IN-MEMORY COMPUTING FOR LARGE LANGUAGE MODELS AT THE EDGE

NEUROMORPHIC HARDWARE AND ALGORITHMS WORKSHOP AT THE UNIVERSITY OF SUSSEX

11TH OF JUNE, 2026 – DR.-ING. SEBASTIAN SIEGEL
PETER-GRÜNBERG-INSTITUTE (PGI-14)
FORSCHUNGSZENTRUM JÜLICH GMBH

JÜLICH
Forschungszentrum

Software development

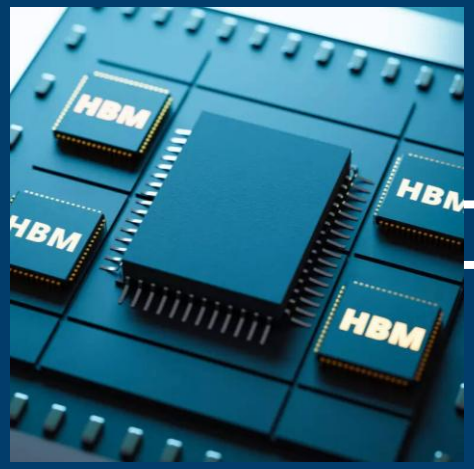
AI

Knowledge retrieval

Artistic creation

NEURAL NETWORK INFERENCE

Recap of memory hierarchy in conventional “von-Neumann” computers



Latency:
- random: 200ns
- pipelined: 30ns

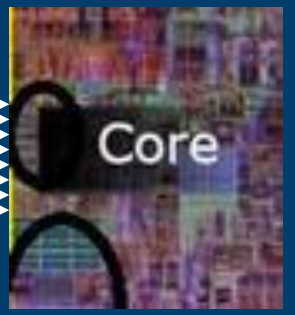
Energy:
- 100 pJ

Latency:
- 3 ns

Energy:
- 5 pJ

Latency:
- <1 ns

Energy:
- 1 pJ



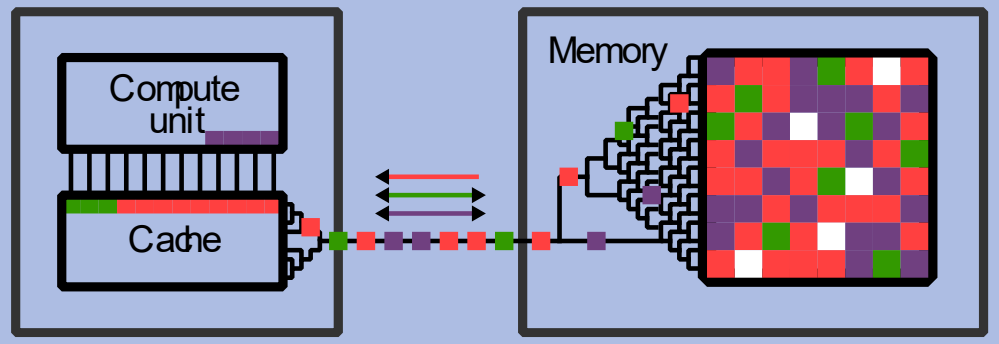
Latency:
- ADD: 1 ns
- MUL: 3 ns

Energy:
- ADD: 0.5 pJ
- MUL: 1 pJ

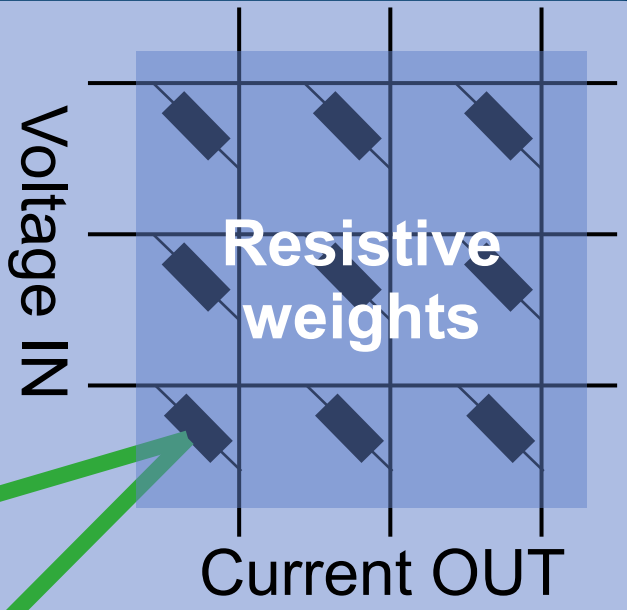
All assumptions for single FP16 value

COMPUTE-IN-MEMORY IS ALL YOU NEED?

Von-Neumann architecture



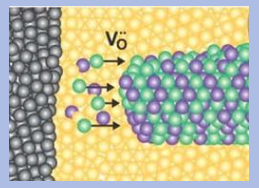
Compute -in- Memory



Capacitive

- 10^{15} writes
- 10^{15} reads
- E_{write} low, E_{read} low
- 1s retention

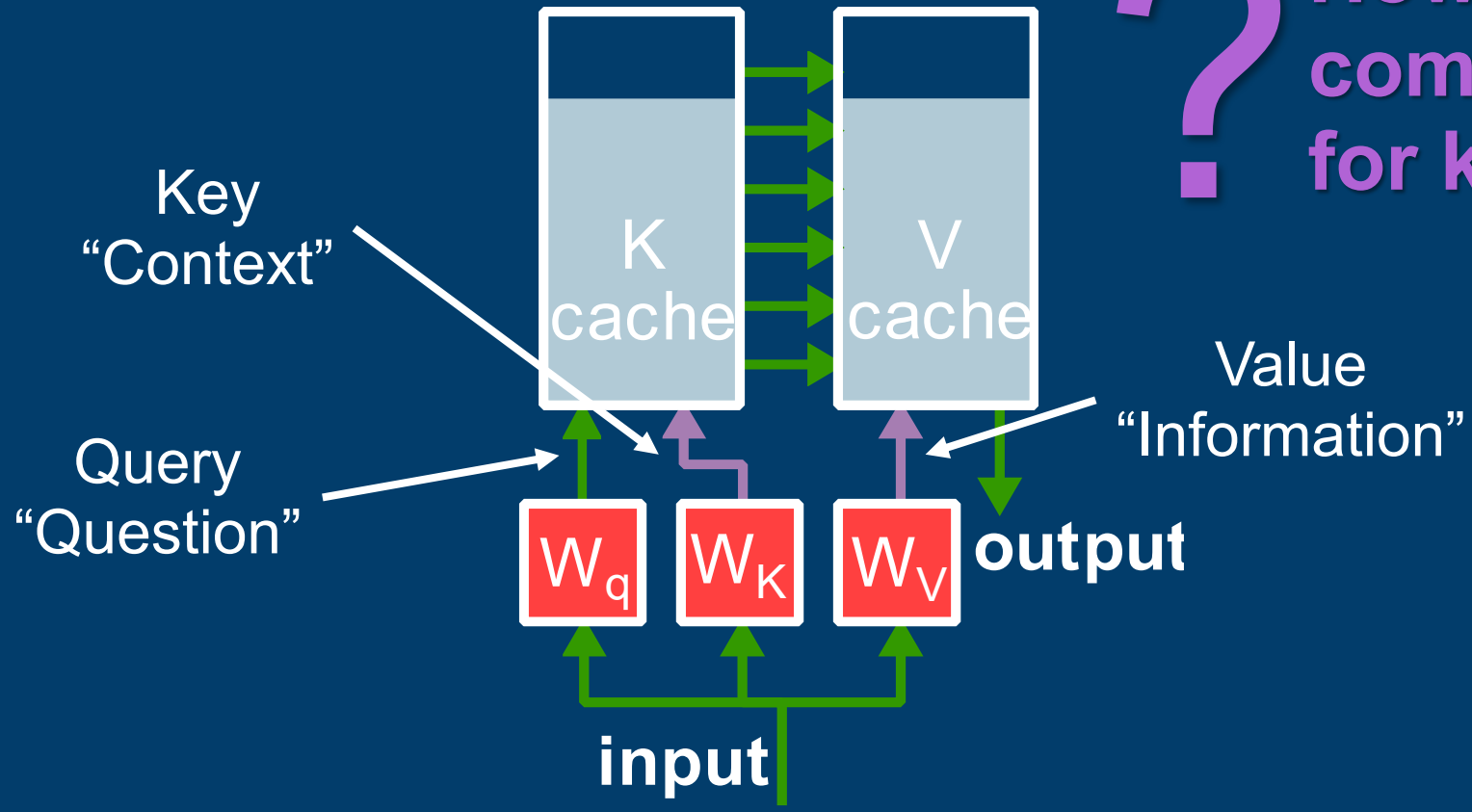
Memristive



- 10^{6} writes
- 10^{15} reads
- E_{write} high, E_{read} low
- 10 years retention

THE TRANSFORMER MODEL

The backbone of modern AI algorithms



? How can we realize compute-in-memory for key / value caches?

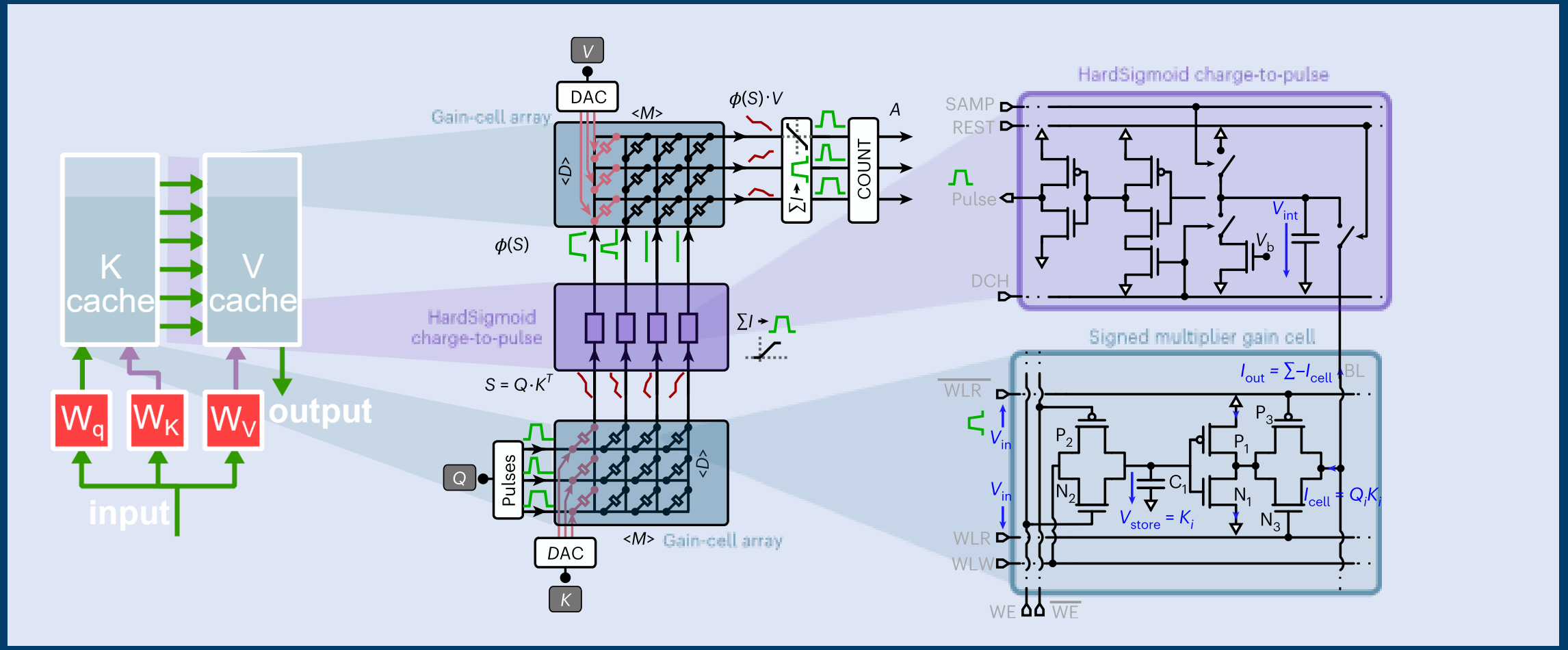
COMPUTE-IN-MEMORY IS ALL YOU NEED?

	MLP / CNN		Self-Attention		RNN / SSM	
Memory write:	Once		Once per sample		Once per token	
	capacitive	**memristive**	**capacitive**	**memristive**	**capacitive**	**memristive**
Write lifetime	190B years	10 years	190B years	1.9 years	19M years	17 hours
Read energy	Low	Low	Low	Low	Low	Low
Write energy	High	Very Low	Low to moderate	Moderate	Moderate	Extremely High
Limiting factor	Retention	None! Match!	None! Match!	Possible, but write limited	Potential match!	Write limited

Use case:
- 1 sample per minute, 1000 token per sample

IN-MEMORY COMPUTING FOR TRANSFORMERS

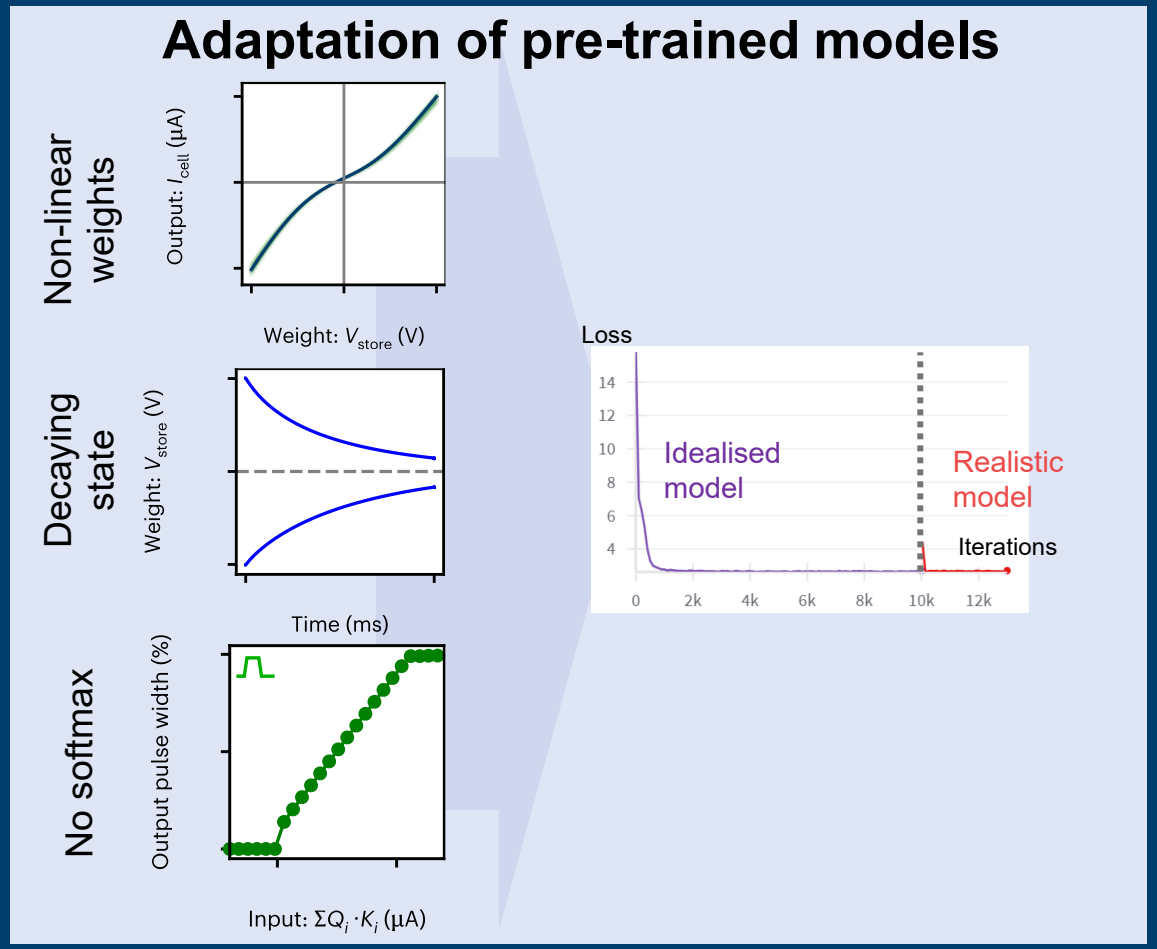
Gain-cell memory for KV-caches

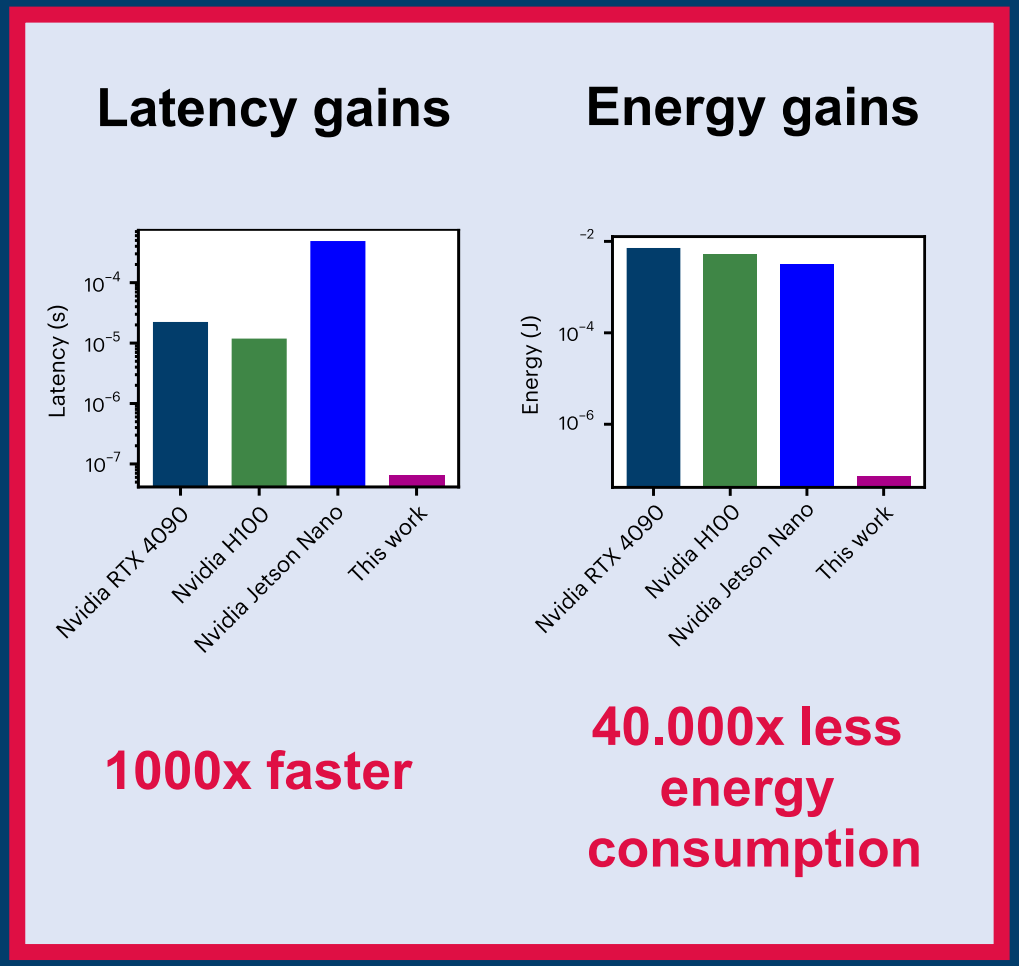


Leroux, N., Manea, P. P., Sudarshan, C., Finkbeiner, J., Siegel, S., Strachan, J. P., & Neftci, E. (2025). Analog in-memory computing attention mechanism for fast and energy-efficient large language models. *Nature Computational Science*, *5*(9), 813-824.

IN-MEMORY COMPUTING FOR TRANSFORMERS

Gain-cell memory for KV-caches



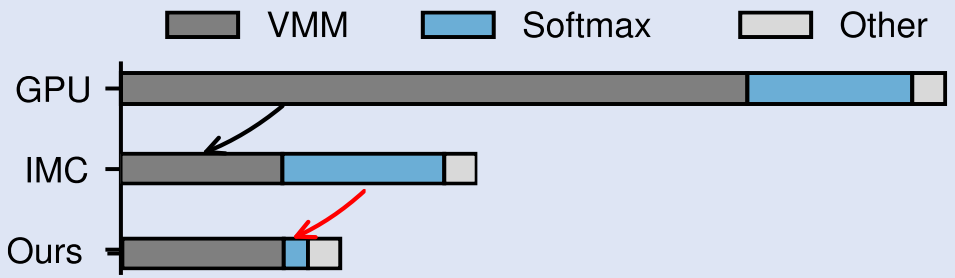


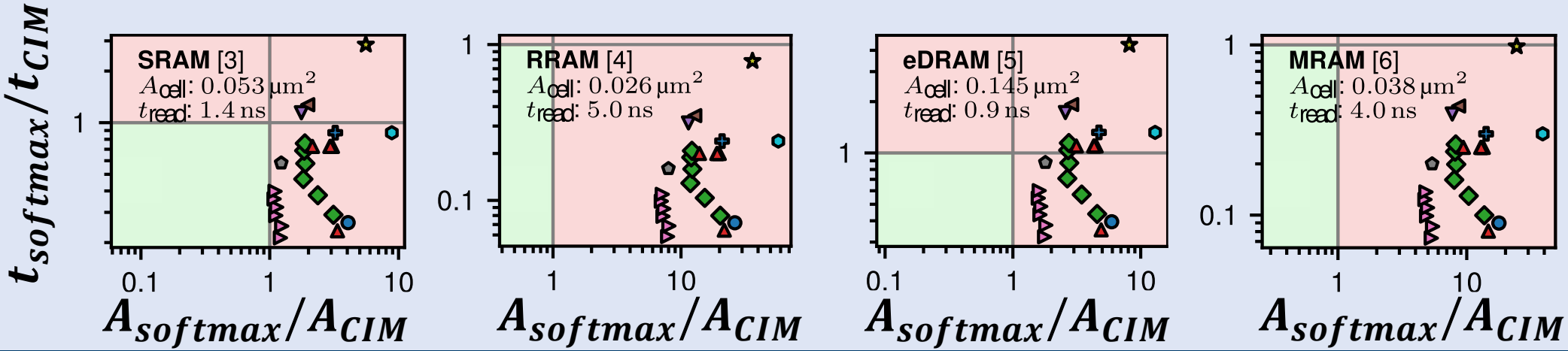
Leroux, N., Manea, P. P., Sudarshan, C., Finkbeiner, J., Siegel, S., Strachan, J. P., & Neftci, E. (2025). Analog in-memory computing attention mechanism for fast and energy-efficient large language models. *Nature Computational Science*, *5*(9), 813-824.

IN-MEMORY COMPUTING FOR TRANSFORMERS

Exact softmax attention for direct Transformer model deployment

$$attention(\vec{q}, \boldsymbol{K}, \boldsymbol{V}) = \frac{\exp(\vec{q}\boldsymbol{K})}{\sum_{m=0}^{N} \exp(\vec{q}\boldsymbol{K})_m} \boldsymbol{V}$$





Finkbeiner, J., Siegel, S., Sudarshan, C., Zhao, Y., Strachan, J. P., & Neftci, E. (2026).
Algorithm-Hardware Implications of Softmax Approximations for In-Memory Computing based LLM Accelerators.
IEEE Journal on Emerging and Selected Topics in Circuits and Systems.

IN-MEMORY COMPUTING FOR TRANSFORMERS

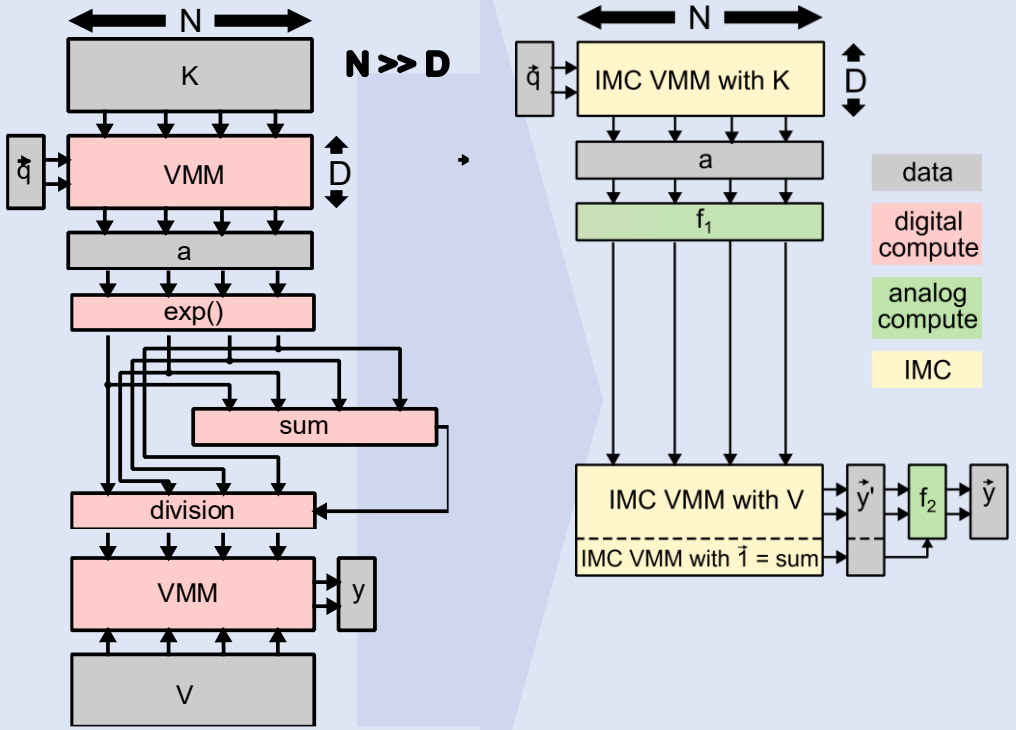
Exact softmax attention for direct Transformer model deployment

$$attention(\vec{q}, \boldsymbol{K}, \boldsymbol{V}) = \frac{\exp(\vec{q}\boldsymbol{K})}{\sum_{m=0}^{N} \exp(\vec{q}\boldsymbol{K})_m} \boldsymbol{V}$$

Non-linearity

Normalization over sequence length

$$attention(\vec{q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{f}_2(\mathrm{f}_1(\vec{q}\boldsymbol{K})\,\boldsymbol{V})$$



Finkbeiner, J., Siegel, S., Sudarshan, C., Zhao, Y., Strachan, J. P., & Neftci, E. (2026).
Algorithm-Hardware Implications of Softmax Approximations for In-Memory Computing based LLM Accelerators.
IEEE Journal on Emerging and Selected Topics in Circuits and Systems.

IN-MEMORY COMPUTING FOR TRANSFORMERS

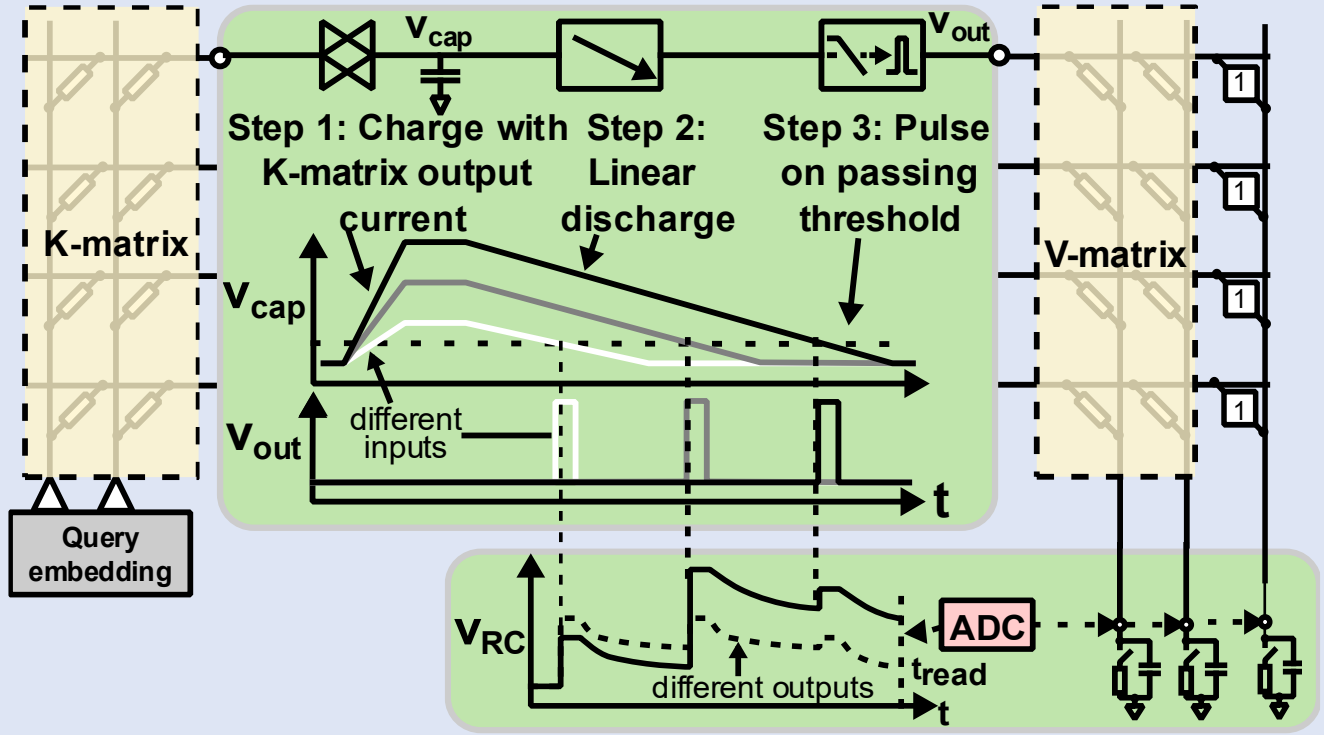
Exact softmax attention for direct Transformer model deployment

$$attention(\vec{q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{f}_2(\mathrm{f}_1(\vec{q}\boldsymbol{K})\,\boldsymbol{V}) = \frac{\exp(\vec{q}\boldsymbol{K})}{\sum_{m=0}^{N}\exp(\vec{q}\boldsymbol{K})_m}\boldsymbol{V}$$

$$t_{pulse} = t_{max} - \mathrm{d}\vec{q}\boldsymbol{K}^T$$

$$\begin{aligned}\vec{v}_{RC} &= \exp(-(t_{read}+t_{spike}))\boldsymbol{V}\\ &= \exp(-t_{read} - t_{max} + \mathrm{d}\vec{q}\boldsymbol{K}^T)\,\boldsymbol{V}\\ &= const.\exp(\mathrm{d}\vec{q}\boldsymbol{K}^T)\,\boldsymbol{V}\\ &= f_1(\vec{q}\boldsymbol{K}^T)\boldsymbol{V}\end{aligned}$$

$$f_2 = \frac{f_1\boldsymbol{V}}{f_1 * \vec{1}} = attention(\vec{q}, \boldsymbol{K}, \boldsymbol{V})$$

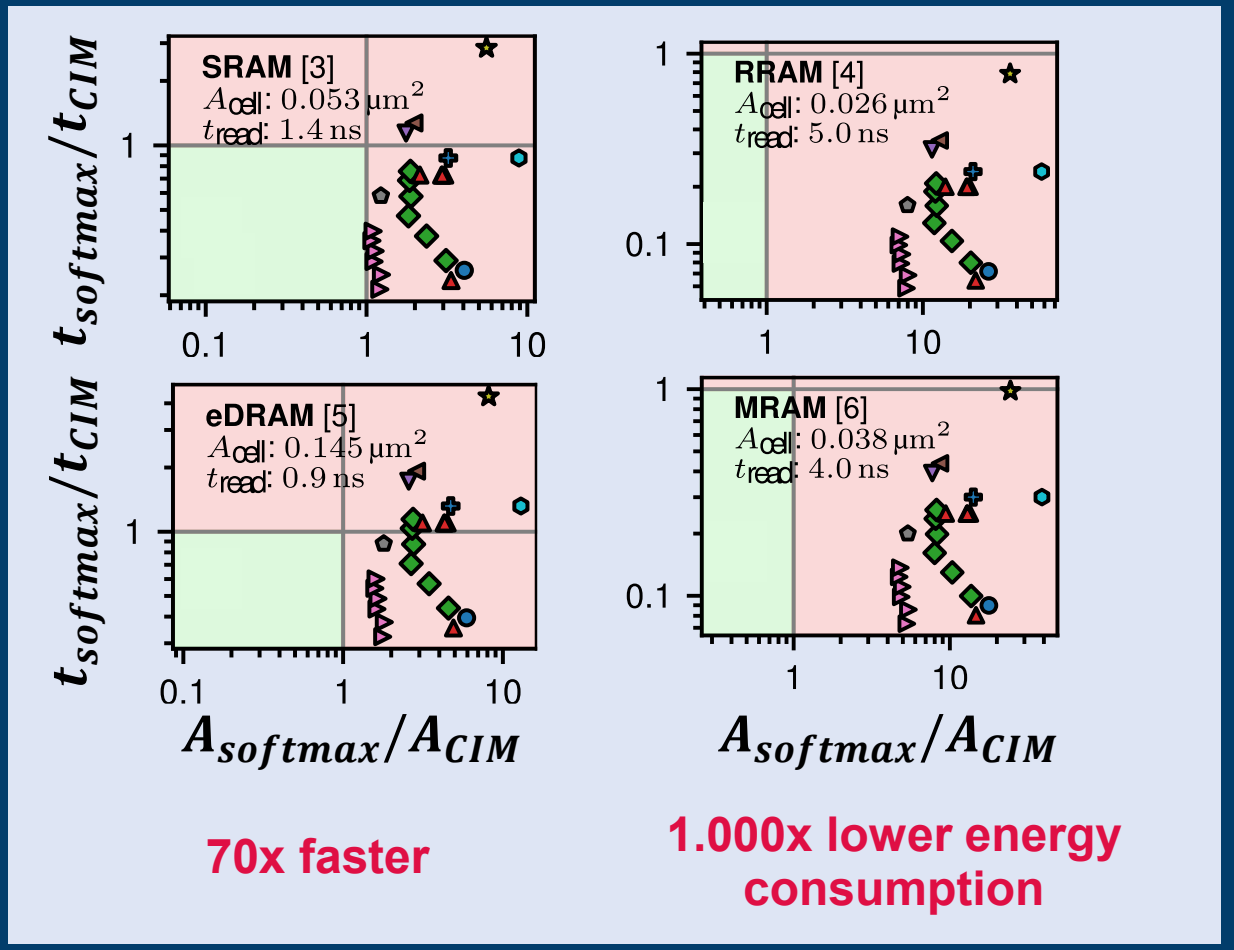


Finkbeiner, J., Siegel, S., Sudarshan, C., Zhao, Y., Strachan, J. P., & Neftci, E. (2026).
Algorithm-Hardware Implications of Softmax Approximations for In-Memory Computing based LLM Accelerators.
IEEE Journal on Emerging and Selected Topics in Circuits and Systems.

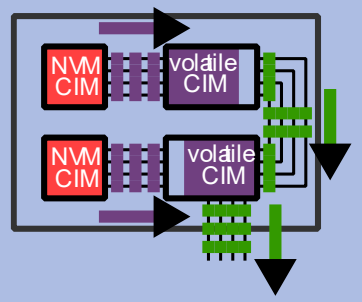
IN-MEMORY COMPUTING FOR TRANSFORMERS

Exact softmax attention for direct Transformer model deployment

Model	Softmax Implementation	WikiText2 ppl ↓	LAMBADA ppl ↓
OPT 125M	Softmax	33.837	27.696
	Smoothquant [38]	48.817	26.679
	Temporal (Analog)	34.354	27.145
	LUT + Shift	33.842	27.588
	1st Order + Shift (Digital)	33.842	27.769
OPT 30B	Softmax	10.757	3.616
	Smoothquant [38]	22.078	3.717
	Temporal (Analog)	10.766	3.640
	LUT + Shift	10.762	3.617
	1st Order + Shift (Digital)	10.758	3.615
Llama3 1B	Softmax	8.630	5.745
	Smoothquant [38]	11.593	5.744
	Temporal (Analog)	8.638	5.888
	LUT + Shift	8.628	5.752
	1st Order + Shift (Digital)	8.636	5.759
Llama3 405B	Softmax	1.257	2.379
	Smoothquant [38]	177.772	2.407
	Temporal (Analog)	1.258	2.376
	LUT + Shift	1.257	2.376
	1st Order + Shift (Digital)	1.256	2.377

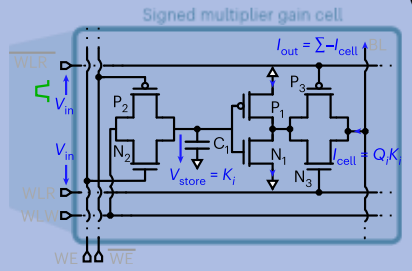


Finkbeiner, J., Siegel, S., Sudarshan, C., Zhao, Y., Strachan, J. P., & Neftci, E. (2026).
Algorithm-Hardware Implications of Softmax Approximations for In-Memory Computing based LLM Accelerators.
IEEE Journal on Emerging and Selected Topics in Circuits and Systems.

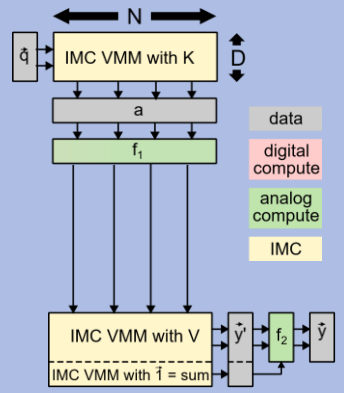


Compute-in-Memory can benefit Large Language Model deployment

Leroux, N., Manea, P. P., Sudarshan, C., Finkbeiner, J., Siegel, S., Strachan, J. P., & Neftci, E. (2025). Analog in-memory computing attention mechanism for fast and energy-efficient large language models. *Nature Computational Science*, *5*(9), 813-824.



Capacitive gain-cell memory device can effectively implement volatile weight



Peripheral functions must be merged with compute-in-memory to gain maximum efficiency

Thank you for your attention!

Finkbeiner, J., Siegel, S., Sudarshan, C., Zhao, Y., Strachan, J. P., & Neftci, E. (2026). Algorithm-Hardware Implications of Softmax Approximations for In-Memory Computing based LLM Accelerators. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*.