-shot	8-shot	16-shot	32-shot	*Δ(32-shot)*
FP16	-	63.73	72.18	76.95	79.74	81.70	-
INT4 g128	RTN	60.24	68.07	72.46	74.09	77.13	-4.57
	GPTQ	59.72	67.68	72.53	74.98	74.98	-6.72
	AWQ	**62.57**	**71.02**	**74.75**	**78.23**	**80.53**	**-1.17**
INT3 g128	RTN	46.07	55.13	60.46	63.21	64.79	-16.91
	GPTQ	29.84	50.77	56.55	60.54	64.77	-16.93
	AWQ	**56.33**	**64.73**	**68.79**	**72.86**	**74.47**	**-7.23**

- Improved quantized performance for both 4-bit and 3-bit quantization.
- Big improvement even under **4-bit** quantization (not very noticeable with QA benchmarks)

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ Results

Quantization of instruction-tuned models



- Comparing quantized Vicuna with FP16 counterparts
- Test under both orderings (quantized vs FP16, FP16 vs quantized) to get rid of ordering bias

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ Results

Quantization of multi-modal LMs (OpenFlamingo, captioning)

W4-RTN: A model airplane flying in the sky.
W4-AWQ: Two toy airplanes sit on a grass field.

W4-RTN: A man is holding a baby elephant in his arms.
W4-AWQ: A man and his daughter pose with an elephant.

W4-RTN: A man and a dog walking past some bushes.
W4-AWQ: Two dogs are walking on the street.

- Improved quantized performance for both 4-bit and 3-bit quantization.
- Big improvement even under **4-bit** quantization (not very noticeable with QA benchmarks)

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ Results

Quantization of multi-modal LMs (LLaVA, visual reasoning)

Question: Can you explain this meme in detail?
INT4-g128 RTN: The meme in the image features a plate of food, likely fried chicken, with a unique twist. There are small pictures of the Earth and other planets placed on top of the food, creating an interesting and amusing visual. Alongside the plate, a caption reads, "Sometimes I look at pictures of the Earth from space and I marvel at how beautiful it all is." This combination of food and planet images adds a touch of humor and curiosity, making the viewer wonder about the context and intent behind the meme.
INT4-g128 AWQ: The meme in the image is a lighthearted and humorous take on the concept of looking at pictures of the Earth from space. A plate of fried food, specifically chicken nuggets, is presented with the caption, "Sometimes I look at pictures of the Earth from space and I marvel at how beautiful it all is." The playful combination of a food item and an Earth-related context creates a fun and entertaining visual.

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

AWQ Results

Quantization of multi-modal LMs (LLaVA, visual reasoning)

Question: Do you know who drew this painting?
INT4-g128 RTN: The painting depicts a woman, who some believe to be a representation of the famous artist, Mona Lisa. However, it is not certain that the woman in the painting is indeed Mona Lisa. The artwork is a popular subject of artistic analysis and admiration, often considered a symbol of feminine beauty and enigma. The portrait is characterized by its intricate details, captivating the viewer with the woman's expressive gaze and the subtle, enigmatic smile that has become synonymous with the Mona Lisa.
INT4-g128 AWQ: The painting depicts a woman, who is believed to be a portrait of Mona Lisa, the famous artwork by the artist Leonardo da Vinci. Although the painting is widely recognized as a representation of Mona Lisa, it is not possible to confirm the artist who drew this particular painting without additional information. Leonardo da Vinci was a renowned artist known for his exceptional skill in painting detailed, realistic portraits, and the Mona Lisa is one of his most famous works.

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

SmoothQuant and AWQ Impacts

NVIDIA	TensorRT-LLM	https://github.com/NVIDIA/TensorRT-LLM#key-features		Transformer Quantization API	https://huggingface.co/docs/transformers/main_classes/quantization
IBM	Granite	IBM's internal code model, Granite, utilizes AWQ for quantization.		lmdeploy	https://github.com/InternLM/lmdeploy/blob/main/lmdeploy/lite/quantization/awq.py
Berkeley UNIVERSITY OF CALIFORNIA	vLLM	https://github.com/vllm-project/vllm/blob/main/vllm/model_executor/layers/quantization/awq.py	FriendliAI		https://friendli.ai/blog/Unlocking-Efficiency-of-Serving-LLMs-with-Activation-aware-Weight-Quantization-AWQ-on-PeriFlow/
	lm-sys/FastChat	https://github.com/lm-sys/FastChat/blob/main/docs/awq.md	replicate		https://github.com/replicate/vllm-with-loras/blob/main/vllm/model_executor/quantization_utils/awq.py

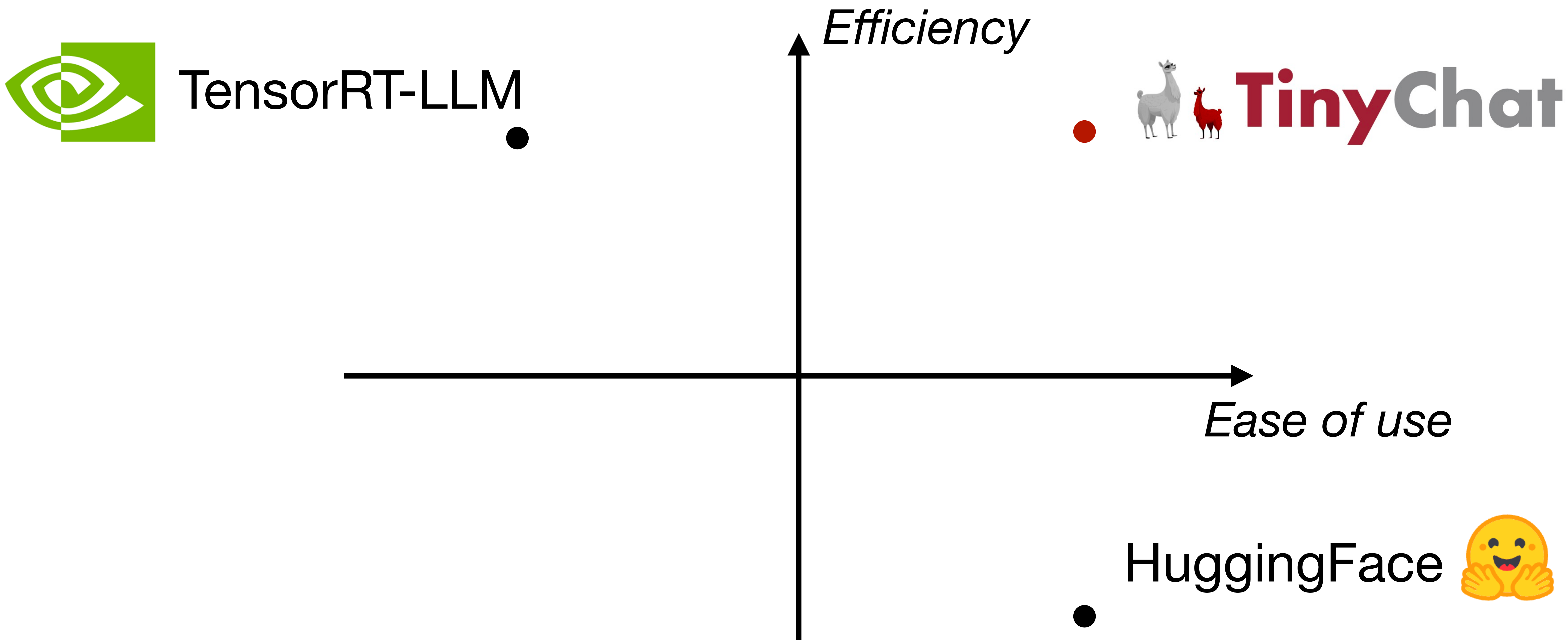
AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

TinyChat: Efficient LLMs Inference Engine

TinyChat: A Lightweight Serving Infra

Pythonic, lightweight, efficient

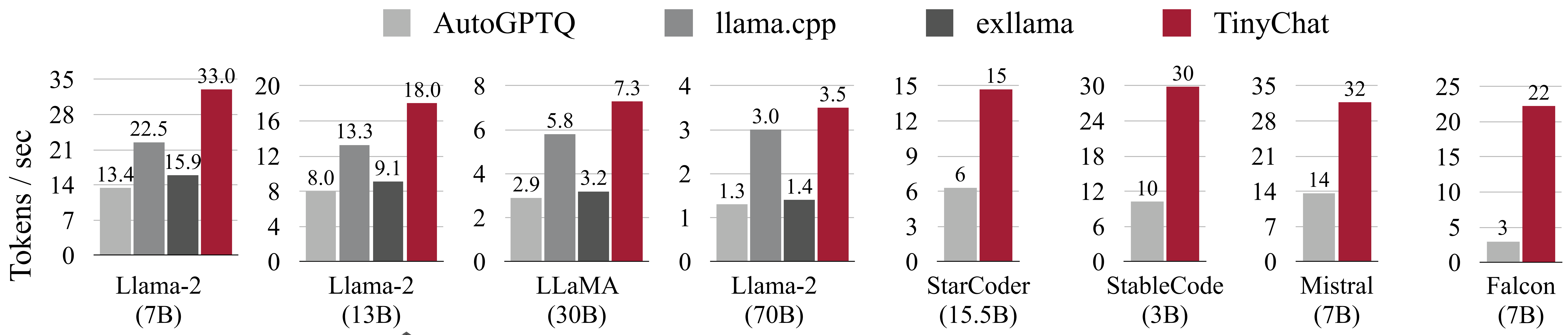
- We need a framework to serve the quantized model to achieve low latency
  - HuggingFace: easy to use, but slow
  - TensorRT-LLM: high efficiency, but harder to use
- **TinyChat**: efficient, lightweight, Python-native (composable with other stacks like vLLM)



AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

TinyChat: A Lightweight Serving Infra

Supporting a wide range of models on NVIDIA Jetson Orin



Latency comparison on Jetson Orin (64G) mobile GPU

- TinyChat achieves up to **1.5x** faster runtime for Meta's Llama models compared with systems specialized for this model.
- Compared with the only competitor that can support a diverse range of models, TinyChat is up to **7x** faster.
- Remarkably, TinyChat's front end is **fully PyTorch-based**.

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

TinyChat seamlessly supports VLMs

Accelerating visual-language models across different GPU platforms

- TinyChat also seamlessly supports VILA, delivering ~3x speedup over FP16 on Orin and allows interactive VLM deployment on the edge (laptops and AIoT).

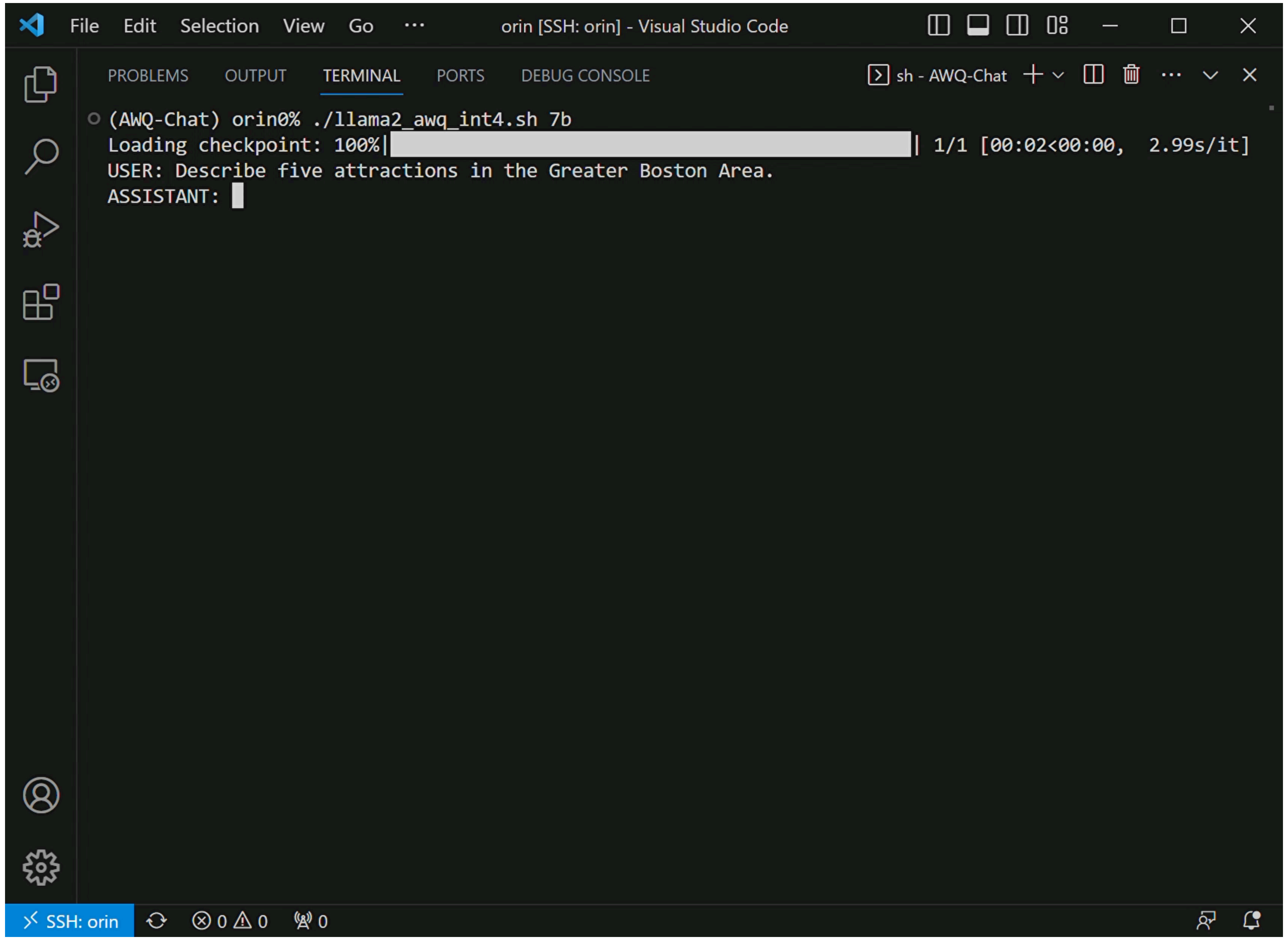
Model	Precision	A100 Tok/sec	4090 Tok / sec	Orin Tok / sec
VILA-7B	FP16	81.6	58.5	11.5
VILA-7B-AWQ	INT4	155.3	168.1	35.6
VILA-13B	FP16	48.5	OOM	6.1
VILA-13B-AWQ	INT4	102.1	99.0	17.5

AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

TinyChat: A Lightweight Serving Infra

Demo on AGX Orin (edge LLM inference)

- Orin Nano has 200GB/s memory bandwidth; even more memory-bounded
- Model size: 7B. ~**30 token/s** generation

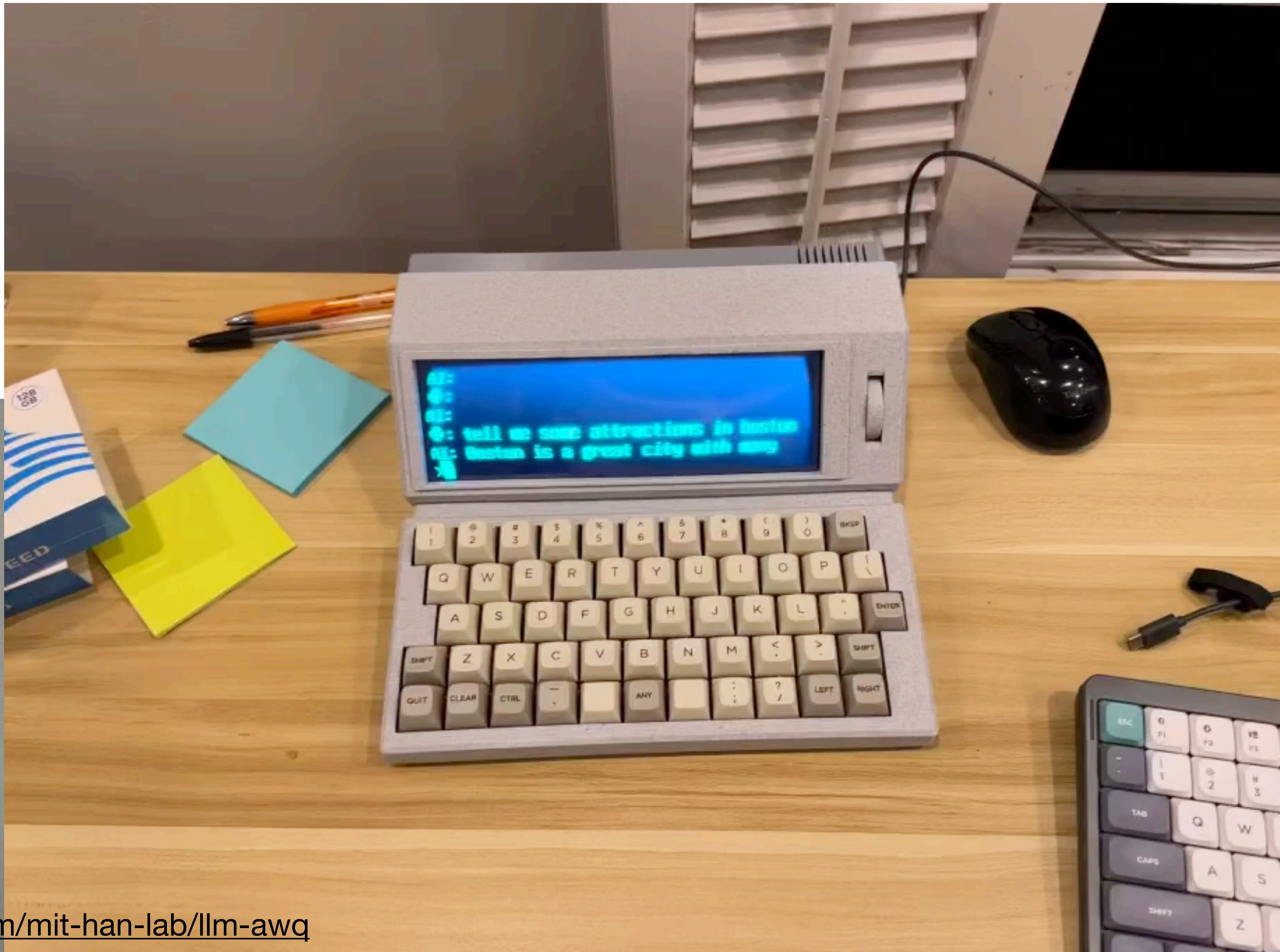


TinyChat: A Lightweight Serving Infra

Demo on TinyChatComputer, powered by NVIDIA Jetson Orin Nano

- On a GPU board with just ~7G available memory, TinyChat enables efficient deployment of 7B large language models, thanks to AWQ quantization.

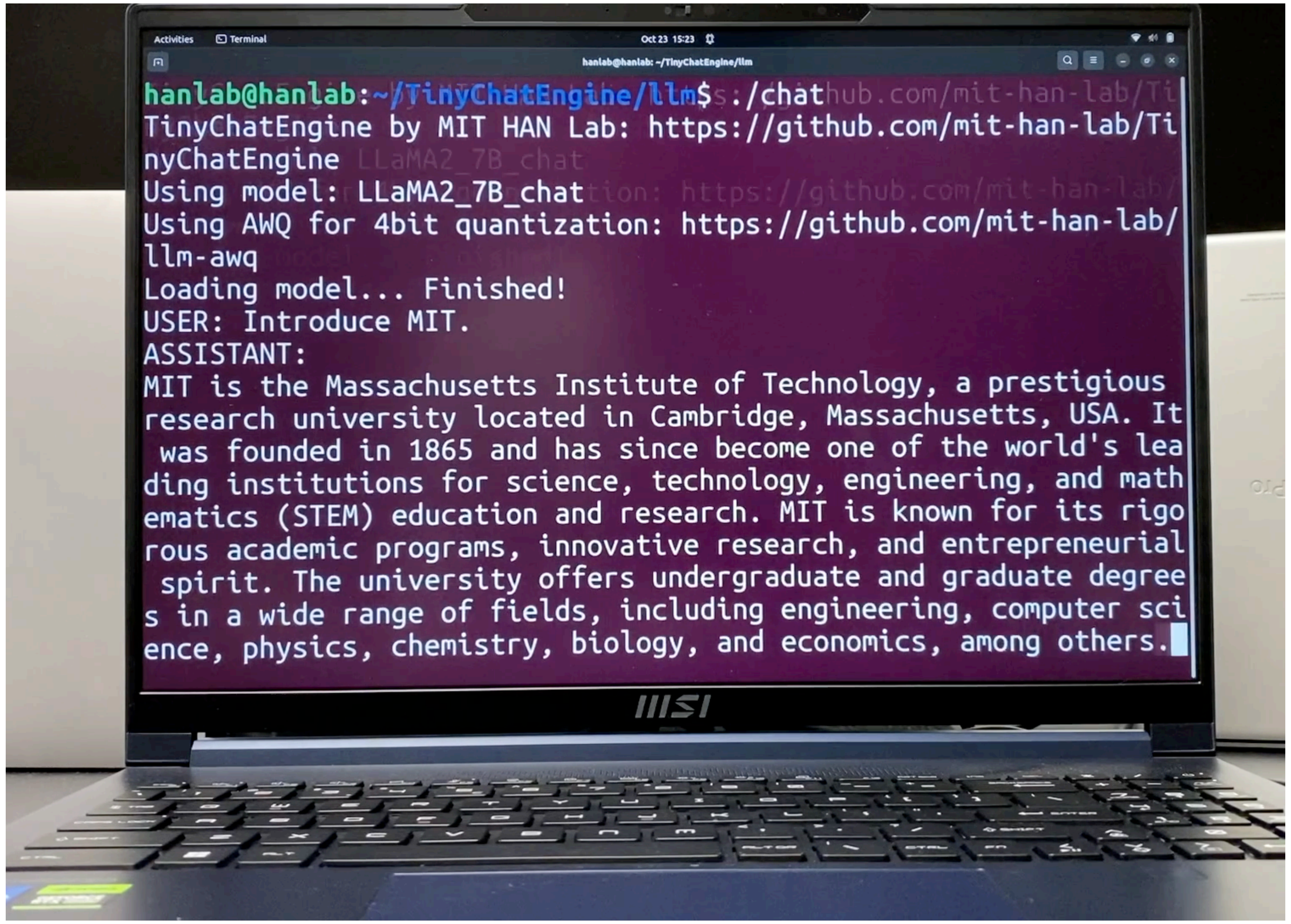
- Worked with students from Harvard Graduate School of Design to manufacture a physical **TinyChatComputer** demo.



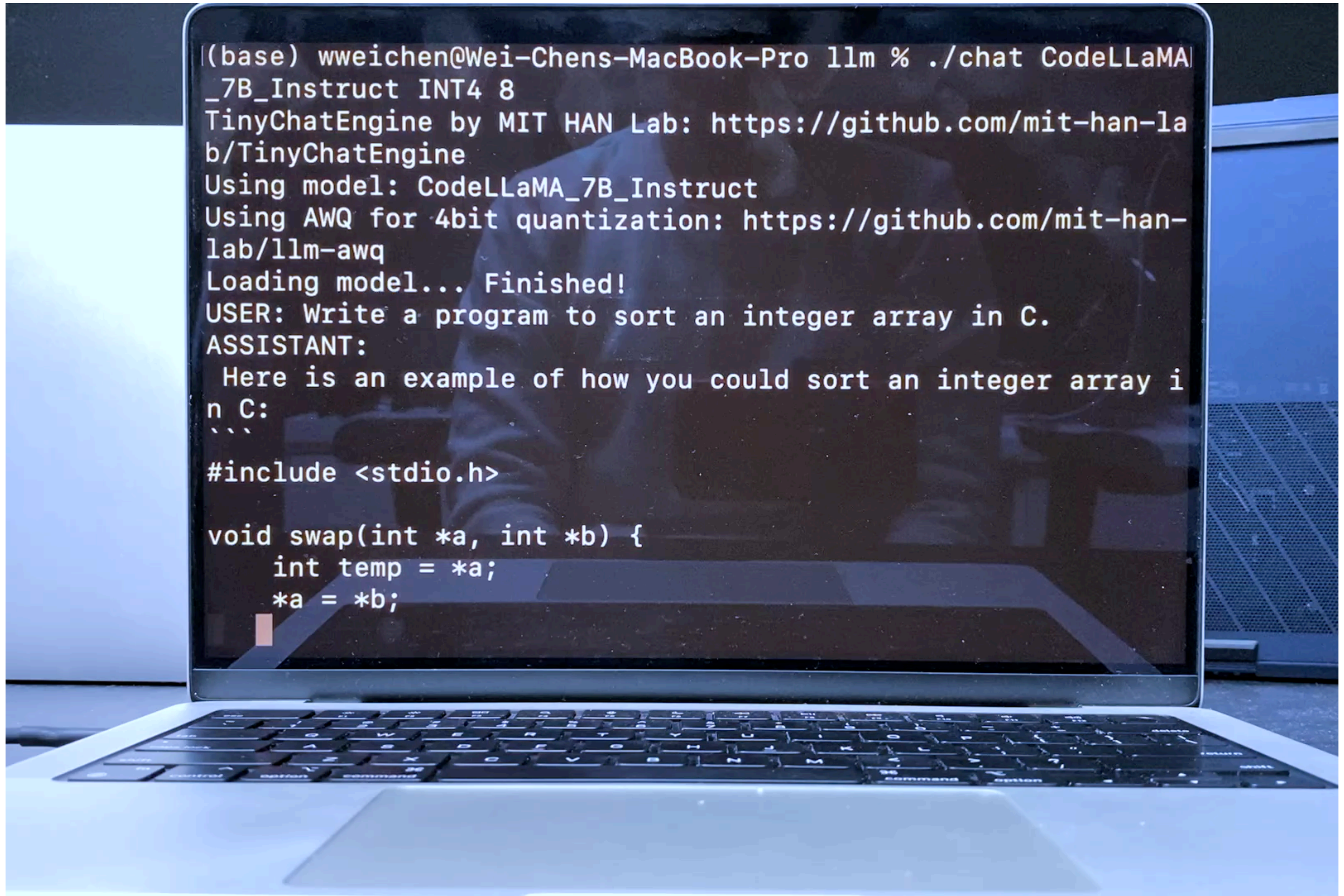
https://github.com/mit-han-lab/llm-awq

TinyChat: A Lightweight Serving Infra

TinyChat seamlessly supports personal laptops with Intel / ARM CPUs



MSI Laptop (RTX 4070 GPU)

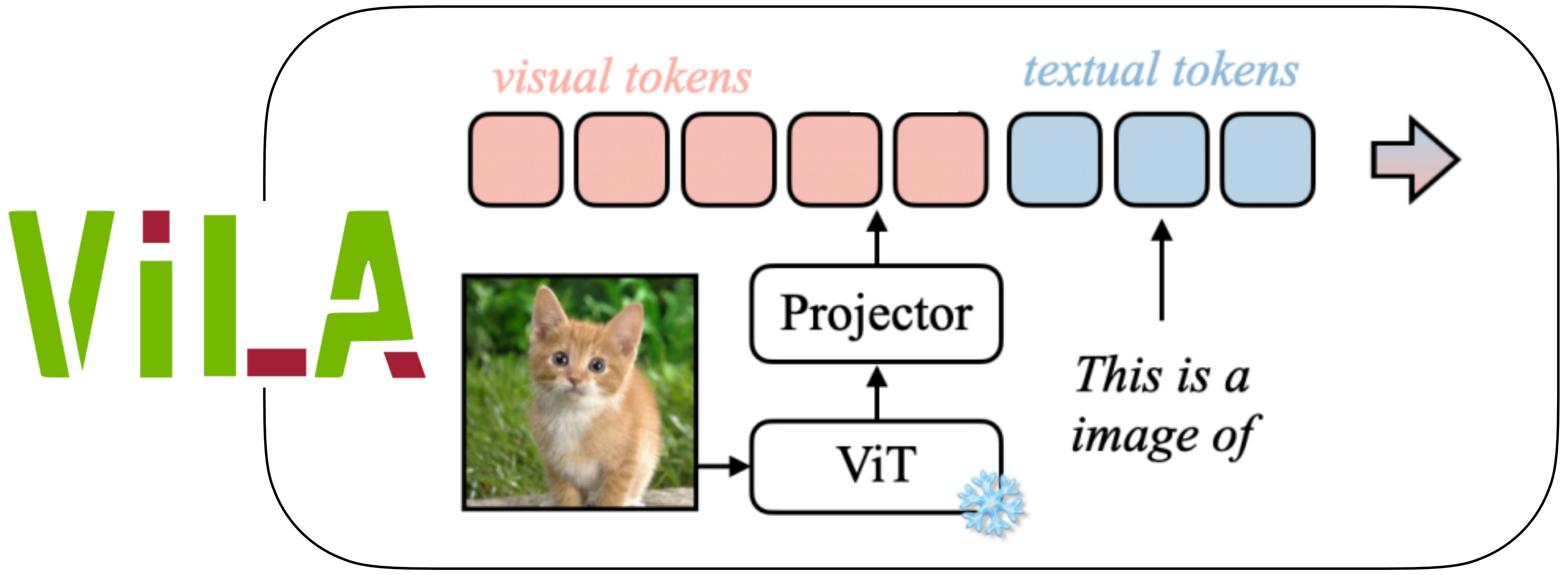


Macbook (ARM CPU)

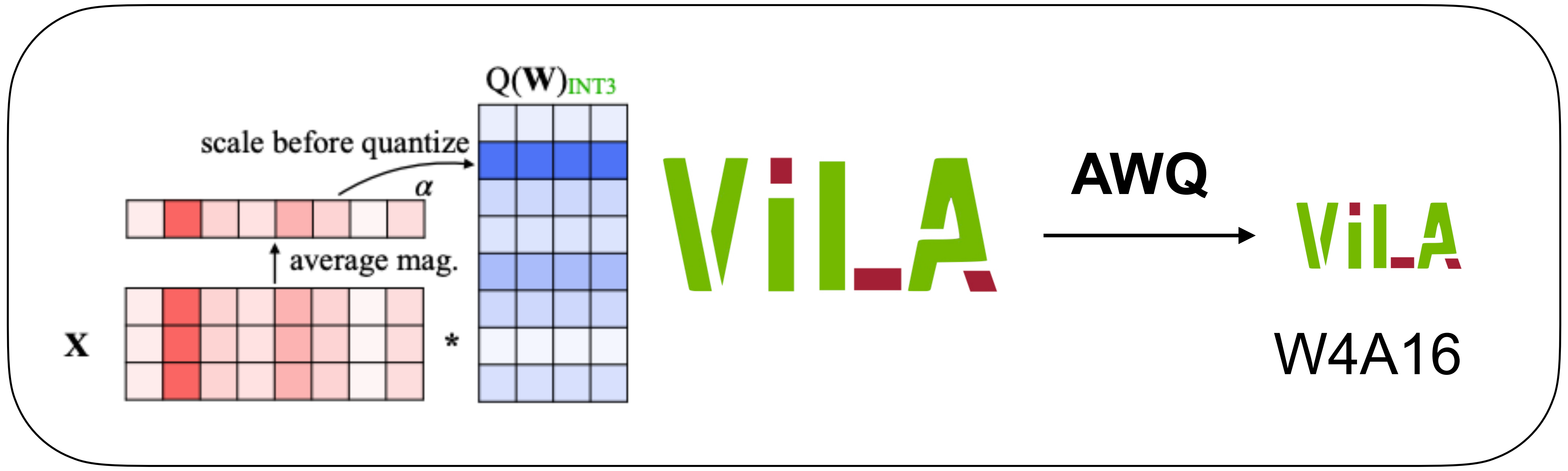
AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Lin et al., MLSys 2024]

TinyChat VILA and Edge AI 2.0

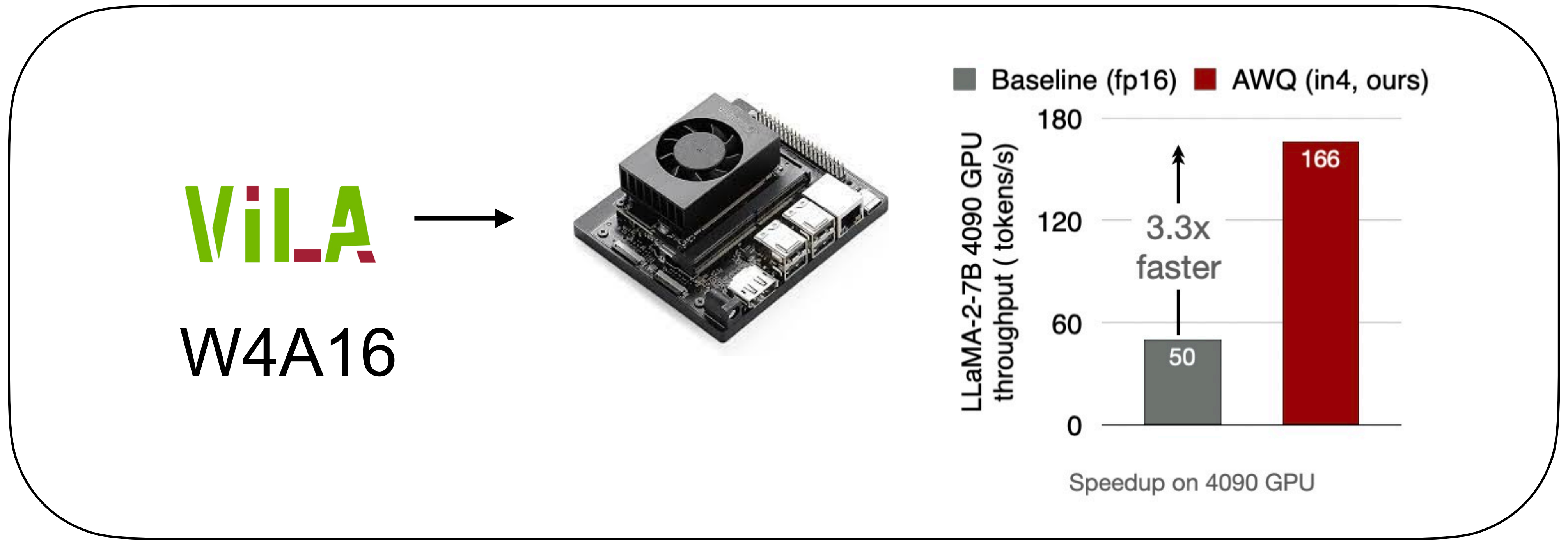
Edge AI 2.0: Foundation models running on the edge efficiently



VILA: Multi-modal capability for LLMs



AWQ: LLM quantization, 4x weight reduction



TinyChat: Efficient framework for LLM deployment

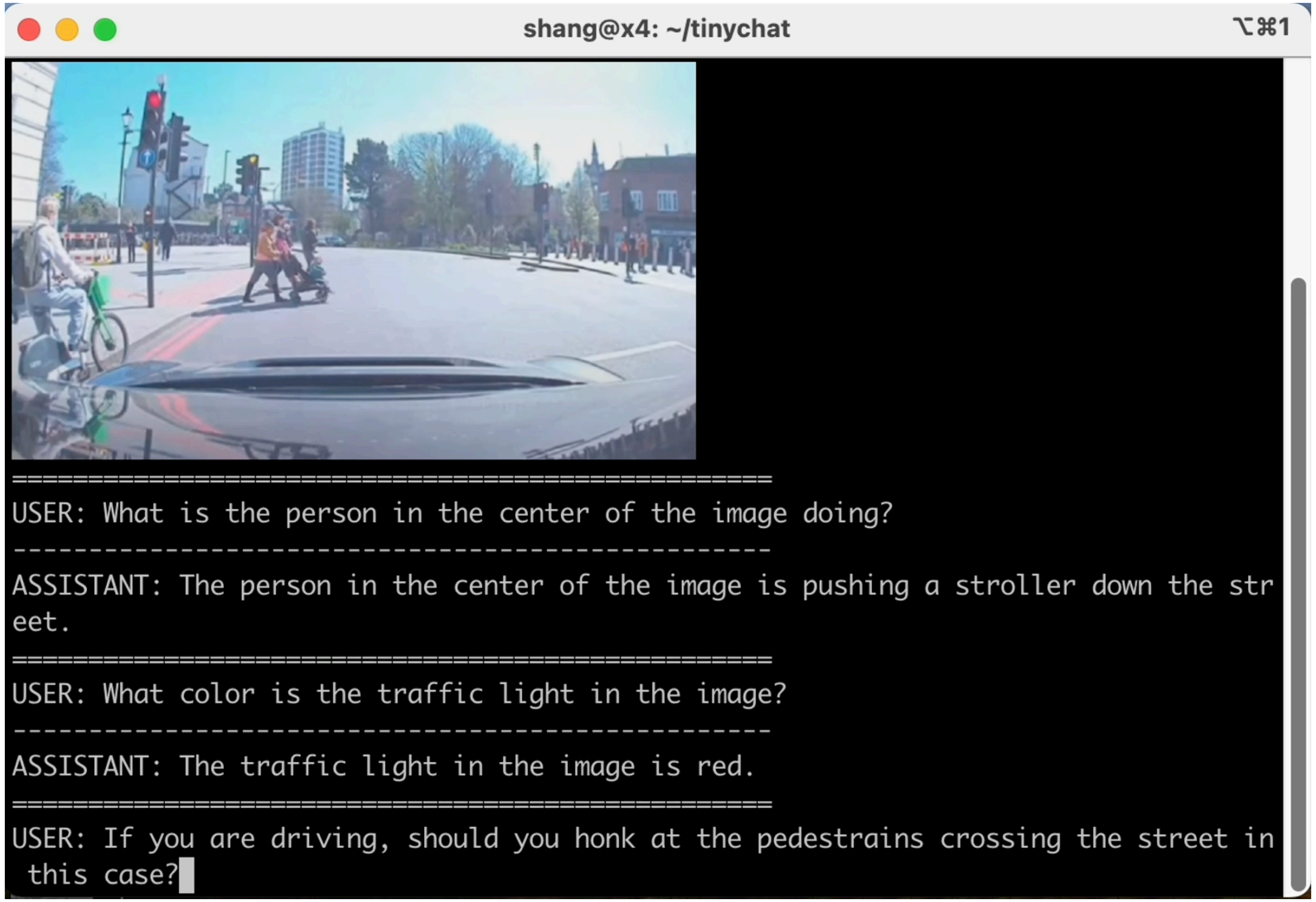
Edge AI 2.0: Multi-model LM on the edge!

TinyChat — Visual language models (VILA)

Efficient image reasoning on Jetson Orin: TinyChat w/ VILA model family

TinyChat for visual language model

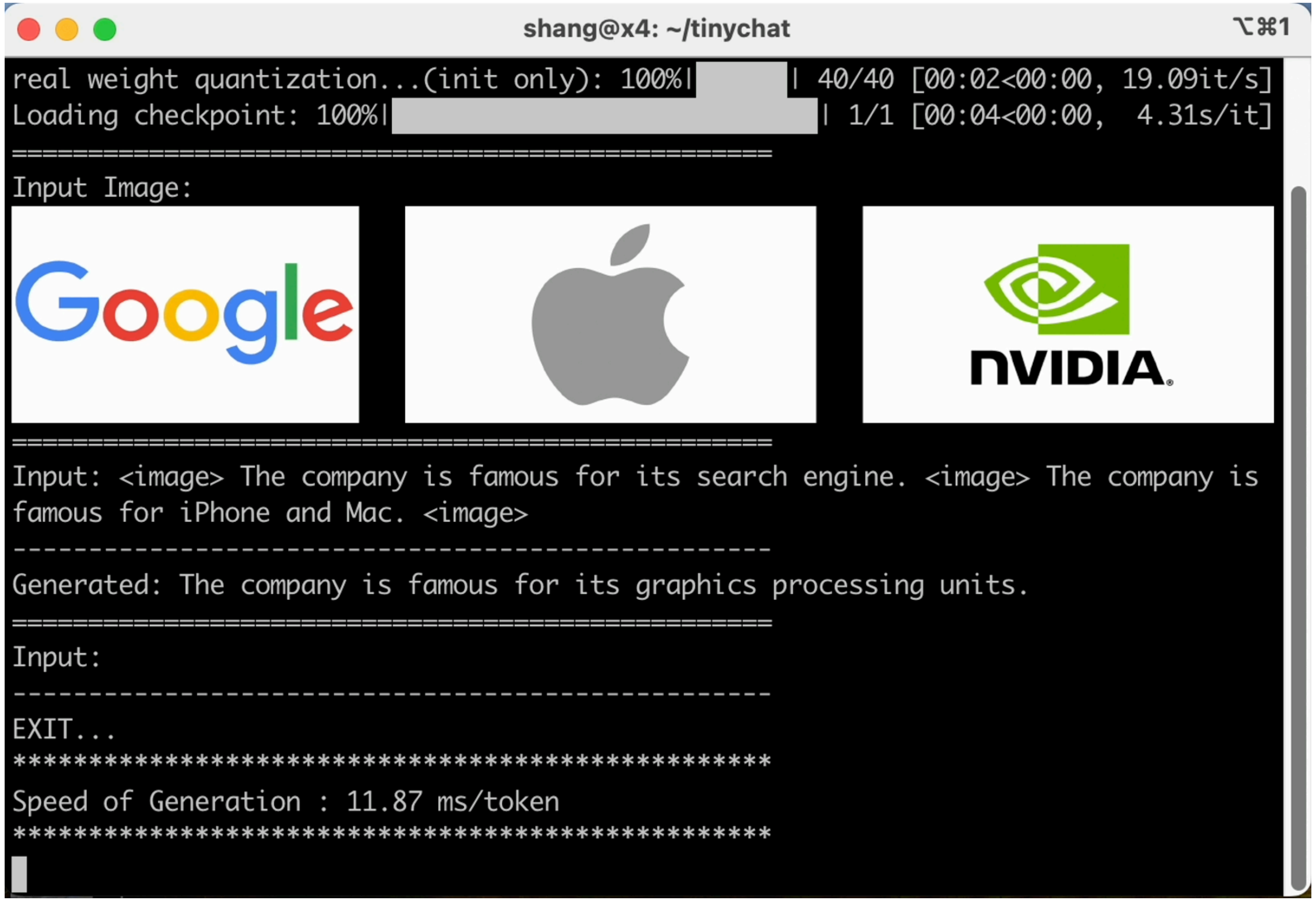
Single image for multi-round Q&A: A driving scenario



VILA-13B + AWQ: 100 tokens/s on 4090

TinyChat for visual language model

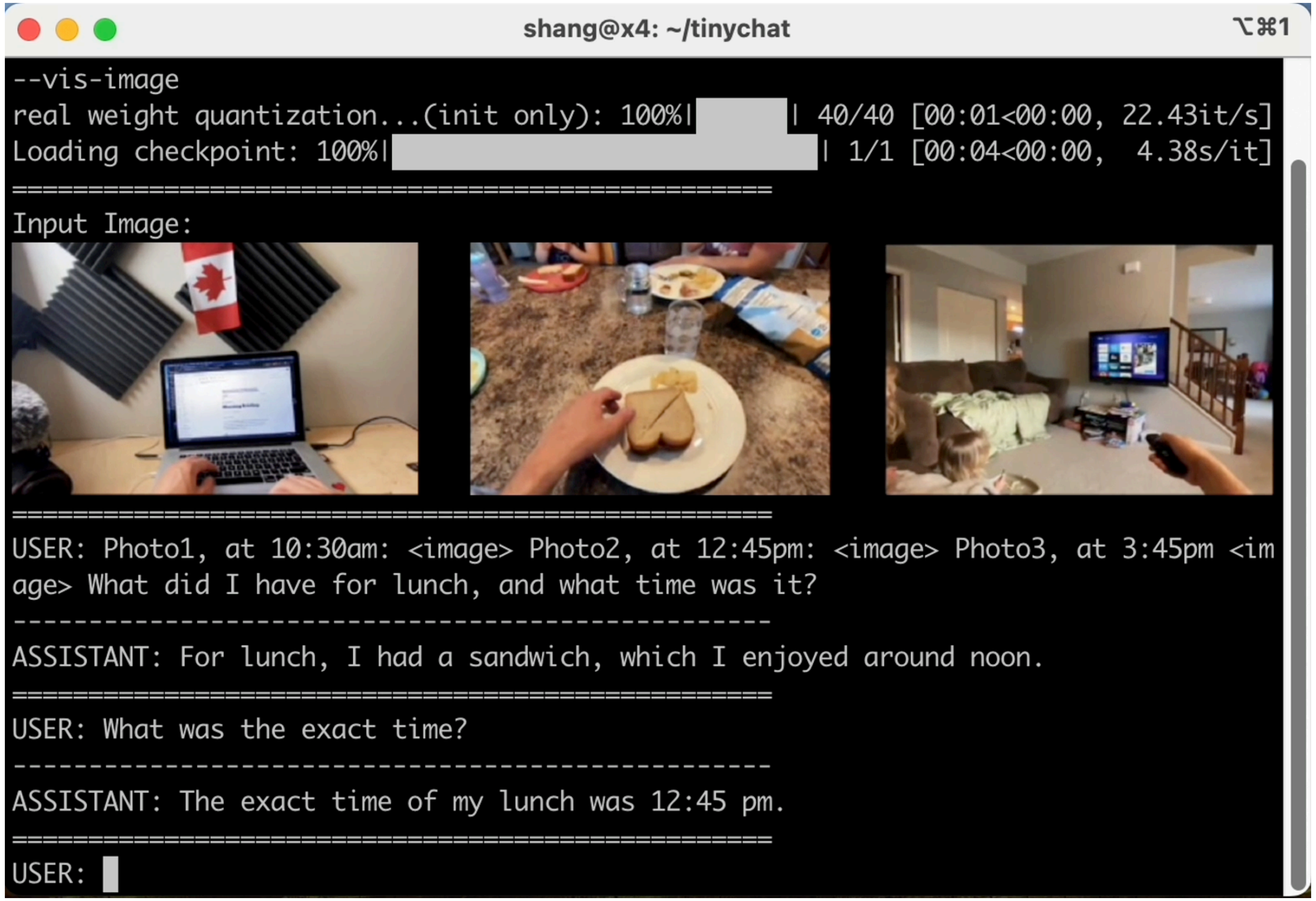
In context learning with multiple image inputs



VILA-13B + AWQ: 84 tokens/s (3 image inputs) on RTX 4090

TinyChat for visual language model

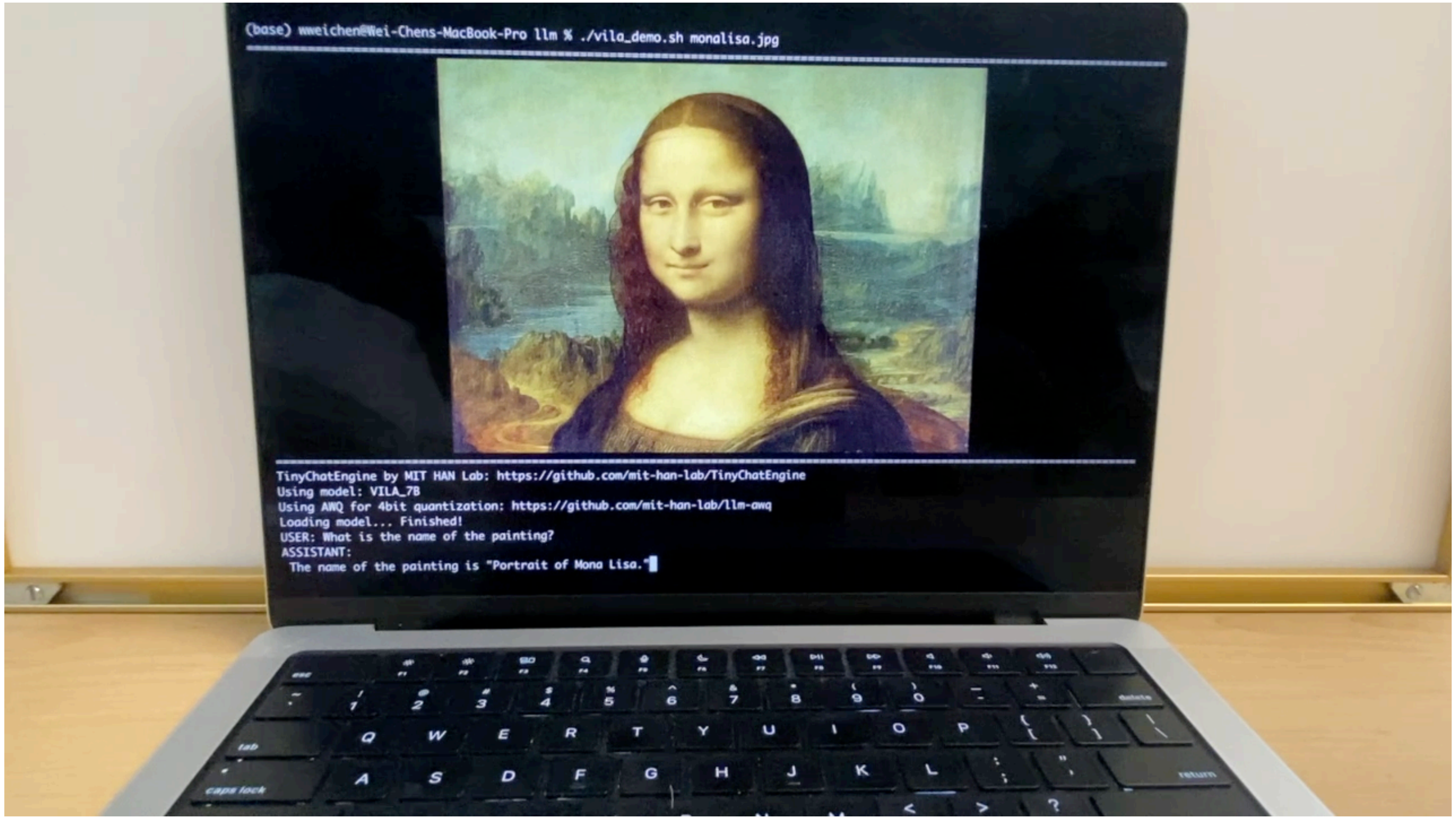
Multi-round Q&A with multi-image inputs



VILA-13B + AWQ: 83 tokens/s (3 image inputs) on RTX 4090

TinyChat for visual language model

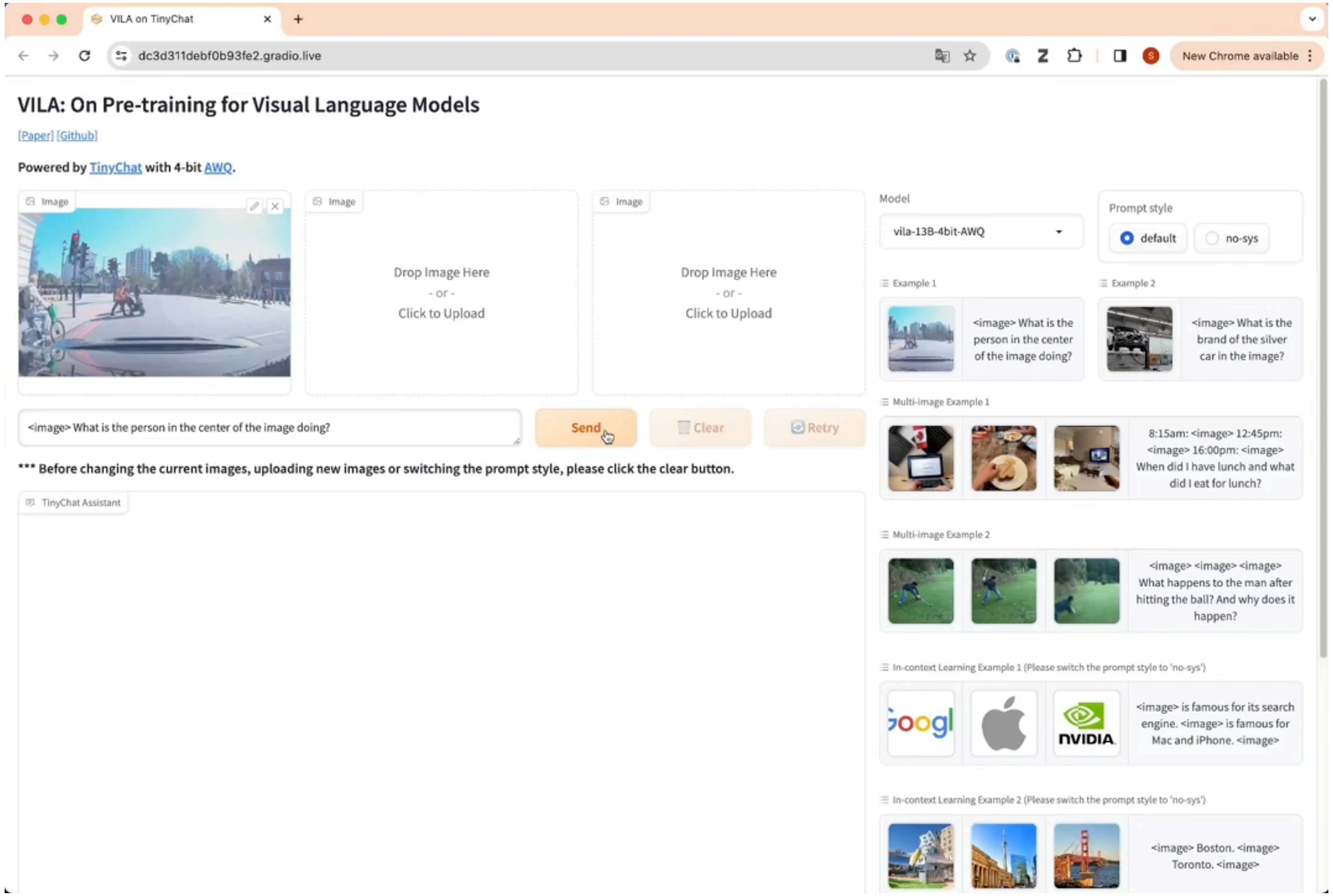
Run visual language models on personal laptops

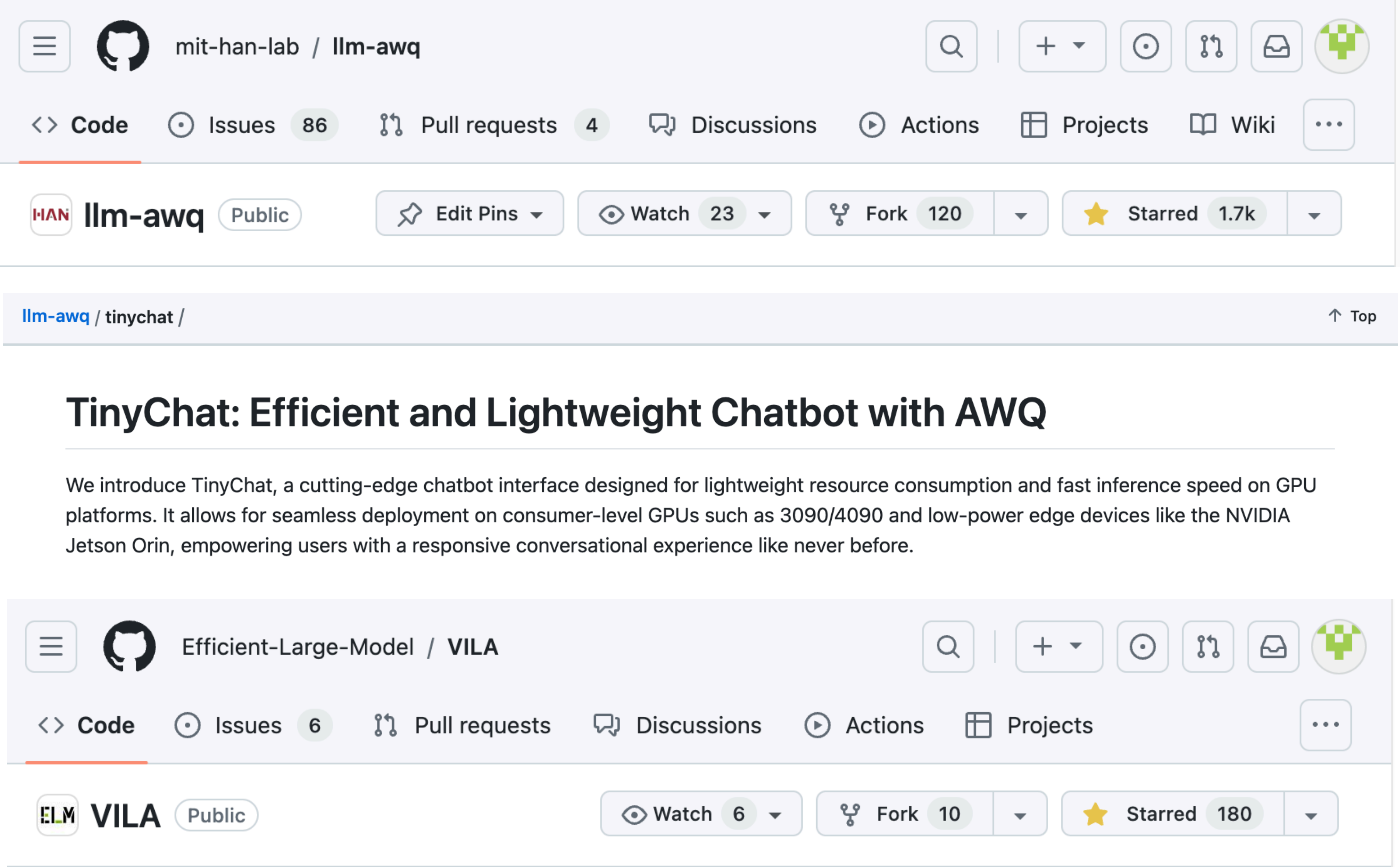


VILA-7B + AWQ: Running on MacBook Arm CPU

TinyChat for visual language model

Try out our online demo for VILA models!





https://vila.hanlab.ai/

Summary

Edge AI 2.0 Requires Full-Stack Optimization

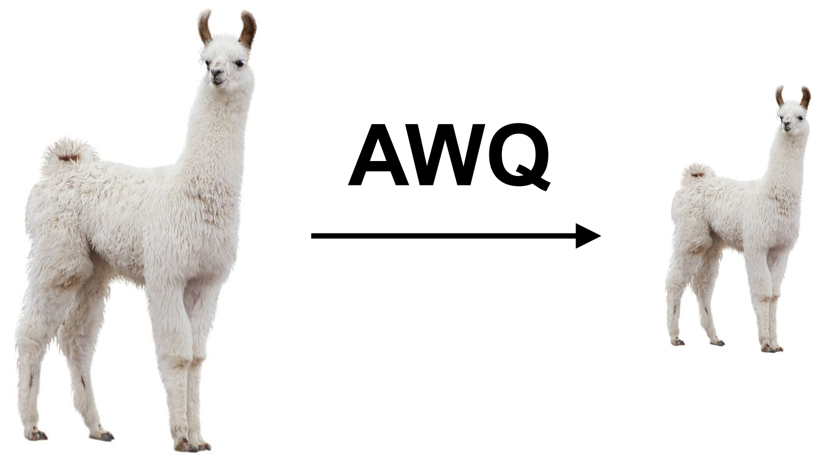
Application

(**Demand** for Computation)

VILA

Model Compression

(Bridging the gap between **demand** and **supply** for computation)



System and Hardware

(**Supply** of Computation)

TinyChat