Lab Seminar

Conditional Memory via Scalable Lookup: A New Axis of Sparsity for Large Language Model

MOAI Lab Internship
Park Chaewon

What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.

1 **Embeddings**
32000 × 4096 = 131.072.000
embedding size · shared parameters
Token embedding lookup

2 **Attention**
32 × 41.943.040 = 1.342.177.280
repeated decoder blocks · (q, k, v) · shared parameters

3 **Router**
8 × 4096 = 32.768
experts · shared parameters
Scores each token
& selects top-k 'experts'

4 **Experts**
8 × 5.637.144.576 = 45.097.156.608
experts · total parameters
2 × 5.637.144.576 = 11.274.289.152
experts · expert size · active parameters

Non-experts consume zero FLOPs
- total parameters: 45B
- active parameters: 11B

5 **LM Head**
32000 × 4096 = 131.072.000
shared parameters

model size ↑
w/o proportional compute cost

What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

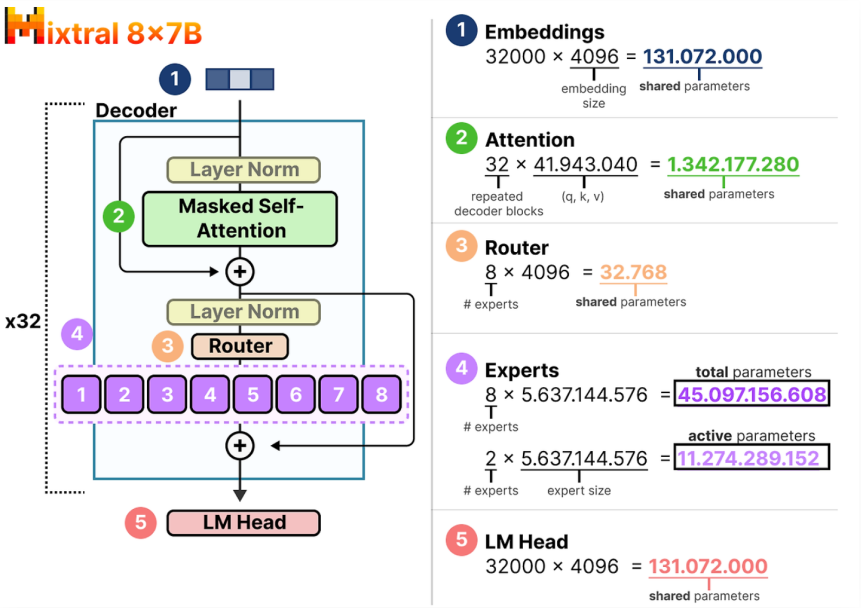
MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.

What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.

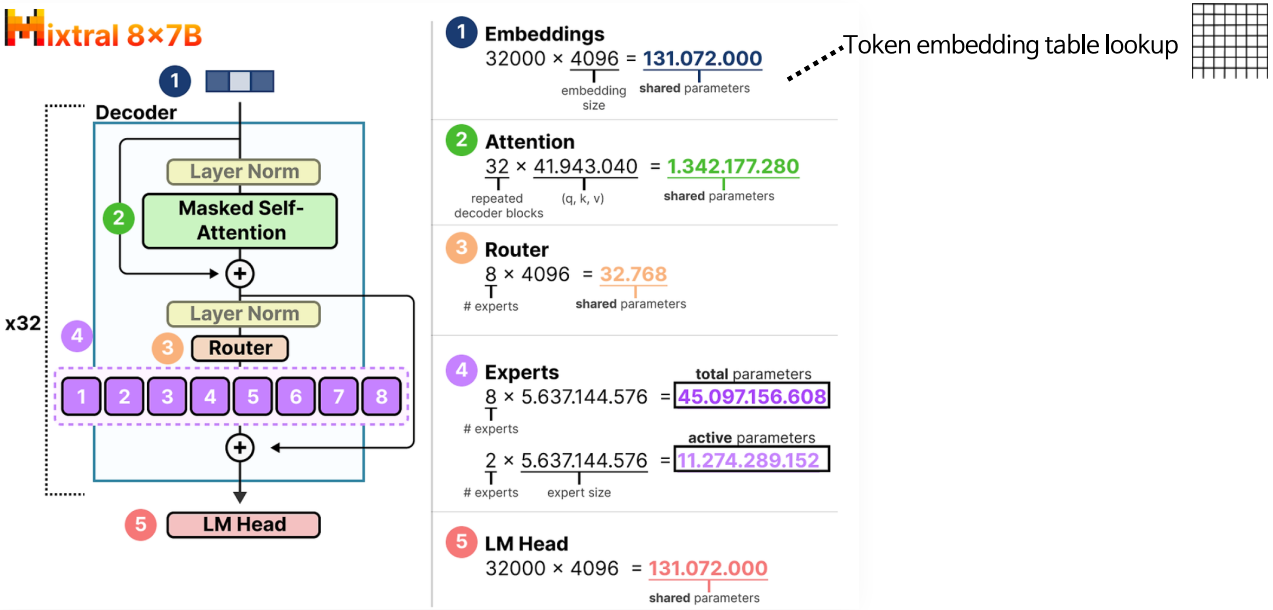


What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.

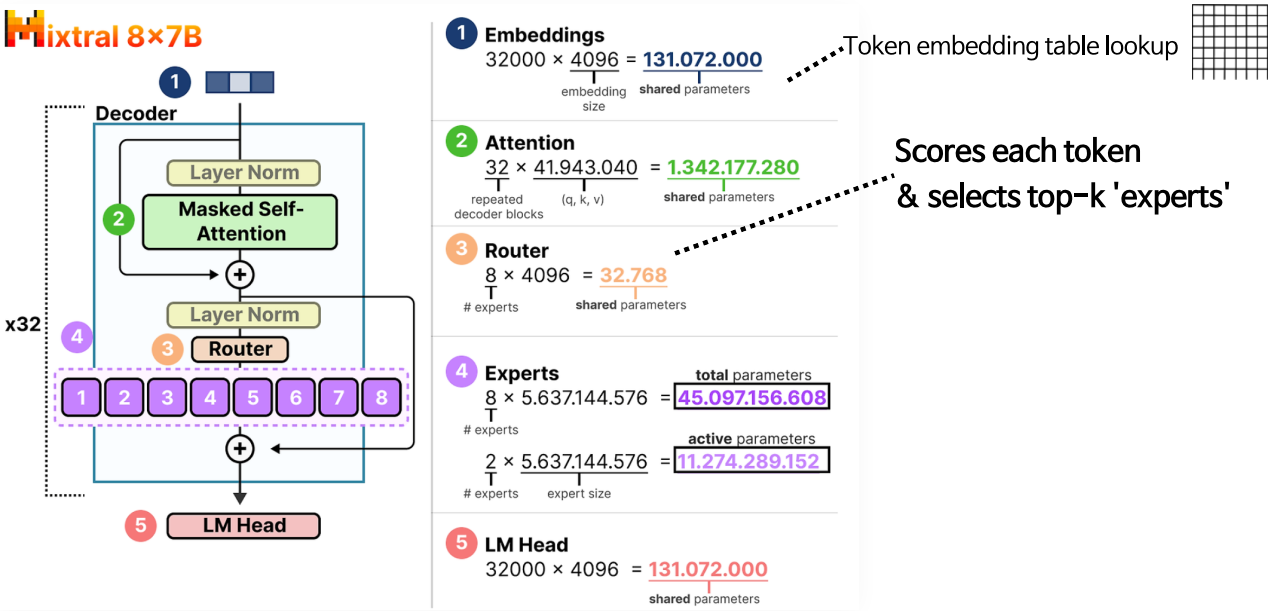


What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.

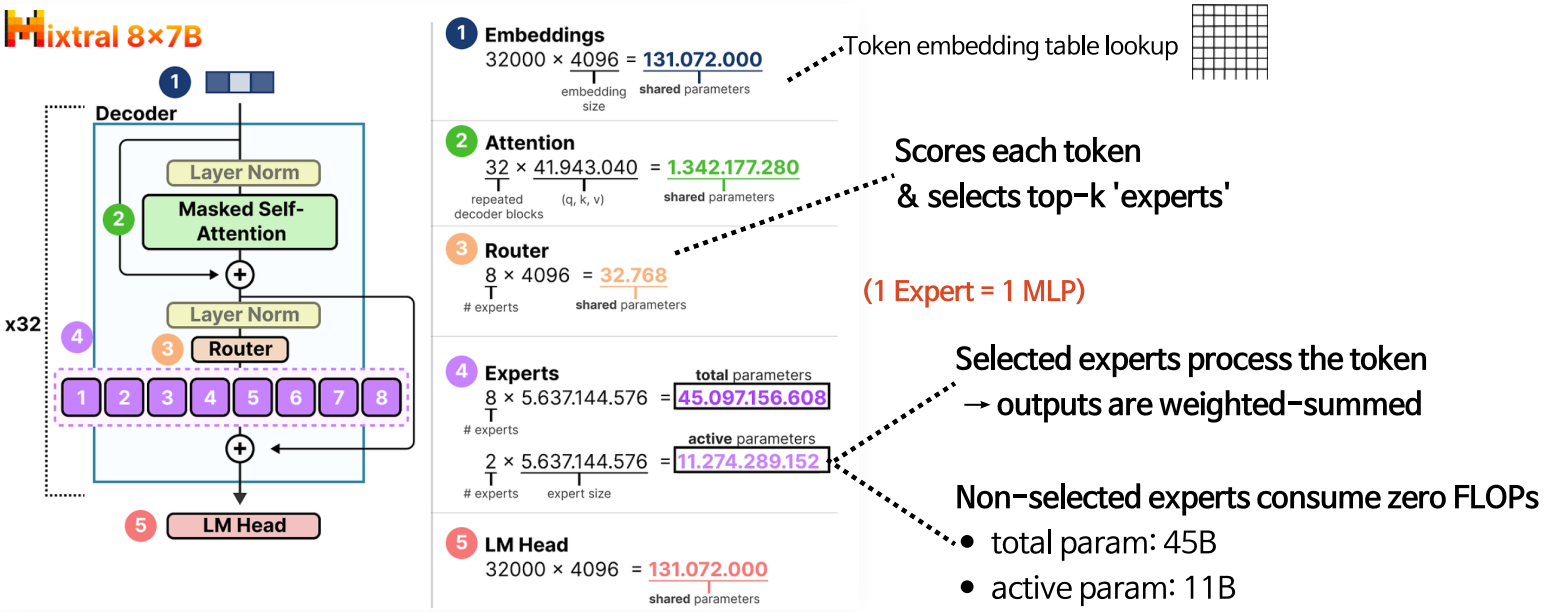


What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.

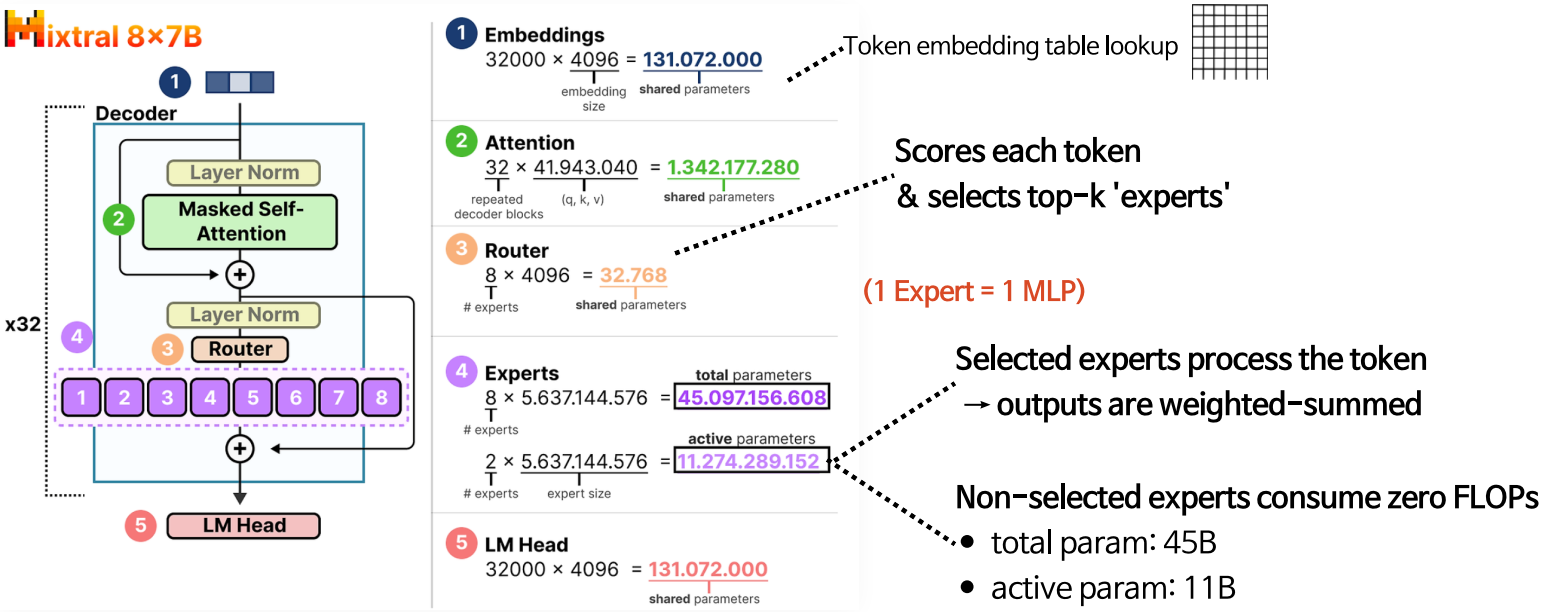


What is Mixture-of-Experts (MoE)?

Standard dense Transformer: every token activates all parameters
→ compute scales linearly with model size.

Mixture-of-Experts (MoE)

MoE's key idea: **conditional computation** — only a subset of parameters ('experts') is activated per token.



- total param: 45B
- active param: 11B

model size ↑
w/o proportional compute cost

In MoE, transformers lack knowledge lookup, resulting in wasted computation.

Language modeling involves two sub-tasks:

1. **Compositional reasoning**
   - Dynamic, context-dependent
   - Ideal Method: **Deep computation (MoE ✓)**

2. **Knowledge retrieval**
   - Static, local, stereotyped
   - Ideal Method: **Direct lookup (N-gram ✓)**

$$P(w_n \mid w_{n-1}, w_{n-2}, \ldots, w_{n-n+1})$$

- Probability of next token given previous N-1 token
- **Pre-computed frequency table → O(1) lookup**

In MoE, transformers lack knowledge lookup, resulting in wasted computation.

Language modeling involves two sub-tasks:

1. **Compositional reasoning**
   - Dynamic, context-dependent
   - Ideal Method: **Deep computation (MoE ✓)**

2. **Knowledge retrieval**
   - Static, local, stereotyped
   - Ideal Method: **Direct lookup (N-gram ✓)**

Transformers have no native way to "look up" static knowledge
→ simulate retrieval through computation.

The inefficiency

"I love Diana, Princess of Wales"
→How 'Wales' encode its surrounding entity
(Tool: PatchScope)

Layer	Latent State Translation	Explanation
1-2	: Country in the United Kingdom	**Wales**
3	: Country in Europe	**Wales**
4	: Title held by female sovereigns in their own right or by queens consort	**Princess of Wales** (unspecific)
5	: Title given to the wife of the Prince of Wales (and later King)	**Princess of Wales** (unspecific)
6	: Diana, Princess of Wales (1961-1997), the first wife of Prince Charles, Prince of Wales, who was famous for her beauty and humanitarian work	**Diana, Princess of Wales**

In MoE, transformers lack knowledge lookup, resulting in wasted computation.

Language modeling involves two sub-tasks:

1. Compositional reasoning
   - Dynamic, context-dependent
   - Ideal Method: **Deep computation (MoE ✓)**
2. Knowledge retrieval
   - Static, local, stereotyped
   - Ideal Method: **Direct lookup (N-gram ✓)**

Transformers have no native way to "look up" static knowledge
→ simulate retrieval through computation.

Token embedding table: one token → one vector.

Why not an N-gram table: token combinations → one vector?

The inefficiency

"I love Diana, Princess of Wales"

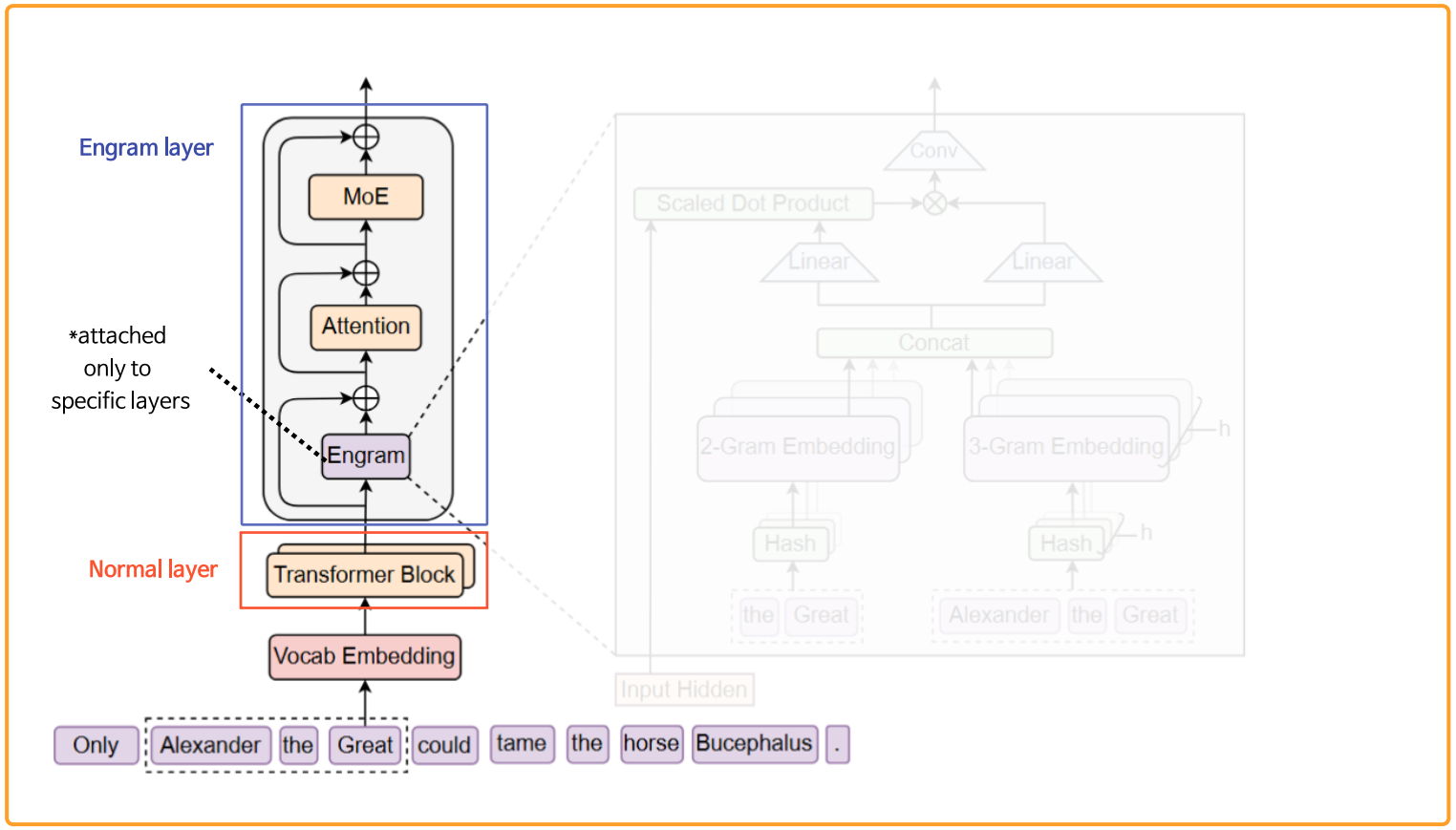
→How 'Wales' encode its surrounding entity

(Tool: PatchScope)

Layer	Latent State Translation	Explanation
3	: Country in Europe	Wales
5	: Title given to the wife of the Prince of Wales (and later King)	Princess of Wales (unspecific)
6	: Diana, Princess of Wales (1961-1997), the first wife of Prince Charles, Prince of Wales, who was famous for her beauty and humanitarian work	Diana, Princess of Wales

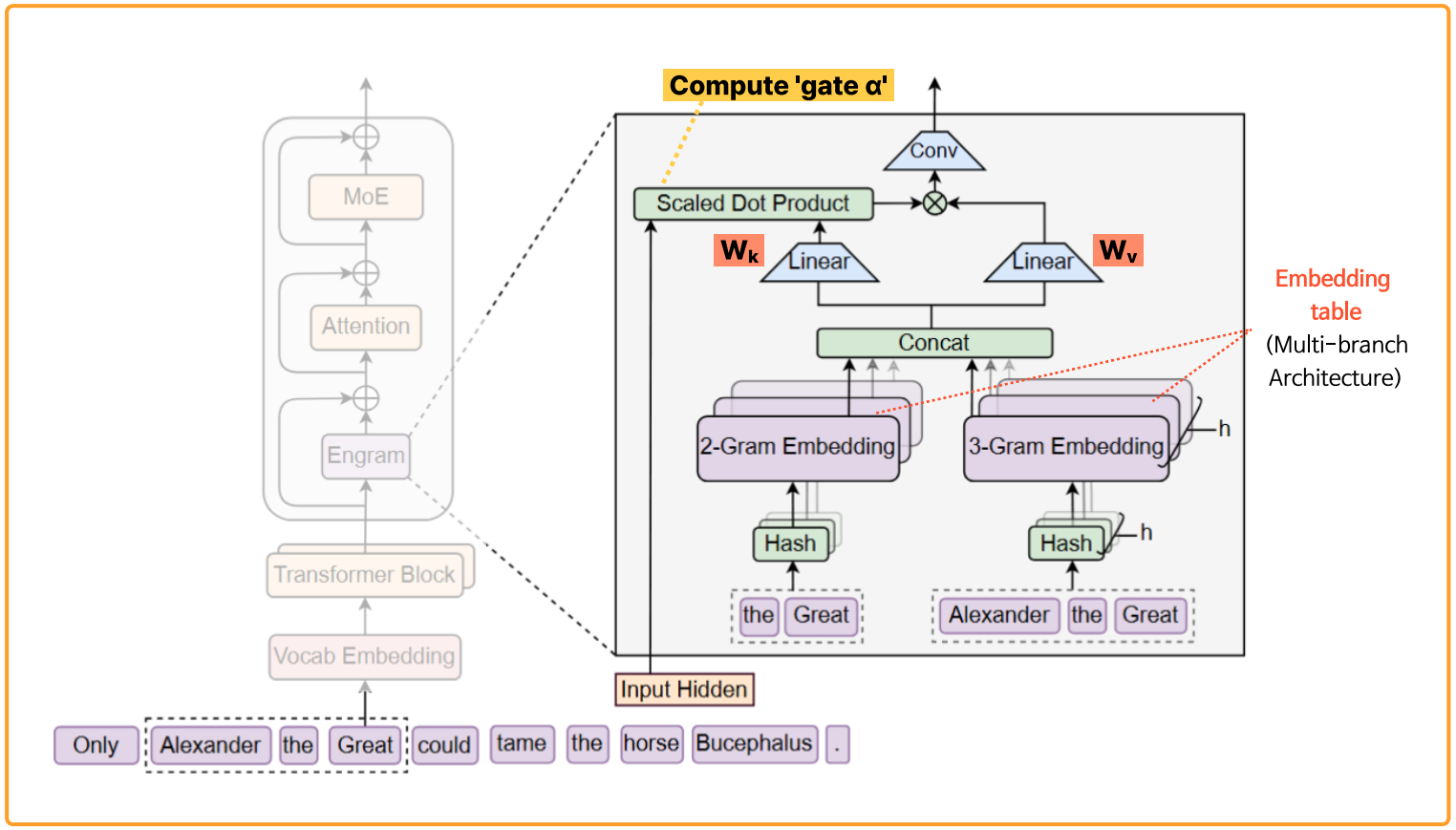
Engram, Engram a conditional memory module

Engram augments the backbone by separating static pattern storage from dynamic computation



Engram, Engram a conditional memory module

Engram augments the backbone by separating static pattern storage from dynamic computation



input seq $X = (x_1, \ldots, x_T)$
& hidden states $H^{(\ell)} \in R^{T \times d}$
at layer ℓ

1. Sparse Retrieval via Hashed -grams

2. Fusion: Context-aware Gating

Architecture – 1. Sparse Retrieval via Hashed -grams

Tokenizer Compression

Rank	Merge Count	Normalized Token	Original Tokens
1	163	’␣’	’\t’, ’\n’, ’\r’, ’␣’, ’␣␣’, ’\n\n’, ’␣␣␣’, ’␣\n’, …
2	54	’a’	’A’, ’a’, ’␣a’, ’␣A’, ’á’, ’ä’, ’ã’, ’ą’, ’␣à’, ’␣å’, ’â’, …
3	40	’o’	’O’, ’o’, ’␣o’, ’␣O’, ’ó’, ’ö’, ’ô’, ’õ’, ’ő’, ’ò’, …
4	35	’e’	’E’, ’e’, ’␣e’, ’␣E’, ’é’, ’è’, ’␣é’, ’ę’, ’ě’, ’ê’, …
5	30	’i’	’I’, ’i’, ’␣I’, ’␣i’, ’í’, ’ì’, ’î’, ’ɪ’, ’ï’, …

- 23% reduction in the effective vocabulary size
- N-gram of position t, N=n → $g_{t,n} = (x'_{t-n+1}, \ldots, x'_t)$

  e.g. $g_{\text{"Great", 3}}$ = *("Alexander", "the", "Great")*

Multi-Head Hashing (XOR hash)

$$z_{t,n,k} \triangleq \varphi_{n,k}(g_{t,n}), \qquad e_{t,n,k} = E_{n,k}[z_{t,n,k}].$$

table index of position t, N=n, kth head

hash function of N=n, kth head

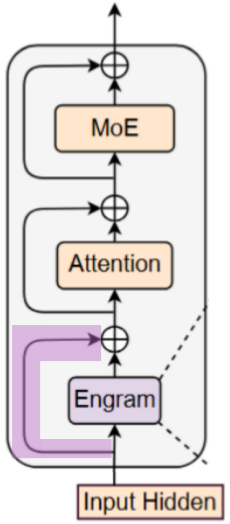
Embedding table of N=n, kth head

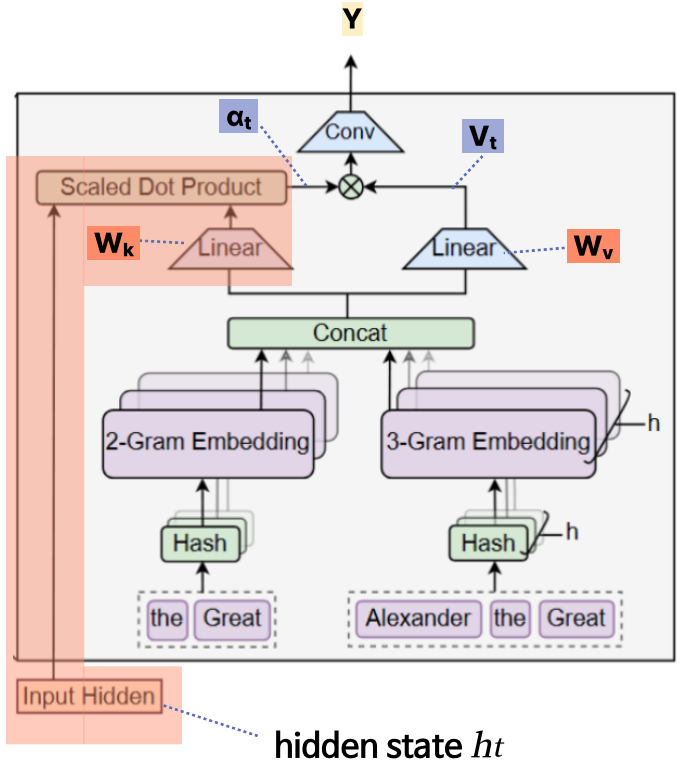
concat

$$e_t \triangleq \Big\Vert_{n=2}^{N} \Big\Vert_{k=1}^{K} e_{t,n,k}.$$

Architecture – 2. Context-aware Gating

Retrieved embeddings (e_t) serve as static, context-independent priors
— but lack adaptability suffering from noise due to polysemy or hash collisions.





hidden state ht
= dynamic query
(containing global context)

scalar gate $\alpha_t \in (0, 1)$

$$\alpha_t = \sigma\left(\frac{\mathrm{RMSNorm}(h_t)^\top \mathrm{RMSNorm}(k_t)}{\sqrt{d}}\right).$$

$$\tilde{\mathbf{v}}_t = \alpha_t \cdot \mathbf{v}_t$$

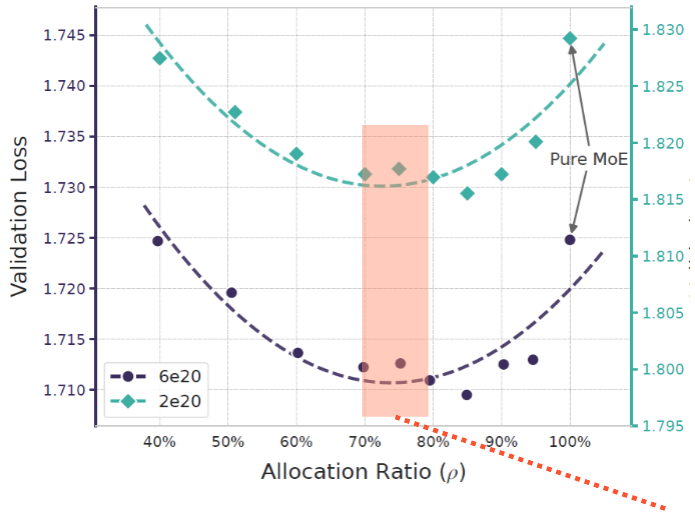
$$Y = \mathrm{SiLU}(\mathrm{Conv1D}(\mathrm{RMSNorm}(\tilde{V}))) + \tilde{V}.$$

Sparsity Allocation: U-shaped Scaling Law

1. Allocation under Finite Constraints (Fixed FLOPs)

- **non-retrieved(Engram) or inactivated(MoE) parameters = 'Sparse budget'**
  - ρ = fraction of sparse budget assigned to MoE experts

    ρ = 1.0 → pure MoE, ρ = 0.0 → pure Engram



Optimal: $\rho \approx$ 75~80% — allocating ~20-25% to Engram consistently improves over pure MoE

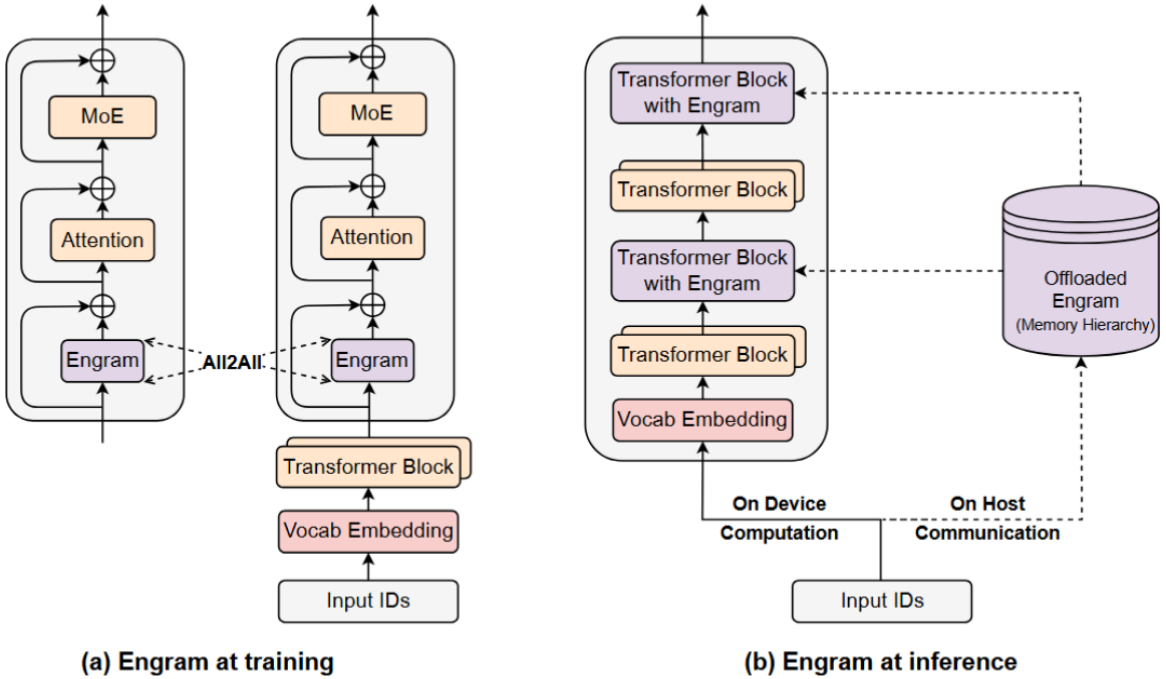
maintain P_tot/P_act≈10

*Engram params increase P_tot without increasing P_act (lookup ≠ computation)

System Efficiency: Decoupling Compute and Memory

Scaling memory-augmented models is often constrained by the limited capacity of GPU high-bandwidth memory (HBM).

MoE routing depends on hidden states → determined during forward pass → prefetch impossible



(a) Engram at training

(b) Engram at inference

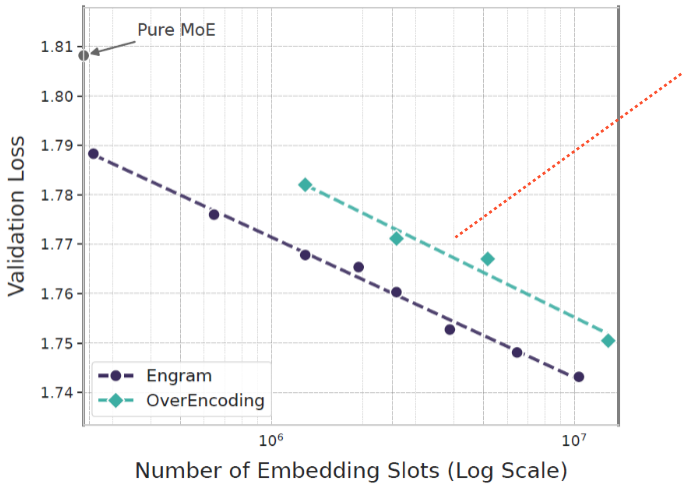
Engram index is determined solely by input token sequence
→ known before forward pass
→ prefetch possible

1. Input IDs arrive → n-gram indices computed immediately
2. Prefetch embedding rows from host DRAM (async)
3. GPU computes preceding layers in parallel
4. Engram layer reached → transfer already done

Engram under Infinite Memory Regime

2. Infinite Memory Regime

- **Fixed 3B MoE backbone → attach an Engram table # of slots: 2.58 × 105 ~ 1.0 × 107 (≈ 13 B param).**
  - validation loss decreases log-linearly (**power law**)
  - A larger memory capacity is always helpful



Baseline – OverEncoding (Huang et al., 2025):
integrates N-gram embeddings by simply averaging them into the vocabulary embedding at the input layer.
(No gating, no context-awareness)

Large Scale Pre-training

	Benchmark (Metric)	# Shots	Dense-4B	MoE-27B	Engram-27B	Engram-40B
	# Total Params		4.1B	26.7B	26.7B	39.5B
	# Activated (w/o token embed)		3.8B	3.8B	3.8B	3.8B
	# Trained Tokens		262B	262B	262B	262B
	# Experts (shared + routed, top-k)		-	2 + 72 (top-6)	2 + 55 (top-6)	2 + 55 (top-6)
	# Engram Params		-	-	5.7B	18.5B
Language Modeling	Pile (loss)	-	2.091	1.960	1.950	1.942
	Validation Set	-	1.768	1.634	1.622	1.610
Knowledge & Reasoning	MMLU (Acc.)	5-shot	48.6	57.4	60.4	60.6
	MMLU-Redux (Acc.)	5-shot	50.7	60.6	64.0	64.5
	MMLU-Pro (Acc.)	5-shot	21.8	28.3	30.1	31.3
	CMMLU (Acc.)	5-shot	47.9	57.9	61.9	63.4
	C-Eval (Acc.)	5-shot	49.3	58.0	62.7	63.3
	AGIEval (Acc.)	0-shot	29.1	38.6	41.8	45.9
	ARC-Easy (Acc.)	25-shot	76.8	85.6	87.1	90.1
	ARC-Challenge (Acc.)	25-shot	59.3	70.1	73.8	76.4
	TriviaQA (EM)	5-shot	33.0	48.1	50.0	51.8
	TriviaQA-ZH (EM)	5-shot	62.8	74.8	76.3	77.9
	PopQA (EM)	15-shot	15.1	19.2	19.4	21.2
	CCPM (Acc.)	0-shot	72.2	83.9	86.9	87.7
	BBH (EM)	3-shot	42.8	50.9	55.9	57.5
	HellaSwag (Acc.)	0-shot	64.3	71.8	72.7	73.1
	PIQA (Acc.)	0-shot	63.8	71.9	73.5	76.5
	WinoGrande (Acc.)	5-shot	64.0	67.6	67.8	68.1
Reading Comprehension	DROP (F1)	1-shot	41.6	55.7	59.0	60.7
	RACE-Middle (Acc.)	5-shot	72.4	80.9	82.8	83.3
	RACE-High (Acc.)	5-shot	66.0	75.4	78.2	79.2
	C3 (Acc.)	0-shot	57.7	60.1	63.6	61.8
Code & Math	HumanEval (Pass@1)	0-shot	26.8	37.8	40.8	38.4
	MBPP (Pass@1)	3-shot	35.4	46.6	48.2	46.2
	CruxEval-i (EM)	0-shot	27.6	30.7	32.2	36.2
	CruxEval-o (EM)	0-shot	28.7	34.1	35.0	35.3
	GSM8K (EM)	8-shot	35.5	58.4	60.6	62.6
	MGSM (EM)	8-shot	27.0	46.8	49.4	52.4
	MATH (EM)	4-shot	15.2	28.3	30.7	30.6

baseline model

replaces the standard dense FFN with a DeepSeekMoE module

reduce routed experts &add 5.7B-param embedding module (ρ = 74.3%)

Large Scale Pre-training

	Benchmark (Metric)	# Shots	Dense-4B	MoE-27B	Engram-27B	Engram-40B
	# Total Params		4.1B	26.7B	26.7B	39.5B
	# Activated (w/o token embed)		3.8B	3.8B	3.8B	3.8B
	# Trained Tokens		262B	262B	262B	262B
	# Experts (shared + routed, top-k)		-	2 + 72 (top-6)	2 + 55 (top-6)	2 + 55 (top-6)
	# Engram Params		-	-	5.7B	18.5B
Language Modeling	Pile (loss)	-	2.091	1.960	1.950	1.942
	Validation Set (loss)	-	1.768	1.634	1.622	1.610
Knowledge & Reasoning	MMLU (Acc.)	5-shot	48.6	57.4	60.4	60.6
	MMLU-Redux (Acc.)	5-shot	50.7	60.6	64.0	64.5
	MMLU-Pro (Acc.)	5-shot	21.1	28.3	30.1	31.3
	CMMLU (Acc.)	5-shot	47.9	57.9	61.9	63.4
	C-Eval (Acc.)	5-shot	46.9	58.0	62.7	63.3
	AGIEval (Acc.)	0-shot	29.1	38.6	41.8	45.9
	ARC-Easy (Acc.)	25-shot	76.8	86.5	89.0	90.1
	ARC-Challenge (Acc.)	25-shot	59.3	70.1	73.8	76.4
	TriviaQA (EM)	5-shot	33.0	48.8	50.7	51.8
	TriviaQA-ZH (EM)	5-shot	62.8	74.8	76.3	77.9
	PopQA (EM)	15-shot	15.1	19.2	19.4	21.2
	CCPM (Acc.)	0-shot	72.2	79.6	87.1	87.7
	BBH (EM)	3-shot	42.8	50.9	55.9	57.5
	HellaSwag (Acc.)	0-shot	64.3	71.8	72.7	73.1
	PIQA (Acc.)	0-shot	63.8	71.9	73.5	76.5
	WinoGrande (Acc.)	5-shot	64.0	67.6	67.8	68.1
Reading Comprehension	DROP (F1)	1-shot	41.6	55.7	59.0	60.7
	RACE-Middle (Acc.)	5-shot	72.4	80.9	82.8	83.3
	RACE-High (Acc.)	5-shot	66.0	75.4	78.2	79.2
	C3 (Acc.)	0-shot	57.7	60.1	63.6	61.8
Code & Math	HumanEval (Pass@1)	0-shot	26.8	37.8	40.8	38.4
	MBPP (Pass@1)	3-shot	35.4	46.6	48.2	46.2
	CruxEval-i (EM)	0-shot	27.6	30.7	32.2	36.2
	CruxEval-o (EM)	0-shot	28.7	34.1	35.0	35.3
	GSM8K (EM)	8-shot	35.5	58.4	60.6	62.6
	MGSM (EM)	8-shot	27.0	46.8	49.4	52.4
	MATH (EM)	4-shot	15.2	28.3	30.7	30.6

knowledge-intensive task

general-reasoning domain

Code & Math

Large Scale Pre-training

	Benchmark (Metric)	# Shots	Dense-4B	MoE-27B	Engram-27B	Engram-40B
	# Total Params		4.1B	26.7B	26.7B	39.5B
	# Activated (w/o token embed)		3.8B	3.8B	3.8B	3.8B
	# Trained Tokens		262B	262B	262B	262B
	# Experts (shared + routed, top-k)		-	2 + 72 (top-6)	2 + 55 (top-6)	2 + 55 (top-6)
	# Engram Params		-	-	5.7B	18.5B
Language Modeling	Pile (loss)	-	2.091	1.960	1.950	1.942
	Validation Set (loss)	-	1.768	1.634	1.622	1.610
Knowledge & Reasoning	MMLU (Acc.)	5-shot	48.6	57.4	60.4	60.6
	MMLU-Redux (Acc.)	5-shot	50.7	60.6	64.0	64.5
	MMLU-Pro (Acc.)	5-shot	21.1	28.3	30.1	31.3
	CMMLU (Acc.)	5-shot	47.9	57.9	61.9	63.4
	C-Eval (Acc.)	5-shot	46.9	58.0	62.7	63.3
	AGIEval (Acc.)	0-shot	29.1	38.6	41.8	45.9
	ARC-Easy (Acc.)	25-shot	76.8	86.5	89.0	90.1
	ARC-Challenge (Acc.)	25-shot	59.3	70.1	73.8	76.4
	TriviaQA (EM)	5-shot	33.0	48.8	50.7	51.8
	TriviaQA-ZH (EM)	5-shot	62.8	74.8	76.3	77.9
	PopQA (EM)	15-shot	15.1	19.2	19.4	21.2
	CCPM (Acc.)	0-shot	72.2	79.6	87.1	87.7
	BBH (EM)	3-shot	42.8	50.9	55.9	57.5
	HellaSwag (Acc.)	0-shot	64.3	71.8	72.7	73.1
	PIQA (Acc.)	0-shot	63.8	71.9	73.5	76.5
	WinoGrande (Acc.)	5-shot	64.0	67.6	67.8	68.1
Reading Comprehension	DROP (F1)	1-shot	41.6	55.7	59.0	60.7
	RACE-Middle (Acc.)	5-shot	72.4	80.9	82.8	83.3
	RACE-High (Acc.)	5-shot	66.0	75.4	78.2	79.2
	C3 (Acc.)	0-shot	57.7	60.1	63.6	61.8
Code & Math	HumanEval (Pass@1)	0-shot	26.8	37.8	40.8	38.4
	MBPP (Pass@1)	3-shot	35.4	46.6	48.2	46.2
	CruxEval-i (EM)	0-shot	27.6	30.7	32.2	36.2
	CruxEval-o (EM)	0-shot	28.7	34.1	35.0	35.3
	GSM8K (EM)	8-shot	35.5	58.4	60.6	62.6
	MGSM (EM)	8-shot	27.0	46.8	49.4	52.4
	MATH (EM)	4-shot	15.2	28.3	30.7	30.6

Artifact of under-training.

Engram40B − the baselines training loss gap continues to widen toward the end.

→ the expanded memory cap has not yet fully saturated within the current token budget

Engram frees attention capacity for global context — Long-context experiment

Key Idea: By offloading local dependency modeling to static lookups,
the Engram architecture preserves valuable attention capacity for managing global context

- **Iso-Loss setting:** compare when the base model performance is the same (same training loss)
- **Iso-FLOPs Setting:** compare under the same total training compute (same steps)
- **Extreme Setting:** compare Engram trained with only 82% of MoE's compute budget

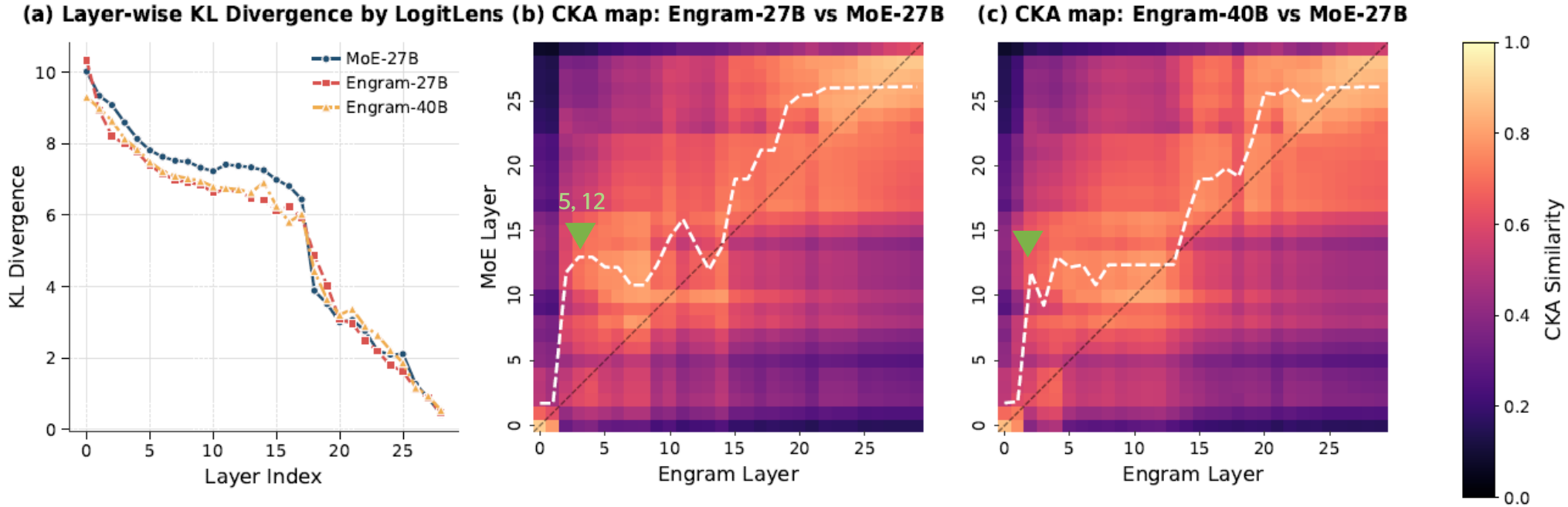
Model	LongPPL (32k)				RULER (32k)							
	Perplexity (↓)				NIAH Accuracy (↑)				Other Tasks (↑)			
	Book	Paper	Code	L-CoT	S	MK	MV	MQ	VT	CWE	FWE	QA
MoE-27B *(50k, 1.63)*	4.38	2.91	2.49	14.16	**100.0**	88.0	92.7	84.2	77.0	4.5	73.0	34.5
Engram-27B *(41k, 1.66)*	4.37	2.92	2.50	14.26	99.6	88.3	93.0	89.5	83.2	3.8	99.6	**44.0**
Engram-27B *(46k, 1.63)*	4.19	2.84	2.45	13.59	97.6	89.0	95.5	**97.0**	87.2	4.3	98.6	37.5
Engram-27B *(50k, 1.62)*	**4.14**	**2.82**	**2.44**	**13.41**	99.3	**89.3**	**96.5**	**97.0**	**89.0**	**5.9**	**99.3**	40.5

information search

» Distinct Architectural advantage

» **Engram frees attention capacity for global context, substantially outperforming baselines in long-context scenarios.**

Engram frees attention capacity for global context — Analysis



Engram finishes feature composition earlier, freeing remaining layers for reasoning.

effective depth increased!

Brighter = more similar representations.

*KL divergence between the intermediate output distribution and the model's final output distribution