

Understanding Transformers through the Lens of Pavlovian Conditioning

Mu Qiao

Meta Platforms, Inc.
muqiao0626@gmail.com

Abstract

Transformer architectures have revolutionized artificial intelligence (AI) through their attention mechanisms, yet the computational principles underlying their success remain opaque. We present a novel theoretical framework that reinterprets the core computation of attention as Pavlovian conditioning. Our model finds a direct mathematical analogue in linear attention, which simplifies the analysis of the underlying associative process. We demonstrate that attention’s queries, keys, and values can be mapped to the three elements of classical conditioning: test stimuli that probe associations, conditional stimuli (CS) that serve as retrieval cues, and unconditional stimuli (US) that contain response information. Through this lens, we suggest that each attention operation constructs a transient associative memory via a Hebbian rule, where CS-US pairs form dynamic associations that test stimuli can later retrieve. Our framework yields several theoretical insights grounded in this linearized model: (1) a capacity theorem showing that attention heads can store $O(\sqrt{d_k})$ associations before interference degrades retrieval; (2) an error propagation analysis revealing fundamental architectural trade-offs of balancing model depth, width, and head redundancy to maintain reliability; and (3) an understanding of how biologically plausible learning rules could enhance transformer architectures. By establishing this deep connection, we suggest that the success of modern AI may stem not from architectural novelty alone, but from implementing computational principles that biology optimized over millions of years of evolution.

1 Introduction

The transformer architecture [30] has revolutionized artificial intelligence (AI), achieving unprecedented performance in language modeling, computer vision, and beyond. At the heart of this revolution lies the attention mechanism, a deceptively simple operation that computes weighted averages of values based on query-key similarities. Yet despite transformers’ ubiquity, we lack a satisfying explanation for a fundamental question: Why does this particular computation work so well?

The standard mathematical description of attention as is operationally clear but intellectually unsatisfying. It tells us *what* attention computes but not *why* this computation captures something essential about intelligence. Current interpretability work [9, 20] has made progress identifying specific computational patterns, but these descriptive accounts still leave the core mystery unresolved.

We propose a fundamental reinterpretation: the core operation of transformer attention can be understood as a form of Pavlovian conditioning, one of the most basic and universal learning mechanisms in nature. Drawing from classical conditioning theory [21], we propose a mapping where attention’s three components correspond to conditioning elements:

- Values (V) \rightarrow Unconditional stimuli (US): Information that directly encodes responses

- Keys (K) \rightarrow Conditional stimuli (CS): Contextual patterns that become associated with US
- Queries (Q) \rightarrow Test stimuli: Patterns that probe learned associations for retrieval

This decomposition illuminates attention’s fundamental operation as a process of dynamic association. During each forward pass, key (CS)-value (US) pairs form associations via a Hebbian rule [12, 10], which queries (test stimuli) then probe through similarity matching. This is not only an analogy; we demonstrate that our conditioning framework is mathematically equivalent to linear attention, a simplified yet powerful variant of the standard mechanism. This provides a tractable foundation for our theoretical analysis.

Beyond technical contributions, this work suggests a profound principle: the convergence between certain AI architectures and neuroscience may not be coincidental. By implementing conditioning principles, these models may tap into computational solutions that biology has optimized through evolution. This perspective not only helps explain current success but also points toward principled architectural improvements inspired by neuroscience.

2 Background and Related Work

2.1 Transformer Attention Mechanisms

Standard transformer attention [30] operates on an input sequence $\mathbf{X} \in \mathbb{R}^{n \times m}$ containing n tokens, each represented by an m -dimensional vector. The attention mechanism projects this input through learned weight matrices to produce queries, keys, and values:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V \quad (1)$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{m \times d}$ project inputs into a d -dimensional latent space. The attention output is computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right) \mathbf{V} \quad (2)$$

which is then transformed through the output projection \mathbf{W}_O . The scaling factor \sqrt{d} prevents gradient instability but can be absorbed into the weight matrices \mathbf{W}_Q and \mathbf{W}_K [9].

Recent work [9, 20] has made progress in mechanistic interpretability by analyzing specific computational patterns. These studies identify "circuits" as functional subgraphs within transformers, often focusing on the composition matrices $\mathbf{W}_Q\mathbf{W}_K^\top$ (determining attention patterns) and $\mathbf{W}_V\mathbf{W}_O$ (determining information flow). However, these approaches primarily provide descriptive accounts of *what* transformers compute rather than explaining *why* these computations are effective.

2.2 Linear Transformer Attention

The quadratic complexity of standard attention poses computational challenges for long sequences. Linear attention methods [29, 14, 6] address this by replacing the softmax with decomposable kernel functions, reducing complexity from $O(n^2)$ to $O(n)$.

The key insight is to approximate the softmax-weighted sum with a kernelized outer-product formulation. A common approach involves a non-linear kernel function ϕ :

$$\text{LinearAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \frac{\phi(\mathbf{Q})(\phi(\mathbf{K})^\top \mathbf{V})}{\phi(\mathbf{Q})\phi(\mathbf{K})^\top \mathbf{1}} \quad (3)$$

where $\mathbf{1}$ is a vector of ones and the $\mathbf{Q}\mathbf{K}^\top$ matrix is never explicitly formed [14, 6].

While early methods included a normalization factor, later work demonstrated that this can lead to instability and that applying normalization to the entire output is more robust [22]:

$$\text{NormAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Norm}\left(\phi(\mathbf{Q})(\phi(\mathbf{K})^\top \mathbf{V})\right) \quad (4)$$

where the normalization can be LayerNorm [2] or RMSNorm [34]. This shifts focus to understanding the core operation $\phi(\mathbf{Q})(\phi(\mathbf{K})^\top \mathbf{V})$.

For a specific token $\mathbf{x}_i \in \mathbb{R}^{1 \times m}$ with corresponding query, key, and value:

$$\mathbf{q}_i = \mathbf{x}_i \mathbf{W}_Q, \quad \mathbf{k}_i = \mathbf{x}_i \mathbf{W}_K, \quad \mathbf{v}_i = \mathbf{x}_i \mathbf{W}_V \quad (5)$$

the above norm attention output \mathbf{o}_i becomes:

$$\mathbf{o}_i = \text{Norm}(\phi(\mathbf{q}_i) \sum_{j=1}^i \phi(\mathbf{k}_j)^\top \mathbf{v}_j) \quad (6)$$

This can be rewritten as:

$$\mathbf{o}_i = \text{Norm}(\phi(\mathbf{q}_i) \mathbf{S}_i) \quad (7)$$

where $\mathbf{S}_i = \sum_{j=1}^i \phi(\mathbf{k}_j)^\top \mathbf{v}_j$. This formulation naturally implements causal masking (only attending to past information) and enables efficient iterative updates [14, 22].:

$$\mathbf{S}_i = \mathbf{S}_{i-1} + \phi(\mathbf{k}_i)^\top \mathbf{v}_i \quad (8)$$

As we will show, the linear formulation provides a direct mathematical realization of a classical conditioning circuit, making it an ideal starting point for our theoretical investigation.

2.3 Classical Conditioning and Neurobiology

Pavlovian conditioning, or classical conditioning [21] represents one of the most fundamental learning mechanisms in biology, where organisms learn to associate neutral stimuli with meaningful outcomes through experience. In classical conditioning experiments, an unconditional stimulus (US) naturally triggers a response (e.g. smell of food causing salivation). When paired with the US, an initially neutral stimulus (e.g. sound from a bell), called the conditional stimulus (CS), emerges to elicit a similar response (e.g. salivation). Importantly, conditioning exhibits stimulus generalization [11]: after conditioning, test stimuli similar to the original CS can elicit the response, with strength proportional to similarity.

In biological neural circuits, this process involves distinct anatomical pathways: sensory neurons carrying conditional stimulus (CS) information converge with those carrying unconditional stimulus (US) information at specific synaptic sites [17], where Hebbian plasticity—"cells that fire together, wire together"—creates lasting associations [12, 10]. This biological foundation suggests that effective learning systems should: (1) separate sensory inputs into distinct processing streams, (2) form associations through Hebbian rule.

3 Theoretical Framework

Having established the biological principles of conditioning in the previous section, we now demonstrate how transformer attention implements these same computational principles in artificial neural networks. The key insight is that attention's queries, keys, and values directly map onto the conditioning paradigm: test stimuli, CS, and US. Just as biological conditioning requires separate CS and US pathways that converge through activity-dependent plasticity, attention employs distinct computational streams for keys (CS), values (US), and queries (test stimuli) that form dynamic associations through multiplicative interactions.

To make this mapping explicit, consider the neural circuit in Figure 1. In the US pathway, an input stimulus \mathbf{z}_j activates a population of neurons in the hidden layer through a fixed (e.g., genetically encoded) weight matrix \mathbf{W}_V , producing responses $g(\mathbf{z}_j \mathbf{W}_V)$ that directly drive downstream activity. In parallel, the CS pathway processes stimulus \mathbf{y}_j through weights \mathbf{W}_K , generating hidden representations $f(\mathbf{y}_j \mathbf{W}_K)$. Initially, CS activation alone cannot trigger the US response. However, repeated CS-US pairing leads to Hebbian strengthening of connections between these pathways: $\Delta \mathbf{S} \propto f(\mathbf{y}_j \mathbf{W}_K) g(\mathbf{z}_j \mathbf{W}_V)^\top$. Once these associations form, a test stimulus \mathbf{x}_i can activate the CS pathway through weights \mathbf{W}_Q , which then retrieves the associated US response through the potentiated connections \mathbf{S} , implementing the conditioned response.

3.1 Classical Conditioning Circuit

This circuit processes three distinct stimulus types through separate pathways:

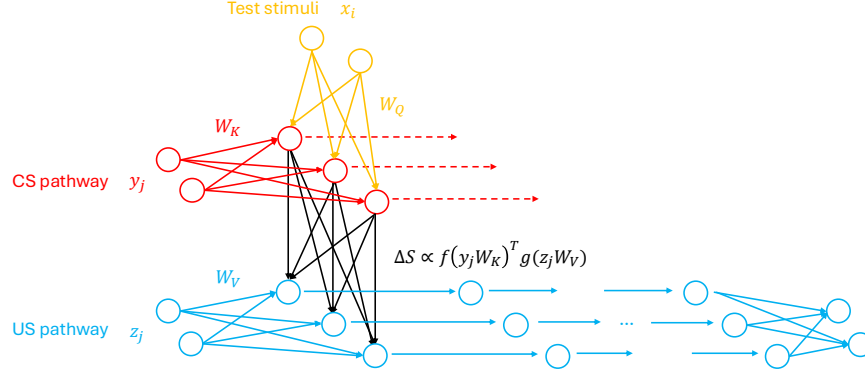


Figure 1: Transformer attention as a conditioning circuit. Conditional stimuli (CS) \mathbf{y}_j and unconditional stimuli (US) \mathbf{z}_j form associations through Hebbian learning: $\Delta \mathbf{S} \propto f(\mathbf{y}_j \mathbf{W}_K)^T g(\mathbf{z}_j \mathbf{W}_V)$. Test stimuli \mathbf{x}_i probe these associations via \mathbf{W}_Q .

Definition 1 (Conditioning Architecture). *Given n time-step sequences for test stimuli $\mathbf{X} \in \mathbb{R}^{n \times m_x}$, conditional stimuli $\mathbf{Y} \in \mathbb{R}^{n \times m_y}$, and unconditional stimuli $\mathbf{Z} \in \mathbb{R}^{n \times m_z}$, the attention mechanism implements:*

$$\text{Test pathway: } f(\mathbf{X} \mathbf{W}_Q) \in \mathbb{R}^{n \times d_k} \quad (9)$$

$$\text{CS pathway: } f(\mathbf{Y} \mathbf{W}_K) \in \mathbb{R}^{n \times d_k} \quad (10)$$

$$\text{US pathway: } g(\mathbf{Z} \mathbf{W}_V) \in \mathbb{R}^{n \times d_v} \quad (11)$$

where f and g are activation functions of the hidden layer neurons, and $\mathbf{W}_Q \in \mathbb{R}^{m_x \times d_k}$, $\mathbf{W}_K \in \mathbb{R}^{m_y \times d_k}$, $\mathbf{W}_V \in \mathbb{R}^{m_z \times d_v}$ are learned projections.

This architecture directly parallels biological conditioning circuits where CS and US information streams remain segregated until they converge at association sites.

3.2 Hebbian Association Formation

The core of our framework is the Hebbian learning principle [12, 10]: at each time point, synaptic strength changes proportionally to the correlation between pre- and post-synaptic activity.

Definition 2 (Hebbian Association). *During the forward pass, CS-US associations accumulate dynamically through Hebbian updates:*

$$\mathbf{S}_i = \alpha \sum_{j=1}^i f(\mathbf{k}_j)^T g(\mathbf{v}_j) = \alpha \sum_{j=1}^i f(\mathbf{y}_j \mathbf{W}_K)^T g(\mathbf{z}_j \mathbf{W}_V) \quad (12)$$

where $f(\mathbf{k}_j)$ and $g(\mathbf{v}_j)$ are the CS and US representations at time j , and α is the association strength factor.

This implements the biological principle $\Delta \mathbf{S} = \alpha f(\mathbf{k}_j)^T g(\mathbf{v}_j)$, where synaptic connections strengthen when CS and US neurons fire together. Crucially, this association is activity-driven: both pre-synaptic (CS) and post-synaptic (US) activations are externally imposed during pairing, creating supervised Hebbian learning. After association, the synaptic matrix \mathbf{S} enables test stimuli to retrieve the US responses previously paired with similar CS patterns.

This can be viewed as inference-time learning—the weight matrices \mathbf{W}_K and \mathbf{W}_V are fixed during inference, but the associative memory \mathbf{S} forms dynamically based on the specific CS-US pairings in the current context. This mirrors biological synapses that form temporary associations based on recent activity patterns.

3.3 Stimulus Generalization through Test Queries

The test pathway enables stimulus generalization, a characteristic of conditioning:

Definition 3 (Test Stimulus Retrieval). *Given a test stimulus at time point i , \mathbf{x}_i , the retrieval process implements:*

$$\mathbf{r}_i = f(\mathbf{q}_i)\mathbf{S}_i = \alpha f(\mathbf{q}_i) \sum_{j=1}^i f(\mathbf{k}_j)^\top g(\mathbf{v}_j) \quad (\text{Retrieve via associations}) \quad (13)$$

$$= \alpha \sum_{j=1}^i (f(\mathbf{q}_i)f(\mathbf{k}_j)^\top)g(\mathbf{v}_j) \quad (\text{Weighted US responses}) \quad (14)$$

where $f(\mathbf{q}_i)$ denotes test stimulus encoding.

This process directly implements stimulus generalization: the similarity $f(\mathbf{q}_i)f(\mathbf{k}_j)^\top$ measures how closely the test stimulus matches each CS pattern. Test stimuli similar to previously paired CS patterns (high $f(\mathbf{q}_i)f(\mathbf{k}_j)^\top$) strongly activate their associated US responses $g(\mathbf{v}_j)$. The retrieval is thus a similarity-weighted combination of all stored US responses, implementing the graded generalization observed in biological conditioning.

3.4 Normalization as Neural Computation

While our framework naturally produces weighted US responses through stimulus generalization, biological neural circuits employ normalization to prevent saturation and enhance selectivity [5]. In our framework, this normalization occurs at the convergence site where retrieved associations activate US pathway neurons.

Definition 4 (Divisive Normalization). *The normalization operation applied to the retrieved response:*

$$\mathbf{o}_i = \text{Norm}(\mathbf{r}_i) = \text{Norm} \left(\alpha f(\mathbf{q}_i) \sum_{j=1}^i f(\mathbf{k}_j)^\top g(\mathbf{v}_j) \right) \quad (15)$$

implements divisive inhibition by normalizing responses across the feature dimension.

This operation serves two critical functions: amplifying differences between competing US responses for enhanced selectivity, and ensuring consistent output magnitudes regardless of sequence length. Biologically, divisive normalization arises through multiple mechanisms, including inhibitory interneurons that pool activity across neural populations and provide divisive feedback [5].

4 Mathematical Analysis

4.1 Equivalence to Linear Attention

We first demonstrate that our conditioning framework finds a direct mathematical realization in linear attention under specific, interpretable conditions:

Theorem 5 (Linear Attention as Conditioning). *When the conditioning framework employs:*

1. *Activation functions:* $f = \phi, g = I$
2. *Association strength factor:* $\alpha = 1$
3. *Self-attention configuration:* $\mathbf{X} = \mathbf{Y} = \mathbf{Z}$
4. *Hidden-layer dimension:* $d_k = d_v = d$

then Equation 15 reduces exactly to the linear attention formulation in Equation 6.

This theorem establishes linear attention as a concrete implementation of our conditioning model. While this formulation does not capture the competitive, winner-take-all dynamics of softmax attention, it provides a tractable foundation to analyze the underlying associative memory formation and its limitations. The principles derived from this model, we argue, offer valuable insights into the general function of attention.

4.2 Memory Capacity and Interference

We analyze how many CS-US associations our conditioning framework can reliably store before interference degrades performance.

Theorem 6 (Associative Memory Capacity). *The number of associations n that can be reliably stored and retrieved from $\mathbf{S} \in \mathbb{R}^{d_k \times d_v}$ is limited by the dimension of the CS representations d_k . The number of associations that can be reliably retrieved is bounded by:*

1. **Average case:** $n < 1 + \frac{d_k}{\gamma}$ where γ is the required signal-to-noise ratio
2. **Worst case (high probability):** $n < \sqrt{\epsilon \delta d_k}$ where ϵ is the failure probability and δ is the noise threshold

See Appendix A.1 for detailed proof.

This capacity limitation has profound implications: as context length increases, earlier associations become progressively harder to retrieve due to interference from newer associations. While increasing head dimensions improves capacity [28], the fundamental constraint remains. Eventually, the memory becomes saturated and retrieval quality degrades.

4.3 Dynamic Association Strength Factor and Temporal Forgetting

The capacity limitation revealed above asks an important question: rather than storing all associations equally, could selective forgetting improve performance? Biological memory systems actively forget older information to make room for new associations, and we can implement this principle through dynamic strength factor.

Definition 7 (Dynamic Association Strength Factor). *Instead of the constant strength factor α , we introduce time-dependent weights that modulate association strength:*

$$\mathbf{S}_i = \sum_{j=1}^i \alpha_{ij} f(\mathbf{k}_j)^\top g(\mathbf{v}_j) \quad (16)$$

where α_{ij} controls the strength of association between positions i and j .

A particularly elegant choice is exponential decay: $\alpha_{ij} = \gamma^{i-j}$ for $\gamma \in (0, 1)$. This yields the recursive update:

$$\mathbf{S}_i = \gamma \mathbf{S}_{i-1} + f(\mathbf{k}_i)^\top g(\mathbf{v}_i) \quad (17)$$

This implements a "forgetting curve" where older associations decay exponentially, preventing memory saturation while maintaining a fixed effective capacity. The decay rate γ controls the trade-off between memory depth and retrieval quality, which is precisely the mechanism used in RetNet [28].

5 Theoretical Implications

5.1 Higher-Order Conditioning: Stacking Multiple Circuits

Our single-layer conditioning framework naturally extends to deep architectures by stacking multiple conditioning circuits. This allows us to model deep networks as performing higher-order conditioning, where the outputs of one associative layer become the inputs for the next.

5.1.1 Attention-Only Transformers

We focus on "attention-only" transformers, where multi-layer perceptron (MLP) layers are omitted, to isolate the associative dynamics of attention. While MLPs are crucial in practice, this simplification allows us to analyze the core conditioning principles. Our analysis thus applies to stacked linear attention layers. Many key transformer behaviors—including in-context learning and pattern matching—are primarily mediated by attention [9], making this a revealing simplification.

Definition 8 (Higher-Order Conditioning). *For an attention-only transformer with total depth L , each layer $\ell \in \{1, \dots, L\}$ with H attention heads implements:*

$$\mathbf{S}_i^{(\ell,h)} = \sum_{j=1}^i f(\mathbf{y}_j^{(\ell-1)} \mathbf{W}_K^{(\ell,h)})^\top g(\mathbf{y}_j^{(\ell-1)} \mathbf{W}_V^{(\ell,h)}) \quad (18)$$

$$\mathbf{o}_i^{(\ell,h)} = \text{Norm}(f(\mathbf{y}_i^{(\ell-1)} \mathbf{W}_Q^{(\ell,h)}) \mathbf{S}_i^{(\ell,h)}) \quad (19)$$

$$\mathbf{y}_i^{(\ell)} = \mathbf{y}_i^{(\ell-1)} + \sum_{h=1}^H \mathbf{o}_i^{(\ell,h)} \mathbf{W}_O^{(\ell,h)} \quad (20)$$

where $\mathbf{y}_i^{(0)} = \mathbf{x}_i$ is the input.

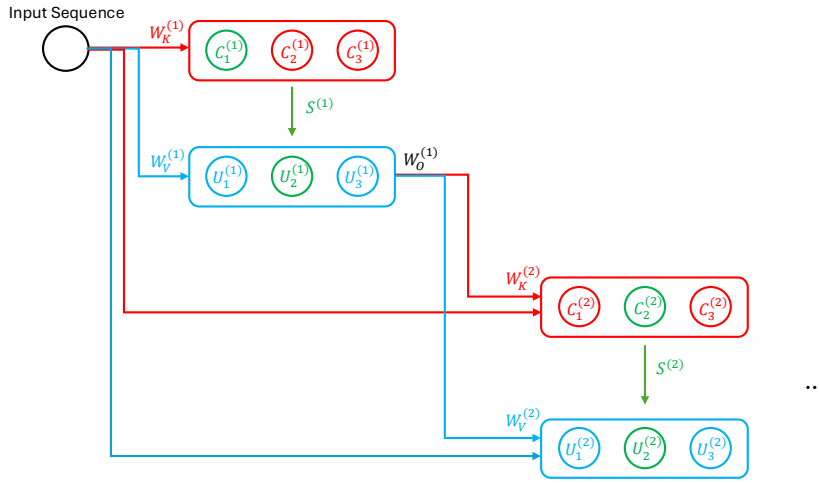


Figure 2: Higher-order conditioning through stacked circuits. The input sequence is processed through multiple layers, each forming its own CS–US associations via $\mathbf{S}^{(\ell)}$. Dynamic routing emerges as different contexts activate different association pathways (e.g., $C_1^{(1)} \rightarrow U_2^{(1)}$ representing “animal” \rightarrow “mammal”, which then informs the second-layer association $C_2^{(2)} \rightarrow U_2^{(2)}$ for “mammal” \rightarrow “dog”), enabling context-dependent information flow.

5.1.2 Higher-Order Association Building

Figure 2 illustrates how higher-order conditioning can enable compositional reasoning. Consider a concrete example: processing the sentence "If an animal is a mammal, check whether it is a dog." Our model provides a mechanistic account for how this reasoning could unfold:

First-Order Associations (Layer 1): $C_1^{(1)}$ encodes "animal" (subject concept) and $U_2^{(1)}$ encodes "mammal" (category concept). The association matrix $\mathbf{S}^{(1)}$ captures the relationship: animal \rightarrow mammal.

Second-Order Associations (Layer 2): $C_2^{(2)}$ (influenced by $U_2^{(1)}$) now represents "mammal" as a conditional stimulus. When processing "...it is a dog," the network forms a new association where $U_2^{(2)}$ encodes "dog" (specific instance) and $\mathbf{S}^{(2)}$ captures: mammal \rightarrow dog.

This two-layer process demonstrates higher-order conditioning: Layer 1 learns a general category membership, while Layer 2 uses that category to learn a more specific instance relationship.

5.1.3 Dynamic Routing and Context-Dependent Processing

The power of higher-order conditioning lies in its dynamic, context-dependent nature. Association matrices $\mathbf{S}^{(\ell)}$ form during inference based on the specific input, creating flexible computational pathways.

Consider an alternative input: "If an animal is a reptile, check whether it is a lizard." This creates a different pathway:

Layer 1: $C_1^{(1)}$ (animal) $\rightarrow U_1^{(1)}$ (reptile)

Layer 2: $C_1^{(2)}$ (reptile-influenced) $\rightarrow U_1^{(2)}$ (lizard)

The same network architecture supports both reasoning paths based on different input context:

Path 1: $C_1^{(1)} \rightarrow U_2^{(1)} \rightarrow C_2^{(2)} \rightarrow U_2^{(2)}$ for animal \rightarrow mammal \rightarrow dog

Path 2: $C_1^{(1)} \rightarrow U_1^{(1)} \rightarrow C_1^{(2)} \rightarrow U_1^{(2)}$ for animal \rightarrow reptile \rightarrow lizard

This dynamic routing mirrors neurobiological findings where cortical pathways adapt based on task demands, with different contexts activating different neural circuits for the same computational goal [20].

5.1.4 Implications for Transformer Capabilities

In-Context Learning: Higher-order conditioning naturally implements few-shot learning. Given examples in context, Layer 1 forms temporary associations between patterns and outcomes. Layer 2 then uses these associations to process new queries, effectively learning new connections from context alone.

Dynamic Task Adaptation: Since associations form during inference, the same network can perform different tasks by building different computational paths based on context. Instructions in the prompt create specific association patterns that guide subsequent processing, explaining how transformers follow diverse instructions without task-specific training.

Multi-Step Inference: This architecture provides a mechanistic account of multi-step reasoning [15]. Analogous to second-order conditioning (e.g. light \rightarrow bell \rightarrow food), the layered structure supports transitive inference by building a cascade of associations. Given premises $A \rightarrow B$, and $B \rightarrow C$, the system can infer $A \rightarrow C$. This perspective suggests that the depth of a transformer L affects the order of associations possible, explaining why larger models exhibit more sophisticated reasoning capabilities [31].

5.2 Error Propagation in Stacked Conditioning Circuits

While higher-order conditioning enables multi-step inference, it introduces a challenge: errors compound through layers. We analyze how reliability constraints limit the depth and complexity of reasoning chains.

Theorem 9 (Error Accumulation in Deep Conditioning). *For a depth- L transformer with H heads per layer, the error rate upper bound r^* for tasks requiring correct retrieval across all layers scales as:*

$$r^* \propto \frac{L \cdot n^H}{d_k^H} \quad (21)$$

where n is the context length and d_k is the head dimension.

See Appendix A.2 for detailed proof.

5.2.1 Architectural Trade-offs

The error scaling law reveals fundamental trade-offs in transformer design:

1. Depth-Width Balance: To reduce the error rate, we could trade the model depth L with width parameters (H or d_k). This theoretical insight aligns with recent findings that transformers often benefit from rebalancing depth and width, and architectures with somewhat fewer layers but wider

attention heads can match or exceed the performance of deeper, narrower models in certain vision and language tasks [23].

2. Head Redundancy: The exponential improvement with H shows that multiple heads provide crucial redundancy. With H heads, error scales as n^H/d_k^H , making the model extremely robust when $d_k > n$. This redundancy allows different heads to specialize while maintaining overall reliability.

This explains why successful transformer architectures balance these parameters carefully, typically using moderate depth with many wide heads rather than extremely deep networks with few narrow heads.

5.2.2 Implications for Model Design

Adaptive Depth Mechanisms: Models could dynamically modulate their effective depth according to task complexity. Incorporating early-exit mechanisms could mitigate unnecessary error propagation on simpler tasks, while preserving advanced reasoning potential for more complex scenarios.

Task-Specific Routing: Architectures should use adaptive routing to direct different problem types through appropriately complex sub-networks, optimizing the performance-reliability trade-off. Mixture of experts (MoE) architectures [25, 7] exemplify this principle: specialized experts can handle different association complexities.

Reliability-Aware Training: Training paradigms may explicitly penalize redundant layer utilization, incentivizing efficient layer deployment aligned with task-specific demands.

5.3 Variants of Hebbian Rule

The error propagation analysis reveals a critical need for mechanisms that can correct mistakes and maintain stability across deep networks. While basic Hebbian learning captures association formation, its variants offer solutions to the reliability challenges identified above.

5.3.1 Delta Rule: Error-Correcting Associations

The Delta rule directly addresses error accumulation by correcting predictions before forming new associations. This "unlearning" step makes it a powerful mechanism for online adaptation [24, 32]:

Definition 10 (Delta Rule). *Instead of purely additive updates, the delta rule corrects existing associations:*

$$\mathbf{S}_i = \mathbf{S}_{i-1} + \alpha f(\mathbf{k}_i)^\top [g(\mathbf{v}_i) - f(\mathbf{k}_i)\mathbf{S}_{i-1}] \quad (22)$$

where $f(\mathbf{k}_i)\mathbf{S}_{i-1}$ represents the current prediction for input \mathbf{k}_i .

Expanding this update rule:

$$\mathbf{S}_i = (\mathbf{I} - \alpha f(\mathbf{k}_i)^\top f(\mathbf{k}_i)) \mathbf{S}_{i-1} + \alpha f(\mathbf{k}_i)^\top g(\mathbf{v}_i) \quad (23)$$

The first term selectively "erases" the old association for pattern \mathbf{k}_i , while the second "writes" the correct association. This error-correcting mechanism directly addresses the reliability issues in deep networks by allowing each layer to fix mistakes from previous retrievals.

5.3.2 Oja's Rule: Stable Learning Through Homeostasis

Oja's rule [19] introduces a homeostatic mechanism that prevents the unbounded growth of synaptic weights, a key cause of training instability. By automatically down-scaling connections based on post-synaptic activity, it maintains stability without external control:

Definition 11 (Oja's Rule). *Oja's rule adds a stabilizing term:*

$$\mathbf{S}_i = \mathbf{S}_{i-1} + \alpha [f(\mathbf{k}_i)^\top g(\mathbf{v}_i) - \mathbf{S}_{i-1} \cdot \text{diag}(g(\mathbf{v}_i)^\top g(\mathbf{v}_i))] \quad (24)$$

Expanding this reveals the homeostatic mechanism:

$$\mathbf{S}_i = \mathbf{S}_{i-1} (\mathbf{I} - \alpha \text{diag}(g(\mathbf{v}_i)^\top g(\mathbf{v}_i))) + \alpha f(\mathbf{k}_i)^\top g(\mathbf{v}_i) \quad (25)$$

Each column of \mathbf{S} (representing connections to one output neuron) is scaled down by that neuron’s squared activity. This creates a self-regulating system: highly active neurons automatically reduce their input weights, preventing saturation while maintaining relative association strengths. This self-regulating mechanism could offer a principled, biologically-inspired alternative to engineering solutions like gradient clipping, which also serve to maintain stable activity levels.

5.3.3 BCM Rule: Threshold-Based Plasticity

The Bienenstock-Cooper-Munro (BCM) rule [3] introduces a dynamic learning threshold, enabling bidirectional plasticity: associations are strengthened for surprising or highly salient inputs (activity above threshold) and weakened for predictable ones (activity below threshold):

Definition 12 (BCM Rule). *Learning depends on whether activity exceeds a dynamic threshold:*

$$\mathbf{S}_i = \mathbf{S}_{i-1} + \alpha f(\mathbf{k}_i)^\top \phi(g(\mathbf{v}_i), \theta_i) \quad (26)$$

where $\phi(x, \theta) = x(x - \theta)$ and θ_i is a threshold based on time-average of recent activity.

This creates bidirectional plasticity: Below threshold ($g(\mathbf{v}_i) < \theta_i$), it causes depression (negative ϕ), while above threshold ($g(\mathbf{v}_i) > \theta_i$), it leads to potentiation (positive ϕ). In transformers, this allows adaptive attention that strengthens associations for surprising tokens while weakening predictable ones.

5.3.4 Implications for Association Rules

Test-Time Adaptation: The "unlearning" mechanism of the delta rule explains the effectiveness of Test-Time Training (TTT) [35], where a model adapts to a new data distribution at inference time by making online updates based on prediction errors.

Gradient Stability in Deep Networks: Oja’s rule provides a solution to gradient instability. By incorporating homeostatic normalization, it could overcome the need for some engineering tricks such as careful initialization, learning rate warmup, and gradient clipping.

Adaptive Attention via BCM: The BCM rule suggests a principled mechanism for implementing adaptive attention. An attention head governed by a BCM-like rule could learn to dynamically focus its capacity on the most informative tokens in a sequence, effectively ignoring redundant information and concentrating on "surprising" content without needing to be explicitly trained for that behavior.

6 Discussion

6.1 Why Attention Works: A Perspective from Classical Conditioning

Our work began with a simple question: *why* is the attention mechanism so effective? This paper provides a new perspective: the success of attention, particularly in its linearized form, can be understood as a direct consequence of implementing one of biology’s most efficient learning algorithms—associative learning through Pavlovian conditioning.

In our framework, which finds a direct mathematical analogue in linear attention, the primary function of an attention head is not merely to "attend" but to *associate* and *generalize*. By mapping queries, keys, and values to test stimuli, CS, and US, we reveal that each forward pass can be seen as an act of inference-time learning. The attention head behaves as a dynamic associative memory, transiently wired by the Hebbian rule. This perspective elevates attention from a simple weighting mechanism to a fundamental computational primitive for learning.

6.2 Different Views of Attention Head: KV Circuit vs QK Circuit

We compare two distinct, yet complementary, ways of analyzing attention heads, which is critical for a complete understanding of how transformers function.

Let us call the core component of our model, the dynamically formed associative memory $\mathbf{S}_i = \alpha \sum_{j=1}^i f(\mathbf{k}_j)^\top g(\mathbf{v}_j)$, a "KV circuit." By summing the outer products of key and value vectors over the temporal (sequence) dimension, we calculate a correlation matrix between features of the hidden

layers. This matrix S represents a learned mapping: which features in the US (value) pathway are associated with which features in the CS (key) pathway. Stacking these KV circuits layer upon layer reveals complex, context-dependent information flow graphs (Figure 2).

Another view focuses on the "QK circuit," a mechanism best understood by analyzing how queries and keys interact to form attention patterns. Induction heads [20] exemplify such circuits, where the computational mechanism lies precisely in *how* attention patterns are dynamically constructed. The QK matrix computes similarities between the current token's query vector and all previous tokens' key vectors, determining the temporal pattern of information retrieval rather than the content being retrieved [9].

Our framework reveals that KV circuits and QK circuits represent two essential aspects of attention: the KV circuit provides a feature-centric view, which is the content of the memory; the QK circuit is a temporal-centric view, showing the addressing mechanism for the memory. An attention head performs both computations simultaneously, and a complete understanding requires appreciating this duality. In essence, the QK circuit acts as the addressing mechanism for the memory, determining which past information to retrieve, while the KV circuit constitutes the content of the memory itself, defining what information is stored in the association.

6.3 Transformer as Dynamic Reasoning Engine

This conditioning lens provides a mechanistic basis for interpreting the transformer's capability of in-context learning [4]. From this viewpoint, it is the expected outcome of forming temporary CS-US associations from examples in the prompt. The model "learns" not by updating its permanent weights, but by building a transient associative matrix that maps new patterns to specified outcomes.

Furthermore, the stacking of conditioning circuits into deep networks (Section 5.1) explains the transformer's capacity for complex, compositional reasoning. We have shown how higher-order associations are built layer by layer, allowing the network to construct internal inferential chains (e.g., $A \rightarrow B$, $B \rightarrow C$, therefore $A \rightarrow C$). This provides a concrete mechanism for the multi-step, internal reasoning observed in large models [15], viewing it as a cascade of second-order and higher-order conditioning events. The "dynamic routing" that emerges from this process, where different inputs activate different associative pathways, mirrors the flexible, context-dependent processing observed in the brain. The transformer, from this viewpoint, is not merely processing sequences; it is dynamically constructing and traversing a graph of learned relationships.

6.4 Reliability, Fragility, and the Limits of Association

Our framework, grounded in the analysis of the linear attention model, also illuminates inherent limitations. The memory capacity analysis (Theorem 6) and the error propagation model (Theorem 9) provide a principled explanation for the fragility observed in large models. The associative memory S is finite and prone to interference; as context length grows, older associations can be drowned out by newer ones.

This reveals a fundamental tension. While stacking layers enables higher-order reasoning, it also creates a longer chain for errors to compound. Our error scaling law ($r^* \propto \frac{L \cdot n^H}{d_k^H}$), derived from the conditioning model, suggests that models may underperform at simple tasks because they engage unnecessarily deep, and therefore noisy, associative pathways [26]. This trade-off between expressive power and reliability appears to be a fundamental constraint.

6.5 Bridge towards Unified Theory of Intelligence

Perhaps the most significant implication of this work is the bridge it builds between AI and neuroscience, suggesting the transformer's success stems not from engineering brilliance alone, but from inadvertently rediscovering computational principles that biology has refined over millions of years [33]. From this point of view, specific architectural choices have biological correspondences. RetNet's temporal decay [28] can be reframed as a biological forgetting curve designed to manage memory interference. Likewise, Hebbian variants like the Delta and Oja's rules are not merely neuroscientific models, but principled solutions to engineering challenges like error correction and gradient

stability. This opens a new frontier for architectural design, moving from brute-force discovery to the principled implementation of biological mechanisms.

Ultimately, our framework suggests that the convergence between artificial and biological computation is not coincidental. The foundational principles of learning, memory, association, and generalization appear to be universal pillars of intelligence, whether instantiated in silicon or carbon. By understanding these principles, we can build AI systems that are not just powerful, but interpretable, efficient, and aligned with the fundamental nature of intelligence itself.

6.6 Limitations and Future Directions

While our conditioning framework provides a powerful explanatory lens, its limitations define the boundaries of our current understanding and highlight critical avenues for future research.

Gap between Linear and Softmax Attention: Our framework finds its most direct mathematical parallel in linear attention. This is the paper’s most significant limitation. Standard transformers, which rely on softmax attention, often achieve superior performance. The exponential nature of the softmax function enforces a much stronger form of competitive selection than the simple additive accumulation in our base model—a "soft winner-take-all" dynamic. This suggests that standard attention may implement a more advanced cognitive process than simple Pavlovian generalization. This competition might correspond to mechanisms like lateral inhibition in brain circuits, where strong activation of one neural representation suppresses its competitors [18, 13]. Understanding the computational necessity of this competitive normalization remains a key open question.

Architectural Simplifications: Our analysis focuses on attention-only architectures to isolate the associative mechanism. In practice, MLP blocks are critical to transformer performance. We hypothesize two potential roles for them within our conditioning framework: (1) they may function as rich, non-linear feature extractors, transforming the raw inputs into more potent CS and US representations for the attention heads to associate; or (2) they may perform essential post-retrieval processing, transforming the retrieved US information into the final output format required by the task. Furthermore, our framework describes the "fast" inference-time learning that occurs during a forward pass. It does not yet address how this interacts with the "slow" learning of the projection matrices (\mathbf{W}_Q , \mathbf{W}_K , \mathbf{W}_V) via backpropagation. Understanding the interplay between these two timescales—how slow gradient-based learning shapes the parameters that enable fast context-dependent association—is a critical area for future research.

Biological Simplifications: Our model is a computational abstraction, not a detailed biological replica. It simplifies the immense complexity of real neural circuits. Biological learning involves a richer tapestry of mechanisms, including dendritic computations [27], diverse forms of synaptic plasticity [1], and the global effects of neuromodulation [16, 8]. The value of our framework lies in extracting a core principle to explain an artificial system, not in claiming to perfectly mirror its biological counterpart.

7 Conclusion

This work reconceptualizes the core associative mechanism of transformer attention through the lens of Pavlovian conditioning, finding a direct mathematical realization in linear attention. We suggest that the success of these architectures may stem from embodying one of biology’s most fundamental learning principles. By mapping queries, keys, and values to test stimuli, CS, and US, we show how these models can be interpreted as implementing dynamic, inference-time associative learning with generalization.

Our theoretical framework, grounded in this linearized model, yields several critical insights. First, it provides a mechanistic lens for interpreting transformer capabilities like in-context learning and compositional reasoning. Second, our analyses of memory capacity and error propagation reveal fundamental architectural trade-offs of balancing model depth, width, and head redundancy to maintain reliability. Finally, we demonstrate how biologically inspired variants of the Hebbian rule, such as the Delta and Oja’s rules, offer principled solutions to contemporary engineering challenges.

The implications extend beyond technical contributions. By revealing deep mathematical connections between a class of attention mechanisms and classical conditioning, we suggest that intelligence,

whether biological or artificial, may be governed by shared computational principles. This perspective offers a bridge between neuroscience and AI, paving the way for building more capable, interpretable, and robust systems [33].

References

- [1] Wickliffe C. Abraham. Metaplasticity: Tuning synapses and networks for plasticity. *Nat Rev Neurosci*, 9(5):387–387, May 2008.
- [2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization, July 2016.
- [3] E. L. Bienenstock, L. N. Cooper, and P. W. Munro. Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex. *J Neurosci*, 2(1):32–48, January 1982.
- [4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners, July 2020. Comment: 40+32 pages.
- [5] Matteo Carandini and David J. Heeger. Normalization as a canonical neural computation. *Nat Rev Neurosci*, 13(1):51–62, January 2012.
- [6] Krzysztof Marcin Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking Attention with Performers. In *International Conference on Learning Representations*, October 2020.
- [7] Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models, January 2024.
- [8] Peter Dayan. Twenty-Five Lessons from Computational Neuromodulation. *Neuron*, 76(1):240–256, October 2012.
- [9] Elhage et al. A Mathematical Framework for Transformer Circuits. *Transformer Circuits Thread*, 2021.
- [10] Wulfram Gerstner and Werner M. Kistler. Mathematical formulations of Hebbian learning. *Biol Cybern*, 87(5):404–415, December 2002.
- [11] STEFANO Ghirlanda. Intensity Generalization: Physiology and Modelling of a Neglected Topic. *Journal of Theoretical Biology*, 214(3):389–404, February 2002.
- [12] D. O. Hebb. *The Organization of Behavior; a Neuropsychological Theory*. The Organization of Behavior; a Neuropsychological Theory. Wiley, Oxford, England, 1949.
- [13] Jeffry S. Isaacson and Massimo Scanziani. How inhibition shapes cortical activity. *Neuron*, 72(2):231–243, October 2011.
- [14] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention, August 2020. Comment: ICML 2020, project at <https://linear-transformers.com/>.
- [15] Authors Jack Lindsey†, Wes Gurnee*, Emmanuel Ameisen*, Brian Chen*, Adam Pearce*, Nicholas L. Turner*, Craig Citro*, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermy, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson*‡. Affiliations Anthropic Published March 27. On the Biology of a Large Language Model. <https://transformer-circuits.pub/2025/attribution-graphs/biology.html>.

- [16] Eve Marder. Neuromodulation of Neuronal Circuits: Back to the Future. *Neuron*, 76(1):1–11, October 2012.
- [17] S. Maren. Neurobiology of Pavlovian fear conditioning. *Annu Rev Neurosci*, 24:897–931, 2001.
- [18] Shreesh P. Mysore and Eric I. Knudsen. The role of a midbrain network in competitive stimulus selection. *Curr Opin Neurobiol*, 21(4):653–660, August 2011.
- [19] E. Oja. A simplified neuron model as a principal component analyzer. *J Math Biol*, 15(3):267–273, 1982.
- [20] Olsson et al. In-context Learning and Induction Heads. *Transformer Circuits Thread*, 2022.
- [21] P Ivan Pavlov (1927). Conditioned reflexes: An investigation of the physiological activity of the cerebral cortex. *Ann Neurosci*, 17(3):136–141, July 2010.
- [22] Zhen Qin, XiaoDong Han, Weixuan Sun, Dongxu Li, Lingpeng Kong, Nick Barnes, and Yiran Zhong. The Devil in Linear Transformer, October 2022. Comment: accepted to EMNLP2022.
- [23] Hemanth Saratchandran, Damien Teney, and Simon Lucey. Leaner Transformers: More Heads, Less Depth, May 2025.
- [24] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear Transformers Are Secretly Fast Weight Programmers, June 2021.
- [25] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer, January 2017.
- [26] Parshin Shojaei, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The Illusion of Thinking: Understanding the Strengths and Limitations of Reasoning Models via the Lens of Problem Complexity.
- [27] Edited by Greg Stuart, Nelson Spruston, and Michael Hausser, editors. *Dendrites*. Oxford University Press, Oxford, New York, third edition, third edition edition, June 2016.
- [28] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive Network: A Successor to Transformer for Large Language Models, August 2023.
- [29] Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer Dissection: A Unified Understanding of Transformer’s Attention via the Lens of Kernel, November 2019. Comment: EMNLP 2019.
- [30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2017.
- [31] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent Abilities of Large Language Models, October 2022. Comment: Transactions on Machine Learning Research (TMLR), 2022.
- [32] Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing Linear Transformers with the Delta Rule over Sequence Length, 2024. Comment: Final camera ready.
- [33] Anthony Zador, Sean Escola, Blake Richards, Bence Ölveczky, Yoshua Bengio, Kwabena Boahen, Matthew Botvinick, Dmitri Chklovskii, Anne Churchland, Claudia Clopath, James Di-Carlo, Surya Ganguli, Jeff Hawkins, Konrad Körding, Alexei Koulakov, Yann LeCun, Timothy Lillicrap, Adam Marblestone, Bruno Olshausen, Alexandre Pouget, Cristina Savin, Terrence Sejnowski, Eero Simoncelli, Sara Solla, David Sussillo, Andreas S. Tolias, and Doris Tsao. Catalyzing next-generation Artificial Intelligence through NeuroAI. *Nat Commun*, 14(1):1597, March 2023.

- [34] Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization, October 2019. Comment: NeurIPS 2019.
- [35] Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T. Freeman, and Hao Tan. Test-Time Training Done Right, May 2025. Comment: 32 pages, 11 figures.

A Detailed Mathematical Proofs

A.1 Proof of Memory Capacity Theorem 6

Proof. Consider the associative memory formed by n CS-US pairs:

$$\mathbf{S} = \alpha \sum_{j=1}^n f(\mathbf{k}_j)^\top g(\mathbf{v}_j) \quad (27)$$

We analyze the retrieval process from the associative memory matrix \mathbf{S} . For simplicity and without loss of generality, we assume $\alpha = 1$ and the activation functions f and g are identity mappings. The memory matrix \mathbf{S} is formed by the sum of outer products of n key-value pairs:

$$\mathbf{S} = \sum_{j=1}^n \mathbf{k}_j^\top \mathbf{v}_j \quad (28)$$

where each key $\mathbf{k}_j \in \mathbb{R}^{1 \times d_k}$ and each value $\mathbf{v}_j \in \mathbb{R}^{1 \times d_v}$ are row vectors. This results in \mathbf{S} being a $d_k \times d_v$ matrix.

A.1.1 Signal and Noise Power

For a query $\mathbf{q} = \mathbf{k}_m$ aimed at retrieving the value \mathbf{v}_m . The retrieved output vector $\mathbf{r} \in \mathbb{R}^{1 \times d_v}$ is given by:

$$\mathbf{r} = \mathbf{q}\mathbf{S} = \sum_{j=1}^n (\mathbf{k}_m \mathbf{k}_j^\top) \mathbf{v}_j = \underbrace{(\mathbf{k}_m \mathbf{k}_m^\top) \mathbf{v}_m}_{\text{Signal}} + \underbrace{\sum_{j \neq m} (\mathbf{k}_m \mathbf{k}_j^\top) \mathbf{v}_j}_{\text{Noise/Interference}} \quad (29)$$

We model keys $\mathbf{k}_j \in \mathbb{R}^{d_k}$ and values $\mathbf{v}_j \in \mathbb{R}^{d_v}$ as random vectors drawn uniformly from the surface of the unit hypersphere in their respective dimensions. Thus, $\|\mathbf{k}_j\| = 1$ and $\|\mathbf{v}_j\| = 1$ for all j .

The **Signal** is the term for $j = m$: $\mathbf{r}_{\text{signal}} = (\mathbf{k}_m \mathbf{k}_m^\top) \mathbf{v}_m = \|\mathbf{k}_m\|^2 \mathbf{v}_m = \mathbf{v}_m$. The **Signal Power** is the squared magnitude of this vector:

$$P_S = E[\|\mathbf{r}_{\text{signal}}\|^2] = \|\mathbf{r}_{\text{signal}}\|^2 = 1 \quad (30)$$

With unit normalization, the signal power is constant and does not scale with dimension.

The **Noise** is the sum over all other terms where $j \neq m$: $\mathbf{r}_{\text{noise}} = \sum_{j \neq m} (\mathbf{k}_m \mathbf{k}_j^\top) \mathbf{v}_j$. The **Noise Power** is the expected squared magnitude of the noise vector. Let $c_j = \mathbf{k}_m \mathbf{k}_j^\top$. For two random unit vectors in high dimensions, their dot product c_j is approximately distributed as $\mathcal{N}(0, 1/d_k)$. Therefore, $\mathbb{E}[c_j] = 0$ and $\mathbb{E}[c_j^2] = \text{Var}(c_j) \approx 1/d_k$.

$$P_N = \mathbb{E}[\|\mathbf{r}_{\text{noise}}\|^2] = \mathbb{E}\left[\left\|\sum_{j \neq m} c_j \mathbf{v}_j\right\|^2\right] \quad (31)$$

$$= \sum_{j \neq m} \mathbb{E}[c_j^2] \mathbb{E}[\|\mathbf{v}_j\|^2] \quad (\text{due to orthogonality of cross-terms}) \quad (32)$$

$$\approx \sum_{j \neq m} \left(\frac{1}{d_k}\right) (1) = \frac{n-1}{d_k} \quad (33)$$

The noise power is inversely proportional to the key dimension d_k .

A.1.2 Average Case SNR Analysis

We ask the signal-to-noise ratio (SNR) to be above a certain threshold γ :

$$\text{SNR} = \frac{P_S}{P_N} = \frac{d_k}{n-1} > \gamma \quad (34)$$

This suggests a linear relationship between n and d_k :

$$n < 1 + \frac{d_k}{\gamma} = O(d_k) \quad (35)$$

A.1.3 Worst Case Concentration Inequalities and Union Bound

To make the argument rigorous, we must ensure that the retrieval works for any of the n items with high probability. We first define the error condition for a single retrieval as $\|\mathbf{r}_{\text{noise}}\|^2 \geq \delta \|\mathbf{r}_{\text{signal}}\|^2 = \delta$, where δ is a predefined threshold.

With this condition, we can bound the probability of a single retrieval failure $P(F_m)$ using Markov's inequality:

$$P(F_m) = P(\|\mathbf{r}_{\text{noise}}\|^2 \geq \delta) \leq \frac{E[\|\mathbf{r}_{\text{noise}}\|^2]}{\delta} = \frac{n-1}{\delta d_k} \quad (36)$$

We are interested in the probability that any of the n retrievals fails. This is the probability of the union of all error events, and the union bound states:

$$P(\text{any failure}) = P(\cup_{m=1}^n F_m) \leq \sum_{m=1}^n P(F_m) = \frac{n(n-1)}{\delta d_k} \quad (37)$$

For the memory to be considered reliable, the total probability of error must be small. Let's say we want this probability to be less than some small constant ϵ :

$$\frac{n(n-1)}{\delta d_k} < \epsilon \quad (38)$$

For large n , this is approximately $\frac{n^2}{\delta d_k} < \epsilon$, which implies:

$$n < \sqrt{\epsilon \delta d_k} = O(\sqrt{d_k}) \quad (39)$$

This demonstrates that the number of associations n must scale as less than the square root of the key dimension d_k to ensure that all memories can be retrieved with high fidelity. \square

A.2 Proof of Error Propagation Theorem 9

Proof. Consider a depth- L network where each layer ℓ has H attention heads. We analyze the probability of successful retrieval through all layers.

A.2.1 Single Head Failure Probability

From the analysis in Theorem 6, we have that for a single retrieval:

$$P(\text{head } h \text{ fails}) < \frac{n}{\delta d_k} \quad (40)$$

where δ is the threshold for successful retrieval. This bound is meaningful when $n/(\delta d_k) < 1$, which is the regime where reliable retrieval is possible.

A.2.2 Layer Success with Multiple Heads

For a layer with H heads to fail, all heads must fail to retrieve the correct association. Assuming the weight matrices for each head are initialized independently, we can treat their failures as approximately independent events. This gives us an upper bound:

$$P(\text{layer fails}) = \prod_{h=1}^H P(\text{head } h \text{ fails}) \quad (41)$$

$$< \left(\frac{n}{\delta d_k} \right)^H \quad (42)$$

Therefore, the probability that the layer succeeds is bounded below by:

$$P(\text{layer succeeds}) > 1 - \left(\frac{n}{\delta d_k} \right)^H \quad (43)$$

A.2.3 Multi-Layer Success Probability

For a successful reasoning chain through all L layers, each layer must successfully retrieve its associations. Assuming layer-wise independence of successes:

$$P(\text{complete success}) = \prod_{\ell=1}^L P(\text{layer succeeds}) \quad (44)$$

$$> \left[1 - \left(\frac{n}{\delta d_k} \right)^H \right]^L \quad (45)$$

A.2.4 Error Rate Upper Bound

The overall error rate r for the entire deep network is bounded by:

$$r = 1 - P(\text{complete success}) < 1 - \left[1 - \left(\frac{n}{\delta d_k} \right)^H \right]^L \quad (46)$$

For cases where the single-layer failure probability is small, we can use the Taylor expansion $(1 - x)^L \approx 1 - Lx$ for small x . Let $x = (n/(\delta d_k))^H$:

$$r < 1 - \left(1 - L \left(\frac{n}{\delta d_k} \right)^H \right) \quad (47)$$

$$\lesssim L \left(\frac{n}{\delta d_k} \right)^H \quad (48)$$

$$= \frac{L \cdot n^H}{\delta^H d_k^H} \quad (49)$$

Since δ is a fixed threshold parameter, the error rate upper bound r^* scales as:

$$r^* \propto \frac{L \cdot n^H}{d_k^H} \quad (50)$$

This upper bound on the error rate reveals the scaling behavior: it grows linearly with depth L and polynomially with the ratio n/d_k to the power of H . The approximation holds when $(n/\delta d_k)^H \ll 1/L$, which requires:

$$n \ll \delta d_k \cdot L^{-1/H} \quad (51)$$

This confirms the architectural trade-offs discussed in the main text. \square