

Test-time regression: a unifying framework for designing sequence models with associative memory

Ke Alexander Wang

*Department of Computer Science
Stanford University*

ALXWANG@CS.STANFORD.EDU

Jiaxin Shi

*Department of Statistics
Stanford University*

JIAXINS@STANFORD.EDU

Emily B. Fox

*Department of Statistics
Department of Computer Science
Stanford University*

EBFOX@STANFORD.EDU

Editor: Kilian Weinberger

Abstract

Sequence models lie at the heart of modern deep learning. However, rapid advancements have produced a diversity of seemingly unrelated architectures, such as Transformers and recurrent alternatives. In this paper, we introduce a unifying framework to understand and derive these sequence models, inspired by the empirical importance of associative recall, the capability to retrieve contextually relevant tokens. We formalize associative recall as a two-step process, memorization and retrieval, casting memorization as a regression problem. Layers that combine these two steps perform associative recall via “test-time regression” over its input tokens. Prominent layers, including linear attention, state-space models, fast-weight programmers, online learners, and softmax attention, arise as special cases defined by three design choices: the regression weights, the regressor function class, and the test-time optimization algorithm. Our approach clarifies how linear attention fails to capture inter-token correlations and offers a mathematical justification for the empirical effectiveness of query-key normalization in softmax attention. Further, it illuminates unexplored regions within the design space, which we use to derive novel higher-order generalizations of softmax attention. Beyond unification, our work bridges sequence modeling with classic regression methods, a field with extensive literature, paving the way for developing more powerful and theoretically principled architectures.

Keywords: neural networks, sequence modeling, test-time, regression, associative memory

1. Introduction

Sequences play a vital role in modern machine learning by providing a powerful abstraction: any computational task can be viewed as transforming one sequence into another (Sutskever et al., 2014). This sequential perspective has spread across diverse domains, including

natural language processing (Sutskever et al., 2014; Devlin et al., 2019; Brown et al., 2020), computer vision (Dosovitskiy et al., 2021; Bertasius et al., 2021), time series analysis (Salinas et al., 2020; Gruver et al., 2023; Ansari et al., 2024), and computational biology (Jumper et al., 2021; Zhou and Troyanskaya, 2015; Nguyen et al., 2024), highlighting the importance of building generically applicable sequence layers (Vaswani et al., 2017).

This development has produced a diversity of architectures, each with its own unique characteristics and performance trade-offs. While these architectures have achieved considerable success, they have largely emerged through separate lines of investigation. Such a fragmented and often empirically-driven approach to model development limits our ability to systematically understand and improve design choices. Moreover, the idiosyncratic notations of each architecture obscures their underlying connections (Rush, 2024). Given the wide variety of sequence models, a natural question to ask is whether there is an underlying principle that explains why some sequence models work better than others.

One empirical discovery that ties together disparate architectures is the strong correlation between an architecture’s associative recall ability and its language modeling performance (Olsson et al., 2022; Arora et al., 2023a). Associative recall is the act of retrieving contextually relevant information based on an association with a query. Arora et al. (2023a) gives the following example of how associative memory and retrieval can improve language modeling. Consider the example sentence from Arora et al. (2023b): “Hakuna Matata, it means no worries for the rest of your days. Hakuna ____”. To predict the next word, we can first memorize the previous occurrence of “Hakuna” and its associated value “Matata”; the next time we encounter “Hakuna” again we can retrieve its associated value from memory as our prediction. Notice that we can perform this task using only in-context information and we can make an accurate prediction even if we have never encountered this strange phrase before. Indeed, Transformer-based language models have been discovered to exhibit this kind of behavior via “induction heads” that emerge during training (Olsson et al., 2022).

Given the empirical importance of associative recall, **how can we systematically design neural network layers that can perform associative recall (AR)?** In this paper, we introduce a simple but principled framework for deriving sequence layers *designed to perform associative recall*, which we call “test-time regression layers”. From the “Hakuna Matata” example, we see that **associative recall has two-steps: memorization and retrieval**. Our crucial observation is that we can implement the memorization step by solving a weighted regression problem. We can then generate an output by applying the regressor to a cue/query token, retrieving the most relevant token from our associative memory. *Combining both memorization and retrieval into the forward pass of a sequence layer* results in a layer that performs regression over the input tokens, a procedure that we call “test-time regression”, visualized in Figure 1. Our terminology reflects that the regressor is regenerated with each forward pass and depends only on the *input tokens* rather than a fixed training dataset.

Since any regression method can be used, our framework provides a general recipe to derive a large class of sequence layers. **Under our framework, an AR-based sequence layer is a mathematical consequence of choosing (1) the regression weights, (2) the regressor parameterization, and (3) the optimization algorithm for finding the regressor.** In this paper, we consider a few choices of these three “ingredients”,

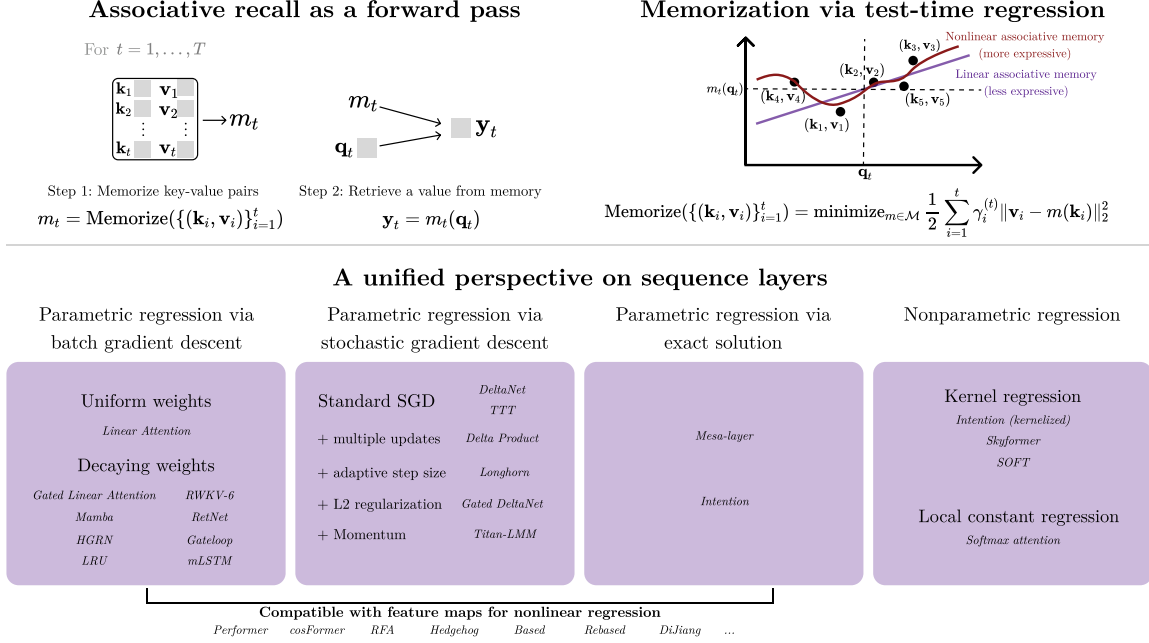


Figure 1: Our framework provides a systematic way to derive sequence models that can perform associative recall, following a two step process of memorization and retrieval. The memorization step can be formalized as a solving a regression problem at test-time. Our perspective results in a “recipe” for deriving sequence layers by making three choices: the importance of each association $\{\gamma_i^{(t)}\}_{i=1}^t$, the regressor function class $\mathcal{M}$, and the optimization algorithm. Parametric regression layers tend to have an efficient recurrence for updating the memory m_t , while non-parametric layers like softmax-attention do not. For simplicity, we discuss causal sequence models where prefix key-value pairs are memorized, but the same principles also apply to non-causal ones (e.g. generically masked attention).

and show how to *derive* many recently proposed classes of sequence layers, illustrating the generality of our framework. Our derivations reveal that linear attention (Katharopoulos et al., 2020), its feature-mapped variants (e.g. (Peng et al., 2020; Qin et al., 2021; Kasai et al., 2021; Zhang et al., 2023a; Aksenov et al., 2024; Chen et al., 2024)), its gated variants (Sun et al., 2023; Orvieto et al., 2023; Katsch, 2024; De et al., 2024; Qin et al., 2024; Peng et al., 2024; Yang et al., 2024b; Beck et al., 2024), state-space model layers (Gu and Dao, 2024; Dao and Gu, 2024), fast-weights layers (Schlag et al., 2021; Yang et al., 2024a,a), online learning layers (Liu et al., 2024; Sun et al., 2024; Yang et al., 2024a; Behrouz et al., 2024), and softmax self-attention (Vaswani et al., 2017) *are all test-time regression layers*, implicitly performing memorization followed by retrieval in their forward passes, despite being developed from different perspectives. Figure 1 previews how existing architectures are instantiations of test-time regression layers within our framework.

Table 1: Test-time regression layers. These layers memorize key-value pairs by minimizing a weighted regression loss $m_t = \operatorname{argmin}_{m \in \mathcal{M}} \sum_{i=1}^t \gamma_i^{(t)} \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2$ as part of their forward pass to memorize $\{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^t$. They compute the output by querying the memory m_t to get $m_t(\mathbf{q}_t)$.

Sequence layer	Optimizer in the forward pass	Regression weights	Function class $\mathcal{M}$
Linear attention (Katharopoulos et al., 2020)	1 step of batch GD	Uniform $\gamma_i^{(t)} = 1$	Linear
Mamba-2 (Dao and Gu, 2024)	1 step of batch GD	Decaying $\gamma_i^{(t)} = \prod_{j=i+1}^t \gamma_j$	Linear
Mesa-layer (von Oswald et al., 2023)	1 step of Newton’s method/analytic solution	Uniform or decaying	Linear
DeltaNet (Schlag et al., 2021)	t steps of SGD	Uniform $\gamma_i^{(t)} = 1$	Linear
Longhorn (Liu et al., 2024)	t steps of SGD w/ adaptive step size	Uniform $\gamma_i^{(t)} = 1$	Linear
GatedDeltaNet (Yang et al., 2024a)	t steps of SGD w/ L2 reg.	Uniform $\gamma_i^{(t)} = 1$	Linear
Titans (Behrouz et al., 2024)	t steps of SGD w/ momentum	Uniform $\gamma_i^{(t)} = 1$	Linear
TTT-MLP (Sun et al., 2024)	t steps of mini-batch SGD	Uniform $\gamma_i^{(t)} = 1$	Nonlinear MLP
Attention (Vaswani et al., 2017)	Analytic solution	Similarity-weighted $\gamma_i^{(t)} = \frac{\exp(\mathbf{k}_i^\top \mathbf{q}_t)}{\sum_j \exp(\mathbf{k}_j^\top \mathbf{q}_t)}$	Non-parametric regressor

Our derivations also lead to new understandings of existing sequence layers. We show that layers based on linear attention underperform because they fail to account for the correlation between tokens. We also show that query-key normalization, an important technique in stabilizing the training of large language models (Dehghani et al., 2023; Wortsman et al., 2023), is mathematically necessary to ensure that softmax self-attention is a proper local constant regressor. Finally, we propose a higher order generalization of softmax attention, motivated by local linear regression.

Outline of our paper. We start in Section 2 by examining a few of the most prominent classes of sequence layers. Although each class of layers was developed from distinct motivations, their similar computation pattern hints at a common unifying theme that underlies their effectiveness. We then introduce our test-time regression framework in Section 3 to formalize the connection between associative memorization and regression. Our perspective provides a systematic approach to designing sequence layers that produce their outputs via associative recall. In Section 4, we show that indeed all of the aforementioned classes of sequence layers can be understood from a *single unified perspective* using the principles of test-time regression and associative recall. We demonstrate the generality of our framework by deriving these layers simply by varying how we minimize the regression objective. In Section 5, we examine how to construct effective key-value pairs for associative recall in next-token prediction tasks. Section 6.1 empirically validates the framework, demonstrating that sequence layers implicitly perform regression in their forward pass, and that higher-order attention is more robust to suboptimal attention scalings in language modeling. We discuss prior work related to associative recall and memory in Section 7. We finish in Section 8 by discussing the broader implications of viewing sequence models through the lens of regression, including future directions for new architectures.

2. Background on existing sequence modeling layers

The goal of sequence modeling is to transform a sequence of input tokens $\mathbf{x}_1, \dots, \mathbf{x}_T$ into output tokens $\mathbf{y}_1, \dots, \mathbf{y}_T$ (Sutskever et al., 2014). A common design pattern, first

proposed by Vaswani et al. (2017) is to transform the input tokens into key-value pairs $(\mathbf{k}_1, \mathbf{v}_1), \dots, (\mathbf{k}_T, \mathbf{v}_T) \in \mathbb{R}^{D_k} \times \mathbb{R}^{D_v}$ and queries $\mathbf{q}_1, \dots, \mathbf{q}_T \in \mathbb{R}^{D_k}$ before applying operations that “mix” the sequences together to produce the output tokens. Typically, each $\mathbf{q}_t, \mathbf{k}_t$, and $\mathbf{v}_t$ is a linear transform of the corresponding input token $\mathbf{x}_t$, which can be thought of as different “views” of $\mathbf{x}_t$. Most subsequent sequence layers have kept this pattern but propose other ways of “mixing” queries, keys, and values. We begin by reviewing a few classes of existing sequence layers that follow this pattern before unifying them under our test-time regression framework.

Softmax attention. Given input tokens $\mathbf{x}_1, \dots, \mathbf{x}_T$, and their query, key, value vectors, softmax attention (Vaswani et al., 2017) computes each output token by

$$\mathbf{y}_t = \sum_{i=1}^t \mathbf{v}_i \frac{s(\mathbf{k}_i, \mathbf{q}_t)}{\sum_{j=1}^t s(\mathbf{k}_j, \mathbf{q}_t)} \quad (1)$$

for $t \in [T]$ where $s(\mathbf{k}, \mathbf{q}) = \exp(\mathbf{k}^\top \mathbf{q} / \sqrt{D_k})$. Each output $\mathbf{y}_t$ is a weighted sum of values $\{\mathbf{v}_i\}_{i=1}^t$, weighted by the similarity between the query $\mathbf{q}_t$ and the keys $\{\mathbf{k}_i\}_{i=1}^t$ associated with the values. This mechanism can be seen as a continuous-valued lookup table, using $\mathbf{q}_t$ to lookup an interpolated value for $\mathbf{y}_t$. Though highly effective, self-attention requires $\mathcal{O}(T^2)$ time to autoregressively compute the entire length T sequence even after caching the keys and values, which can be prohibitively slow for generating long sequences.

Linear attention. To address the computational challenges of self-attention, there has been a wealth of methods that seek to develop more efficient alternatives; see the survey by Tay et al. (2022). One particular class of more efficient sequence models has its origins in a recurrent variant of attention, known as linear attention (Katharopoulos et al., 2020). To achieve this efficiency, linear attention uses the standard inner product $\mathbf{k}^\top \mathbf{q}$, rather than using the exponentiated inner product $s(\mathbf{k}_i, \mathbf{q}_t)$, producing outputs of the form $\mathbf{y}_t \propto \sum_{i=1}^t \mathbf{v}_i \mathbf{k}_i^\top \mathbf{q}_t$ up to a normalization. This change allows all T outputs to be recurrently computed in $\mathcal{O}(T)$ time, an asymptotic speedup, by updating a $D_v \times D_k$ “matrix-valued state” $\mathbf{M}_t = \sum_{i=1}^t \mathbf{v}_i \mathbf{k}_i^\top$, following the recurrence

$$\mathbf{M}_t = \mathbf{M}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top, \quad \mathbf{y}_t = \mathbf{M}_t \mathbf{q}_t = \sum_{i=1}^t \mathbf{v}_i \mathbf{k}_i^\top \mathbf{q}_t. \quad (2)$$

Each rank-1 outer product $\mathbf{v}_t \mathbf{k}_t^\top$ can be thought of as writing a new key-value pair into memory. Although Katharopoulos et al. (2020)’s original formulation of linear attention included a normalization like softmax attention, we drop the normalization here since later works found that this normalization method causes training instabilities (Qin et al., 2022; Dao and Gu, 2024).

To increase expressiveness, Katharopoulos et al. (2020) also proposed a variant of linear attention, inspired by the kernel trick (Schölkopf and Smola, 2001), that replaces the inner product $\mathbf{k}^\top \mathbf{q}$ with an inner product in a feature space $\phi(\mathbf{k})^\top \phi(\mathbf{q})$ where ϕ is a feature map $\phi : D_k \rightarrow D_\phi$. This modification results in the recurrence

$$\mathbf{M}_t = \mathbf{M}_{t-1} + \mathbf{v}_t \phi(\mathbf{k}_t)^\top, \quad \mathbf{y}_t = \mathbf{M}_t \phi(\mathbf{q}_t) = \sum_{i=1}^t \mathbf{v}_i \phi(\mathbf{k}_i)^\top \phi(\mathbf{q}_t). \quad (3)$$

Katharopoulos et al. (2020) used $\phi = 1 + \text{ELU}$, but numerous alternative choices of feature maps have since been proposed, including the cosine function (Qin et al., 2021), deterministic parameter-free projections (Schlag et al., 2021), polynomial maps (Zhang et al., 2023a; Aksenov et al., 2024; Arora et al., 2024), random features (Peng et al., 2020; Choromanski et al., 2020; Chen et al., 2024), and more.

State-space models and linear attention with forgetting. Since the recurrent state $\mathbf{M}_t$ of linear attention has finite size ($D_v \times D_k$ numbers), $\mathbf{M}_t$ can only store a bounded amount of information. In Equation 2, $\mathbf{M}_t$ weighs all key value pairs equally across timesteps, trying to compress all key-value pairs equally well into its finite-sized state. As a result, the performance of linear attention can degrade as the sequence length increases. To address this limitation, various works have proposed adding a forgetting factor γ_t to allow $\mathbf{M}_t$ to retain only the most recent information, decaying out key-value pairs over time. This produces the recurrence

$$\mathbf{M}_t = \gamma_t \mathbf{M}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top, \quad \mathbf{y}_t = \mathbf{M}_t \mathbf{q}_t. \quad (4)$$

At each step t , all previous values in $\mathbf{M}_{t-1}$ are attenuated by a factor of $\gamma_t \in [0, 1]$ before the new value is stored. When γ_t depends on $\mathbf{x}_t$, this forgetting factor can be seen as a forgetting gate, akin to that of an LSTM (Hochreiter and Schmidhuber, 1997). This class of linear attention with forgetting includes RetNet (Sun et al., 2023), gated linear attention (Yang et al., 2024b), GateLoop (Katsch, 2024), mLSTM (Beck et al., 2024), LRU (Orvieto et al., 2023), HGRN (Qin et al., 2024), and RWKV-6 (Peng et al., 2024). See Table 1 of Yang et al. (2024b) for how each model implements γ_t .

The forgetting factor in linear attention is also equivalent to another class of sequence layers known as state-space models (SSMs), which includes Mamba (Gu and Dao, 2024) and Mamba-2 (Dao and Gu, 2024). State-space models originated from a line of work on optimal online orthogonal polynomial projections (Gu et al., 2020, 2021, 2022) that enabled the architecture to retain information over long sequences. However, many recent works (Han et al., 2024; Trockman et al., 2024; Ren et al., 2024), including the Mamba-2 followup (Dao and Gu, 2024), have since shown that SSM layers are equivalent to linear attention with forgetting; the forgetting factor maps onto the time-varying step size and transition matrix used in SSMs.

Fast-weight layers and test-time training layers. An alternative to attenuating *all* past information in $\mathbf{M}_{t-1}$ is to forget only the values whose keys are most similar to the new key $\mathbf{k}_t$. This was the motivation behind DeltaNet (Schlag et al., 2021), which is a revival of classic fast weight layers (Schmidhuber, 1992). DeltaNet follows a recurrence

$$\mathbf{M}_t = \mathbf{M}_{t-1} - \underbrace{(\mathbf{M}_{t-1} \mathbf{k}_t)}_{\mathbf{v}_t^{\text{old}}} \mathbf{k}_t^\top + \underbrace{[\beta_t \mathbf{v}_t + (1 - \beta_t) \mathbf{M}_{t-1} \mathbf{k}_t]}_{\mathbf{v}_t^{\text{new}}} \mathbf{k}_t^\top \quad (5)$$

$$= \mathbf{M}_{t-1} (\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) + \beta_t \mathbf{v}_t \mathbf{k}_t^\top \quad (6)$$

where the first equation follows the notation of Schlag et al. (2021) in defining $\beta_t \in [0, 1]$, $\mathbf{v}_t^{\text{old}}$, and $\mathbf{v}_t^{\text{new}}$. Like previous layers, the output at step t is also $\mathbf{y}_t = \mathbf{M}_t \mathbf{q}_t$. Comparing

the recurrence of DeltaNet (Equation 5) to that of linear attention (Equation 2), we can interpret the update rule as rewriting the value that was bound to key $\mathbf{k}_t$. The update rule deletes an old value $\mathbf{v}_t^{\text{old}} = \mathbf{M}_{t-1}\mathbf{k}_t$ retrieved from the previous timestep’s memory $\mathbf{M}_{t-1}$ but using the new key $\mathbf{k}_t$. The layer then updates the recurrent state with a new interpolated value $\mathbf{v}_t^{\text{new}}$, binding it to $\mathbf{k}_t$.

The iterates $\mathbf{M}_t$ are also known as “fast-weights”, originally proposed by Schmidhuber (1992). These weights/iterates are “fast” because they change with each timestep, rapidly adapting to the input tokens, in contrast to the “slow-weights” in the rest of the neural network, such as the linear projections weights, which are only adjusted at training time. The traditional “slow-weights” can be seen as the weights of a metalearner (Clark et al., 2022), learning to use the “fast-weights” to learn from the input sequence (Sun et al., 2024).

Online learning layers. The recurrences we have reviewed so far all share a common high-level behavior: at each timestep t , the recurrent state must be adapted to incorporate information from the latest key-value pair $(\mathbf{k}_t, \mathbf{v}_t)$. This property can be seen as making an online sequence of decisions, balancing between retaining historical information and adapting to new information.

Motivated by this sequential decision making perspective, Liu et al. (2024) proposed to derive recurrences by minimizing an online sequence of objectives:

$$\mathbf{M}_t = \underset{\mathbf{M}}{\operatorname{argmin}} d(\mathbf{M}_{t-1}, \mathbf{M}) + \ell_t(\mathbf{M}) \quad (7)$$

where $d(\mathbf{M}_{t-1}, \mathbf{M})$ is a measure of the discrepancy between $\mathbf{M}$ and the previous iterate $\mathbf{M}_{t-1}$, encouraging retention of older information, and ℓ_t is a loss function that encourages adapting to new information. Once again, the output is computed by $\mathbf{y}_t = \mathbf{M}_t\mathbf{q}_t$. Recurrent layers derived via Equation 7 are called “online learning layers” or “online learners”. Recent examples of such layers include Longhorn (Liu et al., 2024), Gated DeltaNet (Yang et al., 2024a), and Titans (Behrouz et al., 2024), each of these layers was derived from a specific choice of functions d and ℓ_t . See Table 2 from Yang et al. (2024a) for each layer’s choice of d and ℓ_t .

3. Test-time regression as a framework for designing sequence layers

Having discussed a number of existing sequence layers, each motivated by an atomic architectural modification, we now present a unified framework to clarify their relationship to each other. Our approach is motivated by empirical findings that a model’s associative recall ability strongly correlates with its performance and in-context learning capabilities (Olsson et al., 2022; Arora et al., 2023a). We formalize the two step process of memorization and retrieval, presenting a principled recipe for deriving sequence layers with mathematically grounded recall abilities.

Step 1: Memorization as regression. From our everyday experience, we already have an intuition for how associative memory should behave; for example, hearing a friend’s name should trigger a mental impression of that friend. This pairing of a “cue” and a “response” is known as an association. To reflect terminology in deep learning (Vaswani et al., 2017), we will refer to cues as “keys” and responses as “values”. With this intuition, we can define

a mathematical model for associative memory. Given key-value pairs $(\mathbf{k}_1, \mathbf{v}_1), \dots, (\mathbf{k}_t, \mathbf{v}_t) \in \mathbb{R}^{D_k} \times \mathbb{R}^{D_v}$ to be memorized, an *associative memory system* is a function $m : \mathbb{R}^{D_k} \rightarrow \mathbb{R}^{D_v}$ such that $m(\mathbf{k}_i) \approx \mathbf{v}_i$ for $i = 1, \dots, t$ (Kohonen, 1972, 1989). Such a mapping m between two sets of related objects is also known as a hetero-associative memory in classic signal processing and neurocomputing literature (Hinton and Anderson, 1989).

Our main observation is that finding a memory map that satisfies these constraints reduces to finding a vector-valued regressor m . We can then store a set of key-value associations into an associative memory system by solving the weighted regression problem

$$\text{Memorize associations:} \quad m_t \approx \underset{m \in \mathcal{M}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^t \gamma_i^{(t)} \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2, \quad (8)$$

where the importance of each association is controlled by adjusting the relative weights $\{\gamma_i^{(t)}\}_{i=1}^t$. Classic works on associative memory are restricted to linear memory maps, corresponding to linear regression (Kohonen, 1989, Chapter 6.6), but here we allow $\mathcal{M}$ to be any function class. We refer to this relationship between regression and associative memory as the “regression-memory correspondence”.

Step 2: Memory retrieval as function application. Having derived a regression-based procedure to memorize a set of associations into m , we can now retrieve values from memory by applying m to a new vector, which we call a “query” $\mathbf{q}$, producing output

$$\text{Memory retrieval:} \quad \mathbf{y}_t = m_t(\mathbf{q}_t). \quad (9)$$

The retrieval performance depends on how well Equation 8 is minimized. When Equation 8 attains zero-loss, m_t memorizes all associations perfectly, and we get perfect retrieval: $m_t(\mathbf{k}_i) = \mathbf{v}_i$ for $i = 1, \dots, t$. For certain function classes $\mathcal{M}$, e.g. smooth functions, m can also extrapolate outside of the existing set of keys. In those cases, $m_t(\mathbf{q})$ instead retrieves a value based on the proximity of $\mathbf{q}$ to the existing set of keys.

Sequence layers that memorize then retrieve. Having formalized memorization and retrieval via our regression framework, we can now combine the two to produce sequence layers that *directly perform associative recall as a forward pass*. Recall that we are given $\{\mathbf{q}_i\}_{i=1}^T$, $\{\mathbf{k}_i\}_{i=1}^T$, and $\{\mathbf{v}_i\}_{i=1}^T$, and we seek to produce a set of outputs $\{\mathbf{y}_i\}_{i=1}^T$ with our sequence layer. The forward pass of a test-time regression layer first *memorizes* a set of key-value pairs and then produces $\mathbf{y}_t$ by *retrieving* a value from its associative memory. More specifically, in causal modeling, at each timestep t , we first solve Equation 8 to get a regressor m_t that memorizes the prefix $\{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^t$; then we apply m_t to $\mathbf{q}_t$ to get $\mathbf{y}_t = m_t(\mathbf{q}_t)$. This naturally generalizes to non-causal tasks where m_t can depend on suffix key-value pairs $\{\mathbf{k}_i, \mathbf{v}_i\}_{i>t}$ as well. Our procedure is compatible with multihead design patterns inspired by multihead attention (Vaswani et al., 2017), multi-query attention (Shazeer, 2019), and grouped query attention (Ainslie et al., 2023). For example, multihead test-time regression can be implemented using H independent sets of queries, keys, and values, resulting in H independent associative memory maps at each timestep.

A recipe for designing your own sequence layer. Although Equation 8 succinctly summarizes the memorization step of our framework, it leaves many details unspecified.

For example, over which class of regressors $\mathcal{M}$ should we consider? Additionally, which associations should we prioritize memorizing? Finally, which optimization procedure should we use to minimize the loss? The regression-memory correspondence thus allows for a vast design space. We partition this wide landscape into three simple design choices:

1. the relative importance of each association, specified through weights in Equation 8,
2. the function class $\mathcal{M}$ over which we search,
3. the minimization algorithm,

giving us a formulaic “recipe” for deriving sequence layers that can perform associative recall.

4. Deriving existing architectures from regression

To demonstrate how our framework unifies our understanding of sequence models, we succinctly derive a swath of recently proposed sequence layers. We categorize existing sequence layers into five distinct classes: linear attention (possibly with feature maps), linear attention with gating (which includes state-space models like Mamba), fast-weight layers, online learners, and softmax self-attention.

Notation. We first explain the mathematical notation we will use in our derivations. We use bold lowercase letters to refer to vectors (e.g. $\mathbf{x}, \mathbf{k}, \mathbf{v}$) and bold uppercase letters to refer to matrices (e.g. $\mathbf{M}, \mathbf{K}, \mathbf{V}$). Assume we have a sequence of T inputs $\mathbf{x}_1, \dots, \mathbf{x}_T$ which have already transformed into a sequence of keys and values $(\mathbf{k}_1, \mathbf{v}_1), \dots, (\mathbf{k}_T, \mathbf{v}_T)$. We will use these keys and values as the dataset for our test-time regression problem. Let $\mathbf{K}_t = [\mathbf{k}_1 \dots \mathbf{k}_t]^\top \in \mathbb{R}^{t \times D_k}$ and $\mathbf{V}_t = [\mathbf{v}_1 \dots \mathbf{v}_t]^\top \in \mathbb{R}^{t \times D_v}$ be matrices that contain t keys and t values respectively, for $t = 1, \dots, T$. Each key/value vector is a row of the key/value matrix.

Vignette 1: Linear attention is (suboptimal) linear least squares regression

We start by deriving an architecture from linear least squares, the most classic regression method. Linear least squares corresponds to the following choices in our recipe.

1. Choice of weights: assign equal weight $\gamma_i^{(t)} = 1$ to each association,

$$m_{\text{LINEAR}, t} = \underset{m \in \mathcal{M}_{\text{LINEAR}}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^t \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2. \quad (10)$$

2. Choice of function class: linear functions $\mathcal{M}_{\text{LINEAR}} = \{m \mid m(\mathbf{k}) = \mathbf{M}\mathbf{k}, \mathbf{M} \in \mathbb{R}^{D_v \times D_k}\}$.
3. Choice of minimization algorithm: analytical solution equivalent to one step of Newton’s method¹.

1. Since the objective function is quadratic, the analytical solution is equivalent to one iteration of Newton’s method initializing from the origin.

Since $m_{\text{LINEAR}, t}$ is a linear function, it is equivalent to a linear map $\mathbf{M}_t \in \mathbb{R}^{D_v \times D_k}$. Minimizing the linear least squares objective results in the solution:

$$\mathbf{M}_t = \underset{\mathbf{M}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^t \|\mathbf{v}_i - \mathbf{M} \mathbf{k}_i\|_2^2 = \begin{cases} \mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1}, & \text{overdetermined, e.g. } t \geq D_k, \\ \mathbf{V}_t^\top (\mathbf{K}_t \mathbf{K}_t^\top)^{-1} \mathbf{K}_t, & \text{underdetermined, e.g. } t < D_k, \end{cases} \quad (11)$$

where we choose the min-norm solution in the underdetermined case. The overdetermined case corresponds to $\mathbf{K}_t$ having rank D_k (linearly independent columns) and the underdetermined case corresponds to $\mathbf{K}_t$ having rank t (linearly independent rows). In general, we can express the solution in terms of a Moore-Penrose pseudoinverse: $\mathbf{M}_t = \mathbf{V}_t^\top (\mathbf{K}_t^\dagger)^\top$. We restrict our discussion to the case of $t \geq D_k$ with linearly independent columns, which is the case for typical applications.

Relationship to linear attention. Although $m_{\text{LINEAR}, t}$ is the optimal linear memory with respect to our objective function, it can be expensive to compute, requiring us to invert a $D_k \times D_k$ matrix at every timestep. One crude but hardware-efficient simplification would be to assume that $\mathbf{K}_t^\top \mathbf{K}_t = \mathbf{I}$, avoiding matrix inversion completely, at the cost of some recall ability. After making this rough approximation, *we arrive exactly at the equations for linear attention*:

$$\mathbf{y}_t = m_{\text{LINEAR}}(\mathbf{q}_t) = \mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1} \mathbf{q}_t \approx \mathbf{V}_t^\top \mathbf{K}_t \mathbf{q}_t = \sum_{i=1}^t \mathbf{v}_i \mathbf{k}_i^\top \mathbf{q}_t. \quad (12)$$

Our derivation clarifies exactly how linear attention underperforms as an associative memory. When the keys are zero mean, $\mathbf{K}_t^\top \mathbf{K}_t$ is proportional to the empirical covariance matrix of the keys. Thus linear attention *is a crude associative memory that ignores the covariance between the dimensions of the key vectors*. In fact, it is an optimal linear memory only when $t \leq D_k$ and the keys are orthonormal $\mathbf{k}_i^\top \mathbf{k}_j = \delta_{ij}$.

From a training stability perspective, the inverse covariance factor helps prevent the output $\mathbf{y}_t$ from becoming unbounded. We can bound the norm of the output of a least squares layer (Equation 11) by

$$\|\mathbf{y}_t\|_2 = \|\mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1} \mathbf{q}_t\|_2 \leq \frac{\|\mathbf{q}_t\|_2 (\sum_{i=1}^t \|\mathbf{v}_i\|_2 \|\mathbf{k}_i\|_2)}{\lambda_{\min}(\mathbf{K}_t^\top \mathbf{K}_t)} \quad (13)$$

where $\lambda_{\min}(\mathbf{K}_t^\top \mathbf{K}_t)$ is the minimum eigenvalue of $\mathbf{K}_t^\top \mathbf{K}_t$. See Section A for the derivation. When we ignore the covariance matrix, approximating it with the identity, the denominator disappears and the output $\mathbf{y}_t$ can grow arbitrarily large with the sequence length, leading to training instability. This instability indeed shows up in practice for linear attention, and Qin et al. (2022) proposed output normalization to stabilize training. Our derivation shows that output normalization works by approximately restoring the self-normalizing property of Equation 11 that linear attention loses.

Regardless of training stability, we will see in Section 5 that ignoring the covariance causes linear attention to perform worse at associative recall.

Efficient computation. Conveniently, we can compute $\mathbf{M}_t$ in Equation 11 for $t = 1, \dots, T$ in parallel or sequentially. To compute them in parallel, we solve a batch of T independent linear systems. To compute them sequentially, we note that $(\mathbf{K}_t^\top \mathbf{K}_t)^{-1} = (\mathbf{K}_{t-1}^\top \mathbf{K}_t + \mathbf{k}_t \mathbf{k}_t^\top)^{-1}$, allowing us to autoregressively compute the inverses via the Woodbury matrix identity. The sequential form is known as online linear regression or recursive least squares (RLS) (Haykin, 2014). Thus during training we can parallelize across the sequence axis by solving a batch of T linear systems, and during testing we can unroll the recurrence into a recurrent neural network. In prior deep learning literature, this idea of linear-regression-as-a-layer has been explored by Garnelo and Czarnecki (2023) and von Oswald et al. (2023) under the names “Intention” and “Mesa-layer”, respectively. The Intention layer was proposed as an alternative computation pattern with queries, keys, and values that attention cannot directly represent. The Mesa-layer was proposed as the closed-form optimal solution to solving the least-squares problem along the sequence dimension within the forward pass, imitating the in-context learning ability of softmax-attention. The computation of the Mesa-layer is equivalent to that of recursive least squares, as derived here; our derivation here provides a complementary perspective through the lens of associative memory.

Nonlinear associative memory. So far we have only considered *linear* associative memory implemented by linear functions in $\mathcal{M}_{\text{LINEAR}}$, which have limited expressiveness. We can derive *nonlinear* memory by using a feature map $\phi : \mathbb{R}^{D_k} \rightarrow \mathbb{R}^{D_\phi}$, replacing each key $\mathbf{k}_i$ with $\phi(\mathbf{k}_i)$. Then our function class is defined by $\mathcal{M}_\phi = \{m \mid m(\mathbf{k}) = \mathbf{M}\phi(\mathbf{k}), \mathbf{M} \in \mathbb{R}^{D_v \times D_\phi}\}$ and we solve the least-squares objective $\arg\min_{\mathbf{M}} \sum_{i=1}^t \|\mathbf{v}_i - \mathbf{M}\phi(\mathbf{k}_i)\|_2^2 / 2$. Solving this least squares objective for $\mathbf{M}$ results in an optimal associative memory that is nonlinear with respect to the input keys. Parameterizing associative memory with a nonlinear feature map has a long history, dating back to Poggio (1975) which found polynomial feature maps to be highly effective, consistent with more recent works (Zhang et al., 2023a; Arora et al., 2024).

If we make the same crude approximation as linear attention of disregarding the covariance among $\{\phi(\mathbf{k}_i)\}_{i=1}^t$, we arrive at a suboptimal nonlinear memory. Thus, recent works that apply linear attention with a nonlinear feature map, such as Schlag et al. (2021); Kasai et al. (2021); Zhang et al. (2023a); Chen et al. (2024), and others, can be understood within our framework as suboptimal nonlinear memory as well.

Vignette 2: Gated linear attention and state-space models are (suboptimal) weighted linear least squares regression

For many kinds of sequential data, such as time-series data and text data, recent tokens are more informative of predicting the next token. Thus a natural requirement is to encourage our memory m_t to focus more on the recent history by downweighting the past. Fortunately, this can be straightforwardly enforced through classic *weighted* linear least squares regression together with geometrically decaying weights to *forget* older key-value pairs.

To develop an architecture from weighted linear least squares, we only need to add weights to the optimization objective relative to ordinary least squares:

1. Choice of weights: assign time-decaying weights to associations,

$$m_{\text{LINEAR}, t} = \underset{m \in \mathcal{M}_{\text{LINEAR}}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^t \gamma_i^{(t)} \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2, \quad (14)$$

Here, to connect to past literature, we focus on weights are geometrically decaying in time: $\gamma_i^{(t)} = \prod_{j=i+1}^t \gamma_j$ and $\gamma_i \in [0, 1]$ for $i = 1, \dots, t$. Each association is down-weighted based on how far it is from the current timestep t : $\gamma_1^{(t)} \leq \dots \gamma_{t-1}^{(t)} \leq \gamma_t^{(t)} = 1$.

2. Choice of function class: linear functions $\mathcal{M}_{\text{LINEAR}} = \{m \mid m(\mathbf{k}) = \mathbf{M}\mathbf{k}, \mathbf{M} \in \mathbb{R}^{D_v \times D_k}\}$.
3. Choice of minimization algorithm: analytical solution equivalent to one step of Newton’s method.

We can solve weighted linear regression by instead solving an unweighted regression problem involving rescaled keys and values, $\mathbf{V}_t \mapsto \mathbf{\Gamma}_t^{1/2} \mathbf{V}_t$ and $\mathbf{K}_t \mapsto \mathbf{\Gamma}_t^{1/2} \mathbf{K}_t$, where we defined a diagonal matrix $\mathbf{\Gamma}_t \in \mathbb{R}^{t \times t}$ with entries $(\mathbf{\Gamma}_t)_{ii} = \gamma_i^{(t)}$. Making the rescaled substitution and reusing the analytical min-norm solution that we had before, we see that

$$\mathbf{M}_t = \begin{cases} \mathbf{V}_t^\top \mathbf{\Gamma}_t \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{\Gamma}_t \mathbf{K}_t)^{-1}, & \text{overdetermined, e.g. } t \geq D_k \\ \mathbf{V}_t^\top (\mathbf{K}_t \mathbf{K}_t^\top)^{-1} \mathbf{K}_t, & \text{underdetermined, e.g. } t < D_k. \end{cases} \quad (15)$$

The underdetermined case has the same solution in both the weighted and unweighted case; intuitively this is because when the system is underdetermined, there are infinitely many solutions that interpolate the data points and attain zero regression loss, rendering the weights irrelevant.

Relationship to linear attention with gating and state-space models. Once again, computing $\mathbf{M}_t$ requires matrix inversion which is not efficient on matrix-multiplication based hardware. Making a crude approximation similar to before of $\mathbf{K}_t^\top \mathbf{\Gamma}_t \mathbf{K}_t \approx \mathbf{I}$ results in $\mathbf{M}_t \approx \mathbf{V}_t^\top \mathbf{\Gamma}_t \mathbf{K}_t = \sum_{i=1}^t \gamma_i^{(t)} \mathbf{v}_i \mathbf{k}_i^\top$. We can further unroll this equation into a recurrence due to the multiplicative structure of our weights:

$$\mathbf{M}_t \approx \sum_{i=1}^t \gamma_i^{(t)} \mathbf{v}_i \mathbf{k}_i^\top = \sum_{i=1}^t \left(\prod_{j=i+1}^t \gamma_j \right) \mathbf{v}_i \mathbf{k}_i^\top = \gamma_t \sum_{i=1}^{t-1} \gamma_i^{(t-1)} \mathbf{v}_i \mathbf{k}_i^\top + \mathbf{v}_t \mathbf{k}_t^\top = \gamma_t \mathbf{M}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top. \quad (16)$$

This is exactly the recurrence implemented by gated variants of linear attention, such as GLA (Yang et al., 2024b), RetNet (Sun et al., 2023), Gateloop (Katsch, 2024), RWKV-6 (Peng et al., 2024), mLSTM (Beck et al., 2024), LRU (Orvieto et al., 2023), as well as state-space models like Mamba-2 (Dao and Gu, 2024)². Forgetting older tokens significantly

2. Some architectures such as GLA (Yang et al., 2024b) and Mamba-1 (Gu and Dao, 2024) have also used vector-valued gating, but their empirical performance are comparable to scalar gating. Vector-valued gating can be derived by replacing the regression objective’s L2 norm $\|\cdot\|_2$ with a diagonal Mahalanobis norm.

improves performance, which was first discovered *empirically* (Hochreiter and Schmidhuber, 1997; Greff et al., 2017; van der Westhuizen and Lasenby, 2018). Our derivation of gating from weighted least squares provides a mathematical grounding for the forgetting heuristic upon which these sequence layers were based.

Efficient computation. Like before, we can compute the memory of our architecture (without approximation) sequentially or in parallel as long as we are willing to invest FLOPs in matrix inversion. Parallel computation can be done by solving a batch of T $D_k \times D_k$ linear systems. The sequential version can be derived by applying the Woodbury matrix inversion formula while taking into account our geometrically-decaying weights. Geometrically-weighted recursive least squares (RLS) is a classic method from the adaptive filtering literature (Johnstone et al., 1982).

Vignette 3: Fast weight programmers and online learners are first-order methods for solving streaming least-squares

Up until now we have derived linear associative memory architectures through direct analytical solution, equivalent to one step of Newton’s method, a second-order optimization method. However, second-order iterative methods are computationally expensive, requiring matrix inversions at each step. On the other hand, disregarding the matrix inversion as done by linear attention leads to poor performance, since it ignores the covariance structure of the data.

A computationally cheaper alternative is to apply a first-order gradient-descent method. We consider this here with the following design choices:

1. Choice of weights: assign equal weight $\gamma_i^{(t)} = 1$ to each association,

$$m_{\text{LINEAR}, t} = \underset{m \in \mathcal{M}_{\text{LINEAR}}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^t \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2. \quad (17)$$

2. Choice of function class: linear functions $\mathcal{M}_{\text{LINEAR}} = \{m \mid m(\mathbf{k}) = \mathbf{M}\mathbf{k}, \mathbf{M} \in \mathbb{R}^{D_v \times D_k}\}$.
3. Choice of minimization algorithm: gradient descent, possibly with adaptive step sizes, momentum, or weight decay (L2 regularization).

We can now derive the test-time regression layer corresponding to a first-order iterative method.

Since we consider linear functions, solving Equation 17 is equivalent to minimizing the objective

$$\mathcal{L}_t(\mathbf{M}) = \sum_{i=1}^t L_i(\mathbf{M}) = \sum_{i=1}^t \frac{1}{2} \|\mathbf{v}_i - \mathbf{M}\mathbf{k}_i\|_2^2, \quad (18)$$

using its gradient

$$\nabla \mathcal{L}_t(\mathbf{M}) = \sum_{i=1}^t \nabla L_i(\mathbf{M}) = \sum_{i=1}^t (\mathbf{M}\mathbf{k}_i - \mathbf{v}_i) \mathbf{k}_i^\top = (\mathbf{M}\mathbf{K}_t^\top - \mathbf{V}_t^\top) \mathbf{K}_t. \quad (19)$$

to find $\mathbf{M}_t$. Let the initialization be at the origin $\mathbf{M}_t^{(0)} = \mathbf{0}$. Iteratively solving this objective with full-batch gradient descent and step size $\beta_t^{(i)}$ at each iteration results in

$$\mathbf{M}_t^{(i)} = \mathbf{M}_t^{(i-1)} - \beta_t^{(i)} \nabla \mathcal{L}_t(\mathbf{M}_t^{(i-1)}) \quad (20)$$

$$= \mathbf{M}_t^{(i-1)} - \beta_t^{(i)} \sum_{j=1}^t \nabla L_j(\mathbf{M}_t^{(i-1)}) \quad (21)$$

$$= \mathbf{M}_t^{(i-1)} - \beta_t^{(i)} \sum_{j=1}^t (\mathbf{M}_t^{(i-1)} \mathbf{k}_j - \mathbf{v}_j) \mathbf{k}_j^\top \quad (22)$$

$$= \mathbf{M}_t^{(i-1)} (\mathbf{I} - \sum_{j=1}^t \beta_t^{(i)} \mathbf{k}_j \mathbf{k}_j^\top) + \beta_t^{(i)} \sum_{j=1}^t \mathbf{v}_j \mathbf{k}_j^\top. \quad (23)$$

After I iterations, we produce the linear memory represented by $\mathbf{M}_t = \mathbf{M}_t^{(I)}$.

Linear attention and variants as one-step of batch gradient descent Consider the $I = 1$ case with step size $\beta_t^{(i)} = 1$. Since we initialize at the origin, we obtain the equation for linear attention

$$\mathbf{M}_t^{(1)} = \mathbf{M}_t^{(0)} - \nabla \mathcal{L}_t(\mathbf{M}_t^{(0)}) = \mathbf{V}_t^\top \mathbf{K}_t = \sum_{i=1}^t \mathbf{v}_i \mathbf{k}_i^\top. \quad (24)$$

This is exactly the result of performing *one-step of full batch gradient descent* on the least square objective $\mathcal{L}_t$. A similar result can be shown for layers that implement linear attention with forgetting, performing one step of full batch gradient descent on the weighted least square objective.

A standard technique to accelerate the convergence of gradient descent is to precondition the gradient (Boyd and Vandenberghe, 2004). The optimal preconditioner in this case, due to the quadratic objective, is the *constant* Hessian of $\mathcal{L}_t$, $\mathbf{H}_{\mathcal{L}_t}(\mathbf{M}) = \mathbf{K}_t^\top \mathbf{K}_t$. This preconditioner allows gradient descent to converge in a *single update*

$$\mathbf{M}_t^{(1)} = \mathbf{M}_t^{(0)} - \nabla \mathcal{L}_t(\mathbf{M}_t^{(0)}) \mathbf{H}_{\mathcal{L}_t}(\mathbf{M}_t^{(0)})^{-1} = \mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1}, \quad (25)$$

reproducing the analytic solution from Equation 11. The preconditioned recurrence is equivalent to one step of Newton’s method, a second order iterative method.

By taking this iterative optimization perspective, we see that *linear attention and the ideal linear associative memory lie on two extremes: one-step of batch gradient descent with no preconditioning (Equation 2) versus one-step of batch gradient descent with perfect preconditioning (Equation 11)*. The difference in performance comes from whether the optimizer accounts for the curvature of the objective, governed by the covariance of the keys (Boyd and Vandenberghe, 2004). Exploring other preconditioners, e.g. quasi-Newton methods, that interpolates between the two extremes may be an interesting direction for future research.

Stochastic gradient descent as a layer. An alternative to full-batch gradient descent is stochastic gradient descent (SGD) where we use a subset of the samples to compute the gradient at each step. This is particularly convenient in causal sequence modeling where keys and values typically are presented one at a time, rather than all at once. In this streaming setting, we can instead perform single-example SGD to iteratively minimize Equation 8. In the case of linear memories $\mathcal{M}_{\text{LINEAR}}$, Equation 21 with a single-example and variable step sizes $\beta_t^{(i)}$ becomes

$$\mathbf{M}_t^{(i)} = \mathbf{M}_t^{(i-1)} - \beta_t^{(i)} \nabla L_t(\mathbf{M}_t^{(i-1)}) \quad (26)$$

$$= \mathbf{M}_t^{(i-1)} + \beta_t^{(i)} (\mathbf{v}_t - \mathbf{M}_t^{(i-1)} \mathbf{k}_t) \mathbf{k}_t^\top \quad (27)$$

$$= \mathbf{M}_t^{(i-1)} (\mathbf{I} - \beta_t^{(i)} \mathbf{k}_t \mathbf{k}_t^\top) + \beta_t^{(i)} \mathbf{v}_t \mathbf{k}_t^\top, \quad (28)$$

where $\mathbf{M}_t$ is the matrix representation of m_t . When we take only one gradient update per timestep t and initialize our iterate using the previous timestep’s linear map, $\mathbf{M}_t^{(0)} = \mathbf{M}_{t-1}$, we end up with the recurrence of DeltaNet (Schlag et al., 2021; Yang et al., 2024c) from Equation 6:

$$\mathbf{M}_t = \mathbf{M}_t^{(1)} = \mathbf{M}_{t-1} (\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top) + \beta_t \mathbf{v}_t \mathbf{k}_t^\top. \quad (29)$$

As we perform stochastic gradient descent, we produce an online sequence $\{\mathbf{M}_i\}_{i=1}^t$, each of which is an associative memory m_i for key-value pairs up to $(\mathbf{k}_i, \mathbf{v}_i)$. The stochastic gradient approach to streaming inputs has the advantage that m_t focuses more on recent key-value pairs, rather than uniformly over all timesteps as in the case of unweighted RLS and unweighted linear attention.

Equation 29 is known by many names, such as the delta rule or the Widrow-Hoff algorithm (Widrow and Hoff, 1988). Within the adaptive filtering literature this recurrence is known as Least-Mean Squares (LMS) (Haykin, 2014), serving as one of the most common algorithms for streaming signals processing. From this derivation, we see that the prevalence of the update across disciplines comes from the fact that the update rule is actually an SGD update.

In our case of sequence layers, the SGD update occurs *within* the layer as part of the forward pass. One extension to Equation 29 is to consider SGD with nonlinear parametric regressors m_t . Indeed, this extension is equivalent to test-time training (TTT) layers proposed by Sun et al. (2024), which apply SGD to a nonlinear memory m_t , e.g. multi-layer perceptrons or linear maps with layer normalizations. In contrast to our associative memory derivation here, Sun et al. (2024) motivated their method by viewing the squared loss as an self-supervised auto-encoding objective. Another extension is to take multiple gradient steps per example $(\mathbf{k}_t, \mathbf{v}_t)$, which was recently explored by DeltaProduct (Siems et al., 2025).

Including adaptive step sizes. To further improve the performance of a gradient-descent-based sequence layer, we can consider adaptive step sizes. A common way to derive adaptive step sizes is to use the relationship between gradient descent and online learning (Cesa-Bianchi et al., 2004; Zinkevich, 2003), a connection that has produced many successful algorithms in the past, such as Adagrad (Duchi et al., 2011) and Adam (Kingma and Ba, 2015).

Consider the Longhorn sequence layer derived by Liu et al. (2024) by solving the online learning objective

$$\mathbf{M}_t = \underset{\mathbf{M}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{M} - \mathbf{M}_{t-1}\|_F^2 + \frac{\delta_t}{2} \|\mathbf{v}_t - \mathbf{M}\mathbf{k}_t\|_2^2. \quad (30)$$

This corresponds to $d(\mathbf{M}_{t-1}, \mathbf{M}) = \|\mathbf{M} - \mathbf{M}_{t-1}\|_F^2/2$ and $\ell_t(\mathbf{M}) = \delta_t \|\mathbf{v}_t - \mathbf{M}\mathbf{k}_t\|_2^2/2$ in Equation 7. The solution to Equation 30 is the recurrence

$$\mathbf{M}_t = \mathbf{M}_{t-1} \left(\mathbf{I} - \frac{\delta_t}{1 + \delta_t \|\mathbf{k}_t\|_2^2} \mathbf{k}_t \mathbf{k}_t^\top \right) + \frac{\delta_t}{1 + \delta_t \|\mathbf{k}_t\|_2^2} \mathbf{v}_t \mathbf{k}_t^\top. \quad (31)$$

However, by comparing the Longhorn update in Equation 31 to the SGD update in Equation 29, we see that Longhorn’s recurrence implements SGD but with a conservative adaptive step size of $\beta_t = \delta_t / (1 + \delta_t \|\mathbf{k}_t\|_2^2)$. Thus although the Longhorn layer was derived from an online learning perspective, its global objective is still the least squares regression objective.

We can better understand the effects of this adaptive step size by looking at δ_t , which controls how much we care about memorizing the latest association $(\mathbf{k}_t, \mathbf{v}_t)$. When $\delta_t = 0$, we see that $\mathbf{M}_t = \mathbf{M}_{t-1}$ and the latest association is excluded completely. When $\delta_t \rightarrow \infty$, Equation 30 becomes equivalent to a constrained optimization problem

$$\mathbf{M}_t = \underset{\mathbf{M}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{M} - \mathbf{M}_{t-1}\|_F^2 \quad \text{subject to } \mathbf{M}_t \mathbf{k}_t = \mathbf{v}_t. \quad (32)$$

Solving this constrained optimization problem results in a hyperparameter-free step size $\beta_t = 1/\|\mathbf{k}_t\|_2^2$ with update

$$\mathbf{M}_t = \mathbf{M}_{t-1} \left(\mathbf{I} - \frac{1}{\|\mathbf{k}_t\|_2^2} \mathbf{k}_t \mathbf{k}_t^\top \right) + \frac{1}{\|\mathbf{k}_t\|_2^2} \mathbf{v}_t \mathbf{k}_t^\top. \quad (33)$$

This special case is also known as the normalized LMS (NLMS) update in adaptive filtering (Castoldi and de Campos, 2009) and known as the projection method (Tanabe, 1971) or Kaczmarz’s method (Kaczmarz, 1937) in optimization theory. The advantage of Equation 33 is that it is hyperparameter-free, which may make it easier to tune. However, to our knowledge, the recurrence in Equation 33 has not been explored as a sequence layer.

Including L2 regularization. In addition to adaptive step sizes, we can add regularization. Consider adding L2 regularizing to the least squares objective Equation 18

$$\mathcal{L}_{\text{reg},t}(\mathbf{M}) = \sum_{i=1}^t L_i(\mathbf{M}) + \frac{\lambda_i}{2} \|\mathbf{M}\|_2^2 \quad (34)$$

where $\lambda_i \in [0, 1]$ is the regularization strength. When we minimize this regularized objective with single-example stochastic gradient descent like in Equation 29, we obtain the recurrence

$$\mathbf{M}_t = \mathbf{M}_{t-1} - \beta_t [\nabla L_t(\mathbf{M}_{t-1}) + \lambda_t \mathbf{M}_{t-1}] \quad (35)$$

$$= (1 - \beta_t \lambda_t) \mathbf{M}_{t-1} - \beta_t \nabla L_t(\mathbf{M}_{t-1}) \quad (36)$$

$$= (1 - \beta_t \lambda_t) \mathbf{M}_{t-1} + \beta_t (\mathbf{v}_t - \mathbf{M}_{t-1} \mathbf{k}_t) \mathbf{k}_t^\top, \quad (37)$$

which is known as Leaky LMS in adaptive filtering (Haykin, 2014). Notably, the first term of our new recurrence resembles Equation 27 but also includes a multiplicative factor similar to a forget gate.

Recently, Yang et al. (2024a) proposed Gated DeltaNet motivated by combining DeltaNet (Schlag et al., 2021) with the forget gating of Mamba-2 (Dao and Gu, 2024), following the online learning perspective of Liu et al. (2024). Here we show that the Gated DeltaNet recurrence is actually an equivalent reparameterization of our recurrence in Equation 37. Starting with Equation 37, we have

$$\mathbf{M}_t = (1 - \beta_t \lambda_t) \mathbf{M}_{t-1} + \beta_t (\mathbf{v}_t - \mathbf{M}_{t-1} \mathbf{k}_t) \mathbf{k}_t^\top \quad (38)$$

$$= \mathbf{M}_{t-1} \left[\underbrace{(1 - \beta_t \lambda_t) \mathbf{I}}_{\alpha_t} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top \right] + \beta_t \mathbf{v}_t \mathbf{k}_t^\top \quad (39)$$

$$= \mathbf{M}_{t-1} \alpha_t \left[\mathbf{I} - \underbrace{\frac{\beta_t}{\alpha_t} \mathbf{k}_t \mathbf{k}_t^\top}_{\eta_t} \right] + \frac{\beta_t}{\alpha_t} \underbrace{(\alpha_t \mathbf{v}_t)}_{\mathbf{v}'_t} \mathbf{k}_t^\top \quad (40)$$

$$= \mathbf{M}_{t-1} \alpha_t \left(\mathbf{I} - \eta_t \mathbf{k}_t \mathbf{k}_t^\top \right) + \eta_t \mathbf{v}'_t \mathbf{k}_t^\top \quad (\text{Gated DeltaNet recurrence}) \quad (41)$$

Thus Gated DeltaNet is equivalent to L2-regularized regression via SGD where the regularization strength λ_t is coupled with the step size β_t and $\mathbf{v}_t$ is rescaled by α_t . We can transform Equation 41 into Equation 37 via the invertible mapping $(\alpha_t, \eta_t, \mathbf{v}'_t) = (1 - \beta_t \lambda_t, \beta_t / (1 - \beta_t \lambda_t), (1 - \beta_t \lambda_t) \mathbf{v}_t)$. The inverse map is given by $(\lambda_t, \beta_t, \mathbf{v}_t) = ((1 - \alpha_t) / (\alpha_t \eta_t), \alpha_t \eta_t, \mathbf{v}'_t / \alpha_t)$. Past works have shown that decoupling weight decay from the step size results in better generalization and easier-to-tune hyperparameters (Loshchilov and Hutter, 2019), suggesting that similar modifications to Gated DeltaNet may improve model performance.

Vignette 4: Softmax attention is nonparametric regression

Having exhaustively derived multiple associative memory architectures via linear and non-linear parametric regression, we can now derive self-attention-like architectures using the tools of nonparametric regression.

Unnormalized softmax attention is (suboptimal) kernel regression. We start first from kernel regression as the most common nonparametric regression method (Schölkopf and Smola, 2001), and applying the kernel trick to our feature-mapped linear memory. Let $k : \mathbb{R}^{D_k \times D_k} \rightarrow \mathbb{R}$ be a Mercer kernel such that $k(\mathbf{k}_i, \mathbf{k}_j) = \phi(\mathbf{k}_i)^\top \phi(\mathbf{k}_j) = (\Phi \Phi^\top)_{ij}$ where $\Phi = [\phi(\mathbf{k}_1) \dots \phi(\mathbf{k}_t)]^\top \in \mathbb{R}^{t \times D_\phi}$ and $\phi : \mathbb{R}^{D_k} \rightarrow \mathbb{R}^{D_\phi}$ is a possibly infinite-dimensional feature map. Following our recipe for model design, we make the following design choices:

1. Choice of weights: assign equal weight $\gamma_i^{(t)} = 1$ to each association

$$m_{k,t} = \operatorname{argmin}_{m \in \mathcal{M}_k} \frac{1}{2} \sum_{i=1}^t \|\mathbf{v}_i - \mathbf{M} \phi(\mathbf{k}_i)\|_2^2, \quad (42)$$

where ϕ is possibly an infinite-dimensional feature map.

2. Choice of function class: linear functions $\mathcal{M}_k = \{m \mid m(\mathbf{k}) = \mathbf{M}\phi(\mathbf{k}), \mathbf{M} \in \mathbb{R}^{D_v \times D_\phi}\}$.
3. Choice of minimization algorithm: analytical solution.

The output of the optimal kernelized nonlinear associative memory induced by the kernel k at timestep t is the standard kernel regression prediction equation (Schölkopf and Smola, 2001)

$$m_{k,t}(\mathbf{q}) = \mathbf{V}_t^\top k(\mathbf{K}_t, \mathbf{K}_t)^{-1} k(\mathbf{K}_t, \mathbf{q}). \quad (43)$$

Here we follow standard shorthand notation from the kernel learning literature: $k(\mathbf{K}_t, \mathbf{K}_t)$ is a shorthand for the matrix with $k(\mathbf{k}_i, \mathbf{k}_j)$ in entry (i, j) ; $k(\mathbf{K}_t, \mathbf{q})$ is a shorthand for the vector with $k(\mathbf{k}_i, \mathbf{q})$ in entry i . When we use an exponential kernel $k(\mathbf{k}_i, \mathbf{k}_j) = \exp(\mathbf{k}_i^\top \mathbf{k}_j / \sqrt{D_k})$, corresponding to an infinite-dimensional feature map, and make the crude approximation $k(\mathbf{K}_t, \mathbf{K}_t) \approx \mathbf{I}$, we obtain

$$m_{k,t}(\mathbf{q}) = \mathbf{V}_t^\top k(\mathbf{K}_t, \mathbf{K}_t)^{-1} k(\mathbf{K}_t, \mathbf{q}) \quad (44)$$

$$\approx \mathbf{V}_t^\top k(\mathbf{K}_t, \mathbf{q}) \quad (45)$$

$$= \sum_{i=1}^t \mathbf{v}_i k(\mathbf{k}_i, \mathbf{q}) \quad (46)$$

$$= \sum_{i=1}^t \mathbf{v}_i \exp\left(\frac{\mathbf{k}_i^\top \mathbf{q}}{\sqrt{D_k}}\right), \quad (47)$$

an associative memory map equivalent to an *unnormalized* softmax attention. This relationship with kernel regression has been used by many previous works to derive more efficient versions of self-attention (Peng et al., 2020; Chen et al., 2021; Choromanski et al., 2020; Lu et al., 2021; Chen et al., 2024), borrowing techniques from kernel approximation methods. However unnormalized softmax attention is empirically known to be unstable to train (Lu et al., 2021). Thus we do not stop our derivations here; instead, we will see how the standard normalized softmax attention comes from another kind of nonparametric regressor.

Softmax attention with QKNorm is a locally-constant nonparametric regressor.

Local polynomial regression is another classic class of nonparametric estimators (Wasserman, 2006), which have been historically used as nonlinear smoothing functions in data analysis (Fan, 2018). Here, we consider the function class $\mathcal{M}^p$, which is the set of all vector-valued functions that are *locally* approximated by a degree p polynomial around each point $\mathbf{q} \in \mathbb{R}^{D_k}$. Here locality is measured by some similarity function s that only depends on the distance to $\mathbf{q}$, i.e. $s(\mathbf{k}, \mathbf{q}) = s(\|\mathbf{k} - \mathbf{q}\|_2)$. A classic example is the exponential smoothing kernel $s_{\exp, B}(\mathbf{k}, \mathbf{q}) = \exp(-\|\mathbf{k} - \mathbf{q}\|_2^2 / B)$ where B is a parameter controlling the local neighborhood size around $\mathbf{q}$. Local polynomial regression is most often discussed for one-dimensional data, but here we generalize it to the multivariate case.

Nonparametric regression with local polynomial estimators corresponds to the following design choices:

1. Choice of weights: assign weights to associations based on similarity to $\mathbf{q}$,

$$m_{p,t}(\mathbf{q}) = \operatorname{argmin}_{m \in \mathcal{M}^p} \frac{1}{2} \sum_{i=1}^t \gamma_i^{(t)} \|\mathbf{v}_i - m(\mathbf{k}_i)\|_2^2 \quad (48)$$

with weights given by $\gamma_i^{(t)} = s(\mathbf{k}_i, \mathbf{q}) = s(\|\mathbf{k}_i - \mathbf{q}\|_2)$.

2. Choice of function class: $\mathcal{M}^p$, the class of functions m_p such that for all points $\mathbf{q} \in \mathbb{R}^{D_k}$, the function can be locally approximated by a polynomial

$$\hat{m}_{\mathbf{q}}(\mathbf{k}) = \mathbf{M}^{[0]} + \mathbf{M}^{[1]}(\mathbf{k} - \mathbf{q}) + \mathbf{M}^{[2]}(\mathbf{k} - \mathbf{q}, \mathbf{k} - \mathbf{q}) + \dots + \mathbf{M}^{[p]}(\underbrace{\mathbf{k} - \mathbf{q}, \dots, \mathbf{k} - \mathbf{q}}_{p \text{ arguments}}) \quad (49)$$

where $\mathbf{M}^{[j]}$ is an order j multilinear map. Intuitively, $\mathcal{M}^p$ is the class of functions that can be well approximated by an order p Taylor expansion around each point. Consequently, the value retrieved from $m_{p,t}$ at a point $\mathbf{q}$ is given by the constant term of the polynomial: $m_p(\mathbf{q}) = \hat{m}_{\mathbf{q}}(\mathbf{q}) = \mathbf{M}^{[0]}$.

3. Choice of minimization algorithm: analytical solution. Solving for the minimum is equivalent to solving for the set of tensors $(\mathbf{M}^{[0]}, \dots, \mathbf{M}^{[p]})$ that minimize the least squares objective. Note that we need to jointly solve for all of $(\mathbf{M}^{[0]}, \dots, \mathbf{M}^{[p]})$ to get the retrieved value $\mathbf{M}^{[0]} = m_{p,t}(\mathbf{q})$.

Let us consider the simplest case of $p = 0$, corresponding to the class of *locally constant functions around $\mathbf{q}$* , also known as Nadaraya-Watson estimators (Fan, 2018; Zhang et al., 2024; Goel and Bartlett, 2024; Tsai et al., 2019). With this simplification, the output of our nonparametric associative memory is

$$m_{0,t}(\mathbf{q}) = \operatorname{argmin}_{\mathbf{M}^{[0]} \in \mathbb{R}^{D_k}} \frac{1}{2} \sum_{i=1}^t s(\mathbf{k}_i, \mathbf{q}) \|\mathbf{v}_i - \mathbf{M}^{[0]}\|_2^2 = \sum_{i=1}^t \frac{s(\mathbf{k}_i, \mathbf{q})}{\sum_{j=1}^t s(\mathbf{k}_j, \mathbf{q})} \mathbf{v}_i. \quad (50)$$

We may be disappointed at first that s needs to be a monotonic function of the *distance* $\|\mathbf{k} - \mathbf{q}\|_2$ to enforce the local approximation property of the estimator in Equation 48; in contrast, softmax attention relies on an exponentiated *inner product*. However, if we normalize keys and queries such that $\|\mathbf{k}\|_2 = \|\mathbf{q}\|_2 = 1$, the exponential smoothing kernel with bandwidth $B = 2\sqrt{D_k}$ is equivalent to the exponential scaled dot product:

$$s_{\exp, 2\sqrt{D_k}}(\mathbf{k}, \mathbf{q}) = \exp\left(\frac{-\|\mathbf{k} - \mathbf{q}\|_2^2}{2\sqrt{D_k}}\right) = \exp\left(\frac{-\|\mathbf{k}\|_2^2 - \|\mathbf{q}\|_2^2 + 2\mathbf{k}^\top \mathbf{q}}{2\sqrt{D_k}}\right) \propto \exp\left(\frac{\mathbf{k}^\top \mathbf{q}}{\sqrt{D_k}}\right), \quad (51)$$

allowing us to exactly derive softmax attention in Equation 50. Moreover, this preprocessing step lines up with an existing technique called query-key normalization (QKNorm), empirically known to stabilize the training of large Transformers (Dehghani et al., 2023; Wortsman et al., 2023) but lacks theoretical justification. Our derivation provides a simple explanation: *softmax attention with QKNorm is a locally zeroth order non-parametric approximation of the true key-value map.*

By deriving self-attention from the same framework as the previous recurrent sequence layers, we can more clearly compare them through the perspective of how they do associative recall. Previous sequence layers that we derived maintained an explicit matrix-valued recurrent state to compress the prefix key-value pairs, relying on this constant-sized matrix to perform retrieval. In contrast, self-attention stores the entire set of prefix key-value pairs as its associative memory system, using all of them for retrieval. As a result, its memory (and expressiveness) grows with the sequence length, a typical property of non-parametric methods (Wang et al., 2019; Wasserman, 2006).

Higher-order attention. Equation 48 suggests a path to higher-order generalizations of self-attention by considering $p > 0$ at the cost of more computation. By reducing Equation 48 to a weighted least squares problem, we see the $p = 1$ case of higher-order softmax self-attention is

$$[\mathbf{M}^{[0]} \quad \mathbf{M}^{[1]}] = (\mathbf{V}_t^\top \mathbf{W}_t \mathbf{X}_t)(\mathbf{X}_t^\top \mathbf{W}_t \mathbf{X}_t)^{-1}, \quad \mathbf{W}_t \triangleq \text{diag}(s(\mathbf{k}_1, \mathbf{q}), \dots, s(\mathbf{k}_t, \mathbf{q})) \quad (52)$$

$$\mathbf{X}_t = \begin{bmatrix} 1 & \dots & 1 \\ \mathbf{k}_1 - \mathbf{q} & \dots & \mathbf{k}_t - \mathbf{q} \end{bmatrix}^\top \quad \mathbf{V}_t = [\mathbf{v}_1 \quad \dots \quad \mathbf{v}_t]^\top, \quad (53)$$

where $\mathbf{W}_t \in \mathbb{R}^{t \times t}$, $\mathbf{X}_t \in \mathbb{R}^{t \times (1+D_k)}$, $\mathbf{M}^{[0]} \in \mathbb{R}^{D_v}$, and $\mathbf{M}^{[1]} \in \mathbb{R}^{D_v \times D_k}$. These equations correspond to the standard prediction equations for local linear regression (Fan, 2018).

An alternate form for the prediction $\mathbf{M}^{[0]}$ that is more intuitive is

$$\mathbf{M}^{[0]} = \bar{\mathbf{v}} - \mathbf{M}^{[1]} \bar{\mathbf{d}}, \quad \bar{\mathbf{v}} \triangleq \sum_{i=1}^t \frac{s(\mathbf{k}_i, \mathbf{q})}{\sum_{j=1}^t s(\mathbf{k}_j, \mathbf{q})} \mathbf{v}_i, \quad \bar{\mathbf{d}} \triangleq \sum_{i=1}^t \frac{s(\mathbf{k}_i, \mathbf{q})}{\sum_{j=1}^t s(\mathbf{k}_j, \mathbf{q})} \mathbf{d}_i, \quad \mathbf{d}_i \triangleq \mathbf{k}_i - \mathbf{q}, \quad (54)$$

resulting in an offset softmax attention as the output: $m_{1,t}(\mathbf{q}) = \mathbf{M}^{[0]} = \bar{\mathbf{v}} - \mathbf{M}^{[1]} \bar{\mathbf{d}}$. This alternate form comes from the fact that if we already know $\mathbf{M}^{[1]}$, then we can solve the $p = 0$ case with an alternate regression problem with preprocessed data $\{(\mathbf{k}_i, \mathbf{v}_i - \mathbf{M}^{[1]} \mathbf{d}_i)\}_{i=1}^t$, resulting in $\mathbf{M}^{[0]}$ being a normalized weighted sum of $\mathbf{v}_i - \mathbf{M}^{[1]} \mathbf{d}_i$. Unlike standard softmax attention (the $p = 0$ case), which only considers the interaction *between keys and queries*, the $p = 1$ case in Equation 52 also accounts for the covariance *between keys* through the inverse term, similar to how linear regression improves upon linear attention from before.

Unfortunately, directly computing Equation 52 for $\mathbf{q} \in \{\mathbf{q}_1, \dots, \mathbf{q}_T\}$ in parallel to fully leverage hardware-acceleration is impractical for all but the smallest sequence problems. Materializing all T prefix set of input tokens $\{\mathbf{X}_t\}_{t=1}^T$ requires $\mathcal{O}(T^2 D_k)$ space, and materializing all T prefix weighted sums of outer products $\{\mathbf{X}_t^\top \mathbf{W}_t \mathbf{X}_t\}_{t=1}^T$ requires $\mathcal{O}(T D_k^2)$ space. Computing $\{\mathbf{X}_t^\top \mathbf{W}_t \mathbf{X}_t\}_{t=1}^T$ requires $\mathcal{O}(T^2 D_k)$ time, and solving a batch of T linear systems requires $\mathcal{O}(T D_k^3)$ time. These challenges make the current form computationally impractical for large-scale sequence modeling. Developing hardware-aware optimizations to compute Equation 52 similar to FlashAttention (Dao et al., 2022) and MesaNet (Oswald et al., 2025) is a promising future direction.

5. Constructing effective key-value pairs for next-token recall

The previous sections focused on the test-time regression model: the function class, the optimization algorithm, and the importance weights. However, the quality of any regressor

depends not only on the model but also on the data it is given. We now turn to an equally important design choice: how to construct the key-value pairs that the test-time regressor memorizes.

Historically, query-key-value sequences were constructed via a linear projection on the corresponding input $\mathbf{x}_t$ at each timestep, e.g. $\mathbf{k}_t = \mathbf{W}_K \mathbf{x}_t$ (Vaswani et al., 2017). Recent work on recurrent models, starting with Fu et al. (2022), identified the importance of performing a “short convolution” in addition to the per-timestep projection:

$$\mathbf{k}_t = \sum_{i=0}^{K-1} w_i \mathbf{W}_K \mathbf{x}_{t-i} \quad (55)$$

where K is the length of the short convolution filter. The short convolution imitates the induction head behavior learned by self-attention layers near the start of the neural network, which allows Transformers to “look back” at previous tokens. Setting $w_i = 1$ and $w_j = 0$ for $j \neq i$ in Equation 55 enables other sequence layers to also “look back” at the token from timestep $t - i$. This is because the short convolution can express the “shift” operation that induction heads learn in standard Transformers (Olsson et al., 2022; von Oswald et al., 2023; Fu et al., 2022).

The short convolution is a crucial component of purely recurrent language models; removing the short convolution results in severe performance drops (Yang et al., 2024b,c,a; Sun et al., 2024). This “look back” ability is crucial to setting up the in-context learning objective (Zhang et al., 2025). Including a short convolution before each self-attention layer can even improve the performance of Transformers, by freeing the earlier self-attention layers from having to learn the induction head behavior (Xu et al., 2024; Olsson et al., 2022). We show below that the short convolution enables the sequence layer to memorize and retrieve bigram-like key-value pairs, which has been shown to be important for language modeling (Arora et al., 2023a).

One short conv is all you need for next-token recall. Consider a standard associative recall task introduced by Arora et al. (2023a), in which a model must be able to recall multiple key-value pairings, known as multi-query associative recall (MQAR). In this task, there are P pairs of unique and consistent cue-response³ tokens $\{(\mathbf{u}_j, \mathbf{v}_j)\}_{j \in [P]}$ with each cue $\mathbf{u}_j$ mapping one-to-one to its response $\mathbf{v}_j$. The model is given a contextual sequence of randomly drawn, possibly repeated, cue-response pairs followed by a previously seen cue: $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{T-1}, \mathbf{x}_T, \mathbf{x}_{T+1}) = (\mathbf{u}_{i_1}, \mathbf{v}_{i_1}, \dots, \mathbf{u}_{i_{T/2}}, \mathbf{v}_{i_{T/2}}, \mathbf{u}_{i_j})$. $\mathbf{u}_{i_j}$ is a cue that appeared earlier (the j th pair) and the model is expected to output the paired response $\mathbf{v}_{i_j}$ at timestep $T+1$. This benchmark abstracts the “Hakuna Matata” example “Hakuna Matata, it means no worries for the rest of your days. Hakuna _____” from Arora et al. (2023a), which we also discussed in Section 1.

We now show that a single test-time regression layer (e.g. as simple as linear attention) with one short convolution is sufficient to solve MQAR, without any parameters other than the embeddings, *as long as it is provided with the appropriate key-value pairs to regress*.

To solve this task, it suffices to test-time memorize all the bigram pairing $\{(\mathbf{u}_j, \mathbf{v}_j)\}$. Given $\mathbf{x}_1, \dots, \mathbf{x}_T$, we causally construct our test-time regression dataset as follows. Let the

3. We use the “cue” and “response” terminology to avoid confusion with “keys” and “values” in the context of test-time regression.

keys be constructed via a short convolution $\mathbf{k}_t = w_0 \mathbf{x}_t + w_1 \mathbf{x}_{t-1} = \mathbf{x}_{t-1}$ where $w_0 = 0, w_1 = 1$. Let the queries and values be the same as the inputs: $\mathbf{q}_t = \mathbf{v}_t = \mathbf{x}_t$. In other words, our AR sequence layer will first memorize $\{(\mathbf{k}_t, \mathbf{v}_t)\}_{t=1}^{T+1} = \{(\mathbf{x}_{t-1}, \mathbf{x}_t)\}_{t=1}^{T+1}$ into its associative memory map m_{T+1} . Then we retrieve from m_{T+1} using $\mathbf{q}_{T+1} = \mathbf{x}_{T+1} = \mathbf{u}_{i_j}$, which has previously appeared in the sequence of cues as the i_j th bigram. The output is then

$$\mathbf{y}_{T+1} = m_{T+1}(\mathbf{q}_{T+1}) = m_{T+1}(\mathbf{u}_{i_j}). \quad (56)$$

For simplicity, suppose m_{T+1} is implemented via linear attention, following Equation 2. Then

$$\mathbf{y}_{T+1} = m_{T+1}(\mathbf{u}_{i_j}) = \left(\sum_{t=1}^{T+1} \mathbf{v}_t \mathbf{k}_t^\top \right) \mathbf{u}_{i_j} = \sum_{t=1}^{T+1} \mathbf{x}_t \mathbf{x}_{t-1}^\top \mathbf{u}_{i_j}. \quad (57)$$

When the embedding space is large enough, we can set the embeddings such that all tokens are orthonormal. Then $\mathbf{x}_{t-1}^\top \mathbf{u}_{i_j}$ is nonzero only when $\mathbf{x}_{t-1} = \mathbf{u}_{i_j}$, simplifying the output to a sum of only tokens that are followed by $\mathbf{u}_{i_j}$. Since the cue-response map is one-to-one, the remaining tokens are all $\mathbf{v}_{i_j}$, producing the output $\mathbf{y}_{T+1} \propto \mathbf{v}_{i_j}$, solving the MQAR task.

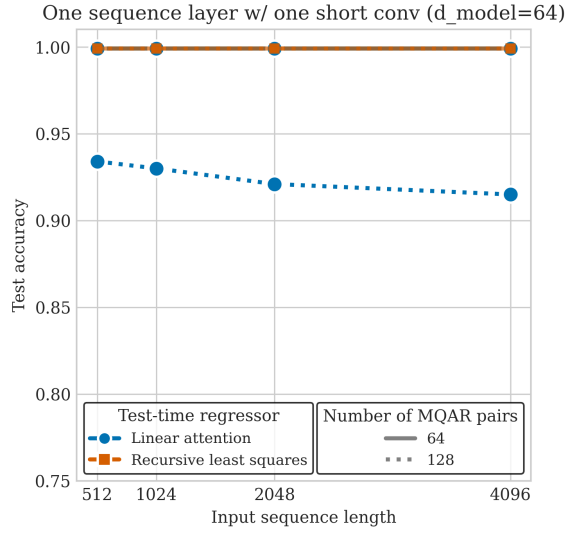


Figure 2: A single test-time regression layer with one short convolution suffices to solve MQAR with $P = 64$ cue-response pairs. When a model has sufficient capacity to memorize all P pairs, it can solve MQAR *regardless* of the sequence length. The regression problem is stationary (the cue-response map doesn’t change over time), hence no forgetting mechanism is needed to solve the task.

Memory capacity, not sequence length, limits model performance. Typical evaluations of models on the MQAR task look at model performance with respect to the length of the sequence, possibly varying the model capacity. A few examples include Figure 2 of

Arora et al. (2023a), Figure 8 of Dao and Gu (2024), Figure 2 of Yang et al. (2024c), and likely others. However, our above construction shows that the sequence length doesn’t matter for memorizers with a large enough memory capacity; once the memory is large enough, the difficulty of MQAR is *independent* of the sequence length T , which may be counter to the intuitive notion of longer sequences being more difficult. Instead, models are *limited by their ability to fully memorize all P key-value pairs*.

Although our construction in Equation 57 relied on the embedding space being large enough to allow orthonormality, trained models can still succeed without perfectly orthonormal embeddings, which we empirically show in Figure 2. We train on the MQAR task following the procedure of (Arora et al., 2023a). However, *we use only one test-time regression layer and one length-2 convolution* (for constructing the keys), following our construction. We also remove the typical MLP block following the sequence layer, since our construction shows that additional nonlinearities are unnecessary for this synthetic associative recall task. We fix our key dimensions D_k to be the same as the embedding size, $d_{\text{model}} = 64$. We compare two test-time regression architectures, linear attention and unweighted recursive least squares (RLS) of Equation 11. We do not use gated variants of linear attention since this task does not require any forgetting; the cue-response mapping is unchanged throughout the sequence. As shown by our previous construction, even linear attention is able to solve the task perfectly when the appropriate keys are constructed. In the case where $d_{\text{model}} = P = 64$, linear attention solves MQAR perfectly, independent of sequence length as predicted. Once the number of keys increases to $P = 128$, linear attention can no longer perfectly solve MQAR, since it is impossible for 128 keys to be orthogonal in a $d_{\text{model}} = 64$ dimensional space. In contrast, an unweighted RLS layer (Equation 11) improves upon linear attention by accounting for the key covariance matrices $\{\mathbf{K}_t^\top \mathbf{K}_t\}$, allowing it to better test-time memorize the cue-response map, even when $P > d_{\text{model}}$. As a result, RLS is able to solve MQAR regardless of the sequence length.

6. Experiments

6.1 Online regression with sequence layers

We now empirically illustrate that the sequence layers discussed so far do in fact perform regression implicitly via a single forward pass. To investigate the impact of feature correlations and non-stationarity, we utilize two distinct data generation settings with $D_k = 64$ and $T = 256$.

Setting 1: Switching process with independent key dimensions. We generate a sequence of keys according to a non-stationary VAR(1) process that switches dynamics at $t = T/2$. We construct two transition matrices, $\mathbf{A}_1$ and $\mathbf{A}_2$, with maximum singular values of 0.9 (fast decay) and 0.999 (slow decay) respectively. The system evolves as:

$$\mathbf{k}_{t+1} = \mathbf{A}_t \mathbf{k}_t + \boldsymbol{\varepsilon}_t, \quad \text{where } \mathbf{A}_t = \begin{cases} \mathbf{A}_1 & t < T/2 \\ \mathbf{A}_2 & t \geq T/2 \end{cases} \quad (58)$$

In both regimes, the noise is drawn from $\boldsymbol{\varepsilon}_t \sim \text{Normal}(0, \mathbf{I} - \mathbf{A}_t \mathbf{A}_t^\top)$. By construction, the noise covariance instantaneously adapts to correlations introduced in $\mathbf{A}_t$, ensuring that $\mathbf{K}_t^\top \mathbf{K}_t \approx \mathbf{I}$ throughout the entire sequence.

Setting 2: Switching process with correlated key dimensions. We generate keys according to a similar switching autoregressive process, but with fixed, correlated noise. We define scalar decay rates $\lambda_1 = 0.9$ (fast decay) and $\lambda_2 = 0.999$ (slow decay), and set the transition matrices as scaled identities $\mathbf{A}_1 = \lambda_1 \mathbf{I}$ and $\mathbf{A}_2 = \lambda_2 \mathbf{I}$. The process evolves as:

$$\mathbf{k}_{t+1} = \mathbf{A}_t \mathbf{k}_t + 0.02 \boldsymbol{\varepsilon}_t, \quad \text{where } \mathbf{A}_t = \begin{cases} \mathbf{A}_1 & t < T/2 \\ \mathbf{A}_2 & t \geq T/2 \end{cases} \quad (59)$$

Here, the noise is drawn from $\boldsymbol{\varepsilon}_t \sim \text{Normal}(0, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is constructed to have unit diagonals and non-zero off-diagonals. Unlike Setting 1, the fixed noise covariance $\boldsymbol{\Sigma}$ creates instantaneous correlation between the dimensions of the keys such that $\mathbf{K}_t^\top \mathbf{K}_t \neq \mathbf{I}$.

Results. In both settings, we set the value vectors to be $\mathbf{v}_t = \mathbf{k}_{t+1} / \|\mathbf{k}_{t+1}\|_2$, such that the dataset $\{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^T$ imitates a next-token prediction task. We make one-step-ahead predictions by setting $\mathbf{q}_t = \mathbf{k}_{t+1}$ so that $\mathbf{y}_t = m_t(\mathbf{q}_t) = m_t(\mathbf{k}_{t+1})$ is the test-time regressor’s out-of-sample prediction of the next timestep, conditioned on the prefix $\{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^t$. To solve this online regression problem, we simply compute a single forward pass through each sequence layer.

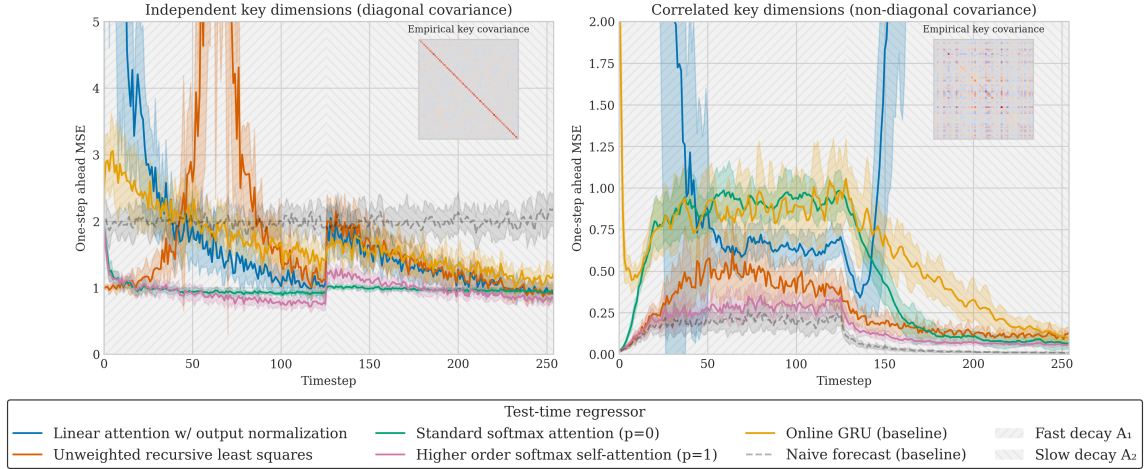


Figure 3: Solving two non-stationary online regression tasks *without any learnable parameters*. The regression-memory correspondence enables us to perform regression via a single forward pass of a test-time regression layer. The inputs change more rapidly in initial $T/2$ timesteps, making it harder for regressors to learn. The latter $T/2$ timesteps are more stable; however, without decaying weights, linear attention and unweighted RLS are unable to adjust to the resulting non-stationary data. We average over 10 random seeds with shaded region representing 1 standard deviation.

We evaluate layers derived both from parametric regression and non-parametric regression. For our parametric regression layers we choose linear attention (Equation 2) and

recursive least squares (Equation 11) to show the effects of accounting for correlations between keys. For our non-parametric regression layers, we chose softmax attention (Equation 1) and our $p = 1$ generalization of softmax attention based on local linear regression (Equation 52). We also show two baselines. One baseline is the naive one-step forecaster baseline where we simply use $\mathbf{v}_{t-1}$ as the prediction given $\mathbf{k}_t$. The other baseline is an online-learning single-layer GRU that takes a gradient step after seeing each example. We use hidden dimension 64 and 3×10^{-3} for the learning rate.

We plot the one-step out-of-sample test loss $\|\mathbf{v}_{t+1} - m_t(\mathbf{k}_{t+1})\|_2^2$ in Figure 3. During the initial $T/2$ timesteps, the dynamical system has a faster decay time with more rapid changes, making it less predictable. In the latter $T/2$ timesteps, the system slows down and is more predictable.

In Setting 1, we see that unweighted recursive least squares performs comparably to linear attention after sufficient training examples, since $\mathbf{K}_t^\top \mathbf{K}_t \approx \mathbf{I}$. Interestingly, there is a double-descent phenomenon for recursive least squares as the system becomes ill conditioned when the number of training points approaches the input dimension D_k . Both softmax attention ($p = 0$) and higher order attention ($p = 1$) perform well at this task, beating even the GRU baseline. Higher order attention outperforms standard softmax attention after sufficient timesteps.

In Setting 2, we see a larger separation between linear attention and recursive least squares. As expected, linear attention performs the worst since $\mathbf{K}_t^\top \mathbf{K}_t \neq \mathbf{I}$ in our dynamical system⁴. Still, unweighted recursive least squares is unable to adapt to the abrupt switching dynamics since it cannot discount older datapoints; as a result, it maintains a high error throughout the latter timesteps. In contrast, the nonparametric regressors are able to better adapt after the transition since earlier keys are dissimilar to the later keys, eventually beating recursive least squares. Interestingly, increasing our nonparametric softmax regressor from $p = 0$ to $p = 1$ allows the model to better adapt to both the initial fast-changing period and the sudden regime change. Here we also see that the $p = 1$ higher-order attention performs the best among the layers shown.

6.2 Language modeling with higher-order attention

The synthetic experiments above showed that the $p = 1$ higher-order attention derived in Equation 52 improves upon standard softmax attention at online regression. A natural question is whether this advantage persists on real-world data. To test this, we compare $p = 1$ higher-order attention to standard softmax attention ($p = 0$) on language modeling, focusing on how the two methods respond to the bandwidth (temperature) of the smoothing kernel.

Setup. We use a single-layer transformer with one attention head and one MLP block, deliberately keeping the architecture minimal to isolate the effect of the attention mechanism from confounders such as depth and inter-layer interactions. We include a “look-back” operation after the key projection, equivalent to a fixed size short-convolution $(w_0, w_1) = (0, 1)$, which provides the induction head capability needed for next-token prediction, as

4. We normalized the predictions of linear attention by t when predicting $\mathbf{y}_t$ to prevent the output from scaling linearly with the sequence length. In a previous version of this paper, we used L2-normalization instead, but it underrepresented how poor the predictions were.

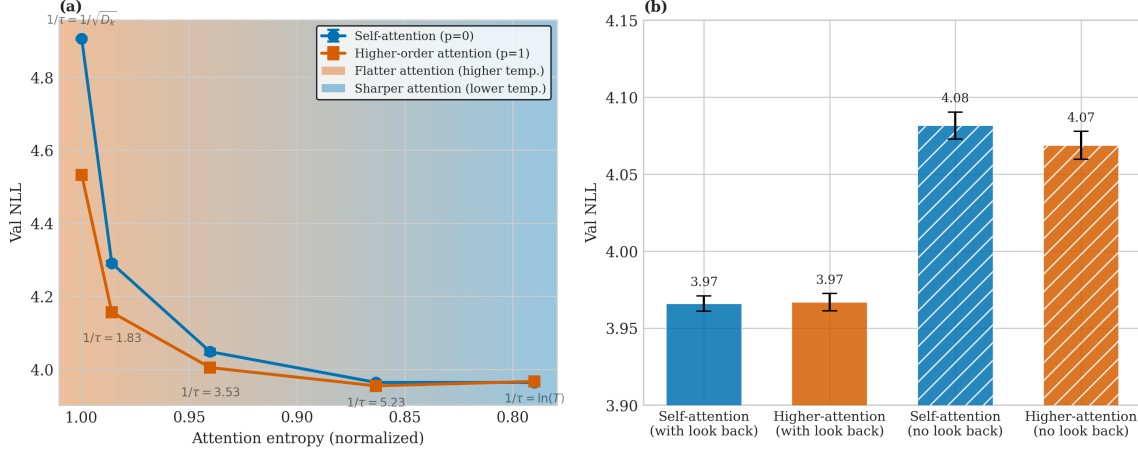


Figure 4: Validation NLL (over 3 seeds) on Wikitext-103. (a) We plot NLL as a function of attention entropy, controlled by the temperature τ . The correction term from higher-order attention ($p = 1$) helps when the temperature is suboptimal and the attention distribution is too flat. It outperforms standard softmax attention ($p = 0$) in the flat attention regime (left), and matches it in the sharp attention regime (right). This is consistent with the known bias reduction properties of local linear regression over the Nadaraya-Watson estimator in nonparametric statistics (Fan, 2018). (b) Ablation of the “look-back” short convolution on the keys at $1/\tau = \ln(T)$. Removing the short convolution removes the mechanism for next-token recall as discussed in Section 5, degrading both methods equally.

discussed in Section 5 (Olsson et al., 2022). Following our interpretation in Section 4 of attention as a local polynomial regressor, we normalize queries and keys to unit norm (QKNorm) before computing attention scores. We train on Wikitext-103 (Merity et al., 2016) for 1600 gradient steps with batch size 64 and sequence length $T = 1024$, totaling 104M tokens. We use AdamW (Loshchilov and Hutter, 2019) with max learning rate 2×10^{-3} , $(\beta_1, \beta_2) = (0.9, 0.95)$, 160 warmup steps, and cosine annealing to 1/10 the max learning rate.

Temperature sensitivity. The temperature parameter τ in softmax attention plays the role of the bandwidth B of the smoothing kernel, controlling the effective neighborhood size around each query. The choice of temperature is important since it directly controls the sharpness of the attention distribution. Past work has observed that sharper attention maps correlate with better (Ye et al., 2024). In classic nonparametric statistics, local linear regression ($p = 1$) is known to have smaller asymptotic bias than the Nadaraya-Watson estimator ($p = 0$), particularly when the bandwidth is large (Fan, 2018). This suggests that higher-order attention should be more robust to suboptimal temperature choices than standard softmax attention.

To test this prediction, we train separate models across a range of inverse temperature scales $1/\tau$, from the standard $1/\sqrt{D_k}$ of Vaswani et al. (2017) up to $\ln(T)$, a scaling mo-

tivated by recent works (Chiang and Cholak, 2022; Bai et al., 2023; Peng et al., 2026). We report the entropy of the resulting attention distributions on the x-axis of Figure 4, where lower entropy corresponds to sharper, more peaked attention maps. We compute the entropy of the attention map at each query of each attention head. We then normalize the entropy of each query by $\log(1 + t)$, the maximum entropy achievable by a distribution over t keys. Finally, we average the normalized entropy over all the queries and heads and sequences in the validation set. Notably, the observation by Chiang and Cholak (2022) that the optimal temperature should scale logarithmically with the sequence length is consistent with classic results in nonparametric statistics, where the optimal bandwidth should decrease as the number of data points increases (Fan, 2018).

Results. Figure 4(a) shows the validation negative log-likelihood (NLL) of both methods, averaged over three seeds, with very tight standard deviations that overlap with the plot markers. In the flat attention regime (high entropy, large bandwidth), $p = 1$ higher-order attention substantially outperforms standard softmax attention; for instance, at the default $1/\tau = 1/\sqrt{D_k}$ scaling, the gap is approximately 0.35 nats. As the temperature decreases and the attention maps become sharper, both methods converge to similar performance. This pattern aligns with the known statistical properties of local polynomial regression: the bias correction from the $p = 1$ term matters most when the bandwidth is large relative to the local structure of the data, and becomes negligible when the bandwidth is already small enough to resolve local variation (Fan, 2018).

We also ablate the “look back” mechanism, since theory suggests that it is important to overall performance. Figure 4(b) shows that removing it degrades both $p = 0$ and $p = 1$ equally at the $1/\tau = \ln(T)$ scaling, confirming that the short convolution provides the induction-head capability needed for next-token recall (Section 5), while the benefit of higher-order attention is orthogonal.

Scaling up to larger models and longer sequences via hardware-efficient implementations of higher-order attention is a promising direction for future work.

7. Related works

Associative memory and neural networks. Designing neural networks motivated by associative memory has a long history, dating back to Willshaw et al. (1969), Willshaw (1989), Kohonen (1972), and Hinton and Anderson (1989). These early works designed parametric memory that could explicitly store patterns (e.g. images) and recall them based on a partial or noisy input. The dominant paradigm for studying these systems has been the energy-based approach, pioneered by the Hopfield network (Hopfield, 1982), which cast associative recall as energy minimization over a quadratic energy function: patterns are stored as local minima via Hebbian learning (outer-product storage $\mathbf{M} = \sum_i \mathbf{v}_i \mathbf{k}_i^\top$) and retrieval proceeds by initializing the network with a noisy cue and letting the dynamics converge to the nearest attractor. However, a fundamental limitation of the classical Hopfield network is that its storage capacity scales only linearly with the number of neurons (Hopfield, 1982). Subsequent works broke this barrier using higher-order (Krotov and Hopfield, 2016) and exponential (Ramsauer et al., 2021) interaction functions, with the latter showing that the modern Hopfield update rule is equivalent to softmax attention. Through our framework, we arrive at the same softmax attention from a complementary direction:

as a locally constant nonparametric regressor in Section 4, where the Boltzmann weights from energy minimization coincide with the normalized kernel weights in local constant regression. While the energy-based approach derives retrieval rules from gradient dynamics of an energy function, our regression-based approach derives them constructively from a memorization objective, directly providing a recipe for designing new sequence layers. Complementary to the test-time associative memory studied in our work, even the feedforward layers of an MLP can be seen as a form of persistent *training-time* associative memory, a perspective studied by recent works and memory-inspired architectures (Sukhbaatar et al., 2019; Zhang et al., 2025; Bietti et al., 2023; Nichani et al., 2024; Cabannes et al., 2023; Behrouz et al., 2024).

Nadaraya-Watson estimator and neural networks A parallel historical thread connects our framework to the famous Nadaraya-Watson (NW) estimator, introduced independently by Nadaraya (1964) and Watson (1964). Structurally, the NW estimator is already an attention mechanism: data points serve as “keys”, outputs serve as “values”, and the input x retrieves a similarity-weighted combination, predating Vaswani et al. (2017) by over fifty years. This connection was operationalized through a series of neural network architectures over the decades. The General Regression Neural Network of Specht (1991) implemented NW estimation as a feedforward pass, radial basis function networks and kernel methods further generalized the similarity computation (Schölkopf and Smola, 2001), and Matching Networks of Vinyals et al. (2016) replaced the fixed kernel with a learned deep embedding for few-shot learning.’

In-context learning. A growing body of work has shown that Transformers can learn linear functions in-context by implicitly performing gradient descent *across layers*, where each layer updates the iterate once (Oswald et al., 2023; Ahn et al., 2023; Garg et al., 2022; Zhang et al., 2023b; Ding et al., 2024). This emergent behavior is sometimes called mesa-optimization (Hubinger et al., 2021). von Oswald et al. (2023) operationalized this insight into the Mesa-layer, which computes the analytical least-squares solution *within* a single layer’s forward pass, rather than optimizing *across* layers, arriving at the same RLS equations as our Equation 11 but from a different motivation. Our framework generalizes this connection: where the Mesa-layer focuses on one specific regression method (analytical OLS), we use regression as a broader design principle. As we showed in Equation 24, standard linear attention implicitly performs one step of batch gradient descent on the least-squares objective. Recurrent layers like DeltaNet go further, performing t steps of stochastic gradient descent *within* the layer (Equation 26). By varying the regression weights, function class, and optimizer, our framework derives a wide range of architectures all connected through the lens of associative memory. These two levels of optimization, within-layer (architectural) and across-layer (learned), provide complementary perspectives on the internal dynamics of sequence models.

8. Conclusion and discussion

We have presented a unifying framework that derives sequence models that can explicitly perform associative recall via test-time regression. Each architecture emerges naturally by specifying three aspects: the relative importance of each association, the function class,

and the optimization algorithm. Through this lens, we derived linear attention and its variants as approximations to the linear least squares solution, online learning layers and fast-weight layers as first-order least-squares solvers, and softmax attention as a local constant regressor. This perspective provided principled explanations for architectural choices like QKNorm and derived higher-order generalizations of attention that go beyond pairwise interactions. We also demonstrated that for multi-query associative recall (MQAR), a single short convolution and a sequence layer suffices, in contrast to typical approaches that use multiple layers and unnecessary feedforward blocks (Arora et al., 2023a).

Our paper focuses specifically on sequence architectures that use query-key-value patterns for associative recall. Several important architectural families lie outside this scope, including structured state-space models with more general structured masks (Dao and Gu, 2024), convolutional architectures (Shi et al., 2023; Poli et al., 2023; Gehring et al., 2017; Gu et al., 2021; Bradbury et al., 2017; Smith et al., 2023; Hochreiter and Schmidhuber, 1997), and more novel computational patterns (Qin et al., 2023; Ren et al., 2023). Moreover, complementary research on model backbones and initialization continues to yield important advances and performance improvements (Gu and Dao, 2024; Dao and Gu, 2024; Orvieto et al., 2023; Mehta et al., 2023; Hua et al., 2022).

Our framework opens rich directions for future research by connecting to the extensive literature on regression and optimization. While initial successes like TTT-MLP (Sun et al., 2024) hint at the potential of neural test-time regressors, the space of nonlinear neural regression models remains largely unexplored. Recent work integrating weight decay and momentum (Behrouz et al., 2024; Yang et al., 2024a) suggests opportunities for developing more effective, stable, and parallelizable test-time optimizers. Concurrent work by Behrouz et al. (2024) also suggests opportunities to more creatively utilize associative memory. The practical success of these approaches will depend heavily on efficient hardware implementations (Hua et al., 2022; Yang et al., 2024c; Sun et al., 2024; Dao and Gu, 2024; Dao et al., 2022; Oswald et al., 2025), aligning with established compute-scaling principles (Sutton, 2019; Hoffmann et al., 2022; Kaplan et al., 2020).

In the words of Kohonen (1989), “associative memory is a very delicate and complex concept which often has been attributed to the higher cognitive processes, especially those taking place in the human brain”. We speculate that test-time associative memory, along with test-time compute more broadly (Sun et al., 2024, 2020; Akyürek et al., 2024), will be fundamental to developing truly adaptive models that can update and learn in changing environments (OpenAI, 2024).

Acknowledgements. This work was supported in part by AFOSR Grant FA9550-21-1-0397, ONR Grant N00014-22-1-2110, NSF Grant 2205084, the Stanford Institute for Human-Centered Artificial Intelligence (HAI). EBF is a Chan Zuckerberg Biohub – San Francisco Investigator.

References

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36:45614–45650, December 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8ed3d610ea4b68e7afb30ea7d01422c6-Abstract-Conference.html.

- Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, December 2023. URL <https://openreview.net/forum?id=hm0w0ZWzYE>.
- Yaroslav Aksenov, Nikita Balagansky, Sofia Maria Lo Cicero Vaina, Boris Shaposhnikov, Alexey Gorbatsovski, and Daniil Gavrilov. Linear Transformers with Learnable Kernel Functions are Better In-Context Models, February 2024. URL <http://arxiv.org/abs/2402.10644>. arXiv:2402.10644 [cs] version: 1.
- Ekin Akyürek, Mehul Damani, Linlu Qiu, Han Guo, Yoon Kim, and Jacob Andreas. The Surprising Effectiveness of Test-Time Training for Abstract Reasoning, November 2024. URL <http://arxiv.org/abs/2411.07279>. arXiv:2411.07279 [cs] version: 1.
- Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Bernie Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL <https://openreview.net/forum?id=gerNCVqqtR>.
- Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. Zoology: Measuring and Improving Recall in Efficient Language Models. In *The Twelfth International Conference on Learning Representations*, October 2023a. URL <https://openreview.net/forum?id=LY3ukUANKo>.
- Simran Arora, Brandon Yang, Sabri Eyuboglu, Avani Narayan, Andrew Hojel, Immanuel Trummer, and Christopher Ré. Language Models Enable Simple Systems for Generating Structured Views of Heterogeneous Data Lakes, April 2023b. URL <http://arxiv.org/abs/2304.09433>. arXiv:2304.09433 [cs].
- Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Re. Simple linear attention language models balance the recall-throughput tradeoff. In *Proceedings of the 41st International Conference on Machine Learning*, pages 1763–1840. PMLR, July 2024. URL <https://proceedings.mlr.press/v235/arora24a.html>.
- Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen Technical Report, September 2023. URL <http://arxiv.org/abs/2309.16609>. arXiv:2309.16609 [cs].
- Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended Long Short-Term Memory, May 2024. URL <http://arxiv.org/abs/2405.04517>. arXiv:2405.04517 [cs, stat].
- Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to Memorize at Test Time, December 2024. URL <http://arxiv.org/abs/2501.00663>. arXiv:2501.00663 [cs].

- Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is Space-Time Attention All You Need for Video Understanding? In *Proceedings of the 38th International Conference on Machine Learning*, pages 813–824. PMLR, July 2021. URL <https://proceedings.mlr.press/v139/bertasius21a.html>.
- Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a Transformer: A Memory Viewpoint. In *Thirty-seventh Conference on Neural Information Processing Systems*, November 2023. URL <https://openreview.net/forum?id=3X2EbBLNsk>.
- Stephen Boyd and Lieven Vandenbergh. *Convex optimization*. Cambridge University Press, Cambridge, 2004.
- James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. Quasi-recurrent neural networks. In *International conference on learning representations*, 2017. URL <https://openreview.net/forum?id=H1zJ-v5x1>.
- 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. URL <http://arxiv.org/abs/2005.14165>. arXiv:2005.14165 [cs].
- Vivien Cabannes, Elvis Dohmatob, and Alberto Bietti. Scaling Laws for Associative Memories. In *The Twelfth International Conference on Learning Representations*, October 2023. URL <https://openreview.net/forum?id=Tzh6xAJS11>.
- Fabiano T. Castoldi and Marcello L. R. de Campos. Minimum-disturbance description for the development of adaptation algorithms and a new leakage least squares algorithm. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3129–3132, April 2009. doi: 10.1109/ICASSP.2009.4960287. URL <https://ieeexplore.ieee.org/document/4960287>.
- N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, September 2004. ISSN 1557-9654. doi: 10.1109/TIT.2004.833339. URL <https://ieeexplore.ieee.org/document/1327806/?arnumber=1327806>. Conference Name: IEEE Transactions on Information Theory.
- Hanting Chen, Liuzhicheng, Xutao Wang, Yuchuan Tian, and Yunhe Wang. DiJiang: Efficient Large Language Models through Compact Kernelization. In *Forty-first International Conference on Machine Learning*, June 2024. URL [https://openreview.net/forum?id=OuUHfhXdnH&referrer=%5Bthe%20profile%20of%20Yuchuan%20Tian%5D\(%2Fprofile%3Fid%3D~Yuchuan_Tian1\)](https://openreview.net/forum?id=OuUHfhXdnH&referrer=%5Bthe%20profile%20of%20Yuchuan%20Tian%5D(%2Fprofile%3Fid%3D~Yuchuan_Tian1)).
- Yifan Chen, Qi Zeng, Heng Ji, and Yun Yang. Skyformer: remodel self-attention with Gaussian kernel and nyström method. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS ’21, pages 2122–2135, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 978-1-7138-4539-3.
- David Chiang and Peter Cholak. Overcoming a Theoretical Limitation of Self-Attention. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7654–7664, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.527. URL <https://aclanthology.org/2022.acl-long.527/>.

- 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. URL <https://openreview.net/forum?id=Ua6zuk0WRH>.
- Kevin Clark, Kelvin Guu, Ming-Wei Chang, Panupong Pasupat, Geoffrey Hinton, and Mohammad Norouzi. Meta-Learning Fast Weight Language Models. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9751–9757, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.661. URL <https://aclanthology.org/2022.emnlp-main.661>.
- Tri Dao and Albert Gu. Transformers are SSMS: Generalized Models and Efficient Algorithms Through Structured State Space Duality. In *Forty-first International Conference on Machine Learning*, June 2024.
- Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Re. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems*, October 2022. URL <https://openreview.net/forum?id=H4DqfPSibmx>.
- Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models, February 2024. URL <http://arxiv.org/abs/2402.19427>. arXiv:2402.19427 [cs].
- Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, Rodolphe Jenatton, Lucas Beyer, Michael Tschannen, Anurag Arnab, Xiao Wang, Carlos Riquelme Ruiz, Matthias Minderer, Joan Puigcerver, Utku Evci, Manoj Kumar, Sjoerd Van Steenkiste, Gamaleldin Fathy Elsayed, Aravindh Mahendran, Fisher Yu, Avital Oliver, Fantine Huot, Jasmijn Bastings, Mark Collier, Alexey A. Gritsenko, Vighnesh Birodkar, Cristina Nader Vasconcelos, Yi Tay, Thomas Mensink, Alexander Kolesnikov, Filip Pavetic, Dustin Tran, Thomas Kipf, Mario Lucic, Xiaohua Zhai, Daniel Keysers, Jeremiah J. Harmsen, and Neil Houlsby. Scaling Vision Transformers to 22 Billion Parameters. In *Proceedings of the 40th International Conference on Machine Learning*, pages 7480–7512. PMLR, July 2023. URL <https://proceedings.mlr.press/v202/dehghani23a.html>.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL <https://aclanthology.org/N19-1423/>.
- Nan Ding, Tomer Levinboim, Jialin Wu, Sebastian Goodman, and Radu Soricut. CausalLM is not optimal for in-context learning. In *The twelfth international conference on learning representations*, 2024. URL <https://openreview.net/forum?id=guRNebwZBb>.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recogni-

- tion at scale. In *International conference on learning representations*, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
- John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL <http://jmlr.org/papers/v12/duchi11a.html>.
- Jianqing Fan. *Local Polynomial Modelling and Its Applications: Monographs on Statistics and Applied Probability 66*. Routledge, New York, May 2018. ISBN 978-0-203-74872-5. doi: 10.1201/9780203748725.
- Daniel Y. Fu, Tri Dao, Khaled Kamal Saab, Armin W. Thomas, Atri Rudra, and Christopher Re. Hungry Hungry Hippos: Towards Language Modeling with State Space Models. In *The Eleventh International Conference on Learning Representations*, September 2022. URL <https://openreview.net/forum?id=COZDyOWYGg>.
- Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? A case study of simple function classes. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in neural information processing systems*, 2022. URL <https://openreview.net/forum?id=f1NZJ2e0et>.
- Marta Garnelo and Wojciech Marian Czarnecki. Exploring the Space of Key-Value-Query Models with Intention, May 2023. URL <http://arxiv.org/abs/2305.10203>. arXiv:2305.10203.
- Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional Sequence to Sequence Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1243–1252. PMLR, July 2017. URL <https://proceedings.mlr.press/v70/gehring17a.html>.
- Gautam Goel and Peter Bartlett. Can a transformer represent a Kalman filter? In Alessandro Abate, Mark Cannon, Kostas Margellos, and Antonis Papachristodoulou, editors, *Proceedings of the 6th annual learning for dynamics & control conference*, volume 242 of *Proceedings of machine learning research*, pages 1502–1512. PMLR, July 2024. URL <https://proceedings.mlr.press/v242/goel24a.html>.
- Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, October 2017. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2016.2582924. URL <http://arxiv.org/abs/1503.04069>. arXiv:1503.04069 [cs].
- Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G. Wilson. Large Language Models Are Zero-Shot Time Series Forecasters. *Advances in Neural Information Processing Systems*, 36:19622–19635, December 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/3eb7ca52e8207697361b2c0fb3926511-Abstract-Conference.html.
- Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In *First Conference on Language Modeling*, August 2024. URL <https://openreview.net/forum?id=tEYskw1VY2#discussion>.
- Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent Memory with Optimal Polynomial Projections. In *Advances in Neural Information Processing Systems*, volume 33, pages 1474–1487. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/102f0bb6efb3a6128a3c750dd16729be-Abstract.html>.

- Albert Gu, Karan Goel, and Christopher Re. Efficiently Modeling Long Sequences with Structured State Spaces. In *International Conference on Learning Representations*, October 2021. URL <https://openreview.net/forum?id=uYLFoz1v1AC>.
- Albert Gu, Ankit Gupta, Karan Goel, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, pages 35971–35983, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 978-1-7138-7108-8.
- Dongchen Han, Ziyi Wang, Zhuofan Xia, Yizeng Han, Yifan Pu, Chunjiang Ge, Jun Song, Shiji Song, Bo Zheng, and Gao Huang. Demystify Mamba in Vision: A Linear Attention Perspective, May 2024. URL <https://arxiv.org/abs/2405.16605v1>.
- Simon S. Haykin. *Adaptive Filter Theory*. Pearson, 2014. ISBN 978-0-13-267145-3. Google-Books-ID: J4GRKQEACAAJ.
- Geoffrey E. Hinton and James A. Anderson. *Parallel Models of Associative Memory*. Psychology Press, 1989.
- Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL <https://ieeexplore.ieee.org/abstract/document/6795963>. Conference Name: Neural Computation.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training Compute-Optimal Large Language Models, March 2022. URL <http://arxiv.org/abs/2203.15556>. arXiv:2203.15556 [cs].
- J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, April 1982. doi: 10.1073/pnas.79.8.2554. URL <https://www.pnas.org/doi/abs/10.1073/pnas.79.8.2554>.
- Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer Quality in Linear Time. In *Proceedings of the 39th International Conference on Machine Learning*, pages 9099–9117. PMLR, June 2022. URL <https://proceedings.mlr.press/v162/hua22a.html>.
- Evan Hubinger, Chris van Merwijk, Vladimir Mikulik, Joar Skalse, and Scott Garrabrant. Risks from Learned Optimization in Advanced Machine Learning Systems, December 2021. URL <http://arxiv.org/abs/1906.01820>. arXiv:1906.01820 [cs].
- Richard M. Johnstone, C. Richard Johnson, Robert R. Bitmead, and Brain D. O. Anderson. Exponential convergence of recursive least squares with exponential forgetting factor. In *1982 21st IEEE Conference on Decision and Control*, pages 994–997, December 1982. doi: 10.1109/CDC.1982.268295. URL <https://ieeexplore.ieee.org/document/4047398>.
- John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Židek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold.

- Nature*, 596(7873):583–589, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2. URL <https://www.nature.com/articles/s41586-021-03819-2>.
- S. Kaczmarz. Angenaherte auflösung von systemen linearer gleichungen. *Bull. Int. Acad. Pol. Sic. Let., Cl. Sci. Math. Nat.*, pages 355–357, 1937. URL <https://cir.nii.ac.jp/crid/1570009749812709888>.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models, January 2020. URL <http://arxiv.org/abs/2001.08361>. arXiv:2001.08361 [cs].
- Jungo Kasai, Hao Peng, Yizhe Zhang, Dani Yogatama, Gabriel Ilharco, Nikolaos Pappas, Yi Mao, Weizhu Chen, and Noah A. Smith. Finetuning Pretrained Transformers into RNNs. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10630–10643, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.830. URL <https://aclanthology.org/2021.emnlp-main.830>.
- Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5156–5165. PMLR, November 2020. URL <https://proceedings.mlr.press/v119/katharopoulos20a.html>.
- Tobias Katsch. GateLoop: Fully Data-Controlled Linear Recurrence for Sequence Modeling, January 2024. URL <http://arxiv.org/abs/2311.01927>. arXiv:2311.01927 [cs].
- Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. 2015. URL <https://arxiv.org/abs/1412.6980>.
- Teuvo Kohonen. Correlation Matrix Memories. *IEEE Transactions on Computers*, C-21(4):353–359, April 1972. ISSN 1557-9956. doi: 10.1109/TC.1972.5008975. URL <https://ieeexplore.ieee.org/abstract/document/5008975>. Conference Name: IEEE Transactions on Computers.
- Teuvo Kohonen. *Self-Organization and Associative Memory*, volume 8 of *Springer Series in Information Sciences*. Springer, Berlin, Heidelberg, 1989. ISBN 978-3-540-51387-2 978-3-642-88163-3. doi: 10.1007/978-3-642-88163-3. URL <http://link.springer.com/10.1007/978-3-642-88163-3>.
- Dmitry Krotov and John J. Hopfield. Dense Associative Memory for Pattern Recognition. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/eaee339c4d89fc102edd9dbdb6a28915-Abstract.html.
- Bo Liu, Rui Wang, Lemeng Wu, Yihao Feng, Peter Stone, and Qiang Liu. Longhorn: State Space Models are Amortized Online Learners, October 2024. URL <http://arxiv.org/abs/2407.14207>. arXiv:2407.14207.
- Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International conference on learning representations*, 2019. URL <https://openreview.net/forum?id=Bkg6RiCqY7>.
- Jiachen Lu, Jinghan Yao, Junge Zhang, Xiatian Zhu, Hang Xu, Weiguo Gao, Chunjing XU, Tao Xiang, and Li Zhang. SOFT: Softmax-free Transformer with Linear Complexity. In *Advances in Neural Information Processing Systems*, volume 34, pages 21297–21309. Curran Associates, Inc., 2021. URL <https://proceedings.neurips.cc/paper/2021/hash/b1d10e7bafa4421218a51b1e1f1b0ba2-Abstract.html>.

- Harsh Mehta, Ankit Gupta, Ashok Cutkosky, and Behnam Neyshabur. Long range language modeling via gated state spaces. In *The eleventh international conference on learning representations*, 2023. URL <https://openreview.net/forum?id=5MkYIYCbva>.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer Sentinel Mixture Models, September 2016. URL <http://arxiv.org/abs/1609.07843>. arXiv:1609.07843 [cs].
- E. A. Nadaraya. On Estimating Regression. *Theory of Probability & Its Applications*, 9(1):141–142, January 1964. ISSN 0040-585X. doi: 10.1137/1109020. URL <https://epubs.siam.org/doi/10.1137/1109020>.
- Eric Nguyen, Michael Poli, Matthew G. Durrant, Brian Kang, Dhruva Katrekar, David B. Li, Liam J. Bartie, Armin W. Thomas, Samuel H. King, Garyk Brixi, Jeremy Sullivan, Madelena Y. Ng, Ashley Lewis, Aaron Lou, Stefano Ermon, Stephen A. Baccus, Tina Hernandez-Boussard, Christopher Ré, Patrick D. Hsu, and Brian L. Hie. Sequence modeling and design from molecular to genome scale with Evo. *Science*, 386(6723):eado9336, November 2024. doi: 10.1126/science.ado9336. URL <https://www.science.org/doi/10.1126/science.ado9336>.
- Eshaan Nichani, Jason D. Lee, and Alberto Bietti. Understanding Factual Recall in Transformers via Associative Memories, December 2024. URL <http://arxiv.org/abs/2412.06538>. arXiv:2412.06538 [cs].
- Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context Learning and Induction Heads, September 2022. URL <http://arxiv.org/abs/2209.11895>. arXiv:2209.11895 [cs].
- OpenAI. OpenAI o1 System Card, 2024. URL <https://openai.com/index/openai-o1-system-card/>.
- Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting Recurrent Neural Networks for Long Sequences. In *Proceedings of the 40th International Conference on Machine Learning*, pages 26670–26698. PMLR, July 2023. URL <https://proceedings.mlr.press/v202/orvieto23a.html>.
- Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers Learn In-Context by Gradient Descent. In *Proceedings of the 40th International Conference on Machine Learning*, pages 35151–35174. PMLR, July 2023. URL <https://proceedings.mlr.press/v202/von-oswald23a.html>.
- Johannes von Oswald, Nino Scherrer, Seijin Kobayashi, Luca Versari, Songlin Yang, Maximilian Schlegel, Kaitlin Maile, Yanick Schimpf, Oliver Sieberling, Alexander Meulemans, Rif A. Saurous, Guillaume Lajoie, Charlotte Frenkel, Razvan Pascanu, Blaise Agüera y Arcas, and João Sacramento. MesaNet: Sequence Modeling by Locally Optimal Test-Time Training, June 2025. URL <http://arxiv.org/abs/2506.05233>. arXiv:2506.05233 [cs].
- Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, Przemysław Kazienko, Kranthi Kiran GV, Jan Kocoń, Bartłomiej Koptyra, Satyapriya Krishna, Ronald McClelland Jr, Jiaju Lin, Niklas Muennighoff, Fares Obeid, Atsushi Saito, Guangyu Song, Haoqin Tu, Cahya Wirawan, Stanisław Woźniak, Ruichong Zhang, Bingchen Zhao, Qihang Zhao, Peng Zhou, Jian Zhu, and Rui-Jie Zhu. Eagle and Finch: RWKV with Matrix-Valued States and Dynamic Recurrence, September 2024. URL <http://arxiv.org/abs/2404.05892>. arXiv:2404.05892.

- Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient Context Window Extension of Large Language Models, February 2026. URL <http://arxiv.org/abs/2309.00071>. arXiv:2309.00071 [cs].
- Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random Feature Attention. In *International Conference on Learning Representations*, October 2020. URL <https://openreview.net/forum?id=QtTKTdVrFBB>.
- T. Poggio. On optimal nonlinear associative recall. *Biological Cybernetics*, 19(4):201–209, September 1975. ISSN 1432-0770. doi: 10.1007/BF02281970. URL <https://doi.org/10.1007/BF02281970>.
- Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: towards larger convolutional language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML ’23*, pages 28043–28078, Honolulu, Hawaii, USA, July 2023. JMLR.org.
- Zhen Qin, Weixuan Sun, Hui Deng, Dongxu Li, Yunshen Wei, Baohong Lv, Junjie Yan, Lingpeng Kong, and Yiran Zhong. cosFormer: Rethinking Softmax In Attention. In *International Conference on Learning Representations*, October 2021. URL <https://openreview.net/forum?id=B18CQrx2Up4>.
- Zhen Qin, Xiaodong Han, Weixuan Sun, Dongxu Li, Lingpeng Kong, Nick Barnes, and Yiran Zhong. The Devil in Linear Transformer. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7025–7041, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.473. URL <https://aclanthology.org/2022.emnlp-main.473>.
- Zhen Qin, Xiaodong Han, Weixuan Sun, Bowen He, Dong Li, Dongxu Li, Yuchao Dai, Lingpeng Kong, and Yiran Zhong. Toeplitz neural network for sequence modeling. In *The eleventh international conference on learning representations*, 2023. URL <https://openreview.net/forum?id=IxmWsm4xrua>.
- Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. HGRN2: Gated Linear RNNs with State Expansion. In *First Conference on Language Modeling*, August 2024. URL <https://openreview.net/forum?id=y6SqbJfCSk#discussion>.
- Hubert Ramsauer, Bernhard Schöfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K. Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield Networks is All You Need. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=tL89RnzIiCd>.
- Liliang Ren, Yang Liu, Shuohang Wang, Yichong Xu, Chenguang Zhu, and ChengXiang Zhai. Sparse modular activation for efficient sequence modeling. In *Thirty-seventh conference on neural information processing systems*, 2023. URL <https://openreview.net/forum?id=TfbzX6I14i>.
- Ruifeng Ren, Zhicong Li, and Yong Liu. Can Mamba Always Enjoy the "Free Lunch"?, October 2024. URL <http://arxiv.org/abs/2410.03810>. arXiv:2410.03810 [cs].
- Sasha Rush. There are like 4 more linear RNN papers out today, April 2024. URL https://x.com/srush_nlp/status/1780231813820002771.

- David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191, July 2020. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2019.07.001. URL <https://www.sciencedirect.com/science/article/pii/S0169207019301888>.
- Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear Transformers Are Secretly Fast Weight Programmers. In *Proceedings of the 38th International Conference on Machine Learning*, pages 9355–9366. PMLR, July 2021. URL <https://proceedings.mlr.press/v139/schlag21a.html>.
- Jürgen Schmidhuber. Learning to Control Fast-Weight Memories: An Alternative to Dynamic Recurrent Networks. *Neural Computation*, 4(1):131–139, January 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.1.131. URL <https://doi.org/10.1162/neco.1992.4.1.131>.
- Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, December 2001. ISBN 978-0-262-25693-3. doi: 10.7551/mitpress/4175.001.0001. URL <https://direct.mit.edu/books/monograph/1821/Learning-with-KernelsSupport-Vector-Machines>.
- Noam Shazeer. Fast Transformer Decoding: One Write-Head is All You Need, November 2019. URL <http://arxiv.org/abs/1911.02150>. arXiv:1911.02150 [cs].
- Jiaxin Shi, Ke Alexander Wang, and Emily B. Fox. Sequence modeling with multiresolution convolutional memory. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML ’23*, pages 31312–31327, Honolulu, Hawaii, USA, July 2023. JMLR.org.
- Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazi. DeltaProduct: Increasing the Expressivity of DeltaNet Through Products of Householders, February 2025. URL <http://arxiv.org/abs/2502.10297>. arXiv:2502.10297 [cs].
- Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *The eleventh international conference on learning representations*, 2023. URL <https://openreview.net/forum?id=Ai8Hw3AXqks>.
- D.F. Specht. A general regression neural network. *IEEE Transactions on Neural Networks*, 2(6): 568–576, November 1991. ISSN 1941-0093. doi: 10.1109/72.97934. URL <https://ieeexplore.ieee.org/document/97934>.
- Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. Augmenting Self-attention with Persistent Memory, July 2019. URL <http://arxiv.org/abs/1907.01470>. arXiv:1907.01470.
- Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-Time Training with Self-Supervision for Generalization under Distribution Shifts. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9229–9248. PMLR, November 2020. URL <https://proceedings.mlr.press/v119/sun20b.html>.
- Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (Learn at Test Time): RNNs with Expressive Hidden States, August 2024. URL <http://arxiv.org/abs/2407.04620>. arXiv:2407.04620.
- 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. URL <http://arxiv.org/abs/2307.08621>. arXiv:2307.08621 [cs].

- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL <https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html>.
- Richard Sutton. The Bitter Lesson, 2019. URL <http://www.incompleteideas.net/IncIdeas/BitterLesson.html>.
- Kunio Tanabe. Projection method for solving a singular system of linear equations and its applications. *Numerische Mathematik*, 17(3):203–214, June 1971. ISSN 0945-3245. doi: 10.1007/BF01436376. URL <https://doi.org/10.1007/BF01436376>.
- Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient Transformers: A Survey. *ACM Comput. Surv.*, 55(6):109:1–109:28, December 2022. ISSN 0360-0300. doi: 10.1145/3530811. URL <https://dl.acm.org/doi/10.1145/3530811>.
- Asher Trockman, Hrayr Harutyunyan, J. Zico Kolter, Sanjiv Kumar, and Srinadh Bhojanapalli. Mimetic Initialization Helps State Space Models Learn to Recall, October 2024. URL <http://arxiv.org/abs/2410.11135>. arXiv:2410.11135 [cs].
- Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer Dissection: An Unified Understanding for Transformer’s Attention via the Lens of Kernel. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4344–4353, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1443. URL <https://aclanthology.org/D19-1443/>.
- Jos van der Westhuizen and Joan Lasenby. The unreasonable effectiveness of the forget gate, September 2018. URL <http://arxiv.org/abs/1804.04849>. arXiv:1804.04849 [cs, stat].
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.
- Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL <https://papers.nips.cc/paper/2016/hash/90e1357833654983612fb05e3ec9148c-Abstract.html>.
- Johannes von Oswald, Eyvind Niklasson, Maximilian Schlegel, Seijin Kobayashi, Nicolas Zucchet, Nino Scherrer, Nolan Miller, Mark Sandler, Blaise Agüera y Arcas, Max Vladymyrov, Razvan Pascanu, and João Sacramento. Uncovering mesa-optimization algorithms in Transformers. *CoRR*, abs/2309.05858, 2023. URL <https://doi.org/10.48550/arXiv.2309.05858>. tex.cdate: 1672531200000 tex.pubtype: informal.
- Ke Alexander Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q. Weinberger, and Andrew Gordon Wilson. Exact Gaussian Processes on a Million Data Points. *Advances in Neural Information Processing Systems*, 32, 2019. URL <https://papers.nips.cc/paper/2019/hash/01ce84968c6969bdd5d51c5eeaa3946a-Abstract.html>.
- Larry Wasserman. *All of Nonparametric Statistics*. Springer Texts in Statistics. Springer, New York, NY, 2006. ISBN 978-0-387-25145-5. doi: 10.1007/0-387-30623-4. URL <http://link.springer.com/10.1007/0-387-30623-4>.

- Geoffrey S. Watson. Smooth Regression Analysis. *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)*, 26(4):359–372, 1964. ISSN 0581-572X. URL <https://www.jstor.org/stable/25049340>.
- Bernard Widrow and Marcian E. Hoff. Adaptive switching circuits. In *Neurocomputing: foundations of research*, pages 123–134. MIT Press, Cambridge, MA, USA, January 1988. ISBN 978-0-262-01097-9.
- D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Non-Holographic Associative Memory. *Nature*, 222(5197):960–962, June 1969. ISSN 1476-4687. doi: 10.1038/222960a0. URL <https://www.nature.com/articles/222960a0>.
- David Willshaw. Holography, Associative Memory, and Inductive Generalization. In *Parallel Models of Associative Memory*. Psychology Press, 1989. ISBN 978-1-315-80799-7. Num Pages: 22.
- Mitchell Wortsman, Peter J. Liu, Lechao Xiao, Katie E. Everett, Alexander A. Alemi, Ben Adlam, John D. Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, Jeffrey Pennington, Jascha Sohl-Dickstein, Kelvin Xu, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Small-scale proxies for large-scale Transformer training instabilities. In *The Twelfth International Conference on Learning Representations*, October 2023. URL <https://openreview.net/forum?id=d8w0pmvXbZ>.
- Mingyu Xu, Wei Cheng, Bingning Wang, and Weipeng Chen. KV Shifting Attention Enhances Language Modeling, December 2024. URL <http://arxiv.org/abs/2411.19574>. arXiv:2411.19574 [cs].
- Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with Delta Rule, December 2024a. URL <http://arxiv.org/abs/2412.06464>. arXiv:2412.06464 [cs] version: 1.
- Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated Linear Attention Transformers with Hardware-Efficient Training. In *Proceedings of the 41st International Conference on Machine Learning*, pages 56501–56523. PMLR, July 2024b. URL <https://proceedings.mlr.press/v235/yang24ab.html>.
- Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing Linear Transformers with the Delta Rule over Sequence Length, August 2024c. URL <http://arxiv.org/abs/2406.06484>. arXiv:2406.06484.
- Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential Transformer. October 2024. URL <https://openreview.net/forum?id=0voCm1gGhN>.
- Jianyu Zhang, Niklas Nolte, Ranajoy Sadhukhan, Beidi Chen, and Leon Bottou. Memory mosaics. In *The thirteenth international conference on learning representations*, 2025. URL <https://openreview.net/forum?id=IiagjrJNwF>.
- Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Re. The Hedgehog & the Porcupine: Expressive Linear Attentions with Softmax Mimicry. In *The Twelfth International Conference on Learning Representations*, October 2023a. URL <https://openreview.net/forum?id=4g0212N2Nx>.
- Yufeng Zhang, Fengzhuo Zhang, Zhuoran Yang, and Zhaoran Wang. What and How does In-Context Learning Learn? Bayesian Model Averaging, Parameterization, and Generalization, October 2023b. URL <http://arxiv.org/abs/2305.19420>. arXiv:2305.19420 [stat].

Yufeng Zhang, Boyi Liu, Qi Cai, Lingxiao Wang, and Zhaoran Wang. An Analysis of Attention via the Lens of Exchangeability and Latent Variable Models, April 2024. URL <http://arxiv.org/abs/2212.14852>. arXiv:2212.14852 [cs].

Jian Zhou and Olga G. Troyanskaya. Predicting effects of noncoding variants with deep learning-based sequence model. *Nature Methods*, 12(10):931–934, October 2015. ISSN 1548-7105. doi: 10.1038/nmeth.3547. URL <https://www.nature.com/articles/nmeth.3547>.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ICML’03, pages 928–935, Washington, DC, USA, August 2003. AAAI Press. ISBN 978-1-57735-189-4.

Appendix A. Bounding the norm of a linear regression layer

Let $\mathbf{y}_t = \mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1} \mathbf{q}_t$ and let $\|\mathbf{A}\|_2$ indicate the spectral norm of a matrix $\mathbf{A}$, a matrix norm induced by the L2 norm on vectors. Then

$$\|\mathbf{y}_t\|_2 = \|\mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1} \mathbf{q}_t\|_2 \quad (60)$$

$$\leq \|\mathbf{q}_t\|_2 \sup_{\|\mathbf{q}\|_2=1} \|\mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1} \mathbf{q}\|_2 \quad (61)$$

$$= \|\mathbf{q}_t\|_2 \|\mathbf{V}_t^\top \mathbf{K}_t (\mathbf{K}_t^\top \mathbf{K}_t)^{-1}\|_2 \quad (62)$$

$$\leq \|\mathbf{q}_t\|_2 \|\mathbf{V}_t^\top \mathbf{K}_t\|_2 \|(\mathbf{K}_t^\top \mathbf{K}_t)^{-1}\|_2 \quad (63)$$

$$= \|\mathbf{q}_t\|_2 \left\| \sum_{i=1}^t \mathbf{v}_i \mathbf{k}_i^\top \right\|_2 \|(\mathbf{K}_t^\top \mathbf{K}_t)^{-1}\|_2 \quad (64)$$

$$\leq \|\mathbf{q}_t\|_2 \left(\sum_{i=1}^t \|\mathbf{v}_i\|_2 \|\mathbf{k}_i\|_2 \right) \|(\mathbf{K}_t^\top \mathbf{K}_t)^{-1}\|_2 \quad (65)$$

$$= \|\mathbf{q}_t\|_2 \left(\sum_{i=1}^t \|\mathbf{v}_i\|_2 \|\mathbf{k}_i\|_2 \right) \max_i \lambda_i((\mathbf{K}_t^\top \mathbf{K}_t)^{-1}) \quad (66)$$

$$= \|\mathbf{q}_t\|_2 \left(\sum_{i=1}^t \|\mathbf{v}_i\|_2 \|\mathbf{k}_i\|_2 \right) \max_i \frac{1}{\lambda_i(\mathbf{K}_t^\top \mathbf{K}_t)} \quad (67)$$

$$= \frac{\|\mathbf{q}_t\|_2 \sum_{i=1}^t \|\mathbf{v}_i\|_2 \|\mathbf{k}_i\|_2}{\lambda_{\min}(\mathbf{K}_t^\top \mathbf{K}_t)}. \quad (68)$$

Hence $\|\mathbf{y}_t\|_2 \leq \|\mathbf{q}_t\|_2 \sum_{i=1}^t \|\mathbf{v}_i\|_2 \|\mathbf{k}_i\|_2 / \lambda_{\min}(\mathbf{K}_t^\top \mathbf{K}_t)$. When we approximate $\mathbf{K}_t^\top \mathbf{K}_t \approx \mathbf{I}$, as in linear attention, we lose the self-normalizing property of dividing by $\lambda_{\min}(\mathbf{K}_t^\top \mathbf{K}_t)$ since the denominator becomes 1. This explains how output normalization by Qin et al. (2022) is an attempt at restoring this intrinsic self-normalizing property of linear regression.