Rethinking Sequence Modeling with HyperMLP: An Integrated Architectural Perspective

Attention as dynamic MLPs, expressive routing, and learnable sequence mixing

Jiecheng Lu

Georgia Institute of Technology
Tsinghua University Talk

March 2026

Based primarily on **HyperMLP**, with broader context from our prior series of works.

Talk goals

- Explain **why architecture matters for scaling**: not only a cheaper path along the same curve, but potentially a better curve.
- Reinterpret autoregressive attention as a **dynamic two-layer MLP** whose hidden width grows with context.
- Show how **HyperMLP / HyperGLU** make the sequence-axis representation learnable while preserving autoregressive semantics.
- Highlight the empirical story and application frontiers.

Running theme

Same asymptotic order, more ability. The goal is an **expressivity-oriented lift** in sequence modeling, not only an efficiency-oriented left shift.

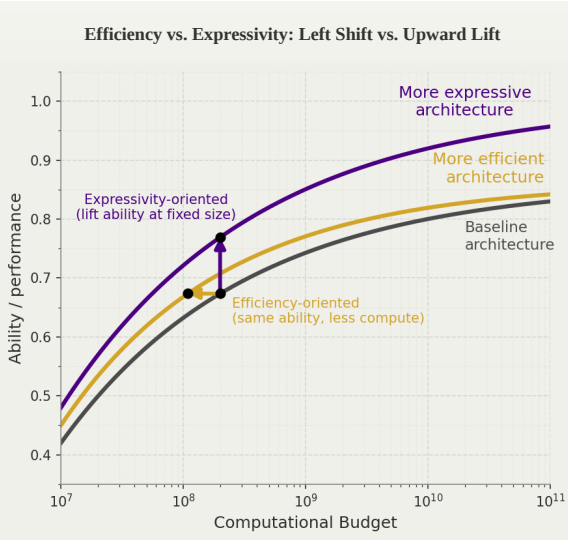
Scaling laws are not architecture-agnostic

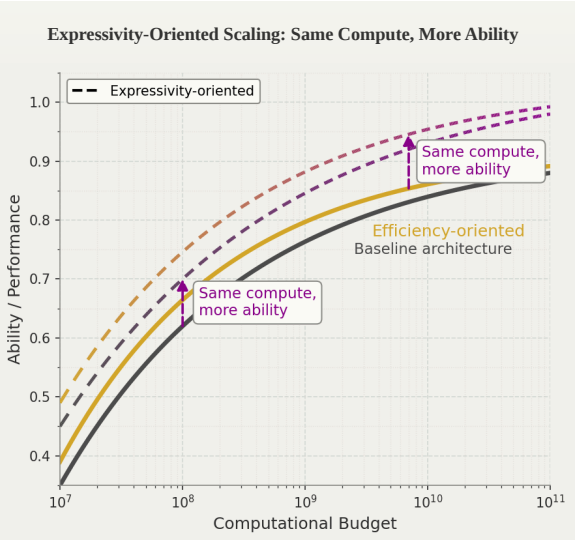
- Scaling laws tell us performance improves smoothly with more resources.
- But the **architecture determines which compute-ability curve** we are on.
- Small constant-factor improvements in the core sequence block may compound into large gains at scale.

Two complementary goals

- **Efficiency**: preserve ability while reducing cost.
- **Expressivity**: improve ability at the same budget.

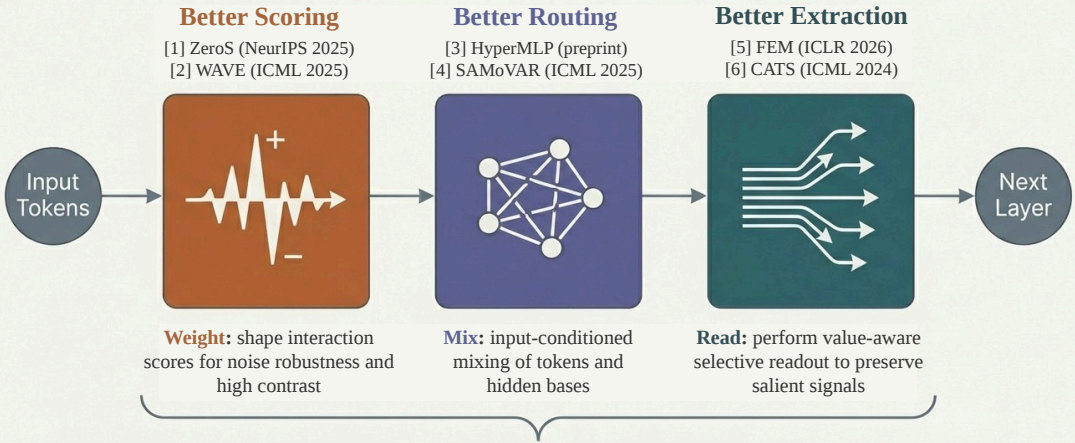
Scaling laws are not architecture-agnostic





Advancing Efficient, Expressive Sequence Models

An Expressivity-Oriented Framework for Modern AI Systems



Sequence computation decomposes into three fundamental operators: Weight (scoring), Mix (routing), and Read (extraction)

The New Sequence Model: Same Complexity, Greater Expressivity

Three operator bottlenecks

$$o_t = \sum_{i \le t} \alpha_{t,i} \, v_i$$

Scoring: generate α

- constraints on weights
- signed / contrastive scoring
- stability in long contexts

[1] ZeroS
[2] WAVE

Routing: match α to memory

- what a score coordinate points to
- routing beyond raw positions
- input-conditioned addressing

[3] HyperMLP - today's focus
[4] SAMoVAR

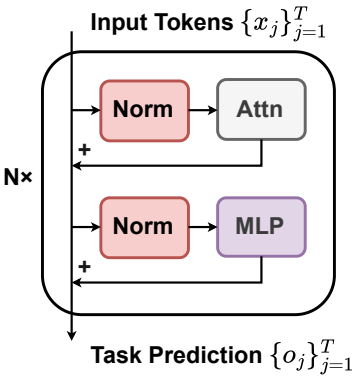
Extraction: combine (α, v)

- value-blind vs. selective readout
- preserve rare events
- channel-wise extraction

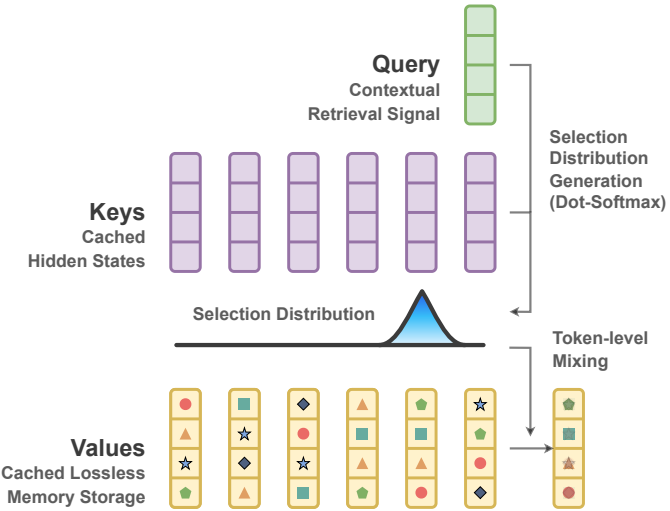
[5] FEM
[6] CATS

Rather than treating attention as a monolith, redesign the sequence block operator by operator.

The structural overview: Transformer blocks and the attention mechanism



Transformer Architecture

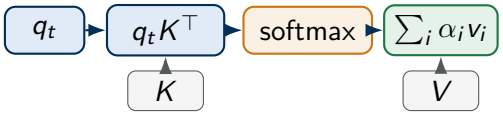


How attention is usually taught

Classical autoregressive head

$$q_t = x_t W_q,\ K = XW_k,\ V = XW_v$$

$$o_t = \mathrm{softmax}(q_t K^\top)\, V\, W_o^\top$$



Three assumptions often inherited

- scores become a **probability distribution**
- hidden coordinates are tied to **token positions**
- the readout is an **expectation-style average**

Key question

This story is intuitive and computationally stable – but it is not obvious that all three assumptions should remain hard constraints for every sequence problem.

Prior works: attention layer structure

$$o_t = \sigma\left(x_t\, W_q\ [1]\, W_k^\top X_{1:t}^\top\, [2]\ \right)\ [3]\ X_{1:t} W_v\ [4]\, W_o^\top$$

Prior works: attention layer structure

1. Softmax
2. ReLU
3. Sigmoid
4. Linearized Kernel
...

$$o_t = \overline{\sigma}\left(x_t W_q \ [1] \ W_k^\top X_{1:t}^\top \ [2] \ \right) \ [3] \ X_{1:t} W_v \ [4] \ W_o^\top$$

Prior works: attention layer structure

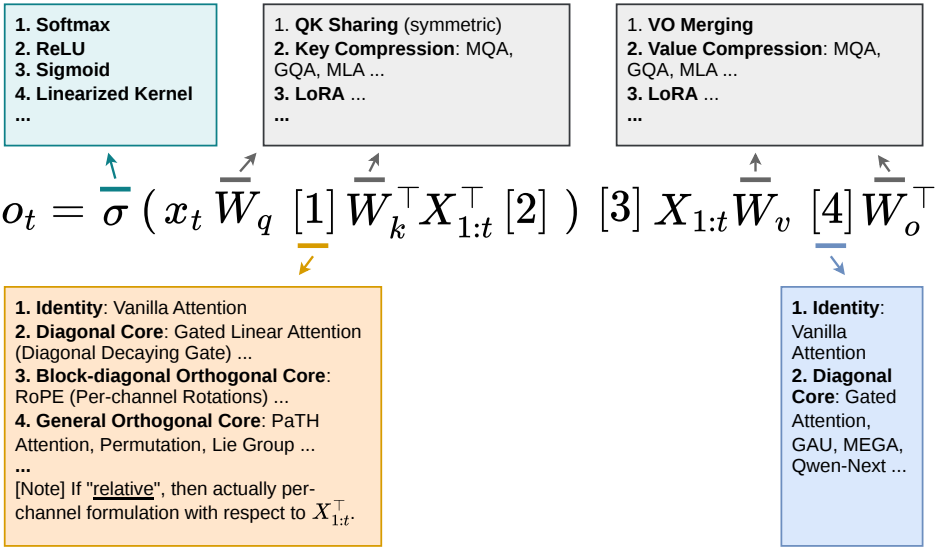
1. Softmax
2. ReLU
3. Sigmoid
4. Linearized Kernel
...

$$o_t = \overline{\sigma}\left(x_t W_q \ \underline{[1]} \ W_k^\top X_{1:t}^\top \ [2] \right) \ [3] \ X_{1:t} W_v \ \underline{[4]} \ W_o^\top$$

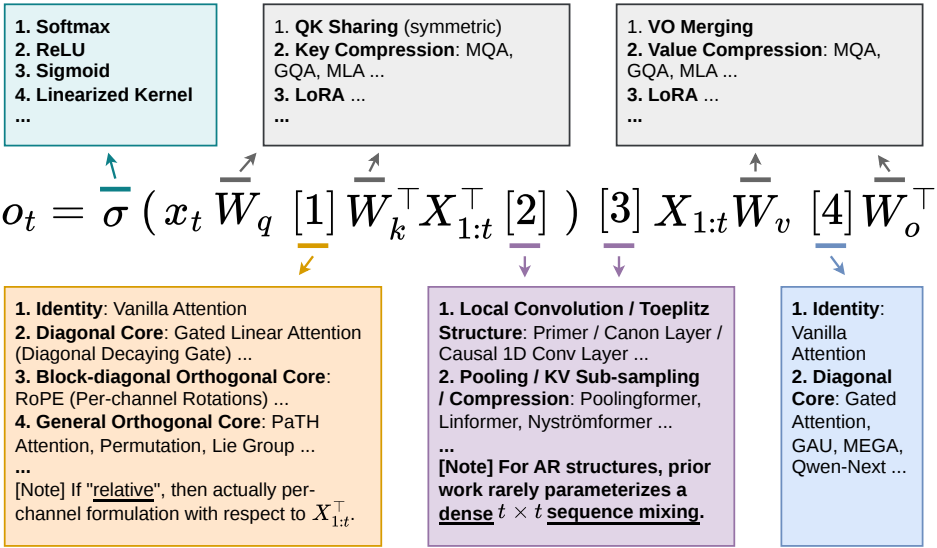
1. Identity: Vanilla Attention
2. Diagonal Core: Gated Linear Attention (Diagonal Decaying Gate) ...
3. Block-diagonal Orthogonal Core: RoPE (Per-channel Rotations) ...
4. General Orthogonal Core: PaTH Attention, Permutation, Lie Group ...
...
[Note] If "relative", then actually per-channel formulation with respect to $X_{1:t}^\top$.

1. Identity: Vanilla Attention
2. Diagonal Core: Gated Attention, GAU, MEGA, Qwen-Next ...

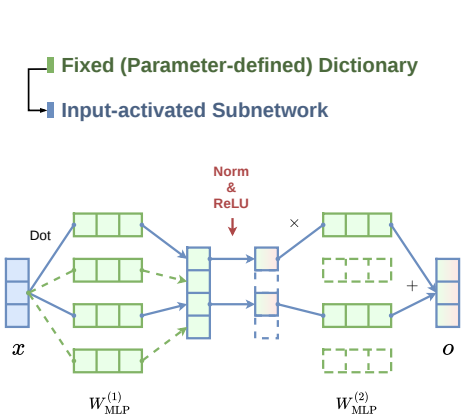
Prior works: attention layer structure



Prior works: attention layer structure

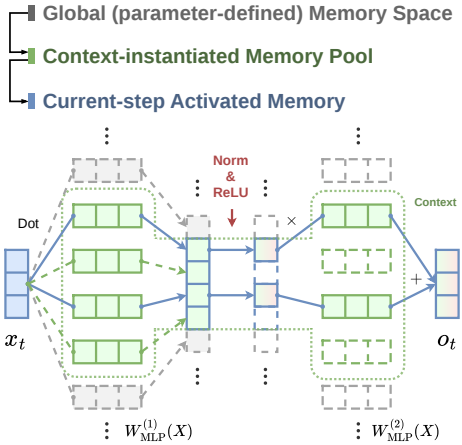


Structural Comparison: ReLU MLP v.s. ReLU Attention



$$o = \text{ReLU}(\text{Norm}(xW^{(1)}_{\text{MLP}}))W^{(2)}_{\text{MLP}}$$

(c) 2-layer ReLU MLP



$$o = \text{ReLU}(\text{Norm}(x_t W_q W_k^\top X^\top))XW_v W_o^\top$$

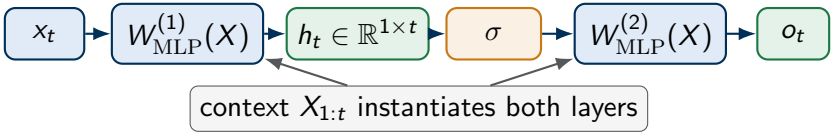
(d) ReLU attention as a dynamic 2-layer ReLU MLP

Attention as a dynamic two-layer MLP

Rewriting one head

$$W^{(1)}_{\mathrm{MLP}}(X) = W_q W_k^\top X^\top, \qquad W^{(2)}_{\mathrm{MLP}}(X) = X W_v W_o^\top$$

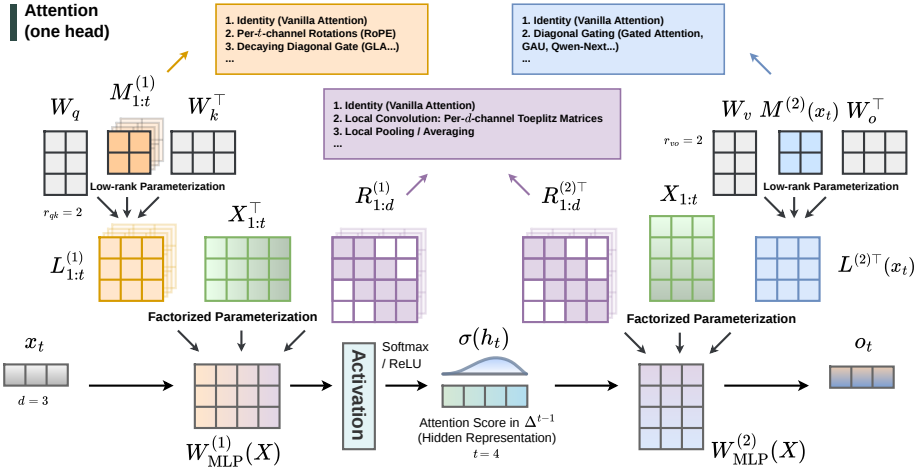
$$o_t = \sigma\big(x_t W^{(1)}_{\mathrm{MLP}}(X)\big)\, W^{(2)}_{\mathrm{MLP}}(X), \qquad h_t := x_t W^{(1)}_{\mathrm{MLP}}(X) \in \mathbb{R}^{1\times t}$$



Interpretation

- A head is a **dynamic MLP**: its weights are instantiated by the prefix.
- The attention map is just a **hidden pre-activation vector** of width t.
- Multi-head attention is a sum of parallel dynamic MLP heads.

The Attention-as-MLP view



(a) The Attention-as-MLP View: Parameterization of the Attention Baselines

Scores are hidden units, not necessarily probabilities

Softmax interpretation

- simplex-constrained weights
- competition across positions
- expectation-style readout

$$\alpha = \mathrm{softmax}(h_t)$$

Dynamic-MLP interpretation

- ReLU / GLU select an active subset of slots
- routing comes from sign patterns and gates
- normalization stabilizes magnitude, not geometry

$$\sigma(z) = \mathrm{ReLU}(\mathrm{L2Norm}_t(z))$$

Useful identity

$$\mathrm{ReLU}(\mathrm{L2Norm}_t(z)) = \frac{1}{\rho_t(z)}\,\mathrm{ReLU}(z)$$

Scalar L2 normalization preserves the selected set and only rescales magnitudes.

The bottleneck in standard attention

$$o_t = \sigma\big(x_t W_q W_k^\top X^\top\,[?]\,\big)\,[?]\,X W_v W_o^\top$$

Fixed positional basis

coordinate i always means
"match to token i"

Why this is restrictive

- The hidden space $\mathbb{R}^t$ is tied to positions, not a learned basis.
- No trainable operator acts along the sequence axis before activation or readout.
- A standard MLP would learn a task-adapted hidden basis.
- **Opportunity:** make the sequence-axis basis learnable.

HyperMLP: make the sequence-axis basis learnable

Core form

$$h_t = x_t\, W^{(1)}_{\mathrm{MLP}}(X_{t:1}), \qquad o_t = \sigma(h_t)\, W^{(2)}_{\mathrm{MLP}}(X_{t:1})$$

$$W^{(1)}_{\mathrm{MLP}}(X_{t:1}) = L^{(1)}(x_t)\, X_{t:1}^{\top}\, R^{(1)}(x_t), \qquad W^{(2)}_{\mathrm{MLP}}(X_{t:1}) = R^{(2)\top}(x_t)\, X_{t:1}\, L^{(2)\top}(x_t)$$

Feature side

$L^{(1)}(x_t), L^{(2)}(x_t)$ provide low-rank, input-conditioned **feature-space mixing**.

Sequence side

$R^{(1)}(x_t), R^{(2)}(x_t)$ make the hidden basis on $\mathbb{R}^t$ **learnable** for routing and readout.

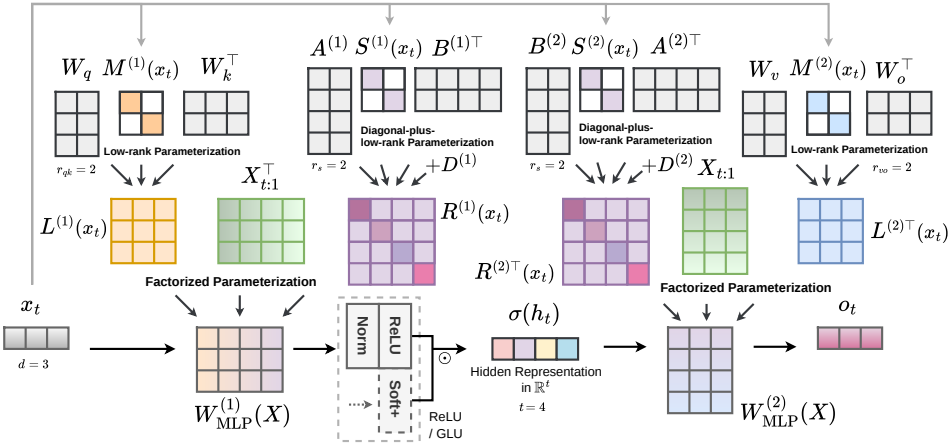
Autoregressive input

Use the lag-ordered history $X_{t:1} = [x_t; x_{t-1}; \ldots; x_1]$.

1. The hidden space is no longer locked to raw positions.
2. Both feature and sequence mixing are conditioned on the current state.
3. HyperMLP is a **richer dynamic-MLP family** than token-wise ReLU attention.

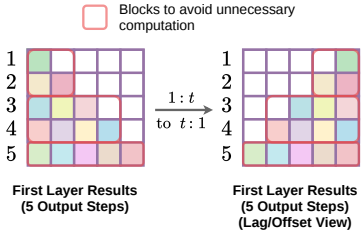
HyperMLP: make the sequence-axis basis learnable

HyperMLP / HyperGLU (one head)



(b) Parameterization of HyperMLP / HyperGLU

Why reverse-offset (lag) order matters



(d) Implementation of the lag/offset layout

Canonical top-left extension preserves the *prefix* of the row ordering. Under lag order, that prefix is exactly the recent window used in autoregressive decoding.

$$[\,0 \,|\, \ldots \,|\, \mathrm{doc}_3 \,|\, \mathrm{doc}_2 \,|\, \mathrm{doc}_1 \,|\, x\,]$$

Canonical extension preserves the prefix

$$R_T^{(m)}(x) = P_{t\to T} R_t^{(m)}(x) P_{t\to T}^\top \Rightarrow o_T(x; \widetilde{X}) = o_t\big(x; P_{t\to T}^\top \widetilde{X}\big).$$

- **Forward order** $X_{1:T}$: prefix = **oldest** t tokens $\Rightarrow$ **wrong invariance** for AR generation.
- **Lag / offset order** $X_{T:1}$: prefix = **most recent** t tokens $\Rightarrow$ adding farther past rows has **zero effect**.
- Therefore lag order is the coordinate system where the **canonical top-left extension** matches the **AR truncation window**.

This is a coordinate-alignment result, not a representational limitation.

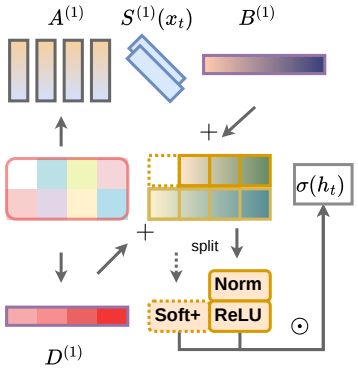
Efficient sequence-space mixing with DPLR operators

Per-head temporal mixer

$$R^{(j)}(x_t) = D^{(j)} + A^{(j)}\, S^{(j)}(x_t)\, B^{(j)\top},\ j \in \{1, 2\}$$

- D: learned per-lag baseline structure.
- $AS(x_t)B^\top$: low-rank interaction across lags, conditioned on the current state.
- $R^{(1)}$ shapes routing-side slots; $R^{(2)}$ shapes readout-side slots.

Extra cost: $O(tr_s)$ per step per head. When $r_s \ll d$, the asymptotic order remains that of quadratic attention, up to lower-order terms.

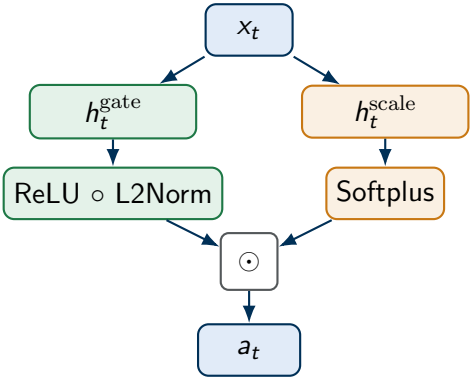


(e) Implementation of the DPLR $R^{(1)}$ and the activation

HyperGLU: separate routing from magnitude

GLU-style activation

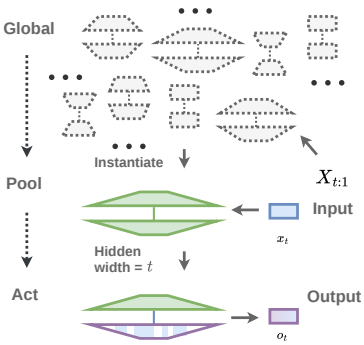
$$a_t = \text{Softplus}(h_t^{\text{scale}}) \odot \text{ReLU}(\text{L2Norm}_t(h_t^{\text{gate}}))$$



What HyperGLU changes

- The **gate branch** determines which slots are active.
- The **scale branch** controls their strength.
- Selection and magnitude no longer share one scalar score.
- Empirically this is usually the strongest matched-budget variant.

An attention head becomes a three-stage memory system



(c) Three-stage memory view

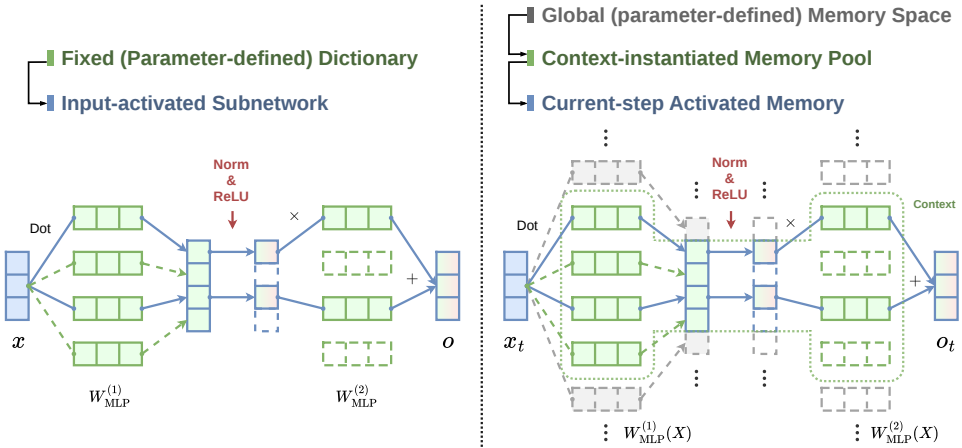
Three-stage view

1. **Global memory space**: the parameter-defined function class the head can instantiate.
2. **Context-instantiated pool**: the current prefix builds candidate slots (u_i, v_i).
3. **Activated memory**: ReLU / GLU choose an input-conditioned subset for readout.

Key implication

Sequence mixing builds **context-wide slots** before routing, so the model selects from a richer memory pool than raw positions alone.

An attention head becomes a three-stage memory system



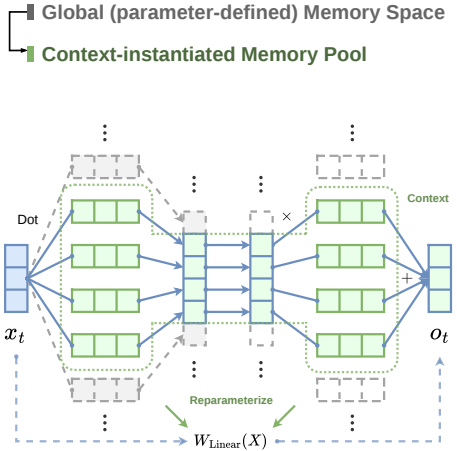
$$o = \text{ReLU}(\text{Norm}(xW^{(1)}_{\text{MLP}}))W^{(2)}_{\text{MLP}}$$

(c) 2-layer ReLU MLP

$$o = \text{ReLU}(\text{Norm}(x_t W_q W_k^\top X^\top))XW_v W_o^\top$$

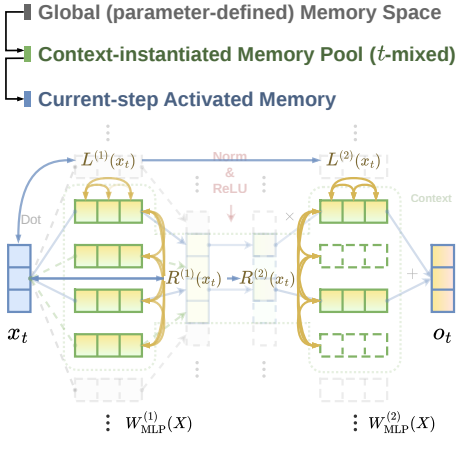
(d) ReLU attention as a dynamic 2-layer ReLU MLP

An attention head becomes a three-stage memory system



$$o = \text{Norm}(x_t W_q W_k^\top X^\top) X W_v W_o^\top$$
$$= x_t W_{\text{Linear}}(X)$$

(e) Linearized attention as a dynamic linear map



$$o$$
$$= \text{ReLU}(\text{Norm}(x_t L^{(1)}(x_t) X^\top R^{(1)}(x_t))) R^{(2)\top}(x_t) X L^{(2)\top}(x_t)$$

(f) HyperMLP with learnable sequence and feature mixing

What the dynamic-MLP view explains about attention design

1. Preserve readout rank

Shrinking VO directly limits the **update subspace**; shrinking QK mostly limits routing.

2. LoRA and gates are efficient on V/O

Readout-side adapters add new **output directions** without changing the address geometry.

3. Linear attention collapses routing

Ungated normalization reads from the **full pool all the time**; there is no active-set pruning.

4. Registers enlarge hidden width

Appending registers adds extra **memory slots** in the same dynamic-MLP hidden space.

The viewpoint gives a common language for routing, gating, low-rank adaptation, and memory augmentation across attention variants.

Controlled design study: what is being varied?

Experimental setting

- **MAD**: mechanistic diagnostics for recall, copy, memorization, and noise robustness.
- **NanoGPT / OpenWebText2**: language modeling, measured by final loss and training progress.
- Matched parameter budgets unless marked with `!`.

Example label parse

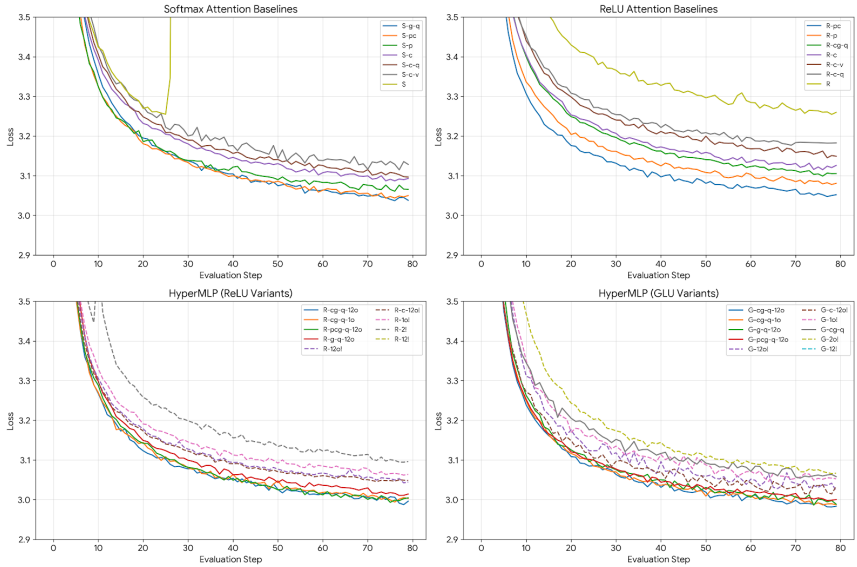
```
G-cg-q-12o = GLU + conv + gates + QK
compression + two-sided mixing + lag
layout
```

Questions isolated by the study

- Is **softmax essential**?
- How much does **sequence-space mixing** matter?
- Is the **lag layout** necessary for autoregressive use?
- Under a fixed budget, should we compress **QK** or **VO**?
- Does GLU-style gating add value beyond ReLU?

Goal: attribute performance changes to specific operator choices, not to scale.

Controlled design study



Controlled study: the key empirical takeaways

1. Sequence mixing is the dominant gain

`R-cg-q` → `R-cg-q-12o`

MAD Avg: 66.78 → 81.01

NanoGPT loss: 3.0828 → 2.9956

Steps to loss < 3.1: 76 → 26.

2. Lag layout is necessary

`R-12!` → `R-12o!`

MAD Avg: 46.27 → 81.82

NanoGPT loss: 4.3567 → 3.0497

3. Softmax is not essential

`S-pc` and `R-pc` are essentially tied:

80.15 vs. 80.50 on MAD Avg

3.0466 vs. 3.0481 on NanoGPT loss

4. HyperGLU is the strongest matched-budget variant

`G-cg-q-12o`

MAD Avg: 81.12, NanoGPT loss: 2.9865

The GLU split helps by decoupling active-set selection from magnitude modulation.

Language modeling results: gains persist at larger scale

340M params / 15B tokens

Model	Avg Rank ↓	#Top1 ↑
SoftmaxAttn	5.28	3
ReLUAttn+KVConv	5.13	1
HyperMLP	2.97	4
HyperGLU	**2.31**	**6**

Selected gains at 340M

HyperGLU improves over SoftmaxAttn on ARC-C, HellaSwag, PIQA, COPA, OBQA, and SciQ, while remaining competitive on reasoning-heavy tasks.

1.3B params / 100B tokens

Model	Avg Rank ↓	#Top1 ↑
SoftmaxAttn	4.22	2
GSA	2.84	4
HyperGLU	**1.88**	**9**

- ReLU attention is already a strong non-softmax baseline.
- Hyper variants improve **broadly**, not just on a single benchmark family.
- The advantage survives - and appears stronger - at the larger scale.

What actually changes? Same complexity order, richer function class

Aspect	Classical attention	HyperMLP / HyperGLU
Hidden geometry	Score coordinates tied to token positions	Sequence-axis basis becomes learnable through $R^{(1)}, R^{(2)}$
Routing	Simplex or token-wise gating over raw positions	Input-conditioned selection over context-wide mixed slots
AR layout	Positional semantics fixed by ordering convention	Lag layout makes extension consistency match AR truncation
Budget use	Often symmetric intuition for QK and VO	Preserve VO rank; "pay" for sequence mixing by shrinking QK first
Complexity	Quadratic attention	Same asymptotic order, plus lower-order $O(tr_s)$ mixing overhead

Interpretation

The gains do not come from more depth or a larger model. They come from changing the **function class per unit compute** of the sequence block.

Application Frontiers

Domains where expressive sequence operators unlock new capabilities

Agentic AI

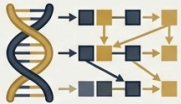
Long-context tool use requiring robust contrastive reasoning over deep historical logs.

Multivariate Time Series

Complex structured signals requiring independent, channel-wise retrieval for varied temporal events.

Multimodal & Vision

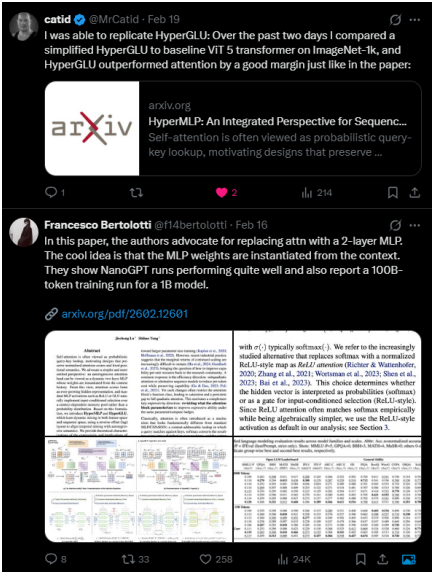
Dense video and patch grids naturally aligned with localized, per-channel feature extraction.

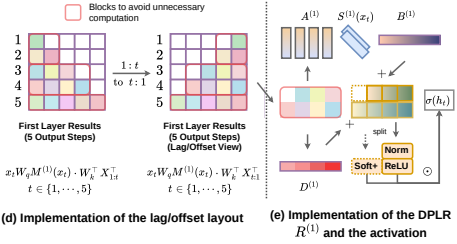


Scientific Sequences

Genomics and telemetry requiring high-fidelity capture of sparse, long-range algorithmic dependencies.

Potential Impact and Limitation





(d) Implementation of the lag/offset layout

(e) Implementation of the DPLR $R^{(1)}$ and the activation

Perspective positioning of the paper

- Most suitable scenario: complex, unordered position basis under a 1D residual stream — Vision, High-dimensional physical data, Complex modal data
- Current limitations:
  1) Cannot directly reuse optimized CUDA kernels such as Flash Attention and its tiling-based computation logic;
  2) Efficient GPU implementation for causal settings: Convolution? FFT? (Proposition G.2)

Takeaways and discussion directions

Three takeaways

1. Attention can be viewed as a **dynamic two-layer MLP**.
2. The fixed positional basis is a real bottleneck; HyperMLP makes it **learnable and AR-consistent**.
3. Under matched budgets, HyperMLP / HyperGLU deliver **consistent ability gains**.

Good directions to explore

- kernel engineering and fused implementations
- multimodal and vision-stream extensions
- scientific sequence modeling
- interpreting learned slots and routing geometry

Thank you!

References

Lu, Jiecheng, et al.
ZeroS: Zero-Sum Linear Attention for Efficient Transformers.
NeurIPS 2025 Spotlight.

Lu, Jiecheng, et al.
WAVE: Weighted Autoregressive Varying Gate for Time Series Forecasting.
ICML 2025.

Lu, Jiecheng, and Shihao Yang.
HyperMLP: An Integrated Perspective for Sequence Modeling.
Preprint, 2026.

Lu, Jiecheng, and Shihao Yang.
Linear Transformers as VAR Models: Aligning Autoregressive Attention Mechanisms with Autoregressive Forecasting.
ICML 2025.

Lu, Jiecheng, and Shihao Yang.
Free Energy Mixer.
ICLR 2026.

Lu, Jiecheng, et al.
CATS: Enhancing Multivariate Time Series Forecasting by Constructing Auxiliary Time Series as Exogenous Variables
ICML 2024.

More information: slides / code / updates

Code: github.com/LJC-FVNR/HyperMLP Personal page: https://ljc-fvnr.github.io/