FASTERWEIGHT-MLP ATTENTION: SEQUENCE MODELING FROM A NEURAL-NETWORK-NATIVE PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

A causal attention layer computes its scores with a network whose weights the context instantiates: a softmax head is a two-layer network whose hidden units and readout come from the prefix. We take that network as the design object and ask two questions: what may it read, and how much learned capacity does it carry? A query position t and a key position i split the causal history into three regions: the prefix the query sees, the prefix the key sees, and the interval between them. FasterWeight-MLP Attention (FW-MLP) reads each region with a small learned network, a fast MLP, and multiplies the three outputs into one score. Reading the interval by subtracting continuous prefix statistics leaves one exact form, a running additive sum, and with it FW-MLP evaluates the score exactly by a recurrence linear in sequence length. Linear attention, retention, and gated decays are exact special cases of the resulting score class, and delta-rule transports and the continuous scores on compact domains lie in its uniform closure. Under a matched attention-side parameter budget, FW-MLP attains the best mean rank among 12 mixers on 26 tasks and the best validation loss in a 123M-parameter nanoGPT setting. On a retrieval task gated by an interval statistic it reaches 34.3% accuracy, against 20.4% for its endpoint-only configuration, and it holds the best average downstream rank at both 340M and 1.3B parameters.

1 INTRODUCTION

A causal sequence mixer is usually designed by choosing a rule that carries information from one position to another: an exponential decay, an input-dependent gate, a delta-rule update, a retention schedule (Katharopoulos et al., 2020; Schlag et al., 2021; Sun et al., 2023; Yang et al., 2024a;b; 2025; Gu & Dao, 2024; Dao & Gu, 2024). The forward computation of attention admits a reading in which the design object is the computation itself. With the row-vector convention, one causal softmax head is a two-layer network, applied to the current token, whose hidden units and readout the prefix instantiates, and identity-feature linear attention is a single context-instantiated linear map (Propositions C.1 and C.2). These are identities, and they hold for every input. Read this way, the layer is a network that the context builds. We call a weight *fast* when the forward pass computes it from the input prefix, and *slow* when it is a trained parameter. We take this *neural-network-native* view as the subject of the paper, and the architecture we report is what the view yields when its two questions are answered.

The view asks two questions. The first is what the context-instantiated network may read. A causal score is a function of a pair of positions, so its argument is whatever the causal history makes available to that pair, and the two boundaries of a pair (t, i) with $i \le t$ delimit exactly three regions of that history:

$$\underbrace{X_{1:t}}_{\text{query-side prefix}} \quad , \qquad \underbrace{X_{1:i}}_{\text{key-side prefix}} \quad , \qquad \underbrace{X_{i+1:t}}_{\text{between-boundary interval}} \quad . \tag{1}$$

Each region answers a distinct question about the pair: the state available where the score is read, the state available where the candidate value was written, and the events that occurred between writing and reading. The key-side prefix and the interval together make up the query-side prefix, so the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

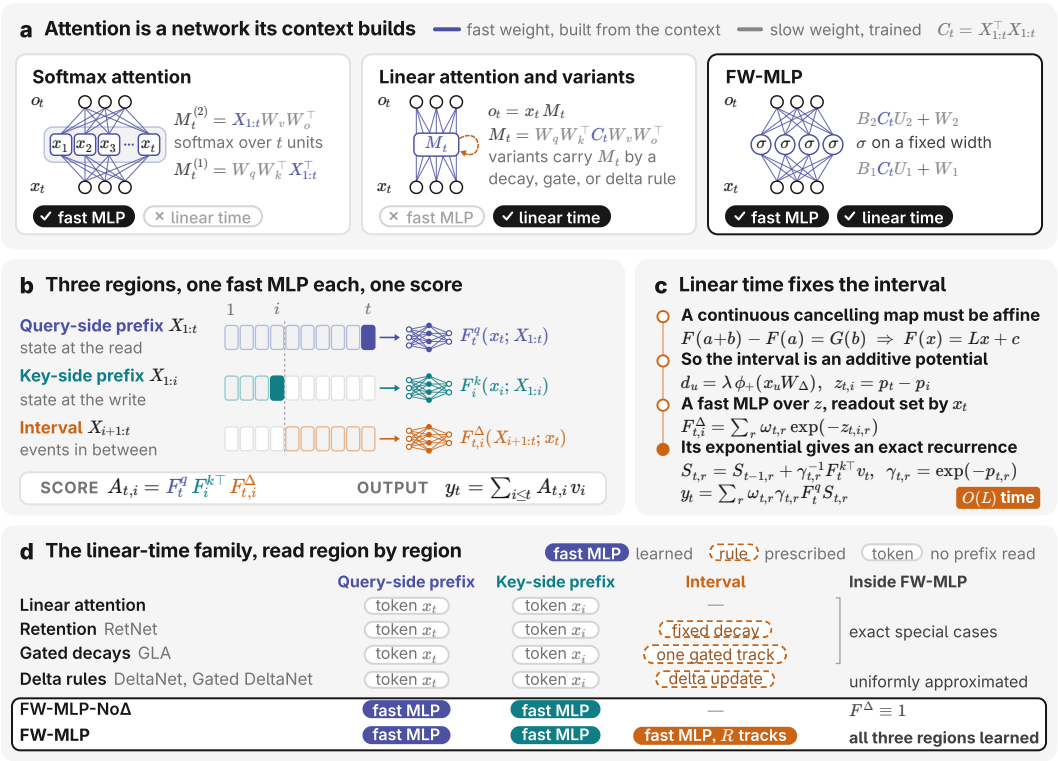

Figure 1: FW-MLP in one view: **(a)** the neural-network-native view (Section 2), **(b)** the score it yields, **(c)** the interval factor and its recurrence, and **(d)** the family it contains (Section 4).

list is complete. The interval is read on its own, because a score that reaches it only through the difference of its endpoint representations pays for it in width (Section 2). The second question is how much learned capacity each region gets. The view answers with the smallest object it recognizes as a learned network: a context-instantiated linear map, an activation, and a second such map, which is the fast-MLP template of Section 2.

Linear time fixes the form of each factor. Cost linear in sequence length constrains how the context may enter a fast MLP: only through statistics that carry forward in a running form. At the two endpoints that statistic is the associative prefix statistic $C_u := X_{1:u}^\top X_{1:u}$. Over the interval, reached by prefix subtraction, the constraint removes the choice: by Cauchy's functional equation the only continuous such statistic is an additive potential (Section 3.2), and its exponential turns the pairwise sum into a recurrence (Section 3.3). Composing the answers gives one learned fast-weight factor per region, whose product is the score (Figure 1). For head h,

$$A_{t,i,h}^{\text{FW-MLP}} = F_{t,h}^q(x_t; X_{1:t})\, F_{i,h}^k(x_i; X_{1:i})^\top\, F_{t,i,h}^\Delta(X_{i+1:t}; x_t). \tag{2}$$

The endpoint factors are fast MLPs on the two boundary tokens, instantiated by the prefix statistic at each boundary, which read different prefixes and so form an asymmetric pair. The interval factor is a fast MLP over the additive potential, with a readout whose track weights are set by x_t. All three factors are formed inside one block, so a single FW-MLP layer presents the interval to its score generator, and Appendix K records what a stack of token-local layers reaches at each depth. Setting $F^\Delta \equiv 1$ gives the reading that stops at the endpoints, which we carry through the first three experimental settings as the ablation FW-MLP-NoΔ.

The view organizes the existing family. Each linear-time mixer answers the two questions at a smaller setting. Linear attention reads the endpoints without an activation, retention and gated linear attention read the interval through one prescribed track, and delta rules prescribe the update carried across it. Softmax attention forms every score at quadratic cost, and its row normalization exposes

the interval statistic as a sequence-conditioned linear term. Section 4 states the first three as exact containments and the delta-rule transports as uniform limits, and Table 3 gives the map.

Contributions. We state the neural-network-native view as three identities about the forward computation of attention (Section 2). We show that the view and the linear-time requirement, with the interval reached by prefix subtraction, fix the structure of the layer as three regions read by fast MLPs with an additive potential over the interval, and leave the activation, the head and track counts, and the rank allocation as design choices (Section 3). We characterize the resulting score class, from exact separable rank for endpoint scores to density of the full score on compact domains; it contains linear attention and gated decays exactly and approximates delta-rule and other matrix transports (Section 4). Under a matched attention-side budget, FW-MLP ranks first among 12 mixers on 26 tasks, on interval-gated retrieval and nanoGPT, and downstream at 340M and 1.3B (Section 5).

2 THE NEURAL-NETWORK-NATIVE VIEW

Attention instantiates its own network. With the row-vector convention $q_t = x_t W_q$, $K_t = X_{1:t} W_k$, and $V_t = X_{1:t} W_v$, one causal softmax head computes

$$o_t = \mathrm{softmax}(q_t K_t^\top) V_t W_o^\top = \mathrm{softmax}\big(x_t W_q W_k^\top X_{1:t}^\top\big)\, X_{1:t} W_v W_o^\top, \tag{3}$$

so the prefix supplies both the hidden units and the readout of a network applied to the single token x_t, with a hidden width that grows with the context (Proposition C.1). The unnormalized numerator of identity-feature linear attention compresses the same context into one dynamic linear map,

$$o_t = x_t \underbrace{W_q W_k^\top X_{1:t}^\top X_{1:t} W_v W_o^\top}_{\text{context-instantiated weight}}, \tag{4}$$

which reaches the token through a linear path (Proposition C.2). Both equations are identities, so the network they describe is the computation the layer already performs, and the design question becomes which fast network to build. Placing an activation between two context-instantiated maps gives the smallest such network that keeps cost linear in sequence length,

$$\sigma\big[x_t\big(W_{q1} W_{k1}^\top X_{1:t}^\top X_{1:t} W_{v1} W_{o1}^\top + W_{\mathrm{static},1}\big)\big]\, \big(W_{q2} W_{k2}^\top X_{1:t}^\top X_{1:t} W_{v2} W_{o2}^\top + W_{\mathrm{static},2}\big)\,, \tag{5}$$

because the context enters only through the associative statistic $X_{1:t}^\top X_{1:t}$ (Proposition C.3). Equation 5 is the fast-MLP template that the view hands to pairwise causal scoring.

What a pairwise score may read. Prefix sums make the three regions of Equation 1 concrete: with $s_t = \sum_{j \le t} x_j$ and $p_i = \sum_{j \le i} x_j$, the difference $r_{t,i} = s_t - p_i$ is the additive statistic of the interval, so $(s_t, p_i, r_{t,i})$ stand for the three arguments a score generator has to read, and a design answers how many of them it takes and with how much learned capacity.

What a smaller reading costs. A design that reads two of the three regions reaches the interval through the difference $r_{t,i}$ of its endpoint representations, at a measurable price. A raw bilinear score $s_t^\top A p_i$ equals a fixed function of $r_{t,i}$ only if that function is zero; endpoint MLP towers reach a nonzero one at a width equal to its *separable rank*, the least number of product terms $u_m(s_t) v_m(p_i)$ that sum to it (Section 4). The next section reads the third region directly, at linear cost.

3 FASTERWEIGHT-MLP ATTENTION

This section builds the layer that the view yields from the fast-MLP template of Equation 5 and the linear-time requirement: the two endpoint regions of Equation 1 in Section 3.1, the interval in Section 3.2, and their recurrence in Section 3.3. The score arrives in two stages. The context instantiates endpoint fast MLPs, and their outputs and the interval factor together form the score that mixes the second-stage values; the name FasterWeight records this nested construction.

Notation. Tokens x_u are row vectors, and $X_{1:u}$ stacks the first u of them. A score compares a query position t with a key position $i \le t$. First-stage heads, indexed by a, build the endpoint fast MLPs; second-stage heads, indexed by h, form the score and mix the values; and each second-stage head carries R interval tracks, indexed by r. Table 28 collects the notation and its shapes, and a scalar example closes the section.

3.1 ASYMMETRIC ENDPOINT FAST MLPS

In the template of Equation 5 the context reaches the first stage through the associative prefix statistic $C_u := X_{1:u}^\top X_{1:u}$, which becomes the first-layer weight of a network applied to a token. A pairwise score has two boundaries and two prefixes: the query endpoint sees $X_{1:t}$, and the key endpoint sees the earlier prefix $X_{1:i}$ that was available where the value at i was written, so the template is applied once per endpoint. The fast-weight generator carries low-rank slow factors, one copy per *first-stage head* a:

$$B_a = W_{q,a}^{(1)} W_{k,a}^{(1)\top},\ U_a = W_{v,a}^{(1)} W_{o,a}^{(1)\top},\ \mathcal{H}_a(x, C) = \sigma\big(x\,[B_a C U_a + W_{\text{static},a}^{(1)}] + b_a^{(1)}\big), \quad (6)$$

where the context-instantiated weight $B_a C U_a$ has B_a in the role of the query/key factor of the template and U_a in that of the value/output factor, and σ is SwiGLU by default (Appendix F.2; ReLU and linear are ablations). Head a thus maps a token x and a prefix statistic C to a prefix-conditioned representation. Each final head h selects two first-stage heads a_h^q, a_h^k and evaluates them at the two boundaries,

$$F_{t,h}^q = \mathcal{H}_{a_h^q}(x_t, C_t)\, W_{q,h}^{(2)}, \qquad F_{i,h}^k = \mathcal{H}_{a_h^k}(x_i, C_i)\, W_{k,h}^{(2)}, \quad (7)$$

with slow projections $W_{q,h}^{(2)}, W_{k,h}^{(2)}$ into the second-stage query/key space. The two towers read different prefixes and carry independent parameters, and that asymmetry strictly enlarges the score class (Proposition F.2); both branches come from one pass over the sequence (Appendix F.2). The product $F_{t,h}^q F_{i,h}^{k\top}$ is already a two-tower score class over $(X_{1:t}, X_{1:i})$; this is the configuration that FW-MLP-NoΔ evaluates, with the interval-generator parameters removed and the budget met through the ranks (Appendix S).

3.2 AN EXACTLY CANCELLATIVE INTERVAL FAST MLP

The third region calls for a factor determined by the tokens $x_{i+1}, \ldots, x_t$ and computed from running quantities. Prefix subtraction is the running route to an interval, and we call a prefix map *exactly cancellative* when its difference depends on the interval alone. Cauchy's functional equation characterizes that class (Aczél, 1966; Appendix S.4).

Proposition E.1. *Let $F : \mathbb{R}^d \to \mathbb{R}^m$ be continuous, and suppose $F(a+b) - F(a) = G(b)$ for all $a, b \in \mathbb{R}^d$ and some G. Then $F(x) = Lx + c$ for a linear map L and a constant c; conversely, every such F satisfies the identity with $G(b) = Lb$.*

An additive potential is therefore the form available here, and the interval branch is built on one. For R tracks, define token increments, prefix potentials, and their interval difference

$$d_{u,h} = \lambda\, \phi_+(x_u W_{\Delta,h}) \in \mathbb{R}^R, \qquad p_{t,h}^\Delta = \sum_{u \le t} d_{u,h}, \qquad z_{t,i,h}^\Delta = p_{t,h}^\Delta - p_{i,h}^\Delta = \sum_{u=i+1}^{t} d_{u,h}, \quad (8)$$

where ϕ_+ is nonnegative, ReLU in our instantiation, and $\lambda > 0$ is a fixed scale. The branch is then a one-hidden-layer fast MLP over that statistic, with an endpoint-conditioned readout:

$$\omega_{t,h} = \frac{1 + \tanh(x_t W_{\omega,h})}{R}, \qquad F_{t,i,h}^\Delta = \omega_{t,h} \exp\big[-z_{t,i,h}^\Delta\big]^\top = \sum_{r=1}^{R} \omega_{t,h,r} \exp\left[-\sum_{u=i+1}^{t} d_{u,h,r}\right]. \quad (9)$$

The identity path in $1 + \tanh(\cdot)$ starts the branch near a decay-like behavior, and the learned projection $W_{\omega,h}$ lets the current token x_t reweight the R tracks. The branch is therefore a learned exponential-basis MLP over an exactly interval-measurable statistic (Lemma G.1, Corollary G.1).

3.3 SCORE, VALUES, AND THE LINEAR-TIME RECURRENCE

The score and the value mixing are

$$A_{t,i,h}^{\text{FW-MLP}} = F_{t,h}^q F_{i,h}^{k\top} F_{t,i,h}^\Delta, \quad v_{i,h}^{(2)} = (x_i W_{v,h}^{(2)}) \odot G_{L,h}^{(2)}(x_i), \quad y_{t,h}^{(2)} = \sum_{i \le t} A_{t,i,h}^{\text{FW-MLP}} v_{i,h}^{(2)}, \quad (10)$$

followed by per-head normalization, a right output gate, an output projection, and an optional first-stage shortcut, all detailed in Appendix O.2.

The exponential form of the interval factor turns the pairwise sum into a recurrence. Writing $\gamma_{t,h,r} = \exp(-p^{\Delta}_{t,h,r})$ gives $\exp[-(p^{\Delta}_{t,h,r} - p^{\Delta}_{i,h,r})] = \gamma_{t,h,r}\gamma^{-1}_{i,h,r}$, so absorbing $\gamma_{t,h,r}$ into the query side and $\gamma^{-1}_{i,h,r}$ into the key side leaves each track an ordinary causal linear mixer, started from $S_{0,h,r} = 0$:

$$S_{t,h,r} = S_{t-1,h,r} + \gamma^{-1}_{t,h,r} F^{k\top}_{t,h} v^{(2)}_{t,h}, \qquad y^{(2)}_{t,h,r} = \gamma_{t,h,r} F^{q}_{t,h} S_{t,h,r}, \qquad y^{(2)}_{t,h} = \sum_{r=1}^{R} \omega_{t,h,r} y^{(2)}_{t,h,r}. \tag{11}$$

Theorem G.1. *For every input sequence and head h, the recurrence of Equation 11, with every track state started from $S_{0,h,r} = 0$, returns exactly the output $y^{(2)}_{t,h} = \sum_{i \le t} A^{\text{FW-MLP}}_{t,i,h} v^{(2)}_{i,h}$ of the materialized score of Equation 10.*

The recurrence runs in time and memory linear in sequence length at fixed ranks and track count (Corollary G.2; Appendix L covers packed batches), and Section 5.5 measures both forms on the same inputs.

A scalar example. Take scalar tokens $x_u = 1$, a ReLU activation, weights that give $F^q_u = u$ and $F^k_u = 2u$, values $v_u = 1$, and one track with $d_u = \log 2$ and $\omega = 1$. The interval factor is $2^{-(t-i)}$, so the materialized score is $A_{t,i} = 2ti\, 2^{-(t-i)}$ and $y_4 = \sum_{i \le 4} A_{4,i} = 1 + 4 + 12 + 32 = 49$. The recurrence reaches the same value without forming a pair: with $\gamma_u = 2^{-u}$, $S_u = S_{u-1} + 2^u \cdot 2u$ gives $S_{1:4} = 4, 20, 68, 196$, and $y_4 = \gamma_4 F^q_4 S_4 = 2^{-4} \cdot 4 \cdot 196 = 49$. Appendix S gives the forward pass as an algorithm and the weights behind this example.

4 What the score class contains

The derivation rests on two claims: reading all three regions is worth more than reading fewer, and the linear-time forms stay expressive. The results behind them come in two kinds. *Exact identities* hold for every input and fix what a score equals. *Approximation statements* hold uniformly on compact domains and say which functions the class contains; Section 5 then measures which member training reaches at a fixed budget. Appendix A gives the full statements and proofs.

Exact identities. Under a rich prefix-difference domain (Assumption B.1), the identity $s^\top A p = g(s - p)$ holds for all (s, p) exactly when $A = 0$ and $g \equiv 0$, and with row normalization softmax expresses a sequence-conditioned linear function of the interval statistic (Propositions D.2 and D.1). Endpoint scores are priced by separable rank:

Theorem D.1. *A scalar $h : S \times P \to \mathbb{R}$ equals $\phi(s)^\top \psi(p)$ for some $\phi : S \to \mathbb{R}^M$, $\psi : P \to \mathbb{R}^M$ exactly when $h(s, p) = \sum_{m=1}^{M} u_m(s) v_m(p)$ for scalar functions u_m, v_m. Hence $g(s - p)$ is a finite two-tower dot product if and only if it has finite separable rank.*

A reading that omits the third region therefore pays for interval dependence in tower width, at the separable rank of the target (Proposition J.3), while bilinear-GLU towers realize every quadratic difference function exactly (Proposition D.4). For the third region, Proposition E.1 and Theorem G.1, stated in Section 3, make every interval track depend on $X_{i+1:t}$ alone, with a readout set by x_t, and evaluate the factor in linear time (Lemma G.1, Corollary G.1). Expanding the dot product and the track sum writes the whole FW-MLP score as a finite *tri-region separable sum*, a sum of products of a query-prefix factor, a key-prefix factor, and an interval factor (Theorem I.1). The family sits inside this class exactly: linear attention is the endpoint-only case with the activation removed, scalar and diagonal gated decays and retention are interval branches with one track per decay rate, and fixed exponential decay is the single unlearned track (Propositions J.1, J.4 and P.2, Corollary H.1, Table 3).

Approximation on compact domains. Two-layer ReLU towers approximate every continuous difference function on a compact set (Theorems D.2 and F.1). Finite mixtures of exponentials are dense in the continuous functions of a nonnegative additive statistic on a compact set, and the endpoint-conditioned version holds when the readout class is dense on the endpoint domain and the track increments approximate the interval statistic (Theorems H.1 and H.2). The three regions combine:

Theorem I.2. *Let $\mathcal{Z}_q, \mathcal{Z}_k, \mathcal{Z}_\Delta$ be compact Hausdorff state domains and $\mathcal{K} \subset \mathcal{Z}_q \times \mathcal{Z}_k \times \mathcal{Z}_\Delta$ a compact set of feasible three-region states. Finite sums $\sum_m u_m(z_q)v_m(z_k)w_m(z_\Delta)$ of continuous factors are uniformly dense in $C(\mathcal{K})$.*

When each branch approximates its factors and heads and tracks are allocated one product term each, FW-MLP inherits this density (Corollary I.1); under the same assumptions it approximates delta-rule and other matrix-state transports, and the lift applies to other mixer backbones (Proposition P.3, Corollary P.2).

5 EXPERIMENTS

The derivation fixes the layer before any task is seen. Four settings test whether it holds up and whether the third region earns its place, at increasing scale: a controlled cross-domain suite, a diagnostic whose target depends on an interval statistic, nanoGPT validation, and downstream evaluation of trained language models. The endpoint-only configuration FW-MLP-NoΔ appears in the first three. Every comparison swaps the sequence-mixing block inside a fixed scaffold, and all training runs use $8\times$A100-80GB. The matched-budget rule (Appendix R.1) gives every mixer about Bd^2 trainable attention-side parameters per layer at model width d, with $B = 8$ on VariantBench and the diagnostic. Each mixer reaches that count through its own width parameter, and the feed-forward block, embeddings, and classifier are shared. An exact-count solver sets FW-MLP's internal ranks and verifies the realized count (Appendices O.1 and O). Section 5.5 reports compute separately. Table 1 states the headline outcome in each setting, Figure 2 shows the individual measurements behind all four, and Appendix R holds the protocols and the full result tables.

5.1 VARIANTBENCH: A CONTROLLED CROSS-DOMAIN COMPARISON

VariantBench collects widely used tasks from the sequence-modeling and linear-attention literature into one matched protocol, covering synthetic reasoning, associative retrieval, compositional generalization, time series, recommendation, language, audio, and vision sequences. The raw suite holds 74 task groups, 23 outside the Time-Series Library and 51 within it (THUML, 2024). Collapsing the library groups into their three families (forecasting, classification, and anomaly detection) gives 26 paper-level tasks and keeps the time-series portion proportionate in the aggregate. The 12 mixers are Softmax attention with learned or rotary positions (Vaswani et al., 2017; Su et al., 2024), Gated Softmax, Differential Transformer and its v2 (Ye et al., 2025; 2026), Linear Attention (Katharopoulos et al., 2020), Gated DeltaNet, GLA, Kimi Delta Attention (Yang et al., 2025; 2024a; Kimi Team et al., 2025), Mamba-3 (Lahoti et al., 2026), FW-MLP, and FW-MLP-NoΔ. With three seeds they give 2664 mixer-task-seed evaluations, and Appendix R.3 lists the coverage of every group together with the shared optimizer settings, task splits, and evaluation protocol.

Table 1: Headline outcome in each setting. Ranks are tie-aware and averaged over tasks, with lower values better. Appendix R gives the protocol and per-task tables behind each row, and Tables 10 and 23 give task-bootstrap intervals and paired tests for the first and last rows.

Setting	Controlled comparison	Outcome for FW-MLP
VariantBench	12 mixers, 26 paper-level tasks, three seeds, matched attention-side budget	Best mean tie-aware rank, 4.538, with 14 task wins; FW-MLP-NoΔ second at 5.019
Interval-gated retrieval	Target set by the conjunction of endpoint key matching and a modular statistic of the interval	$34.34\% \pm 0.56\%$ accuracy, with $20.44\% \pm 6.74\%$ for FW-MLP-NoΔ and 8.33% at chance
nanoGPT	GPT-2-style 123M model trained from scratch on OpenWebText	Best matched-budget validation loss, 3.0014, lowest at 29 of the last 30 checkpoints, and first to 3.03
Language models	340M/15B-token and 1.3B/100B-token checkpoints on 16 downstream tasks	Best average rank at both scales, 1.69 and 1.94, first in 98% and 94% of task resamples

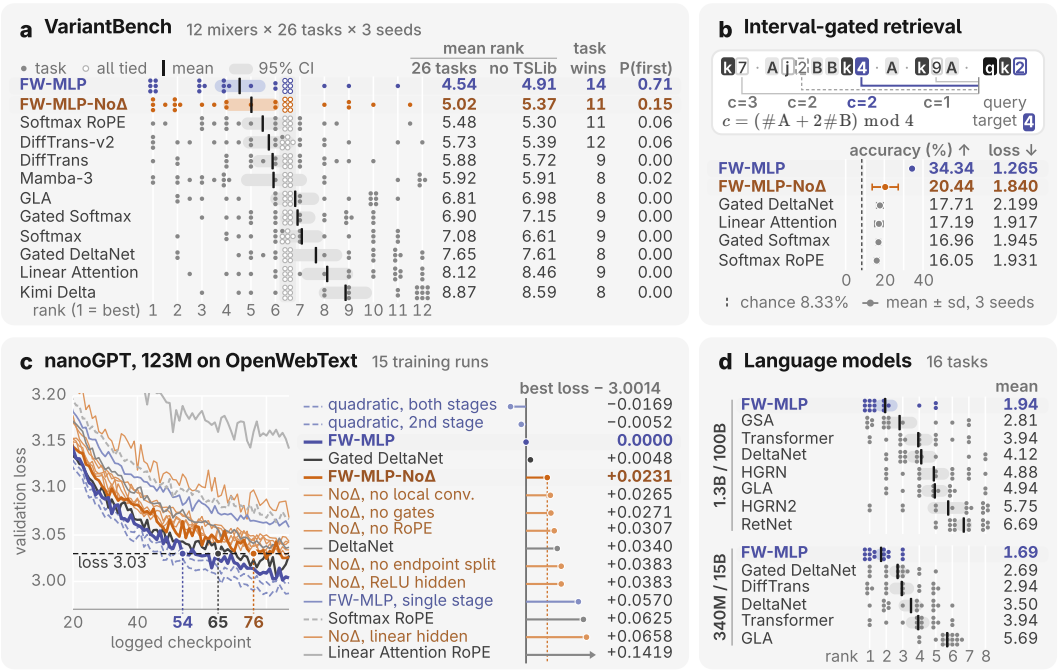

Figure 2: **(a)** Tie-aware rank of each mixer on each of the 26 VariantBench tasks, its mean and 95% task-bootstrap interval, the mean over the 23 tasks outside the Time-Series Library (TSLib), task wins, and the share of task resamples in which it ranks first (Figure 4, Tables 6–11). **(b)** A worked retrieval example, with the different-key same-class distractor dashed, and accuracy and loss over three seeds (Table 19). **(c)** All 15 nanoGPT validation curves, the checkpoints at which FW-MLP, Gated DeltaNet, and FW-MLP-NoΔ first reach 3.03, and each run's best loss relative to FW-MLP (Tables 20 and 21). **(d)** Per-task ranks of the language models, as in (a) (Tables 22 and 23). The labels write DiffTrans for Differential Transformer and Kimi Delta for Kimi Delta Attention.

Panel (a) of Figure 2 gives the aggregate. FW-MLP holds the best mean tie-aware rank and the most task wins, and FW-MLP-NoΔ is second, ahead of Softmax RoPE and the two Differential Transformer variants. The same ordering of the two FW-MLP configurations survives removal of the time-series families: over the 23 remaining tasks FW-MLP stays first, with Softmax RoPE second (Table 7). The gain is spread across domains, and Figure 4 gives every task-level rank: FW-MLP leads the recommendation, audio, and time-series forecasting and classification tasks, and FW-MLP-NoΔ takes two retrieval tasks and time-series anomaly detection. Resampling the 26 tasks keeps FW-MLP first in 71% of draws. It ranks above each of the six weakest variants on 14 to 17 tasks (sign test $p \leq 0.031$; Table 10) and stays first under four other aggregation rules and every leave-one-task-out ranking (Table 11). Appendix R.4 repeats the comparison at tighter budgets.

5.2 A RETRIEVAL TASK GATED BY AN INTERVAL STATISTIC

The region count makes a testable prediction: a target that depends on the interval separates a reading of three regions from a reading of two. We build a task whose target depends on the interval. Each length-56 sequence holds six key–value records, and a query supplies both a key and an interval class $c_{t,i} = (\#\mathrm{A}_{i+1:t} + 2\#\mathrm{B}_{i+1:t}) \bmod 4$ computed from marker tokens between the value position and the query. Three records share the query key and carry distinct interval classes, and a different-key same-class distractor is present, so the target is identified by the conjunction of the endpoint match and the interval statistic. The task realizes the setting of Proposition N.2 at the token level: histories with identical token-local endpoints and different interval descriptors.

Panel (b) of Figure 2 reports accuracy. FW-MLP reaches $34.34\% \pm 0.56\%$, the endpoint-only ablation reaches $20.44\% \pm 6.74\%$, and the four remaining baselines fall in a narrow band above the chance rate of 8.33%. Over three seeds FW-MLP clears each of them (Welch $p \leq 0.004$). The

ordering is consistent with the theory. Endpoint towers reach interval structure through separable rank (Theorem D.2), and the interval factor of the full model approximates continuous functions of the exactly cancellative interval statistic (Theorem H.2, Corollary I.1). Appendix R.5 gives the full protocol.

5.3 NANOGPT VALIDATION

We then train each variant from scratch in a GPT-2-style nanoGPT setting on OpenWebText (Gokaslan & Cohen, 2019; Radford et al., 2019; Karpathy, 2022), sharing the tokenizer, data split, optimization schedule, context length, Transformer scaffold, and feed-forward block across runs. Panel (c) of Figure 2 draws every validation curve and the best loss of each run, the lowest value over its logged checkpoints. FW-MLP reaches the best matched-budget value, 3.0014, ahead of Gated DeltaNet, its own endpoint-only ablation, Softmax RoPE, and Linear Attention RoPE, and it holds the lead through late training. It has the lowest loss of the five runs at 29 of the last 30 logged checkpoints, sits 0.0112 below Gated DeltaNet on average over that window, and is the first to reach a loss of 3.03 (Table 21). Against the endpoint-only ablation, the interval factor lowers the best loss by 0.0231, or 2.33% in perplexity.

The ablation ladder locates the gain inside the block. Measured from FW-MLP-NoΔ, removing the local convolution, the gates, or the rotary embedding raises the best loss by at most 0.0076, and removing the endpoint split or changing the hidden activation raises it more. A single-stage FW-MLP sits 0.0570 above the full model, the largest change of any single ablation, consistent with Proposition O.1: the second-stage value/output rank bounds every residual update. Replacing one or both FW-MLP stages with quadratic computation, outside the linear-time budget, lowers the best loss further and measures the headroom of the design beyond linear time (Table 20).

5.4 DOWNSTREAM LANGUAGE-MODEL EVALUATION

Finally we evaluate trained checkpoints at two scales, 340M parameters on 15B tokens and 1.3B parameters on 100B tokens of FineWeb-Edu (Penedo et al., 2024). Within a scale, models share the tokenizer, context length, optimizer recipe, token budget, and evaluation harness, and differ in the sequence-mixing block. The 16 tasks combine an Open LLM Leaderboard-style reasoning suite (Fourrier et al., 2024; Gao et al., 2024) with a general-ability suite, and we summarize them by average rank within a scale and by the number of tasks a model leads. In panel (d) of Figure 2, FW-MLP holds the best average rank at both scales, ahead of GSA (Zhang et al., 2024), Gated DeltaNet, Differential Transformer, DeltaNet, GLA, HGRN, HGRN2, RetNet (Qin et al., 2023; 2024; Sun et al., 2023), and the softmax Transformer. It scores above each other model on at least 11 of the 16 tasks in a paired comparison and stays first in 94% (1.3B) and 98% (340M) of task resamples (Table 23). In average score across the 16 tasks it leads the softmax Transformer by 2.13 points and the second-ranked GSA by 1.28 at 1.3B, and Gated DeltaNet by 0.84 at 340M. At 1.3B its largest per-task margins are on ARC-Easy and OpenBookQA (Table 22, Appendix R.7).

5.5 COST OF THE TRI-REGION SCORE

At fixed shapes, compiled forward throughput on one A100-80GB puts FW-MLP at $1.03\times$ the median of four recent linear-time mixers and FW-MLP-NoΔ at $1.17\times$ (Table 24, panel (d) of Figure 3), so the larger score class runs at a comparable throughput. For long context, panels (a) and (b) sweep sequence length on one L40S with both forms on identical inputs. The materialized score grows quadratically in memory and reaches its device limit past L=8,192; the recurrence grows linearly and evaluates L=131,072 in under 3 GiB at a roughly constant 0.085M tokens per second. The two forms agree to the float32 accumulation floor (Table 26). Latency grows linearly in the track count R and memory in the head count H (panel (c), Table 27); Appendix R.8 gives the protocols.

6 RELATED WORK

Fast weights and context-instantiated networks. Fast-weight programmers, fast-weight memories, and hypernetworks generate the weights of one network from another signal (Schmidhuber, 1992; Ba et al., 2016; Ha et al., 2017), and attention admits a reading in the same terms, with the prefix

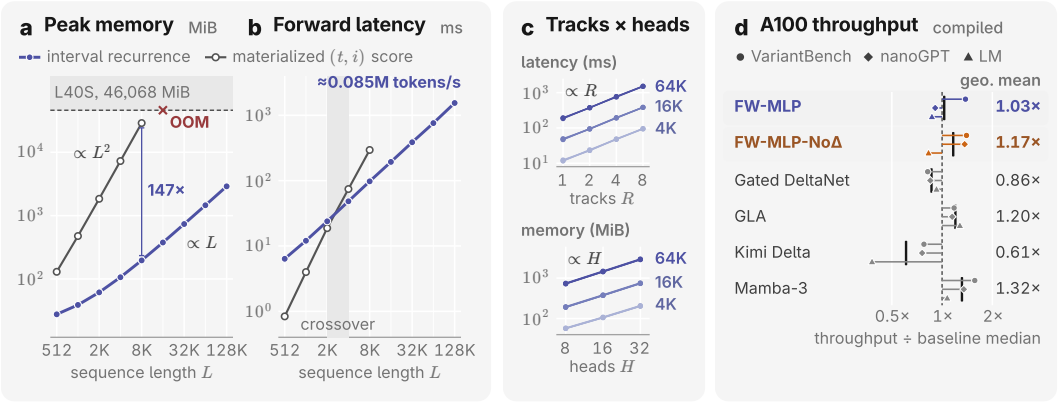

Figure 3: **(a, b)** Memory and latency of the materialized score (Eq. 10) and the recurrence (Eq. 11) of the FW-MLP mixing core on one L40S in BF16 at batch size 1, H=16, d_k=d_v=64, and R=4 (Table 25). **(c)** The recurrence over tracks R and heads H (Table 27). **(d)** Compiled throughput on one A100-80GB at three shapes, over the four-baseline median (Table 24).

supplying the hidden layer of a network applied to the current token (Schlag et al., 2021; Lu & Yang, 2026). FW-MLP extends that lineage from the value path to the pairwise score, with one fast MLP per causal region, and keeps its second stage a value-side memory per track (Appendix S.3).

Linear-time sequence mixers. Feature-map linear attention replaces the softmax kernel with an inner product of finite features, turning the pairwise sum into a running state (Katharopoulos et al., 2020; Choromanski et al., 2021; Zhai et al., 2021). Later designs shape the transport between two positions: retention applies a fixed exponential schedule (Sun et al., 2023), gated linear attention makes the decay input-dependent (Yang et al., 2024a), delta rules write an error-driven update into the state (Schlag et al., 2021; Yang et al., 2024b; 2025), and state-space blocks carry the same recurrence with structured parameterizations (Gu & Dao, 2024; Dao & Gu, 2024; Lahoti et al., 2026). Gated and differential variants of softmax attention shape the score on the quadratic side (Qiu et al., 2025; Ye et al., 2025). Each design prescribes the form in which the interval enters a score; in the view each is a setting of the two questions of Section 1, and Section 4 places several inside the FW-MLP interval branch.

Readings of trained attention and controlled evaluation. Trained attention has been read as in-context regression and gradient descent (Garg et al., 2022; Akyürek et al., 2023; von Oswald et al., 2023), Bayesian updating (Bai et al., 2023; Panwar et al., 2024), and test-time optimization (Sun et al., 2025; von Oswald et al., 2026). VariantBench draws on synthetic recall suites (Arora et al., 2024) and on time-series, recommendation, audio, and vision collections (THUML, 2024; Wang et al., 2026; Kang & McAuley, 2018; Harper & Konstan, 2015; Warden, 2018; Krizhevsky, 2009; Yang et al., 2023).

7 Conclusion

The forward computation of a causal attention layer is a network whose weights the context instantiates. Reading each of the three regions a query-key pair delimits with a learned fast MLP, under linear time, yields FW-MLP: an exact recurrence whose score class is dense on compact domains and contains linear attention and gated decays as smaller settings. Under a matched attention-side parameter budget it takes the best aggregate rank on a 12-mixer cross-domain suite under all five aggregation rules, the largest margin on a retrieval task gated by an interval statistic, the best nanoGPT validation loss, and the best average downstream rank at 340M and 1.3B parameters, with FW-MLP-NoΔ second on the suite and the retrieval task. The theory describes the function class, and the experiments measure which member of it training selects.

ETHICS STATEMENT

This work studies the parameterization of causal attention scores. The experiments train sequence models on public corpora and public benchmark collections: OpenWebText, FineWeb-Edu, the Time-Series Library task collection, MovieLens, Speech Commands, and the vision and reasoning datasets cited in Section 5 and Appendix R. Each dataset is used under its own license and for the purpose its curators describe, and no human-subject data was collected for this paper. A general-purpose sequence mixer inherits the capability profile of the models built on it, including the established risks of language models trained on web text, and it carries no safeguard of its own; deployments therefore need the same evaluation and monitoring that the surrounding system requires. The efficiency results lower the memory cost of long-context inference, which widens access to long-context models for research groups with modest hardware.

REPRODUCIBILITY STATEMENT

The implementation of the FW-MLP block, the parameter-budget solver, the implementation of every baseline mixer in the shared block, the task definitions of the controlled suite, the diagnostic generator, the measurement scripts for both efficiency experiments, and the drawing scripts for all four figures are in the anonymous repository `https://anonymous.4open.science/r/FasterWeight-MLP-FC32/`. Appendix R.1 states the matched-budget rule and the training configuration of each setting; Appendix R.2 records each baseline implementation and its source; Appendix O.1 gives the exact-count solver, including the realized parameter counts it verifies after construction; Appendix R.5 gives the sampling rule of the diagnostic; and Appendix R.8 gives the hardware, the library versions, the warmup and timing counts, and the numerical equivalence check between the materialized score and the recurrence. Every seed used in the paper is reported with its results. The proofs of all statements in Section 4 are in Appendix A, together with the assumptions each one uses.

USE OF LARGE LANGUAGE MODELS

The authors used a large language model as a writing and coding assistant. It drafted and revised manuscript prose, wrote the figure-drawing and table-formatting scripts, and helped with routine refactoring of experiment code. The research question, the architecture, the theoretical statements and their proofs, the experimental design, and the interpretation of results are the authors' own. The authors checked every theorem statement and proof, verified each reported number against the experiment records, inspected every figure, and take responsibility for the content of the paper.

REFERENCES

János Aczél. *Lectures on Functional Equations and Their Applications*, volume 19 of *Mathematics in Science and Engineering*. Academic Press, New York, 1966.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? Investigations with linear models. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=0g0X4H8yN4I`.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models. In *International Conference on Learning Representations*, 2024.

Jimmy Ba, Geoffrey E. Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In *Advances in Neural Information Processing Systems 29*, pp. 4331–4339, 2016. URL `https://papers.nips.cc/paper/6057-using-fast-weights-to-attend-to-the-recent-past`.

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The UEA multivariate time series classification archive, 2018. arXiv preprint arXiv:1811.00075, 2018.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Advances in Neural Information Processing Systems 36*, pp. 57125–57211, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/b2e63e36c57e153b9015fece2352a9f9-Abstract-Conference.html.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Łukasz Kaiser, David Benjamin Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with Performers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *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)*, pp. 2924–2936, 2019. doi: 10.18653/v1/n19-1300.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. arXiv preprint arXiv:1803.05457, 2018.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10041–10071. PMLR, 2024. URL https://proceedings.mlr.press/v235/dao24a.html.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems 35*, pp. 16344–16359, 2022. doi: 10.52202/068431-1189.

Ronen Eldan and Yuanzhi Li. TinyStories: How small can language models be and still speak coherent English? arXiv preprint arXiv:2305.07759, 2023.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open LLM leaderboard v2, 2024. URL https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 2024. URL https://github.com/EleutherAI/lm-evaluation-harness.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? A case study of simple function classes. In *Advances in Neural Information Processing Systems 35*, pp. 30583–30598, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c529dba08a146ea8d6cf715ae8930cbe-Abstract-Conference.html.

Aaron Gokaslan and Vanya Cohen. OpenWebText Corpus, 2019. URL https://skylion007.github.io/OpenWebTextCorpus/.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

David Ha, Andrew M. Dai, and Quoc V. Le. HyperNetworks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkpACe1lx.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):1–19, 2015. doi: 10.1145/2827872.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems 35*, pp. 30016–30030, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c1e2faff6f588870935f114ebe04a3e5-Abstract-Conference.html.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206, 2018. doi: 10.1109/ICDM.2018.00035.

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. arXiv preprint arXiv:2001.08361, 2020. URL https://arxiv.org/abs/2001.08361.

Andrej Karpathy. nanoGPT, 2022. URL https://github.com/karpathy/nanoGPT.

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*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, Wentao Li, Enzhe Lu, Weizhou Liu, Yanru Chen, Weixin Xu, Longhui Yu, Yejie Wang, Yu Fan, Longguang Zhong, Enming Yuan, Dehao Zhang, Yizhi Zhang, Y. T. Liu, Haiming Wang, Shengjun Fang, Weiran He, Shaowei Liu, Yiwei Li, Jianlin Su, Jiezhong Qiu, Bo Pang, Junjie Yan, Zhejun Jiang, Weixiao Huang, Bohong Yin, Jiacheng You, Chu Wei, Zhengtao Wang, Chao Hong, Yutian Chen, Guanduo Chen, Yucheng Wang, Huabin Zheng, Feng Wang, Yibo Liu, Mengnan Dong, Zheng Zhang, Siyuan Pan, Wenhao Wu, Yuhao Wu, Longyu Guan, Jiawen Tao, Guohong Fu, Xinran Xu, Yuzhi Wang, Guokun Lai, Yuxin Wu, Xinyu Zhou, Zhilin Yang, and Yulun Du. Kimi Linear: An expressive, efficient attention architecture. arXiv preprint arXiv:2510.26692, 2025.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles. arXiv preprint arXiv:2603.15569, 2026.

Brenden M. Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2873–2882. PMLR, 2018. URL https://proceedings.mlr.press/v80/lake18a.html.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024.

Jiecheng Lu and Shihao Yang. HyperMLP: An integrated perspective for sequence modeling. arXiv preprint arXiv:2602.12601, 2026. URL https://arxiv.org/abs/2602.12601.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dSprites: Disentanglement testing sprites dataset, 2017. URL https://github.com/google-deepmind/dsprites-dataset.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018. doi: 10.18653/v1/D18-1260.

Nikita Nangia and Samuel R. Bowman. ListOps: A diagnostic dataset for latent tree learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 92–99, 2018. doi: 10.18653/v1/N18-4013.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the Bayesian prism. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=HX5ujdsSon.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems 37*, pp. 30811–30849, 2024. doi: 10.52202/079017-0970.

Hao Peng, Jungo Kasai, Nikolaos Pappas, Dani Yogatama, Zhaofeng Wu, Lingpeng Kong, Roy Schwartz, and Noah A. Smith. ABC: Attention with bounded-memory control. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7469–7483, 2022. doi: 10.18653/v1/2022.acl-long.515.

Michael Poli, Armin W Thomas, Eric Nguyen, Pragaash Ponnusamy, Björn Deiseroth, Kristian Kersting, Taiji Suzuki, Brian Hie, Stefano Ermon, Christopher Ré, Ce Zhang, and Stefano Massaroli. Mechanistic design and scaling of hybrid architectures. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 40908–40950. PMLR, 2024. URL https://proceedings.mlr.press/v235/poli24a.html.

Zhen Qin, Songlin Yang, and Yiran Zhong. Hierarchically gated recurrent neural network for sequence modeling. In *Advances in Neural Information Processing Systems 36*, pp. 33202–33221, 2023. doi: 10.52202/075280-1442.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. HGRN2: Gated linear RNNs with state expansion. arXiv preprint arXiv:2404.07904, 2024.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *Advances in Neural Information Processing Systems 38*, pp. 100092–100118, 2025. doi: 10.52202/085713-3345. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/904e89bb4e632e75fb47f093b620b257-Abstract-Conference.html.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. OpenAI technical report, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. arXiv preprint arXiv:2311.12022, 2023.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, 2011.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9355–9366. PMLR, 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, 1992. doi: 10.1162/neco.1992.4.1.131.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. MuSR: Testing the limits of chain-of-thought with multistep soft reasoning. In *International Conference on Learning Representations*, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

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. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 57503–57522. PMLR, 2025. URL https://proceedings.mlr.press/v267/sun25h.html.

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. arXiv preprint arXiv:2307.08621, 2023. URL https://arxiv.org/abs/2307.08621.

Richard S. Sutton. The bitter lesson. Essay, 2019. URL http://www.incompleteideas.net/IncIdeas/BitterLesson.html.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023. doi: 10.18653/v1/2023.findings-acl.824.

THUML. Time-Series-Library. GitHub repository, 2024. URL https://github.com/thuml/Time-Series-Library.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pp. 5998–6008, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João 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*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35151–35174. PMLR, 2023. URL https://proceedings.mlr.press/v202/von-oswald23a.html.

Johannes von Oswald, Nino Scherrer, Seijin Kobayashi, Luca Versari, Songlin Yang, Sarthak Mittal, 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. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=xa3OnTb6c3.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems 37*, pp. 95266–95290, 2024. doi: 10.52202/079017-3018.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Chen Wang, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(10):11642–11662, 2026. doi: 10.1109/tpami.2026.3690845.

Pete Warden. Speech Commands: A dataset for limited-vocabulary speech recognition. arXiv preprint arXiv:1804.03209, 2018.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, 2017. doi: 10.18653/v1/W17-4413.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards AI-complete question answering: A set of prerequisite toy tasks. arXiv preprint arXiv:1502.05698, 2015.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. MedMNIST v2 – A large-scale lightweight benchmark for 2D and 3D biomedical image classification. *Scientific Data*, 10(1):41, 2023. doi: 10.1038/s41597-022-01721-8.

Songlin Yang and Yu Zhang. FLA: A Triton-based library for hardware-efficient implementations of linear attention mechanism, 2024. URL https://github.com/fla-org/flash-linear-attention.

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*, volume 235 of *Proceedings of Machine Learning Research*, pp. 56501–56523. PMLR, 2024a. 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. In *Advances in Neural Information Processing Systems 37*, pp. 115491–115522, 2024b. doi: 10.52202/079017-3668. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d13a3eae72366e61dfdc7eea82eeb685-Abstract-Conference.html.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=r8H7xhYPwz.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential Transformer. In *International Conference on Learning Representations*, 2025.

Tianzhu Ye, Li Dong, Yutao Sun, and Furu Wei. Differential Transformer V2, 2026. URL https://aka.ms/diff-transformer-v2.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019. doi: 10.18653/v1/P19-1472.

Shuangfei Zhai, Walter Talbott, Nitish Srivastava, Chen Huang, Hanlin Goh, Ruixiang Zhang, and Josh Susskind. An attention free transformer. arXiv preprint arXiv:2105.14103, 2021. URL https://arxiv.org/abs/2105.14103.

Yu Zhang, Songlin Yang, Ruijie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, Peng Zhou, and Guohong Fu. Gated slot attention for efficient linear-time sequence modeling. In *Advances in Neural Information Processing Systems 37*, pp. 116870–116898, 2024. doi: 10.52202/079017-3710. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d3f39e51f5f634fb16cc3e658f8512b9-Abstract-Conference.html.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. arXiv preprint arXiv:2311.07911, 2023.

A ROADMAP AND CLAIM-TO-THEOREM MAP

This appendix gives a mathematical framework for the claims behind *FasterWeight-MLP Attention* (FW-MLP). The central object is the causal score assigned by a query position t to a context position $i \le t$. The pair (t, i) induces three natural regions:

$$X_{1:t}, \qquad X_{1:i}, \qquad X_{i+1:t}. \tag{12}$$

The FW-MLP score exposes all three regions to learned fast MLPs:

$$A^{\text{FW-MLP}}_{t,i,h} = F^{q}_{t,h}(x_t; X_{1:t})\, F^{k}_{i,h}(x_i; X_{1:i})^{\top}\, F^{\Delta}_{t,i,h}(X_{i+1:t}; x_t), \tag{13}$$

where the first two factors are asymmetric endpoint fast MLPs and the third factor is a cancellative interval fast MLP.

Claim-to-theorem map. Table 2 maps the main theoretical claims to precise mathematical statements.

Table 2: Claim-to-theorem map for the FW-MLP theoretical framework.

Claim	Mathematical support	Primary theorem or proposition
Attention is a context-conditioned fast-weight generator.	Softmax attention is algebraically a context-instantiated two-layer network with context-sized hidden width; the unnormalized numerator of identity-feature linear attention is a context-instantiated dynamic linear layer; adding an activation yields a linear-time dynamic MLP template.	Propositions C.1, C.2, C.3
Softmax removes a shared prefix term and turns a bilinear score into a sequence-conditioned linear interval score.	Softmax invariance to common shifts gives $\mathrm{softmax}_i(s^\top A p_i) = \mathrm{softmax}_i(-s^\top A(s - p_i))$, and a fixed interval function g with $\mathrm{softmax}_i(s^\top A p_i) = \mathrm{softmax}_i(g(s - p_i))$ forces $A = 0$ and a constant g.	Proposition D.1, Corollary D.1
The only fixed interval or difference function that a raw bilinear linear attention score implements on its own is zero.	If $s^\top A p = g(s - p)$ on a rich input domain, then $A = 0$ and $g \equiv 0$.	Proposition D.2
Two-tower MLP endpoint scores reach a strictly larger class than raw bilinear scores, and separable rank governs exact representation.	$\phi(s)^\top \psi(p)$ is exactly a finite separable sum; ReLU towers approximate continuous difference functions on compact sets; bilinear-GLU/ReGLU towers exactly realize quadratic difference functions.	Theorems D.1 and D.2, Proposition D.4, Corollaries D.2 and D.3
Among continuous maps of the prefix sum, those whose differences depend on the interval sum alone are exactly the affine maps.	If $F(a + b) - F(a)$ depends only on b for all a, b and F is continuous, then F is affine.	Proposition E.1
The FW interval transport tracks depend exactly on $X_{i+1:t}$ and are linear-time computable.	Additive prefix potentials are exactly cancellative; the exponential transport satisfies $\exp[-(p_t - p_i)] = \gamma_t/\gamma_i$, yielding an exact recurrent/linear-attention implementation. The readout is endpoint-conditioned and enters the tri-region expansion as a query-side factor.	Lemma G.1, Corollary G.1, Theorem G.1
Multi-track interval transport is a learned exponential-basis fast MLP that contains fixed decay as its one-track case.	Finite real-coefficient mixtures of exponentials are dense in continuous functions of nonnegative additive interval statistics on compact sets; the endpoint-conditioned version holds for a readout class that is dense on the endpoint domain. Fixed exponential decay is the one-track case over a constant input coordinate.	Theorems H.1, H.2, Corollary H.1
The full FW-MLP score gives a tri-region neural parameterization.	Expanding the dot product and the track sum gives a finite sum of products over query-prefix, key-prefix, and interval regions. Generic tri-separable sums are dense on compact feasible three-region state domains; realizing arbitrary finite sums by the concrete product form requires enough width/heads/tracks and term-isolating allocation when cross terms would otherwise appear.	Theorems I.1, I.2, Corollary I.1

Logical strength of each statement. The appendix uses four logical strengths. Exact identities include additive cancellation and the recurrent γ_t/γ_i implementation, and they hold for every input. Impossibility results, such as Proposition D.2, are necessity statements for a specified score class under a rich-domain assumption. The two-tower theorem is an exact necessary-and-sufficient separable-rank characterization. Approximation results, such as Theorems D.2, H.2, and I.2, are compact-domain sufficiency statements that hold under the branch-richness assumptions stated with each theorem, and each asserts that a finite width, head count, or track count reaches any given accuracy.

B SETUP: CAUSAL REGIONS, PREFIX STATISTICS, AND SCORE CLASSES

Let $X_{1:T} = (x_1, \dots, x_T)$ be a causal sequence with $x_t \in \mathbb{R}^D$. We use the row-vector convention in neural-network equations. For $i \le t$, write

$$X_{1:t} = (x_1, \dots, x_t), \qquad X_{1:i} = (x_1, \dots, x_i), \qquad X_{i+1:t} = (x_{i+1}, \dots, x_t). \tag{14}$$

When a segment is empty, such as $X_{t+1:t}$, sums over that segment are zero and products over that segment are one.

Definition B.1 (Causal score class)**.** *A causal score class is a family of scalar functions*

$$A_{t,i} = \mathcal{A}_{\theta,t,i}(X_{1:t}), \qquad 1 \le i \le t \le T, \tag{15}$$

where θ denotes trained slow parameters. A value/output branch then maps scores to outputs by

$$y_t = \sum_{i \le t} A_{t,i} v_i, \tag{16}$$

possibly followed by normalization, gates, and output projections.

Definition B.2 (Tri-region score factor)**.** *A score is a* tri-region factor *if it can be written as*

$$A_{t,i} = Q_t(X_{1:t})\, K_i(X_{1:i})\, \Delta_{t,i}(X_{i+1:t}), \tag{17}$$

where $Q_t, K_i, \Delta_{t,i}$ are scalar functions. More generally, it is a finite tri-region separable sum of rank M if

$$A_{t,i} = \sum_{m=1}^{M} Q_{m,t}(X_{1:t}) K_{m,i}(X_{1:i}) \Delta_{m,t,i}(X_{i+1:t}). \tag{18}$$

The FW-MLP score in Equation 13 is a vector-valued version of Equation 17. Expanding the query-key dot product and the interval track mixture turns it into Equation 18; this is formalized in Theorem I.1.

Assumption B.1 (Rich prefix-difference domain)**.** *For impossibility results about prefix differences, we take tokens in $\mathbb{R}^d$ and assume that $t \ge 2$ and that for every pair $(s, p) \in \mathbb{R}^d \times \mathbb{R}^d$ there is a sequence for which*

$$s = \sum_{j=1}^{t} x_j, \qquad p = \sum_{j=1}^{i} x_j. \tag{19}$$

For example, take $x_1 = p$, $x_2 = s - p$, and all remaining tokens equal to zero. The exact-representation results use this assumption, and the approximation results of Sections F–I hold on compact domains.

Definition B.3 (Prefix sum and interval difference)**.** *For the prefix-sum setting, define*

$$s_t := \sum_{j=1}^{t} x_j, \qquad p_i := \sum_{j=1}^{i} x_j, \qquad r_{t,i} := s_t - p_i = \sum_{j=i+1}^{t} x_j. \tag{20}$$

The vector $r_{t,i}$ is the additive interval statistic of $X_{i+1:t}$.

Definition B.4 (Fast weight)**.** *A fast weight at time t is a weight matrix M_t given by a deterministic function of the input prefix and the slow parameters:*

$$M_t = G_\theta(X_{1:t}). \tag{21}$$

A fast-MLP layer applies an elementwise nonlinearity to a linear map whose weight is fast:

$$h_t = \sigma(x_t M_t + b_\theta). \tag{22}$$

The definition separates the trained slow parameters θ from the context-instantiated fast weights M_t. The forward graph produces the dynamic weights.

C ATTENTION AS A CONTEXT-CONDITIONED FAST-WEIGHT GENERATOR

This section formalizes the attention-as-dynamic-MLP viewpoint that motivates FW-MLP. HyperMLP frames autoregressive attention as a dynamic two-layer MLP whose weights are instantiated from context history (Lu & Yang, 2026); linear attention is the corresponding dynamic linear layer (Katharopoulos et al., 2020; Schlag et al., 2021). The propositions below make this algebra explicit.

C.1 SOFTMAX ATTENTION AS A DYNAMIC TWO-LAYER CONTEXT-INSTANTIATED NETWORK

For one head, let

$$q_t = x_t W_q, \qquad K_t = X_{1:t} W_k, \qquad V_t = X_{1:t} W_v. \tag{23}$$

The causal attention output is

$$o_t = \sigma\big(q_t K_t^\top\big)\, V_t W_o^\top = \sigma\big(x_t W_q W_k^\top X_{1:t}^\top\big)\, X_{1:t} W_v W_o^\top, \tag{24}$$

where σ is the row-wise softmax; the same algebra holds for ReLU- or GLU-style activations over the score vector.

Proposition C.1 (Softmax attention is a context-instantiated two-layer network)**.** *For fixed $X_{1:t}$, Equation 24 is a two-layer context-instantiated network applied to input x_t with first-layer weight*

$$M_t^{(1)} = W_q W_k^\top X_{1:t}^\top \tag{25}$$

and second-layer weight

$$M_t^{(2)} = X_{1:t} W_v W_o^\top. \tag{26}$$

Both weights are fast weights generated by the context prefix $X_{1:t}$.

Proof. Substitute the two definitions into a two-layer MLP template

$$o_t = \sigma(x_t M_t^{(1)}) M_t^{(2)}. \tag{27}$$

The dimensions match because $M_t^{(1)}$ maps from the feature dimension to width t, while $M_t^{(2)}$ maps from width t back to the output dimension. The hidden width grows with the prefix length, and both matrices depend on $X_{1:t}$. With softmax, the hidden activation is normalized across the context-width hidden layer, so the hidden units are coupled; with ReLU/SiLU/GLU-style alternatives it is an ordinary elementwise or gated hidden activation. In all cases, the displayed matrices are context-instantiated fast weights. □

C.2 LINEAR ATTENTION AS A DYNAMIC LINEAR LAYER

A simple unnormalized identity-feature linear attention head has the form

$$o_t = q_t \sum_{i \le t} k_i^\top v_i W_o^\top = x_t W_q W_k^\top X_{1:t}^\top X_{1:t} W_v W_o^\top. \tag{28}$$

Proposition C.2 (Identity-feature linear attention is a context-instantiated dynamic linear layer)**.** *Equation 28 can be written as*

$$o_t = x_t M_t, \qquad M_t = W_q W_k^\top X_{1:t}^\top X_{1:t} W_v W_o^\top. \tag{29}$$

The dynamic weight M_t can be updated from the prefix covariance

$$C_t = X_{1:t}^\top X_{1:t} = C_{t-1} + x_t^\top x_t. \tag{30}$$

Proof. The first identity is associativity of matrix multiplication. The update identity follows from

$$X_{1:t}^\top X_{1:t} = X_{1:t-1}^\top X_{1:t-1} + x_t^\top x_t. \tag{31}$$

The effective weight therefore changes with the prefix, and the layer is linear in x_t once the prefix is fixed. For a feature map ϕ, the same algebra holds with $\phi(q_t)$ in place of q_t and with the prefix statistic $\sum_{i \le t} \phi(k_i)^\top v_i$ in place of the covariance; the layer is then linear in $\phi(q_t)$. □

C.3 THE MINIMAL LINEAR-TIME DYNAMIC MLP

The most direct way to move from a dynamic linear layer to a dynamic MLP is to introduce a nonlinear activation after the context-generated first layer and, optionally, another context-generated second layer:

$$\sigma\left[x_t\left(W_{q1}W_{k1}^\top X_{1:t}^\top X_{1:t}W_{v1}W_{o1}^\top + W_{\text{static},1}\right)\right]\left(W_{q2}W_{k2}^\top X_{1:t}^\top X_{1:t}W_{v2}W_{o2}^\top + W_{\text{static},2}\right). \quad (32)$$

Proposition C.3 (Linear-time dynamic MLP template)**.** *For any activation σ, Equation 32 is a context-instantiated two-layer MLP whose fast weights are functions of the additive prefix statistic $C_t = X_{1:t}^\top X_{1:t}$. If C_t is maintained recurrently, the fast weights are available in time linear in the sequence length.*

Proof. Both displayed matrices are deterministic functions of C_t and the slow parameters. Since C_t has the recurrent update in Proposition C.2, one update per token maintains it, and the fast weights at all positions cost time linear in the sequence length. The activation σ turns the first fast linear map into a fast nonlinear layer. □

Corollary C.1 (Nonlinear dynamic MLPs strictly extend dynamic linear layers)**.** *Assume σ can realize a non-affine scalar function, as ReLU and SiLU do. Then the class of functions $x \mapsto \sigma(xM + b)N$ contains nonlinear functions of x. The dynamic MLP template therefore strictly extends the dynamic linear layer template before any context dependence is considered.*

D PREFIX-DIFFERENCE ALGEBRA OF BILINEAR AND TWO-TOWER SCORES

This section compares softmax attention and raw bilinear linear attention on prefix sums, the simplest setting in which prefix exposure and interval exposure can be compared exactly. Section H gives the corresponding compact-domain approximation results for general prefix states.

D.1 SOFTMAX QUOTIENTS OUT A SHARED PREFIX TERM

Let

$$s = s_t = \sum_{j=1}^{t} x_j, \qquad p_i = \sum_{j=1}^{i} x_j, \qquad r_i = s - p_i = \sum_{j=i+1}^{t} x_j. \quad (33)$$

Consider bilinear logits

$$\ell_i = s^\top A p_i. \quad (34)$$

Since $p_i = s - r_i$,

$$\ell_i = s^\top A(s - r_i) = s^\top As - s^\top Ar_i. \quad (35)$$

Proposition D.1 (Softmax removes the common prefix term)**.** *Let $\alpha_i = \text{softmax}_i(\ell_i)$ over $i \le t$. Then*

$$\alpha_i = \text{softmax}_i(-s^\top Ar_i). \quad (36)$$

Thus softmax converts the two-prefix bilinear logit into a sequence-conditioned linear interval score.

Proof. The term $c(s) = s^\top As$ in Equation 35 is shared by all i. Softmax is invariant under adding the same scalar to all logits:

$$\frac{e^{c(s)-s^\top Ar_i}}{\sum_{j\le t} e^{c(s)-s^\top Ar_j}} = \frac{e^{-s^\top Ar_i}}{\sum_{j\le t} e^{-s^\top Ar_j}}. \quad (37)$$

□

The effective interval score in Equation 36 is

$$g_s(r) = -s^\top Ar = -\left\langle A^\top s, r\right\rangle. \quad (38)$$

It is linear in the interval statistic r, with a direction conditioned on the full prefix s.

Corollary D.1 (Interval logits of bilinear softmax are sequence-conditioned and linear)**.** *Under Assumption B.1, suppose there exists a fixed function $g : \mathbb{R}^d \to \mathbb{R}$, independent of the full prefix s, such that for every realizable candidate set containing the terminal interval $r = 0$,*

$$\mathrm{softmax}_i(s^\top A p_i) = \mathrm{softmax}_i(g(s - p_i)). \tag{39}$$

Then $A = 0$ and g is constant on $\mathbb{R}^d$. Every nondegenerate interval logit that bilinear softmax supplies therefore belongs to the sequence-conditioned linear family

$$r \mapsto -s^\top A r, \tag{40}$$

whose direction $A^\top s$ changes with the full prefix s.

Proof. Softmax equality for a fixed candidate set is equivalent to equality of logits up to a common scalar shift. Compare any candidate interval r with the terminal interval 0. The common shift cancels in the log-ratio, giving

$$g(r) - g(0) = -s^\top A r. \tag{41}$$

The left-hand side is independent of s. By Assumption B.1, s can be varied while r is held fixed, so $Ar = 0$ for all $r \in \mathbb{R}^d$. Thus $A = 0$ and $g(r) - g(0) = 0$ for all r. □

D.2 Raw bilinear scores and fixed difference functions

Now remove softmax and ask whether the raw bilinear score itself can be a fixed function of $s - p$.

Proposition D.2 (The zero function is the only fixed difference function of a raw bilinear score)**.** *Under Assumption B.1, if*

$$s^\top A p = g(s - p) \qquad \forall s, p \in \mathbb{R}^d, \tag{42}$$

then $A = 0$ and $g \equiv 0$.

Proof. Set $p = 0$. Then

$$g(s) = s^\top A 0 = 0 \qquad \forall s, \tag{43}$$

so $g \equiv 0$. Substituting back gives $s^\top A p = 0$ for all s, p, which implies $A = 0$. □

Remark D.1 (Centering)**.** *If an external operation subtracts the terminal logit $z_t = s^\top A s$ from each logit $z_i = s^\top A p_i$, then*

$$s^\top A p_i - s^\top A s = -s^\top A (s - p_i). \tag{44}$$

Raw bilinear scoring with this centering step therefore recovers a sequence-conditioned linear interval score, as softmax does in Proposition D.1.

D.3 Two-tower scores and finite separable rank

A more general endpoint score is

$$h(s, p) = \phi(s)^\top \psi(p) = \sum_{m=1}^{M} \phi_m(s) \psi_m(p). \tag{45}$$

Theorem D.1 (Exact two-tower representation equals finite separable rank)**.** *For a scalar function $h : S \times P \to \mathbb{R}$, the following are equivalent:*

(i) There exist $M < \infty$ and maps $\phi : S \to \mathbb{R}^M$, $\psi : P \to \mathbb{R}^M$ such that $h(s, p) = \phi(s)^\top \psi(p)$.

(ii) There exist $M < \infty$ and scalar functions $u_m : S \to \mathbb{R}$ and $v_m : P \to \mathbb{R}$ such that

$$h(s, p) = \sum_{m=1}^{M} u_m(s) v_m(p). \tag{46}$$

Consequently, a difference function $g(s - p)$ is exactly representable by a finite two-tower dot product if and only if it has finite separable rank as a function of (s, p).

Proof. If $h = \phi^\top \psi$, expand coordinates and take $u_m = \phi_m$, $v_m = \psi_m$. Conversely, define $\phi = (u_1, \ldots, u_M)$ and $\psi = (v_1, \ldots, v_M)$. □

Corollary D.2 (Two-tower scores strictly contain raw bilinear scores)**.** *If the tower classes can represent constants and linear maps, then raw bilinear scores are contained in two-tower scores. The containment is strict. For $a \neq 0$, the score*

$$h(s, p) = a^\top (s - p) \tag{47}$$

is represented by

$$\phi(s) = \begin{bmatrix} a^\top s \\ 1 \end{bmatrix}, \qquad \psi(p) = \begin{bmatrix} 1 \\ -a^\top p \end{bmatrix}, \tag{48}$$

and it has no raw bilinear form: $s^\top A p = a^\top (s - p)$ at $p = 0$ would force $a^\top s = 0$ for all s.

D.4 ReLU towers: compact approximation

Proposition D.3 (Finite ReLU two-tower scores are piecewise linear in each endpoint)**.** *Let every coordinate of ϕ and ψ be a finite-width two-layer ReLU network. For fixed p, the function $s \mapsto \phi(s)^\top \psi(p)$ is continuous piecewise linear. In particular, no such score equals $\|s - p\|^2$ on all of $\mathbb{R}^d$.*

Proof. For fixed p, $\psi(p)$ is a constant vector, so $\phi(s)^\top \psi(p)$ is a finite linear combination of continuous piecewise-linear functions, hence continuous piecewise linear. Every continuous piecewise-linear function is affine on some nonempty open set, and $s \mapsto \|s - p\|^2$ is strictly convex on every such set, so the two functions differ. □

Theorem D.2 (Compact approximation by two-tower ReLU scores)**.** *Let $K \subset \mathbb{R}^d$ be compact and $g \in C(K - K)$. For every $\epsilon > 0$, there exist $M < \infty$ and vector-valued two-layer ReLU networks $\phi, \psi : K \to \mathbb{R}^M$ such that*

$$\sup_{s,p \in K} \left| \phi(s)^\top \psi(p) - g(s - p) \right| < \epsilon. \tag{49}$$

Proof. Let $h(s, p) = g(s - p)$, a continuous function on the compact set $K \times K$. The algebra generated by functions of s alone and functions of p alone contains constants and separates points of $K \times K$; by Stone–Weierstrass, finite sums $\sum_{m=1}^{M} u_m(s) v_m(p)$ are dense in $C(K \times K)$. Choose such a sum within $\epsilon/3$ of h. By the universal approximation theorem for two-layer ReLU networks, approximate each u_m and v_m uniformly well enough that the finite-sum error is at most another $\epsilon/3$, so the total error is below ϵ. Put these approximants into the coordinates of ϕ and ψ. □

D.5 Bilinear-GLU/ReGLU towers exactly realize quadratic differences

Define two polynomial-gated scalar blocks:

$$\mathrm{BGLU}(x) = a^\top \left[(U_1 x + b_1) \odot (U_2 x + b_2) \right] + c, \tag{50}$$

$$\mathrm{ReGLU}(x) = a^\top \left[(U_1 x + b_1) \odot \mathrm{ReLU}(U_2 x + b_2) \right] + c. \tag{51}$$

Proposition D.4 (Bilinear-GLU and ReGLU generate arbitrary quadratics)**.** *Finite linear combinations of BGLU or ReGLU features can exactly represent every quadratic polynomial*

$$q(x) = x^\top H x + a^\top x + b. \tag{52}$$

Proof. For BGLU, one hidden feature has the form $(u^\top x + b)(v^\top x + c)$, whose linear combinations generate constants, linear terms, and all monomials $x_k x_\ell$. For ReGLU, linear terms are obtained using a constant positive gate. Squares follow from

$$x_k^2 = x_k \,\mathrm{ReLU}(x_k) + (-x_k)\,\mathrm{ReLU}(-x_k), \tag{53}$$

while cross terms follow by polarization:

$$x_k x_\ell = \frac{1}{4} \left[(x_k + x_\ell)^2 - (x_k - x_\ell)^2 \right]. \tag{54}$$

□

Corollary D.3 (Exact quadratic difference scores)**.** *For every quadratic difference function*

$$g(r) = r^\top H r + a^\top r + b, \tag{55}$$

there exist finite-dimensional BGLU or ReGLU towers ϕ, ψ such that

$$\phi(s)^\top \psi(p) = g(s - p) \qquad \forall s, p \in \mathbb{R}^d. \tag{56}$$

Proof. The function $g(s-p)$ is a polynomial of total degree at most two in (s, p) and hence is a finite sum of products of polynomials in s and polynomials in p. Proposition D.4 realizes each polynomial factor in a tower coordinate; Theorem D.1 then gives the dot-product representation. □

Interpretation for FW-MLP. The endpoint branches in FW-MLP are fast MLP towers. By the results above, two-tower scores strictly contain raw bilinear scores (Corollary D.2), ReLU towers approximate every continuous difference function on a compact set (Theorem D.2), and BGLU or ReGLU towers exactly realize every quadratic difference function (Corollary D.3). The towers represent a difference function through a separable decomposition of finite rank, whose factors they compute from the two prefixes. An interval branch whose first layer is exactly cancellative computes the interval statistic by subtraction (Section E).

E Exact interval extraction by prefix subtraction

For an additive prefix statistic such as the prefix sum, $s_t - p_i = r_{t,i}$ is the statistic of the interval $X_{i+1:t}$. This section characterizes the continuous maps F of the prefix sum whose difference $F(s_t) - F(p_i)$ depends on $r_{t,i}$ alone.

Proposition E.1 (Continuous cancellative maps of the prefix sum are affine)**.** *Let $F : \mathbb{R}^d \to \mathbb{R}^m$ be continuous. Suppose there exists a function $G : \mathbb{R}^d \to \mathbb{R}^m$ such that*

$$F(a + b) - F(a) = G(b) \qquad \forall a, b \in \mathbb{R}^d. \tag{57}$$

Then F is affine:

$$F(x) = Lx + c \tag{58}$$

for some linear map L and constant c. Conversely, every affine map of this form satisfies Equation 57 with $G(b) = Lb$. Consequently, a continuous map of the prefix sum extracts the interval statistic exactly by subtraction precisely when it is affine.

Proof. Set $a = 0$ in Equation 57 to get $G(b) = F(b) - F(0)$. Define $H(x) = F(x) - F(0)$. Then

$$H(a + b) = H(a) + H(b) \qquad \forall a, b. \tag{59}$$

A continuous additive map is linear (Cauchy's functional equation), so H is linear. Hence $F(x) = H(x) + F(0)$ is affine. □

Remark E.1 (Consequence for the branch layout)**.** *For a nonlinear continuous MLP F on the prefix sum, $F(s_t) - F(p_i)$ varies with the endpoint p_i at fixed $r_{t,i}$. FW-MLP therefore applies a token-local nonlinearity before the sum and its cross-token nonlinearity after the subtraction:*

$$z^{\Delta}_{t,i} = p^{\Delta}_t - p^{\Delta}_i = \sum_{u=i+1}^{t} \rho(x_u), \tag{60}$$

with $\rho(x) = \lambda \phi_+(x W_\Delta)$. The interval branch is nonlinear in $z^{\Delta}_{t,i}$, and the operation that extracts the segment is linear and cancellative.

F Asymmetric endpoint fast MLPs

FW-MLP uses two different first-stage heads to produce the query-side and key-side endpoint MLPs. This section records the algebraic form and its implications.

F.1 PREFIX-INSTANTIATED ENDPOINT WEIGHTS

For a first-stage head a, define

$$M_{t,a}^{(1)} = W_{q,a}^{(1)} W_{k,a}^{(1)\top} X_{1:t}^{\top} X_{1:t} W_{v,a}^{(1)} W_{o,a}^{(1)\top} + W_{\text{static}}^{(1)}. \tag{61}$$

The hidden endpoint feature is

$$h_{t,a}^{(1)} = \sigma\left(x_t M_{t,a}^{(1)} + b_a^{(1)}\right). \tag{62}$$

For a second-stage head h, FW-MLP uses independent first-stage heads a_h^q and a_h^k:

$$F_{t,h}^{q} = \sigma\left(x_t M_{t,h}^{q} + b_{a_h^q}^{(1)}\right) W_{q,h}^{(2)}, \tag{63}$$

$$F_{i,h}^{k} = \sigma\left(x_i M_{i,h}^{k} + b_{a_h^k}^{(1)}\right) W_{k,h}^{(2)}. \tag{64}$$

F.2 PARALLEL FIRST-STAGE REALIZATION

A single first-stage computation produces both endpoint branches above, which remain mathematically distinct. Let H_2 denote the number of second-stage heads and N the batch size. The first-stage mixer is run once with a packed head dimension, and the GLU activation is realized by cross-activating paired head groups inside that single pass. Concretely, the mixer allocates $4H_2$ pre-gated head groups at the first stage and splits them into two halves along the head axis:

$$Z^{(1),\text{pre}} \in \mathbb{R}^{N\times T\times 4H_2\times d_r}, \qquad Z^{(1),\text{pre}} = [Z^{q,\text{val}} \,\|\, Z^{q,\text{gate}} \,\|\, Z^{k,\text{val}} \,\|\, Z^{k,\text{gate}}]_{\text{head}}. \tag{65}$$

A GLU activation is then applied head-wise inside each branch by multiplying a value head group by a gate head group of the same branch:

$$Z^{q} = Z^{q,\text{val}} \odot \varsigma\left(Z^{q,\text{gate}}\right), \qquad Z^{k} = Z^{k,\text{val}} \odot \varsigma\left(Z^{k,\text{gate}}\right), \tag{66}$$

where the gate nonlinearity ς is SiLU in the SwiGLU default and ReLU in the ReGLU alternative, and the endpoint activation σ of Equation 62 is the resulting GLU. The GLU activation halves the head axis from $4H_2$ to $2H_2$ post-activation groups, and the two branches are concatenated along the head axis:

$$Z^{(1)} \in \mathbb{R}^{N\times T\times 2H_2\times d_r}, \qquad Z^{(1)} = [Z^{q} \,\|\, Z^{k}]_{\text{head}}. \tag{67}$$

All $4H_2$ pre-gated head groups (equivalently all $2H_2$ post-activation groups) are computed by the same batched causal mixer or recurrent prefix scan. The GLU activation therefore keeps the first stage at a single pass that both branches share; the gate and value heads of each branch are produced side by side and multiplied elementwise. For each second-stage head h,

$$F_{t,h}^{q} = Z_{t,h}^{q} W_{q,h}^{(2)}, \qquad F_{i,h}^{k} = Z_{i,h}^{k} W_{k,h}^{(2)}, \tag{68}$$

with one second-stage head per query-key pair. At position i, the key-side representation $Z_{i,h}^{k}$ comes from the same prefix computation that produced all endpoint features, and the second stage reads this representation when forming scores for later queries. The first-stage head count thus runs $4H_2 \to 2H_2 \to H_2$: four pre-gated groups per second-stage head feed one scan, the GLU reduces them to a query group and a key group, and each second-stage head reads one such pair. Separate head-indexed slow parameters and projections make the two endpoint MLPs asymmetric, and both MLPs are evaluated inside the single first-stage scan.

Proposition F.1 (Endpoint branches are prefix-measurable fast MLPs)**.** *For fixed slow parameters, $F_{t,h}^{q}$ is measurable with respect to $(x_t, X_{1:t})$, and $F_{i,h}^{k}$ is measurable with respect to $(x_i, X_{1:i})$. If $C_t = X_{1:t}^{\top} X_{1:t}$ is maintained by $C_t = C_{t-1} + x_t^{\top} x_t$, then the fast weight in Equation 61 is available from a linear-time prefix statistic.*

Proof. The only data-dependent term in $M_{t,a}^{(1)}$ is $X_{1:t}^{\top} X_{1:t} = C_t$. The recurrence for C_t is additive, and all other operations are fixed matrix multiplications and an activation. The query statement follows directly, and the key statement follows by replacing t with i. □

F.3 Asymmetry is a strict expansion

Let z_t denote the endpoint state that a branch can access, for example $z_t = (x_t, C_t)$.

Definition F.1 (Shared and asymmetric endpoint score classes)**.** *A shared endpoint dot-product class has scores*

$$h_{\text{shared}}(z, z') = \phi(z)^\top \phi(z'). \tag{69}$$

An asymmetric endpoint dot-product class has scores

$$h_{\text{asym}}(z, z') = \phi_q(z)^\top \phi_k(z'), \tag{70}$$

where ϕ_q and ϕ_k are independent learned maps.

Proposition F.2 (Independent query/key endpoint heads strictly expand the score class)**.** *Let the same endpoint state domain $\mathcal{Z}$ be used on both sides, and suppose the branch class can represent constants and a nonconstant scalar function $a : \mathcal{Z} \to \mathbb{R}$. Then the asymmetric class strictly contains the shared class on $\mathcal{Z} \times \mathcal{Z}$.*

Proof. Every shared score is symmetric:

$$h_{\text{shared}}(z, z') = \phi(z)^\top \phi(z') = \phi(z')^\top \phi(z) = h_{\text{shared}}(z', z). \tag{71}$$

The asymmetric class can represent

$$h(z, z') = a(z) - a(z') \tag{72}$$

by pairing a on each side with a constant coordinate in the other tower. This score is antisymmetric and nonzero because a is nonconstant, so it lies outside the symmetric shared class. □

Remark F.1 (The two causal roles)**.** *In a causal score $A_{t,i}$, the query endpoint $(t, 1{:}t)$ and the key endpoint $(i, 1{:}i)$ have different semantics. The query branch conditions on the full current prefix, and the key branch on the prefix that ends at the context token. The asymmetric head split gives each role its own parameters.*

F.4 Endpoint towers as separable approximators over prefix states

The following theorem states what the endpoint dot product reaches when the endpoint branch classes are dense on the compact sets of prefix states that the model encounters.

Theorem F.1 (Endpoint two-tower approximation on compact prefix-state sets)**.** *Let $\mathcal{Z}_q$ and $\mathcal{Z}_k$ be compact sets of query-prefix and key-prefix endpoint states. Suppose the scalar coordinate classes available to the FW query and key branches are dense in $C(\mathcal{Z}_q)$ and $C(\mathcal{Z}_k)$, respectively. Then finite-dimensional endpoint dot products*

$$\Phi_q(z_q)^\top \Phi_k(z_k) \tag{73}$$

are dense in $C(\mathcal{Z}_q \times \mathcal{Z}_k)$.

Proof. The algebra of finite sums $\sum_m u_m(z_q) v_m(z_k)$ contains constants and separates points of $\mathcal{Z}_q \times \mathcal{Z}_k$, hence is dense by Stone–Weierstrass. Approximate each u_m and v_m by scalar coordinates of the query and key branches, and pack the coordinates into Φ_q and Φ_k. □

Remark F.2. *Endpoint fast MLPs thus turn the query-key dot product into a separable neural approximation over prefix states. An endpoint-only score reaches interval information through the relation between its two prefixes. The next section adds a branch that computes an interval statistic exactly by cancellation.*

G Cancellative interval fast MLP

For each head h and track r, FW-MLP defines a nonnegative token-local increment

$$\rho_{h,r}(x_u) = \lambda \phi_+(x_u W_{\Delta,h,r}) \geq 0, \tag{74}$$

and its prefix potential

$$p^{\Delta}_{t,h,r} = \sum_{u \le t} \rho_{h,r}(x_u). \tag{75}$$

The interval preactivation is

$$z^{\Delta}_{t,i,h,r} = p^{\Delta}_{t,h,r} - p^{\Delta}_{i,h,r} = \sum_{u=i+1}^{t} \rho_{h,r}(x_u). \tag{76}$$

The hidden activation is exponential,

$$\tau_{t,i,h,r} = \exp(-z^{\Delta}_{t,i,h,r}), \tag{77}$$

and the dynamic readout is

$$\omega_{t,h,r} = \frac{1 + \tanh(x_t W_{\omega,h,r})}{R}. \tag{78}$$

The interval output is

$$F^{\Delta}_{t,i,h} = \sum_{r=1}^{R} \omega_{t,h,r} \tau_{t,i,h,r}. \tag{79}$$

In the notation of Section 3.2, $\rho_{h,r}(x_u) = d_{u,h,r}$; $p^{\Delta}_{t,h,r}$ and $z^{\Delta}_{t,i,h,r}$ are the track coordinates of $p^{\Delta}_{t,h}$ and $z^{\Delta}_{t,i,h}$; and $\omega_{t,h,r}$ is the rth entry of $\omega_{t,h}$.

G.1 Exact interval dependence

Lemma G.1 (Additive prefix potentials are exactly cancellative)**.** *For any token-local map $\rho : \mathbb{R}^D \to \mathbb{R}^R$, define $p_t = \sum_{u \le t} \rho(x_u)$. Then*

$$p_t - p_i = \sum_{u=i+1}^{t} \rho(x_u), \tag{80}$$

which is a function only of the interval $X_{i+1:t}$.

Proof. All terms with $u \le i$ appear once in p_t and once in p_i and cancel exactly. □

Corollary G.1 (Interval tracks are exactly interval-measurable)**.** *For fixed slow parameters, each transport track*

$$\tau_{t,i,h,r} = \exp\left[- \sum_{u=i+1}^{t} \rho_{h,r}(x_u) \right] \tag{81}$$

is a function only of the interval $X_{i+1:t}$. The full scalar interval output $F^{\Delta}_{t,i,h} = \sum_r \omega_{t,h,r} \tau_{t,i,h,r}$ is an endpoint-conditioned interval factor: its coefficients depend on the current endpoint x_t, and its basis functions are exactly interval-measurable.

Proof. The track statement follows from Lemma G.1 composed with the exponential activation. The coefficient $\omega_{t,h,r}$ depends on x_t and can be absorbed into the query-side factor in the expanded score. The transport basis is therefore interval-only, and the whole branch has the form $F^{\Delta}(X_{i+1:t}; x_t)$. □

G.2 Linear-time implementation of exponential transport

Define

$$\gamma_{t,h,r} = \exp(-p^{\Delta}_{t,h,r}). \tag{82}$$

Then

$$\tau_{t,i,h,r} = \exp[-(p^{\Delta}_{t,h,r} - p^{\Delta}_{i,h,r})] = \frac{\gamma_{t,h,r}}{\gamma_{i,h,r}}. \tag{83}$$

Let $q_{t,h}, k_{i,h} \in \mathbb{R}^{d_q}$ and $v_{i,h} \in \mathbb{R}^{d_v}$ denote the second-stage query, key, and value vectors after endpoint processing. The explicit track output is

$$y_{t,h,r} = \sum_{i \le t} \left(q_{t,h} k_{i,h}^{\top} \right) \tau_{t,i,h,r} v_{i,h}. \tag{84}$$

Theorem G.1 (Exact linear-time track recurrence)**.** *For each head h and track r, define the recurrent matrix state*

$$S_{t,h,r} = S_{t-1,h,r} + \gamma_{t,h,r}^{-1} k_{t,h}^{\top} v_{t,h}, \qquad S_{0,h,r} = 0. \tag{85}$$

Then the explicit output in Equation 84 is exactly

$$y_{t,h,r} = (\gamma_{t,h,r} q_{t,h})\, S_{t,h,r}. \tag{86}$$

The full interval-weighted head output is

$$y_{t,h} = \sum_{r=1}^{R} \omega_{t,h,r} y_{t,h,r}. \tag{87}$$

Proof. Unroll Equation 85:

$$S_{t,h,r} = \sum_{i \le t} \gamma_{i,h,r}^{-1} k_{i,h}^{\top} v_{i,h}. \tag{88}$$

Multiplying by $\gamma_{t,h,r} q_{t,h}$ gives

$$(\gamma_{t,h,r} q_{t,h})\, S_{t,h,r} = \sum_{i \le t} \left(q_{t,h} k_{i,h}^{\top}\right) \frac{\gamma_{t,h,r}}{\gamma_{i,h,r}} v_{i,h}. \tag{89}$$

By Equation 83, the ratio equals $\tau_{t,i,h,r}$, proving Equation 86. Summing over tracks with readout weights gives Equation 87. The implementation carries the scaled state $\gamma_{t,h,r} S_{t,h,r}$, whose update multiplies the old state by the decay $e^{-d_{t,h,r}} \le 1$ (Equation 153). □

Corollary G.2 (Linear-time complexity)**.** *For sequence length T, H heads, R tracks, query-key rank d_q, and value rank d_v, the interval branch can be computed with recurrent memory $O(HRd_qd_v)$ and per-token update cost $O(HRd_qd_v)$, excluding the cost of generating the endpoint queries, keys, values, and track potentials. The total cost is linear in T.*

G.3 Stability and monotonicity of the base transport

Lemma G.2 (Base transport is bounded)**.** *If $\lambda \ge 0$ and $\phi_+ \ge 0$, then*

$$0 < \tau_{t,i,h,r} \le 1. \tag{90}$$

Moreover $\tau_{t,t,h,r} = 1$.

Proof. The interval preactivation $z^{\Delta}_{t,i,h,r}$ is a sum of nonnegative increments and hence is nonnegative. Therefore $\exp(-z^{\Delta}_{t,i,h,r}) \in (0, 1]$. When $i = t$, the sum is empty and $z^{\Delta}_{t,t,h,r} = 0$. □

Remark G.1 (Coefficients of the tracks)**.** *The exponential tracks form a bounded positive transport basis. In the score of Equation 13, each track is weighted by the product of its readout weight and the endpoint product $F^{q}_{t,h} F^{k\top}_{i,h}$, which takes either sign. Every such combination keeps the exact linear-time identity of Theorem G.1.*

H Expressivity of the interval branch

The interval branch is a one-hidden-layer network over a cancellative additive segment statistic, with exponential hidden units and an endpoint-conditioned readout. Its approximation results hold on compact domains, for track increments that realize a sufficiently rich nonnegative additive lift, and for a readout class that approximates the required coefficient functions.

H.1 Exponential bases over compact additive-statistic domains

Theorem H.1 (Density of exponential mixtures)**.** *Let $\mathcal{K}_{\Delta} \subset \mathbb{R}^m_+$ be compact. The linear span*

$$\operatorname{span}\left\{ z \mapsto e^{-\alpha^{\top} z} : \alpha \in \mathbb{R}^m_+ \right\} \tag{91}$$

is uniformly dense in $C(\mathcal{K}_{\Delta})$.

Proof. Let $\mathcal{E}$ be the finite linear span of these exponentials. It is a real algebra because

$$e^{-\alpha^\top z} e^{-\beta^\top z} = e^{-(\alpha+\beta)^\top z}, \tag{92}$$

and $\alpha + \beta \in \mathbb{R}^m_+$. It contains the constants ($\alpha = 0$). It separates points: if $z \neq z'$, then some coordinate j differs, and $e^{-\eta z_j} \neq e^{-\eta z'_j}$ for any $\eta > 0$. The Stone–Weierstrass theorem therefore gives uniform density in $C(\mathcal{K}_\Delta)$. □

Remark H.1 (The domain of the theorem)**.** *The theorem approximates functions of a chosen additive statistic z. Proposition H.1 gives a positional and token lift under which z determines the ordered interval. FW-MLP learns a finite lift that serves as a trainable transport basis.*

H.2 Approximation of endpoint-conditioned interval functions

Let $\rho_0 : \mathcal{X} \to \mathbb{R}^m_+$ be a nonnegative token lift and define

$$Z_{t,i} = \sum_{u=i+1}^{t} \rho_0(x_u). \tag{93}$$

A track that realizes the scalar increment $\rho_\alpha(x) = \alpha^\top \rho_0(x)$ computes

$$e^{-\alpha^\top Z_{t,i}} = \exp\left[- \sum_{u=i+1}^{t} \alpha^\top \rho_0(x_u) \right]. \tag{94}$$

The constructions below use a track family that contains or approximates these nonnegative scalar increments.

Theorem H.2 (Endpoint-conditioned interval approximation)**.** *Let $\mathcal{K}_x$ be a compact endpoint domain and let $\mathcal{K}_\Delta \subset \mathbb{R}^m_+$ be a compact domain of additive interval statistics. Let $G \in C(\mathcal{K}_x \times \mathcal{K}_\Delta)$. For every $\epsilon > 0$, there exist $R < \infty$, nonnegative vectors $\alpha_r \in \mathbb{R}^m_+$, and continuous coefficient functions $c_r \in C(\mathcal{K}_x)$ such that*

$$\sup_{x \in \mathcal{K}_x,\, z \in \mathcal{K}_\Delta} \left| G(x,z) - \sum_{r=1}^{R} c_r(x) e^{-\alpha_r^\top z} \right| < \epsilon. \tag{95}$$

If the readout class is dense in $C(\mathcal{K}_x)$ and, on a fixed finite horizon, the track-increment class can approximate each $x \mapsto \alpha_r^\top \rho_0(x)$ uniformly on the token domain, then the FW interval branch can approximate $G(x_t, Z_{t,i})$ uniformly on the compact domain.

Proof. By Theorem H.1, finite exponential sums are dense in $C(\mathcal{K}_\Delta)$. The algebra generated by functions of x alone and these exponential functions of z contains constants and separates points of $\mathcal{K}_x \times \mathcal{K}_\Delta$: differences in x are separated by continuous functions on $\mathcal{K}_x$, and differences in z are separated by exponential functions. Stone–Weierstrass gives finite sums $\sum_r c_r(x) e^{-\alpha_r^\top z}$ that approximate G. The final statement follows by uniformly approximating each of the finitely many coefficients c_r with the readout class and each scalar increment with the track-increment class. On a fixed finite horizon, per-token increment errors accumulate by at most a finite factor, and the exponential map is uniformly continuous on the resulting compact range. Hence the finitely many approximation errors can be chosen small enough to make the total uniform error arbitrarily small. □

Corollary H.1 (Fixed decay is a strict special case)**.** *A single hand-designed exponential decay*

$$\Delta_{t,i} = \exp\left[- \sum_{u=i+1}^{t} \delta(x_u) \right] \tag{96}$$

is recovered from the FW interval branch by taking $R = 1$, $\omega_{t,h,1} = 1$, and $\rho_{h,1} = \delta$. The parameterization realizes every increment of the form $\delta(x) = \lambda \phi_+(xw)$, which includes a constant decay rate when the token features contain a constant coordinate. It also gives $\omega_{t,h,1} = 1$ at $W_{\omega,h} = 0$ when $\psi(0) = 0$. Multi-track FW interval branches strictly extend this one-track family with finite mixtures of tracks and endpoint-conditioned coefficients.

Proof. The recovery is immediate from Equation 79. Strictness follows on any interval domain with nonempty interior: every nonzero one-track function ce^{-az} has constant logarithmic derivative $-a$, and $e^{-z} + e^{-2z}$ has logarithmic derivative $-(1 + 2e^{-z})/(1 + e^{-z})$, which is strictly increasing in z. Hence $e^{-z} + e^{-2z}$ lies outside the one-track family. □

H.3 Representing order-sensitive intervals by nonnegative positional lifts

Additive statistics become order-sensitive when token features include absolute or relative position or lag features, since the lift can allocate separate nonnegative channels to positions or lags.

Proposition H.1 (Nonnegative injective additive lift on finite horizons)**.** *Fix a maximum length T and a compact token domain $\mathcal{X} \subset \mathbb{R}^D$. Suppose each token is paired with a one-hot position identifier $e_u \in \mathbb{R}^T$. Define the nonnegative split with a mask channel*

$$\eta(x) = \big(1, \mathrm{ReLU}(x_1), \dots, \mathrm{ReLU}(x_D), \mathrm{ReLU}(-x_1), \dots, \mathrm{ReLU}(-x_D)\big) \in \mathbb{R}_+^{2D+1} \tag{97}$$

and the positional lift

$$\rho(u, x_u) = e_u \otimes \eta(x_u) \in \mathbb{R}_+^{T(2D+1)}. \tag{98}$$

Then the additive statistic

$$Z_{t,i} = \sum_{u=i+1}^{t} e_u \otimes \eta(x_u) \tag{99}$$

uniquely determines the ordered, position-indexed interval $X_{i+1:t}$ and the positions that belong to it.

Proof. The channels indexed by different e_u are disjoint. The first coordinate in the block for position u is one if $u \in \{i + 1, \dots, t\}$ and zero otherwise. When the mask coordinate is one, the remaining coordinates recover x_u because $x_j = \mathrm{ReLU}(x_j) - \mathrm{ReLU}(-x_j)$ coordinatewise. Hence the statistic recovers the ordered interval, including zero-valued tokens and the interval length. □

Corollary H.2 (Continuous finite-horizon interval functionals can factor through additive lifts)**.** *On a fixed finite horizon, with the positional split lift of Proposition H.1, every continuous function of the zero-padded ordered interval can be written as a continuous function of a nonnegative additive interval statistic. Therefore Theorem H.2 applies to such functionals with a sufficiently rich positional and token lift and a readout class dense on the endpoint domain.*

Proof. The lift map from the compact zero-padded interval domain to its image is continuous and injective; a continuous injective map from a compact space to a Hausdorff space is a homeomorphism onto its image. Hence any continuous interval functional factors through the additive statistic by composing with the continuous inverse on the image. □

Remark H.2 (Implementing the lift)**.** *With $T(2D+1)$ channels the lift of Proposition H.1 is injective on the finite horizon. Cheaper implementations use causal depthwise convolutions, learned position features, lag-indexed channels, or lower-dimensional learned lifts, and a smaller learned lift reaches a correspondingly smaller class of interval functionals.*

I The full FasterWeight-MLP score as a tri-region separable approximation

This section combines the endpoint and interval branches. For one second-stage head h, write

$$F_{t,h}^q = (F_{t,h,1}^q, \dots, F_{t,h,d_q}^q), \tag{100}$$

$$F_{i,h}^k = (F_{i,h,1}^k, \dots, F_{i,h,d_q}^k), \tag{101}$$

$$F_{t,i,h}^{\Delta} = \sum_{r=1}^{R} \omega_{t,h,r} \tau_{t,i,h,r}. \tag{102}$$

The score is

$$A_{t,i,h}^{\text{FW-MLP}} = \left(\sum_{a=1}^{d_q} F_{t,h,a}^q F_{i,h,a}^k \right) \left(\sum_{r=1}^{R} \omega_{t,h,r} \tau_{t,i,h,r} \right). \tag{103}$$

Theorem I.1 (FW-MLP scores are finite tri-region separable sums)**.** *The score in Equation 103 can be expanded as*

$$A^{\text{FW-MLP}}_{t,i,h} = \sum_{a=1}^{d_q} \sum_{r=1}^{R} \underbrace{F^q_{t,h,a}\omega_{t,h,r}}_{Q_{a,r,t}(X_{1:t})} \underbrace{F^k_{i,h,a}}_{K_{a,i}(X_{1:i})} \underbrace{\tau_{t,i,h,r}}_{\Delta_{r,t,i}(X_{i+1:t})} . \tag{104}$$

Thus one head has tri-region separable rank at most $d_q R$; multiple heads add more terms. The endpoint-conditioned readout is absorbed into the query-side factor, leaving the transport basis exactly interval-measurable.

Proof. Distribute the two finite sums in Equation 103. The query factor $F^q_{t,h,a}\omega_{t,h,r}$ is measurable with respect to the current endpoint and $X_{1:t}$, the key factor is measurable with respect to $X_{1:i}$, and the transport factor is interval-measurable by Corollary G.1. □

Theorem I.2 (Density of tri-region separable scores on compact feasible state domains)**.** *Let $\mathcal{Z}_q$, $\mathcal{Z}_k$, and $\mathcal{Z}_\Delta$ be compact Hausdorff state domains, and let $\mathcal{K} \subset \mathcal{Z}_q \times \mathcal{Z}_k \times \mathcal{Z}_\Delta$ be a compact set of feasible three-region states. Consider finite sums*

$$\sum_{m=1}^{M} u_m(z_q)v_m(z_k)w_m(z_\Delta), \tag{105}$$

with $u_m \in C(\mathcal{Z}_q)$, $v_m \in C(\mathcal{Z}_k)$, and $w_m \in C(\mathcal{Z}_\Delta)$, restricted to $\mathcal{K}$. These sums are uniformly dense in $C(\mathcal{K})$.

Proof. The restrictions of finite sums of products form a unital real algebra on $\mathcal{K}$. They separate points: if two feasible triples in $\mathcal{K}$ differ, then at least one coordinate differs. Continuous functions on compact Hausdorff spaces separate points, so a function on the differing coordinate, multiplied by constants on the other coordinates, separates the two triples. Stone–Weierstrass gives density on $\mathcal{K}$. □

Corollary I.1 (Conditional FW-MLP tri-region approximation)**.** *Assume the query endpoint branch, key endpoint branch, and interval branch can approximate the scalar factors required by Theorem I.2. Also assume the architecture has enough heads, channels, and tracks to allocate product terms without unwanted cross terms. For example, it can use term-isolating heads with one active query-key channel and one active interval track per product term, or an equivalent block-sparse or gated allocation. Then sums of FW-MLP score modules are dense in continuous score functions of the feasible three-region state (z_q, z_k, z_Δ) on compact domains.*

Proof. Theorem I.2 approximates the target by a finite sum of products $u_m(z_q)v_m(z_k)w_m(z_\Delta)$. By the branch approximation assumptions, each scalar factor can be approximated by the corresponding FW branch. The term-isolating allocation assumption prevents the product expansion in Theorem I.1 from introducing unrelated cross products between endpoint channels and interval tracks. Summing the allocated modules then approximates the finite tri-region separable expansion uniformly. The width this construction uses grows with the number of product terms. □

Remark I.1 (Why multiplication is natural)**.** *Multiplying query, key, and transport factors is the elementary tensor-product operation that generates finite separable approximations over product-like state descriptions. The dot-product dimension and the number of interval tracks control the rank of this approximation.*

J Relation to linear attention, softmax attention, and gated/delta variants

This section uses the previous results to position FW-MLP relative to common attention families.

J.1 Static linear attention as a degenerate FW-MLP case

Proposition J.1 (Linear attention is contained as a degenerate endpoint-only case)**.** *Assume the endpoint branch has an identity or bypass path that realizes $x \mapsto x$, or can realize the required linear projections on the compact operating domain. Then an FW-MLP head with its interval branch fixed to $F^{\Delta}_{t,i,h} = 1$ and its endpoint maps set to ordinary query/key projections reduces to*

$$A_{t,i} = q_t k_i^{\top}, \tag{106}$$

the raw score of linear attention.

Proof. Set the interval track preactivation to zero and choose a readout weight of one, so the interval factor is one. Use the identity/bypass or linear-region endpoint maps to output ordinary query and key projections. The FW score becomes the query-key dot product. □

Proposition J.2 (FW-MLP separates middle intervals on which token-local linear attention is constant)**.** *Suppose a token-local linear attention score has the form $A_{t,i} = q(x_t)^{\top} k(x_i)$. Then it takes the same value on any two sequences that share the endpoint tokens x_t and x_i, whatever their middle intervals $X_{i+1:t-1}$. FW-MLP separates such a pair whenever three conditions hold: its interval branch contains a track whose additive interval statistic takes different values on the two middle intervals, its endpoint maps are token-local projections with $q(x_t)^{\top} k(x_i) \neq 0$, and the track readout is nonzero.*

Proof. The token-local score depends only on x_t and x_i, so changing the middle tokens while keeping endpoints fixed leaves it unchanged. For FW-MLP, choose a track with nonzero readout and increment ρ such that $\sum_{u=i+1}^{t} \rho(x_u)$ differs between the two sequences. Then

$$F^{\Delta}_{t,i} = \exp\left[-\sum_{u=i+1}^{t} \rho(x_u)\right] \tag{107}$$

changes, and with token-local endpoint maps the endpoint dot product stays fixed and nonzero. Hence the final score changes. □

J.2 Cancellation in endpoint-only prefix models

The results of Section D show that sufficiently rich endpoint towers can approximate difference functions on compact domains. Such an endpoint-tower representation is a function of two prefixes, and the towers learn to cancel the part that the two prefixes share. The interval branch makes the cancellation exact in the first layer by construction.

Proposition J.3 (Endpoint-only interval dependence has separable-rank cost)**.** *Let $g(s-p)$ be represented by an endpoint-only dot product $\phi(s)^{\top}\psi(p)$ with M coordinates. Then $g(s-p)$ has separable rank at most M as a function of (s,p). Equivalently, if $g(s-p)$ has separable rank N on a domain, every exact endpoint-only dot-product representation has $M \geq N$. Likewise, every representation with uniform error ϵ has M at least equal to the ϵ-approximate separable rank, defined as the least separable rank of a function within uniform distance ϵ of $g(s-p)$.*

Proof. The first part is Theorem D.1. The second part is the contrapositive of the first in exact form. In approximate form it follows because $\phi(s)^{\top}\psi(p)$ has separable rank at most M and lies within uniform distance ϵ of $g(s-p)$. □

Remark J.1. *The branches play complementary roles. Endpoint towers model $X_{1:t}$ and $X_{1:i}$ directly, and the interval branch gives an explicit learned basis over $X_{i+1:t}$. The split frees endpoint rank for endpoint structure, since the interval branch already performs the cancellation.*

J.3 GATED DECAYS AS FORM-CONSTRAINED TRANSPORT

Many linear-time variants introduce a multiplicative transport between write time i and read time t. A simplified gated decay has

$$\Delta_{t,i}^{\mathrm{gate}} = \prod_{u=i+1}^{t} a_u, \qquad 0 < a_u \le 1. \tag{108}$$

Writing $a_u = e^{-\delta_u}$ gives

$$\Delta_{t,i}^{\mathrm{gate}} = \exp\left[-\sum_{u=i+1}^{t} \delta_u\right]. \tag{109}$$

This is exactly the one-track exponential transport form.

Proposition J.4 (One-track gated decay is contained in FW interval transport)**.** *Let $\delta_u = \delta(x_u) \ge 0$ with δ in the track-increment class; for the ReLU instantiation this class is $\{\lambda\phi_+(xw)\}$, which contains constant increments when the token features carry a constant coordinate. Then the transport in Equation 109 is represented by FW interval transport with $R = 1$, $\rho = \delta$, and $\omega_{t,h,1} = 1$.*

Proof. Substitute $\rho = \delta$ into Equation 76 and Equation 79. □

Remark J.2 (Designed update rules and the learned transport class)**.** *Gated linear attention and delta-rule variants add structured memory transport or memory modification to linear attention (Yang et al., 2024a;b; 2025). In the FW-MLP view, transport is one region of a larger neural score parameterization. Scalar decay recurrences are one-track special cases of that region, diagonal decays use one track per decayed coordinate (Proposition P.2). Delta-rule transports are approximated within the region on compact domains (Proposition P.3). FW-MLP expands the transport factor into a multi-track fast MLP with an endpoint-conditioned readout.*

J.4 SOFTMAX NORMALIZATION AND INTERVAL CANCELLATION

Softmax attention uses normalization to quotient out common logit shifts. In the prefix-sum setting of Section D, softmax normalization exposes a sequence-conditioned linear interval score. FW-MLP builds an explicit interval factor and keeps the final score compatible with linear-time causal mixing.

Proposition J.5 (Positions of softmax quotienting and FW interval cancellation in the computation graph)**.** *Softmax quotienting maps logits $\ell_i = c + s_i$ to the same distribution as s_i by removing a common term c. FW interval cancellation constructs $p_t - p_i$ before the score is formed. Softmax quotienting is a normalization invariance over positions, and FW interval cancellation is an architectural feature map over the interval.*

Proof. The softmax statement is $\mathrm{softmax}_i(c+s_i) = \mathrm{softmax}_i(s_i)$. The FW statement is Lemma G.1. Softmax applies its identity after logit construction, and FW-MLP applies its identity before the nonlinear transport activation. □

K LAYER-DEPTH INTERPRETATION

FW-MLP exposes the three causal regions used by its score before the second-stage value mix is formed. The endpoint branches compute prefix statistics $C_t = X_{1:t}^\top X_{1:t}$ and $C_i = X_{1:i}^\top X_{1:i}$, and the interval branch computes a cancellative potential difference $p_t^\Delta - p_i^\Delta$. A first-layer FW score therefore depends on all three causal regions (Proposition K.1).

Proposition K.1 (Graph availability of causal regions)**.** *If $h_j^{(0)}$ is token-local and a layer forms its pre-mixing score as $a_\theta(h_t^{(\ell-1)}, h_i^{(\ell-1)}, \mathrm{pos}(t,i))$, then a first-layer score is a function of x_t, x_i, and $\mathrm{pos}(t,i)$. Prefix-dependent scores arise only after earlier causal aggregation has made $h_j^{(\ell-1)}$ prefix-measurable. FW-MLP computes C_t, C_i, and $p_t^\Delta - p_i^\Delta$ inside the block, so $A_{t,i,h}^{\text{FW-MLP}}$ can depend on $X_{1:t}$, $X_{1:i}$, and $X_{i+1:t}$ before the final value/output mix.*

Proof. For token-local first-layer states, a_θ sees only the endpoint tokens and positions. A causal aggregation creates prefix-measurable states, which later token-local projections can use. FW-MLP forms the relevant prefix and interval statistics inside the first block, before the second-stage score, using Proposition F.1 and Lemma G.1. □

FW-MLP therefore exposes all three causal regions to learned score factors inside one block, from the first layer on.

L SEGMENTED SEQUENCES AND NUMERICAL CONSIDERATIONS

For packed training batches, every cumulative state used by FW-MLP is reset at sequence boundaries. Let $s(t)$ be the segment containing token t and define a segmented prefix potential for a token-local increment ρ_t by

$$p_t = \sum_{u \le t:\, s(u)=s(t)} \rho_u. \tag{110}$$

If i and t lie in the same segment, then

$$p_t - p_i = \sum_{u=i+1}^{t} \rho_u. \tag{111}$$

If they lie in different segments, the causal mask excludes the pair, so every interval the model uses lies within one segment. Thus segmented cumulative sums preserve the interval-cancellation identity exactly within each packed sequence.

The linear-time recurrence also uses ratios $\tau_{t,i,r} = \gamma_{t,r}/\gamma_{i,r}$, where $\gamma_{t,r} = \exp(-p^{\Delta}_{t,r})$. These ratios are invariant to a common positive rescaling of all $\gamma_{u,r}$ values within a segment. In a chunked recurrent evaluation, changing the scale convention at a boundary requires the stored state to be inversely rescaled, since the state stores terms proportional to $\gamma_{i,r}^{-1} k_i^\top v_i$. Log-space computation, chunkwise renormalization, and segmented resets therefore preserve the represented transport when applied consistently, and clamping of prefix potentials preserves it whenever the potentials stay within the clamp bounds. Finally, for nonnegative interval preactivation $z \ge 0$, the base map e^{-z} has derivative magnitude $e^{-z} \le 1$, so the scalar exponential transport itself is non-expansive.

M PUTTING THE PIECES TOGETHER: DESIGN IMPLICATIONS

M.1 THE INTERVAL BRANCH IN THE THREE-REGION VIEW

For a causal pair (t, i), endpoint scores naturally expose the current endpoint and the memory endpoint. In a prefix-enriched representation, these endpoints correspond to $X_{1:t}$ and $X_{1:i}$. The interval $X_{i+1:t}$ is the difference between them, and Section E identifies the affine-additive prefix maps as the ones whose difference depends on the interval alone. FW-MLP therefore gives $X_{i+1:t}$ its own exactly cancellative first layer.

M.2 WHAT SOFTMAX NORMALIZATION CONTRIBUTES

Section D shows that softmax removes a shared prefix term in a bilinear logit and exposes a suffix score. In the simplest bilinear setting, the exposed score is $-s^\top Ar$, a sequence-conditioned linear function of the interval statistic. FW-MLP learns a nonlinear interval MLP and retains a linear-time causal mixer. The two mechanisms compose: softmax supplies a normalization quotient, and FW-MLP supplies an explicit learned interval feature map.

M.3 WHAT THE MULTI-TRACK INTERVAL BRANCH ADDS TO A GATED DECAY

Gated decays introduce an interval transport, and Proposition J.4 shows that the one-track exponential form is contained in FW-MLP. The FW-MLP interval branch generalizes this form in two directions: multiple learned tracks provide a finite exponential dictionary, and endpoint-conditioned readouts make the transport a one-hidden-layer fast MLP over the interval. The universal-approximation

statement holds under the compact-domain, rich-lift, and readout-richness assumptions of Section H. A finite implementation trains a finite dictionary from this family.

M.4 What endpoint asymmetry adds

The query endpoint and the key endpoint correspond to different causal roles. Proposition F.2 shows that the dot-product score class of independent endpoint towers strictly contains that of shared towers. FW-MLP therefore routes different first-stage heads a_h^q and a_h^k into the two boundaries.

M.5 The final product as a neural tensor-product construction

The product in Equation 13 is a neural tensor-product construction. Finite sums of products are dense on compact product domains and, by restriction, on compact feasible three-region state domains, and the FW expansion in Theorem I.1 is exactly such a finite sum over the three causal regions.

N Function-class inclusions and rank accounting

This section collects the architecture-level inclusions in one place and states how FW-MLP relates to common score families. Table 3 summarizes the resulting score-class hierarchy at a glance; the rest of the section gives the formal definitions and containment statements.

Table 3: FW-MLP treats the attention score as a learnable causal score class over three regions. Linear attention, retention, and gated decays are exact special cases of this class, and delta-rule transports are approximated within it on compact domains. The full FW-MLP score exposes all three causal regions.

Mixer family	Score object	Object exposed to learning
Softmax attention	$\exp(q_t k_i^\top)$ with row normalization	Rich normalized endpoint kernel; materialized pairwise scores are quadratic.
Linear attention	$\phi(q_t)^\top \phi(k_i)$	Linear-time endpoint bilinear form; interval dependence is only implicit.
Gated and retentive variants	Endpoint score times a prescribed decay	Hand-designed decay on the interval between the endpoints.
Delta-rule variants	Endpoint score plus a prescribed state update	Hand-designed state update on the interval between the endpoints.
FW-MLP-NoΔ (endpoint-only)	$F_{t,h}^q F_{i,h}^{k\top}$	Query-side and key-side prefixes are learned fast-MLP towers; interval effects enter through separable rank.
Full FW-MLP	$F_{t,h}^q F_{i,h}^{k\top} F_{t,i,h}^{\Delta}$	Query prefix, key prefix, and interval are all learned fast-MLP factors.

Definition N.1 (Four schematic score classes)**.** *Let z_t^q denote a query endpoint state, z_i^k a key endpoint state, and $z_{t,i}^{\Delta}$ an interval state. Define the following schematic classes:*

$$\mathcal{B}_{\mathrm{lin}} = \{(t,i) \mapsto q(x_t)^\top k(x_i)\}, \tag{112}$$

$$\mathcal{B}_{\mathrm{prefix}} = \{(t,i) \mapsto q(z_t^q)^\top k(z_i^k)\}, \tag{113}$$

$$\mathcal{G}_{\mathrm{decay}} = \{(t,i) \mapsto q(z_t^q)^\top k(z_i^k) \exp[-\sum_{u=i+1}^{t} \delta(x_u)]\}, \tag{114}$$

$$\mathcal{F}_{\text{FW-MLP}} = \left\{ (t,i) \mapsto q(z_t^q)^\top k(z_i^k) \sum_{r=1}^{R} c_r(z_t^q) e^{-\sum_{u=i+1}^{t} \rho_r(x_u)} \right\}. \tag{115}$$

The definitions suppress head indices and value/output projections. The four classes are schematic, and the concrete implementation realizes finite-dimensional parameterized subsets of them.

Proposition N.1 (Basic inclusions). *When the branches can realize constant channels and identity or linear endpoint maps,*

$$\mathcal{B}_{\mathrm{lin}} \subseteq \mathcal{B}_{\mathrm{prefix}} \subseteq \mathcal{F}_{\text{FW-MLP}}, \qquad \mathcal{G}_{\mathrm{decay}} \subseteq \mathcal{F}_{\text{FW-MLP}}. \tag{116}$$

Proof. The first inclusion follows because prefix endpoint states can include token-local states. The second follows by setting the FW interval factor to one. The third follows from Proposition J.4: a gated decay is the one-track interval branch with constant readout. □

Proposition N.2 (Strict interval expansion over endpoint-only scores). *Let the endpoint states z_t^q and z_i^k be finite-dimensional compressed prefix states, as in linear-time layers, and assume two sequences share (z_t^q, z_i^k) and differ in their additive interval statistic $Z_{t,i}$. Then every endpoint-only score in $\mathcal{B}_{\mathrm{prefix}}$ takes equal values on the two sequences, and $\mathcal{F}_{\text{FW-MLP}}$ separates them whenever it contains an interval track that takes different values on their two statistics $Z_{t,i}$.*

Proof. Endpoint-only scores are functions only of (z_t^q, z_i^k), so equal endpoint states imply equal scores. If $Z_{t,i}$ differs and an interval track $\tau_r(Z_{t,i})$ (for example $e^{-\alpha^\top Z_{t,i}}$ under the exponential dictionary) separates the two values, then setting the endpoint factors and readout coefficients to nonzero constants makes the FW score differ. □

N.1 Fast-weight rank budget

The first-stage endpoint fast weight has the form

$$M_{t,a}^{(1)} = W_{q,a}^{(1)} W_{k,a}^{(1)\top} C_t W_{v,a}^{(1)} W_{o,a}^{(1)\top} + W_{\mathrm{static}}^{(1)}, \qquad C_t = X_{1:t}^\top X_{1:t}. \tag{117}$$

Proposition N.3 (Low-rank control of dynamic endpoint weights). *Ignoring the static shortcut, the dynamic part of $M_{t,a}^{(1)}$ has rank at most*

$$\min\{r_{qk}^{(1)}, r_{vo}^{(1)}, \mathrm{rank}(C_t)\}. \tag{118}$$

Thus the projection ranks control the dynamic fast-weight rank of the endpoint branch and keep the dynamic part of the instantiated matrix within rank $\min\{r_{qk}^{(1)}, r_{vo}^{(1)}\}$.

Proof. The dynamic part is a product

$$W_{q,a}^{(1)} \left(W_{k,a}^{(1)\top} C_t W_{v,a}^{(1)} \right) W_{o,a}^{(1)\top}. \tag{119}$$

The rank of a matrix product is bounded by the minimum rank of its factors. The middle factor has rank at most $\mathrm{rank}(C_t)$ and at most the two projection ranks. □

Proposition N.4 (Score-rank accounting). *For one head with query-key width d_q and R interval tracks, the expanded FW-MLP score has tri-region separable rank at most $d_q R$. If there are H heads, the total rank contributed before the output projection is at most $\sum_{h=1}^{H} d_{q,h} R_h$.*

Proof. This is Theorem I.1 summed over heads. □

Remark N.1 (Independent rank knobs). *The endpoint width d_q and the interval track count R control different axes of the separable approximation. Increasing d_q enriches query-key endpoint interactions, and increasing R enriches transport basis functions. The interval branch therefore adds interval rank through R, independently of the endpoint dot-product width d_q.*

N.2 A COMPARISON THEOREM

Theorem N.1 (FW-MLP is a strict neural extension of bilinear linear attention under finite endpoint states)**.** *Consider a finite-state setting in which endpoint-only scores are functions of (z_t^q, z_i^k), and two sequences exist that share (z_t^q, z_i^k) and differ in their additive interval statistic $Z_{t,i}$. Then FW-MLP strictly extends endpoint-only bilinear/prefix scores by adding functions of a cancellative interval statistic. One-track gated decay is a special case. Under the compactness, nonnegative-lift, readout-richness, and term-allocation assumptions of Theorem H.2 and Corollary I.1, the multi-track interval construction supplies a sufficient approximation family for continuous functions of nonnegative additive interval statistics.*

Proof. Containment and strictness follow from Propositions N.1 and N.2, the gated-decay case from Proposition J.4, and the approximation statement from Theorem H.2 and Corollary I.1. □

O READOUT RANK, WORKING-RANK ALLOCATION, AND MATCHED MIXER BUDGETS

The FW-MLP parameterization introduces several rank knobs, and the ranks play distinct roles. Under the attention-as-dynamic-MLP view, the readout and action side controls the subspace of residual updates, and the scoring and routing side controls which context-dependent hidden units are selected. HyperMLP makes this distinction explicit: its Theorem 2.5 states that shrinking the second-layer/readout (VO) core directly constrains the update subspace and that shrinking the first-layer/routing (QK) core primarily constrains conditioning and routing geometry (Lu & Yang, 2026). Its experiments also report that VO compression harms tasks requiring diverse update directions more directly than QK compression does, which matches the predicted readout-rank bottleneck.

Proposition O.1 (Readout-rank bottleneck in a dynamic two-layer block)**.** *Consider a residual dynamic two-layer block in row-vector convention*

$$f_X(x) = x + \sigma(xA_X)B_X, \qquad A_X \in \mathbb{R}^{d\times m}, \quad B_X \in \mathbb{R}^{m\times d}. \tag{120}$$

If $\operatorname{rank}(B_X) \le r_{\text{out}}$ for all contexts X, then every residual update lies in the context-dependent subspace $\operatorname{row}(B_X)$, of dimension at most r_{out}:

$$f_X(x) - x \in \operatorname{row}(B_X), \qquad \dim \operatorname{row}(B_X) \le r_{\text{out}}. \tag{121}$$

The rank of A_X acts through the hidden activations, because the updates span the subspace $\operatorname{span}\{\sigma(xA_X)\}B_X$ of $\operatorname{row}(B_X)$. Lowering the rank of A_X changes the routing features and activation boundaries that generate this subspace.

Proof. For every hidden activation $h = \sigma(xA_X) \in \mathbb{R}^m$, the update is hB_X, a linear combination of the rows of B_X. Hence it belongs to $\operatorname{row}(B_X)$, whose dimension is $\operatorname{rank}(B_X) \le r_{\text{out}}$. For the second statement, the updates are $\sigma(xA_X)B_X$, so they span the image under B_X of the span of the hidden activations, which A_X determines through the map $x \mapsto h$. □

For FW-MLP this principle places the second-stage VO rank highest in the allocation order. The second stage is the final score-to-value and action layer, and reducing $r_{vo}^{(2)}$ reduces the working rank of the equivalent dynamic MLP exactly where information is written back into the residual stream. The second-stage QK rank $r_{qk}^{(2)}$ controls final endpoint scoring geometry and is a routing and conditioning rank, which places it second. The first-stage ranks shape the endpoint fast MLPs that produce F^q and F^k, and the second-stage mixer mediates their effect. Within the first stage the same readout-first principle places $r_{vo}^{(1)}$ above $r_{qk}^{(1)}$. The resulting priority is

$$r_{vo}^{(2)} > r_{qk}^{(2)} > r_{vo}^{(1)} > r_{qk}^{(1)}. \tag{122}$$

The solver of Appendix O.1 converts four allocation weights into working ranks. The weights are

$$\left(\alpha_{qk}^{(1)}, \alpha_{qk}^{(2)}, \alpha_{vo}^{(1)}, \alpha_{vo}^{(2)}\right) = (0.10, 0.30, 0.15, 0.45). \tag{123}$$

The solver fixes the mixer parameter budget $B_{\text{mixer}} = \mu d^2$, where μ is the attention budget of Appendix R.1. It subtracts the rank-independent parameter counts, such as those of shortcuts, biases, convolutional terms, and interval-generator overheads, which leaves a variable-rank budget B_{var}. For each rank group $j \in \{qk1, qk2, vo1, vo2\}$, let c_j denote the exact number of trainable parameters contributed by one additional unit of the corresponding working rank, counted over all heads and including the enabled value gates and projections. With normalized weights $w_j = \alpha_j / \sum_\ell \alpha_\ell$, the unrounded target ranks are given by Equation 132, and the solver rounds them to the legal head and RoPE granularities and adjusts them to use the remaining budget slack. The conversion matters because the VO groups carry different per-rank costs from the QK groups. The second-stage VO group includes the value projection, output projection, and two value/output gates. The second-stage QK working-rank group pays for the learned query/key fast-weight projections, and the interval and scoring generator overhead is counted separately as fixed or branch-specific budget terms. The weights follow the priority order above, which rests on Proposition O.1 and the HyperMLP rank asymmetry (Lu & Yang, 2026).

O.1 Exact parameter-budget solver

The empirical protocol fixes a mixer (attention-side) parameter budget before training, and the solver instantiates each variant under that budget. The FW-MLP solver converts the target μd^2 into legal integer working ranks and counts every trainable parameter of the constructed module, so the ranks follow from the budget alone and FW-MLP receives the same mixer budget as every baseline.

Let d be the residual width, d_r the first-stage hidden width, μ the attention budget, and

$$P_{\max} = \lfloor \mu d^2 \rfloor \tag{124}$$

be the target mixer budget. The solver chooses the rank vector

$$r = (r_{qk}^{(1)}, r_{qk}^{(2)}, r_{vo}^{(1)}, r_{vo}^{(2)}) \tag{125}$$

for first-stage QK, second-stage QK, first-stage VO, and second-stage VO groups. The trainable parameter count is affine in these four ranks:

$$P(r) = C + \sum_{j \in \{qk1, qk2, vo1, vo2\}} c_j r_j. \tag{126}$$

Here C collects every term independent of the four ranks, including static shortcuts, first-stage biases, optional output-projection bias, beta maps for delta-rule backbones, fixed depthwise-convolution terms, and the interval-generator/track-readout projections. The coefficients c_j count the exact number of trainable parameters added by one unit of the corresponding working rank, counted over all heads and including enabled gates and projections.

With linear-layer bias indicator $b \in \{0, 1\}$, QK gate multiplicity m_{qk}, VO gate multiplicity m_{vo}, convolution kernel size κ, and binary flags $I_\cdot$ for enabled submodules, the exact-count coefficients used by the solver are

$$\begin{aligned}
c_{qk1} &= (d + b)\big(I_q^{(1)} + I_k^{(1)} + m_{qk} I_{g,qk}^{(1)}\big), \\
c_{qk2} &= d_r\big(I_q^{(2)} + I_k^{(2)}\big) + m_{qk}(d + b) I_{g,qk}^{(2)} + 2\kappa I_{\text{conv}} I_{\text{delta}}^{(2)}, \\
c_{vo1} &= m_{vo}(d + b) I_{g,vo}^{(1)} + (d + b) I_v^{(1)} + d_r I_o^{(1)}, \\
c_{vo2} &= m_{vo}(d + b) I_{g,vo}^{(2)} + (d + b) I_v^{(2)} + d I_o^{(2)} + \kappa I_{\text{conv}} I_{\text{delta}}^{(2)}.
\end{aligned} \tag{127}$$

The last terms in c_{qk2} and c_{vo2} appear when the second-stage backbone is a delta-rule mixer whose depthwise convolutions scale with the second-stage ranks. For the default linear-time FW-MLP configuration with no QK gates, two VO gates, no linear biases, and a linear-attention second stage, these reduce to $c_{qk1} = 2d$, $c_{qk2} = 2d_r$, $c_{vo1} = 3d + d_r$, and $c_{vo2} = 4d$. The interval generator and non-rank convolutions are included in C.

The constant term is also counted from the module configuration. For the default configuration,

$$C = C_{\text{shortcut}} + C_{\text{bias}} + C_{\text{beta}} + C_{\text{interval}} + C_{\text{conv}}, \tag{128}$$

where the two shortcuts contribute dd_r each when enabled, the first-stage bias contributes $H_1 d_r$, an output-projection bias contributes d when enabled, and delta-rule beta maps contribute $H_s(d+b)$ for each delta stage s. If the second-stage backbone is linear attention, the interval generator contributes

$$C_{\text{interval}} = (d+b)a_\Delta RH_2 + 2(d+b)RH_2 + I_{\text{phase}}H_2, \tag{129}$$

where R is the number of interval tracks, H_2 is the number of second-stage heads, $a_\Delta = \max(1, I_{\text{decay}} + I_{\text{phase}})$ is the number of active interval channels, and the final term counts a phase-gain vector when a rotation/phase transport channel is enabled. For the default linear-attention second stage the non-rank convolutional constants are $3\kappa d + 2\kappa d_r H_2 + \kappa d$, from the three first-stage depthwise convolutions, the two second-stage hidden convolutions, and the second-stage value convolution.

The legal rank granularities enforce both head divisibility and even RoPE dimensions. If g is the requested base granularity, H_1 is the first-stage head count, and H_2 is the second-stage head count, the solver uses

$$(g_{qk1}, g_{qk2}, g_{vo1}, g_{vo2}) = (\text{lcm}(g, 2H_1), \text{lcm}(g, 2H_2), \text{lcm}(g, H_1), \text{lcm}(g, H_2)). \tag{130}$$

Thus every returned rank can be reshaped into heads, and every per-head QK dimension is even for RoPE.

After subtracting constants, the remaining variable-rank budget is

$$B_{\text{var}} = P_{\max} - C. \tag{131}$$

The desired allocation weights α_j are normalized over active rank groups, $w_j = \alpha_j / \sum_{\ell \in \mathcal{A}} \alpha_\ell$, and the unrounded ratio-preserving target is

$$r_j^* = w_j \frac{B_{\text{var}}}{\sum_{\ell \in \mathcal{A}} w_\ell c_\ell}. \tag{132}$$

The solver rounds each active rank down to its legal granularity,

$$r_j \leftarrow g_j \left\lfloor r_j^* / g_j \right\rfloor, \tag{133}$$

with a one-step floor for active branches when nonzero ranks are required. If the one-step floors push $P(r)$ above $P_{\max}$, ranks are decremented by legal steps until $P(r) \le P_{\max}$. The decrement order protects the second-stage VO rank: it reduces $qk1$, $qk2$, and $vo1$ first and reduces $vo2$ last.

The remaining slack is then filled in two passes. A greedy pass repeatedly adds a legal rank step that fits the budget and minimally increases the squared deviation from the unrounded target,

$$(r_j + g_j - r_j^*)^2 - (r_j - r_j^*)^2, \tag{134}$$

using the allocation weights as a tie-breaker. A local meet-in-the-middle search then examines a small neighborhood around the greedy allocation and minimizes the lexicographic objective

$$\big(P_{\max} - P(r),\ \sum_j (r_j/g_j - r_j^*/g_j)^2\big). \tag{135}$$

The solver accepts the local correction when it reduces slack by more than a fixed threshold, which keeps capacity from moving between branches for a gain of a few parameters.

After construction, the solver computes

$$P_{\text{actual}} = \sum_{\theta \in \Theta_{\text{module}}} |\theta| \tag{136}$$

over the trainable parameters of the module and requires $P_{\text{actual}} = P(r) \le P_{\max}$. Every reported configuration satisfies this check, so every FW-MLP block is instantiated from the same target budget as the corresponding baseline or ablation, with the trainable count verified before training. The same budget-matching principle applies to the other attention variants: their rank, width, or expansion parameters are chosen under the mixer (attention-side) parameter budget of the experiment, and the realized trainable parameter count is recorded for every configuration.

O.2 VALUE/OUTPUT GATING AS READOUT-SIDE DIAGONAL MODULATION

The second stage uses two value/output gates on the readout side. In simplified notation, suppressing heads and normalizers, the second-stage value path is

$$\tilde{v}_i = (x_i W_v^{(2)}) \odot g_i^L, \qquad y_t = \sum_{i \le t} A_{t,i}^{\text{FW-MLP}} \tilde{v}_i, \qquad o_t = (y_t \odot g_t^R) W_o^{(2)}, \tag{137}$$

where g_i^L is the value-side or write-side gate and g_t^R is the output-side or readout-side gate. These gates are positive and softplus-normalized. Ignoring the normalization inserted for numerical stability, each pairwise contribution can be written as

$$A_{t,i}^{\text{FW-MLP}}\, x_i W_v^{(2)}\, \mathrm{Diag}\big(g_i^L \odot g_t^R\big) W_o^{(2)}. \tag{138}$$

Thus the two gates combine into an input-dependent diagonal core between $W_v^{(2)}$ and $W_o^{(2)}$. This is the readout-side low-rank/diagonal parameterization emphasized by the attention-as-MLP analysis of Lu & Yang (2026): it modulates what the selected value directions write into the residual stream and leaves the QK routing boundaries unchanged. HyperMLP's readout-rank results and its discussion of VO-side gating support placing this multiplicative capacity on the VO side, in particular in the second stage, where the final update directions are produced.

The right/output gate g_t^R also matches a broader empirical trend in gated attention. Qiu et al. (2025) study gated variants of scaled dot-product attention (SDPA) and report that a head-specific gate after SDPA consistently improves performance, stability, and scaling. They attribute the gains to added nonlinearity on the low-rank attention mapping and to query-dependent sparse modulation of the SDPA output. The same form is related to parameterizations of the Attention Free Transformer (AFT), in which a query-side gate multiplies a pooled key-value context vector (Zhai et al., 2021). In FW-MLP the gates are a compact readout-side modulation that complements the tri-region score and transport parameterization.

O.3 BLOCK-LEVEL PARAMETER BUDGETS

A standard attention layer with dense Q, K, V, O projections has the familiar schematic budget $4d^2$. A standard Transformer block often pairs this attention budget with an MLP/channel-mixing budget of roughly $8d^2$, giving a block-level budget around $12d^2$. Modern subquadratic architectures frequently alter the token-mixing block itself, sometimes replacing attention and even standard MLP blocks with a different recurrent or state-space mixer. Mamba, for example, integrates input-selective state-space dynamics into a simplified architecture without attention or standard MLP blocks (Gu & Dao, 2024), and Gated DeltaNet combines Mamba-2-style gating with the delta rule and also studies hybrid architectures (Yang et al., 2025).

The FW-MLP solver allocates the mixer budget so that the first-stage hidden width and the second-stage VO rank stay as large as the budget allows. Fitting the entire nested fast-MLP mixer into the vanilla $4d^2$ attention budget compresses the equivalent dynamic-MLP working ranks, in particular $r_{vo}^{(2)}$, which Proposition O.1 identifies as the working-rank bottleneck. Appendix R.4 reports the variant ordering at mixer budgets $6d^2$ and $4d^2$, which compress $r_{vo}^{(2)}$.

The main setting uses

$$B_{\text{mixer}} = 8d^2, \qquad B_{\text{mlp}} = 4d^2, \qquad B_{\text{block}} = 12d^2. \tag{139}$$

The robustness settings lower B_{mixer} to $6d^2$ and $4d^2$ with the same feed-forward block, $B_{\text{mlp}} = 4d^2$ (MLP ratio 2), so B_{block} is $10d^2$ and $8d^2$, respectively. Within each setting every variant receives the same mixer budget and the same feed-forward block. This protocol is *mixer-budget matching*: the matched quantity is B_{mixer}, with a shared feed-forward block.

P EXPRESSIVITY ENVELOPE AND PLUG-IN MIXER MODULARITY

The preceding sections establish separate facts: the reach of raw bilinear linear attention as a difference or interval score, the shared prefix term that softmax quotients out, gated decay as a special interval transport, and the learned tri-region product form of FW-MLP. This section combines them into

an *envelope statement*: FW-MLP contains several common linear-time transport forms as special cases, approximates more general prescribed transports on compact domains when the interval lift is rich enough, and can lift either a linear-time mixer or a quadratic activated mixer. The softmax numerator kernel has infinite separable rank on open domains (Proposition P.4). On compact domains, a normalized finite-feature linear-time FW mixer uniformly approximates vanilla softmax attention and attention with a general positive kernel (Theorem P.1, Corollary P.1). Choosing a quadratic second-stage mixer recovers quadratic attention exactly (Proposition P.5).

P.1 A COMMON NOTATION FOR MIXER BACKBONES

Let a causal mixer backbone be specified by a pairwise kernel or logit rule κ_η and a normalization operator $\mathcal{N}$. Its output has the schematic form

$$\mathrm{Mix}_\eta(q_t, K_{1:t}, V_{1:t}) = \sum_{i \le t} \mathcal{N}_i(\kappa_\eta(q_t, k_i))\, v_i. \tag{140}$$

Examples include unnormalized linear attention, normalized linear attention, softmax attention with $\kappa(q,k) = \exp(qk^\top)$ or logit $qk^\top$, and quadratic activated attention with $\kappa(q,k) = \rho(qk^\top + b)$ for an activation such as ReLU. The FW-MLP lift replaces token-local endpoint features by the learned endpoint fast-MLP features and optionally multiplies the pairwise score by a tri-region interval factor:

$$\kappa^{\text{FW-MLP}}_{t,i} = \kappa_\eta\big(F^q_t, F^k_i\big)\; F^\Delta_{t,i}, \qquad F^\Delta_{t,i} = \sum_{r=1}^{R} \omega_{t,r} \exp\left[-\sum_{u=i+1}^{t} \rho_r(x_u)\right]. \tag{141}$$

For softmax-logit implementations, the same lift can be used as an additive log-bias or score feature before normalization:

$$\ell^{\text{FW-MLP}}_{t,i} = \ell_\eta(F^q_t, F^k_i) + b^\Delta_{t,i}, \tag{142}$$

where $b^\Delta_{t,i}$ is an interval or tri-region scalar produced by the FW branch. Multiplicative and additive-log forms are equivalent when $F^\Delta_{t,i} > 0$ and $b^\Delta_{t,i} = \log F^\Delta_{t,i}$.

Recurrent-update backbones such as DeltaNet, Gated DeltaNet, and GSA carry a state $S_t = U_\eta(S_{t-1}, x_t, k_t, v_t)$ and read it with the query as $o_t = R_\eta(q_t, S_t)$. Their FW lift replaces q_t and k_t by F^q_t and F^k_t. The FW interval factor in Equation 141 is the scalar exponential branch of Section 3, the case that the linear-time proofs treat.

Proposition P.1 (Backbone containment under FW lifting)**.** *Fix a causal mixer backbone* Mix_η *of either form above, with query, key, and value features as inputs. Assume the FW endpoint branches can realize the backbone's original query/key feature maps, or can realize identity maps before the original query/key projections. For the pairwise form, also assume the interval branch can realize the constant factor* 1*. Then the corresponding FW-lifted architecture contains the original backbone as a subfamily.*

Proof. Set the endpoint fast MLPs to identity or to the original query/key projections and keep the original value branch. For the pairwise form, set the interval factor to the constant $F^\Delta_{t,i} \equiv 1$ by choosing $R = 1$, $\rho_1 \equiv 0$, and $\omega_{t,1} \equiv 1$. The lifted kernel in Equation 141 then reduces to $\kappa_\eta(q_t, k_i)$, so the output equals that of the original backbone. For the recurrent-update form, the identity endpoints recover q_t and k_t, hence the state sequence S_t and the output o_t. □

Applying FW-MLP to an existing linear, softmax, activated-attention, or recurrent-update backbone therefore preserves the backbone's representable class, because a parameter setting recovers the base mixer. The computational complexity is inherited from the chosen backbone. The lifted mixer runs in linear time when the selected mixer is associative/recurrent and the interval factor has the γ_t/γ_i form of Theorem G.1, and it runs in quadratic time when the selected second-stage mixer materializes all pairwise softmax or ReLU-activated scores.

P.2 CONTAINMENT OF COMMON LINEAR-TIME TRANSPORT FORMS

Define the scalar/diagonal FW transport dictionary

$$\tau_{t,i} = \sum_{r=1}^{R} \omega_{t,r} \prod_{u=i+1}^{t} a_{u,r}, \qquad a_{u,r} = \exp[-\rho_r(x_u)] \in (0,1], \tag{143}$$

with real-valued readout coefficients $\omega_{t,r}$. This is the interval branch in exponential-product notation with a general nonnegative increment class. The ReLU instantiation of Section 3 uses increments $\lambda\phi_+(xW_\Delta)$, which include constant rates when the token features carry a constant coordinate. The dictionary subsumes several transport factors used by efficient sequence models.

Proposition P.2 (Linear, retentive, and gated transports are FW special cases)**.** *When the track-increment class contains the required increments ρ_r, the dictionary in Equation 143 contains the following transport factors.*

1. *Ordinary additive linear attention transport (Katharopoulos et al., 2020; Schlag et al., 2021): choose $R = 1$, $\rho_1 \equiv 0$, and $\omega_{t,1} = 1$.*
2. *Fixed exponential decay or RetNet-style multi-scale retention: in head h choose $R = 1$, $\rho_1(x_u) = -\log \gamma_h$ with a constant $0 < \gamma_h \leq 1$, and $\omega_{t,1} = 1$; several tracks give mixtures of such decays.*
3. *Scalar or diagonal gated linear attention transport: for a gate $g_{u,r} \in (0, 1]$, choose $\rho_r(x_u) = -\log g_{u,r}$, giving $\prod_{u=i+1}^{t} g_{u,r}$.*

Proof. Each statement follows by substituting the specified ρ_r into the product form in Equation 143. The first case gives $\tau_{t,i} = 1$. The second gives γ_h^{t-i} in head h. The third gives the product of data-dependent gates along the interval. □

FW-MLP therefore contains, as exact special cases, the scalar and diagonal commuting transport factors used by ordinary linear attention, fixed retentive decays, and gated linear attention (Sun et al., 2023; Yang et al., 2024a). A diagonal gate's score $\sum_a q_{t,a} k_{i,a} \prod_{u=i+1}^{t} g_{u,a}$ is realized by pairing endpoint coordinate a with track a under the term-isolating allocation of Corollary I.1. Selective state-space models use input-dependent diagonal transports (Gu & Dao, 2024), and Mamba-2/SSD restricts them to a scalar times the identity per head (Dao & Gu, 2024), so their transport factors take the scalar and diagonal gated form above. DeltaNet and Gated DeltaNet use richer prescribed updates whose unrolled form contains input-dependent matrix products such as $A_t A_{t-1} \cdots A_{i+1}$ (Yang et al., 2024b; 2025). When the effective pairwise transport or value map induced by such a recurrence is a continuous function of compact endpoint and ordered-interval descriptors, a sufficiently rich FW tri-region parameterization approximates it uniformly (Proposition P.3). Selecting the delta-rule update as the second-stage backbone gives an FW-lifted model that contains the original delta-rule model (Proposition P.1).

Proposition P.3 (Approximation of delta-rule effective transports)**.** *Fix a maximum length $T_{\max}$ and compact token/endpoint domains. Suppose a DeltaNet-style, Gated-DeltaNet-style, or matrix-state recurrent mixer induces a scalar effective pairwise coefficient, or one coordinate of a vector-valued effective value map,*

$$G_\Delta(z_t^q, z_i^k, Z_{t,i}), \tag{144}$$

where $Z_{t,i}$ is a descriptor that determines the ordered interval $X_{i+1:t}$ on the feasible compact domain. For example, $Z_{t,i}$ can be the positional lift of Proposition H.1 when $T_{\max}$ is fixed (Corollary H.2). Under the compactness, rich-lift, readout-richness, and term-allocation assumptions of Theorem I.2 and Corollary I.1, sums of FW-MLP score modules approximate G_Δ uniformly to arbitrary accuracy, and vector-valued maps are handled coordinatewise.

Proof. For fixed $T_{\max}$ and compact token domains, a finite unroll of a delta-rule recurrence is a finite composition of continuous maps, so its effective pairwise coefficient is continuous in the endpoint variables and the ordered interval. If $Z_{t,i}$ determines that ordered interval on the feasible compact domain, the coefficient can be written as a continuous function of $(z_t^q, z_i^k, Z_{t,i})$. Theorem I.2 approximates this continuous function by finite sums of products of query-endpoint, key-endpoint, and interval functions. Corollary I.1 compiles such product sums into FW endpoint coordinates and interval tracks up to arbitrary uniform error. □

Remark P.1 (Approximation and backbone choice)**.** *Increasing the number of FW tracks enriches the scalar interval dictionary and, with sufficiently rich interval lifts, improves the approximation of the* effective transport function*. FW-MLP therefore approximates delta-rule induced transports on compact domains, and selecting the DeltaNet or Gated DeltaNet update as the second-stage backbone gives an FW-lifted model that contains the original algorithm (Proposition P.1).*

P.3 Comparison with quadratic attention

Proposition P.4 (The exponential dot-product kernel has infinite separable rank)**.** *On any nonempty open intervals $I, J \subset \mathbb{R}$, the kernel $K(x, y) = \exp(xy)$ has infinite separable rank. Consequently, on nonempty open query and key sets in $\mathbb{R}^d$, the softmax numerator kernel $\exp(q^\top k)$ has infinite separable rank, so every exact representation $\phi(q)^\top \psi(k)$ needs infinitely many features.*

Proof. If $e^{xy} = \sum_{r=1}^{m} a_r(x) b_r(y)$ on $I \times J$, then for any $m + 1$ distinct $y_j \in J$, the functions $x \mapsto e^{xy_j}$ are linearly dependent. Analytic continuation extends any dependence relation to all real x. Differentiating at $x = 0$ for orders $0, \dots, m$ yields a Vandermonde system in the distinct y_j, forcing all coefficients to be zero, a contradiction. For the multidimensional claim, choose interior points q_0 and k_0 and directions u, v with $u^\top v \neq 0$ such that the small line segments $q = q_0 + xu$ and $k = k_0 + yv$ remain in the open domains. Then

$$\exp(q^\top k) = \exp(q_0^\top k_0) \exp(x u^\top k_0) \exp(y q_0^\top v) \exp(xy\, u^\top v). \tag{145}$$

The first three factors are nonzero separable factors. Hence any fixed finite separable representation of $\exp(q^\top k)$ would imply one for $\exp(cxy)$ with $c = u^\top v \neq 0$ on an interval product, contradicting the one-dimensional result. □

Theorem P.1 (Linear-time FW approximation of softmax attention)**.** *Fix compact query, key, and value domains with $\|q\| \leq B$, $\|k\| \leq B$, and $\|v\| \leq V$. For every $\epsilon > 0$, there exist a finite feature dimension m and endpoint feature maps*

$$\Phi_m : Q \to \mathbb{R}^m, \qquad \Psi_m : K \to \mathbb{R}^m \tag{146}$$

such that the normalized linear-time mixer

$$\widehat{o}_t = \frac{\sum_{i \leq t} \widehat{K}_m(q_t, k_i) v_i}{\sum_{i \leq t} \widehat{K}_m(q_t, k_i)}, \qquad \widehat{K}_m(q, k) = \Phi_m(q)^\top \Psi_m(k) \tag{147}$$

approximates causal vanilla softmax attention

$$o_t = \frac{\sum_{i \leq t} \exp(q_t^\top k_i) v_i}{\sum_{i \leq t} \exp(q_t^\top k_i)} \tag{148}$$

uniformly over all nonempty causal rows whose tokens lie in the compact domain. For fixed m, the approximant is computed in time linear in the sequence length from the recurrent numerator state $S_t = \sum_{i \leq t} \Psi_m(k_i) v_i^\top$ and the denominator state $n_t = \sum_{i \leq t} \Psi_m(k_i)$, and its output is $\Phi_m(q_t)^\top S_t / (\Phi_m(q_t)^\top n_t)$. An FW-MLP instance whose second-stage mixer is normalized linear attention realizes this approximation when its endpoint towers implement, or approximate to sufficiently small additional error, Φ_m and Ψ_m, and when the interval branch is set to a constant.

Proof. The exponential dot-product kernel $K(q, k) = \exp(q^\top k)$ is continuous on the compact set $Q \times K$. By Stone–Weierstrass, or by truncating the Taylor expansion of $\exp(q^\top k)$ and separating each monomial into query and key factors, there are finite feature maps such that

$$\sup_{q,k} |\Phi_m(q)^\top \Psi_m(k) - K(q, k)| \leq \delta. \tag{149}$$

Let $a = e^{-B^2}$, so $K(q, k) \geq a$. Choose $\delta < a/2$. For a row with t keys, the exact denominator D_t satisfies $D_t \geq ta$, and the approximate denominator $\widehat{D}_t$ satisfies $\widehat{D}_t \geq t(a - \delta) \geq ta/2$. The numerator error is at most $t\delta V$ and the denominator error is at most $t\delta$. Using $\| \sum_{i \leq t} K(q_t, k_i) v_i \| \leq D_t V$ gives

$$\|\widehat{o}_t - o_t\| \leq \frac{2V\delta}{a} + \frac{2V\delta}{a} = \frac{4V\delta}{a}. \tag{150}$$

Choosing $\delta \leq a\epsilon/(4V)$ gives the result, with the trivial case $V = 0$ immediate. The recurrent implementation follows from associativity of the finite feature state. The FW realization follows from endpoint-tower universal approximation, with its approximation error absorbed by choosing a smaller kernel error budget, and from the identity/constant-interval construction. □

Corollary P.1 (Continuous positive quadratic kernels)**.** *Let $K(q, k)$ be a continuous kernel on compact query/key domains with a uniform positive lower bound $K(q, k) \geq a > 0$. Then the normalized attention map with kernel K admits the same finite-feature linear-time approximation as in Theorem P.1. More generally, for every nonnegative K and a fixed maximum row length, the same proof applies whenever the exact row denominators are uniformly bounded away from zero and the finite-feature approximation error is chosen small relative to that bound.*

Proof. Replace the exponential kernel in Theorem P.1 by the given kernel. Finite separable approximation follows from Stone–Weierstrass. Under the uniform lower bound, each row denominator is at least ta, so the denominator-stability argument is identical. In the nonnegative fixed-horizon variant, the accumulated denominator approximation error is at most the row length times the pairwise kernel error. Choosing the pairwise kernel error so that this accumulated error lies below the stated denominator margin then gives a ratio perturbation bound of the same form, with a replaced by that margin. □

The construction matches the finite-feature principle of Performer/FAVOR+, which approximates softmax attention kernels in linear time with finite random features (Choromanski et al., 2021). FW-MLP learns the endpoint features and can additionally include endpoint-prefix and interval features.

Quadratic attention and linear-time attention trade expressivity against computation. A quadratic softmax or ReLU-activated mixer can materialize a pairwise score for every (t, i). Softmax uses the exponential dot-product kernel of the Transformer (Vaswani et al., 2017), and activated attention can use kernels such as $\rho(q_t k_i^\top)$. Finite-feature linear attention keeps a fixed-dimensional separable state. By Proposition P.4, the exponential dot-product kernel has infinite separable rank on open domains, and by Theorem P.1, finite-feature linear-time FW-MLP approximates vanilla softmax attention uniformly on compact domains. Linear-time FW-MLP is therefore a compact-domain approximation envelope for softmax and positive-kernel attention, and a quadratic FW lift exactly contains the chosen quadratic backbone (Proposition P.5).

Proposition P.5 (Quadratic FW lift contains its quadratic backbone)**.** *Let $\mathcal{F}_{\mathrm{quad}}$ be the function class of a chosen quadratic causal mixer, such as softmax attention or ReLU-activated attention, with a fixed value branch. Let $\mathcal{F}_{\mathrm{quad}}^{\text{FW-MLP}}$ be the class obtained by replacing its endpoint inputs by FW endpoint fast-MLP features and optionally adding the FW interval score feature. Assume, as in Proposition P.1, that the FW endpoint branches can realize the original endpoint features and that the interval branch can realize the constant factor. Then*

$$\mathcal{F}_{\mathrm{quad}} \subseteq \mathcal{F}_{\mathrm{quad}}^{\text{FW-MLP}}. \tag{151}$$

If the interval branch is allowed to be nonconstant and the data domain admits two histories with identical token-local endpoints and different interval descriptors, then the lifted score class is strictly larger than the endpoint-only quadratic score class.

Proof. Containment is the same identity-branch construction as in Proposition P.1. For strictness relative to endpoint-only scores, hold x_t and x_i fixed while changing a middle token so that $\sum_{u=i+1}^{t} \rho(x_u)$ changes. The endpoint-only quadratic score takes equal values on the two histories, and the FW interval feature takes different values. Choosing a nonzero readout makes the lifted score change. □

The results give three comparison regimes. With a linear-time backbone, FW-MLP broadens linear attention by adding asymmetric endpoint fast MLPs and learned interval transports, and it keeps linear-time execution for finite tracks. With a normalized finite-feature linear-time backbone, FW-MLP approximates vanilla softmax attention and attention with a general positive kernel on compact domains, and the feature dimension m controls the approximation error. With a quadratic backbone, FW-MLP is a plug-in lift of softmax/ReLU attention: it contains the original quadratic attention class exactly and adds learned tri-region score features, at the quadratic cost of the selected mixer.

P.4 Two-stage mixer switching

The FW-MLP computation has two conceptually distinct mixers. The first-stage mixer instantiates endpoint fast MLPs from prefixes such as $X_{1:t}$ or $X_{1:i}$; the second-stage mixer uses the resulting

endpoint and interval features to mix values. Both stages can be parameterized by a mixer choice

$$\mathcal{M}^{(1)}, \mathcal{M}^{(2)} \in \{\text{linear associative mixing},\ \text{softmax quadratic mixing},\ \text{ReLU/activated quadratic mixing},\ \text{other causal mixer}\}. \tag{152}$$

The linear-time equations of Section 3 correspond to choosing linear associative mixing for the first-stage fast-weight generator and for the second-stage value mixer. Softmax or ReLU-activated mixing in either stage changes the computation budget and keeps the same modular structure: FW-MLP supplies learned endpoint and interval features, and the selected mixer decides how those features are aggregated.

Corollary P.2 (Plug-in lift for attention variants)**.** *Any attention or linear-attention variant that can be written as a causal mixer over query/key/value features can be FW-lifted by replacing its query/key features with FW endpoint fast-MLP features and by adding a constant-or-learned interval branch. If the FW endpoint branches can realize the original endpoint features and the interval branch can realize the constant factor, the lifted variant contains the original variant as the constant-interval, identity-endpoint subfamily.*

FW-MLP is a feature and score parameterization that pairs with a linear-time recurrence, quadratic softmax, quadratic ReLU-activated attention, or a prescribed update backbone. By containment, the lifted class includes the backbone class, and the chosen mixer determines the runtime and memory profile.

Q Related work

Attention, fast weights, and dynamic-MLP views. The Transformer made content-dependent sequence mixing a central primitive for modern sequence modeling (Vaswani et al., 2017). The forward computation of attention can also be read as an input-conditioned weight generator. This view is close to HyperNetworks, where one network generates the weights of another (Ha et al., 2017), and to fast-weight programming, where rapidly changing weights are produced and consumed inside the sequence computation (Ba et al., 2016; Schlag et al., 2021). Linear attention makes this fast-weight structure explicit by rewriting causal attention with kernel feature maps and associative prefix states (Katharopoulos et al., 2020), and Performers approximate softmax attention with positive random features (Choromanski et al., 2021). HyperMLP further argues that an autoregressive attention head is equivalent to a dynamic two-layer MLP whose working rank depends strongly on the readout/VO side (Lu & Yang, 2026). FW-MLP adopts this structural perspective: the object of study is the context-conditioned forward graph and its function class.

Efficient recurrent transport and linear-time sequence models. Many recent architectures replace quadratic attention with fixed-size recurrent states or linear-time transport laws. RetNet derives a retention mechanism with parallel, recurrent, and chunkwise recurrent forms (Sun et al., 2023); Mamba introduces selective state-space dynamics whose parameters depend on the input token (Gu & Dao, 2024); and state-space duality connects structured state-space models and attention-like sequence mixing, leading to Mamba-2/SSD (Dao & Gu, 2024). Gated linear attention adds data-dependent gates to linear attention while retaining efficient training and recurrent inference (Yang et al., 2024a). The delta-rule update of Schlag et al. (2021) replaces the additive fast-weight write in linear attention, and DeltaNet (Yang et al., 2024b) develops a parallel training algorithm for that recurrence. Gated DeltaNet combines gating and the delta rule to improve long-context linear-time modeling (Yang et al., 2025). These models supply prescribed transport and update biases. FW-MLP contains ordinary linear attention, fixed exponential retention, and scalar or diagonal gated transport as exact subcases. Matrix-valued delta-rule transports enter either as mixer backbones or as continuous effective maps that the tri-region FW envelope approximates on compact domains.

In-context-learning mechanisms and optimization interpretations. A parallel line of theory studies what algorithms Transformers can implement in context. Transformers can be trained to in-context learn simple function classes such as linear functions, sparse linear functions, small neural networks, and decision trees (Garg et al., 2022). Other work investigates whether in-context learning implements algorithms such as gradient descent, ridge regression, or least-squares regression (Akyürek et al., 2023), and gives constructions where self-attention layers realize gradient-descent-like updates on regression tasks (von Oswald et al., 2023). Statistical and Bayesian perspectives

further show that Transformers can implement broad classes of in-context estimators and perform algorithm selection (Bai et al., 2023), and that they can approximate Bayesian predictors in meta-learning setups (Panwar et al., 2024). These results establish the richness of the Transformer forward graph. This appendix treats such mechanisms as members of the realizable function class that the architecture exposes.

Test-time training and mechanism-defined layers. Recent sequence models build optimization into the layer itself. TTT layers use a hidden state that is itself a machine learning model and update it by self-supervised learning on the test sequence (Sun et al., 2025). MesaNet similarly frames recurrent dynamics through locally optimal test-time training solved with a fast conjugate-gradient procedure (von Oswald et al., 2026). FW-MLP enlarges the neural parameterization available to a linear-time attention-like mixer, and its parameters are trained end to end through the score and value computation of Section 3. Enlarging the learned parameterization and training it end to end draws on the observation that general learned structure benefits from added data and compute (Sutton, 2019; Kaplan et al., 2020; Hoffmann et al., 2022).

Readout rank, VO gating, and parameter-budget allocation. The attention-as-MLP perspective also informs parameter-budget allocation. HyperMLP shows that the readout and VO side of the equivalent MLP has a stronger working-rank effect than the QK and routing side: compressing the readout core directly restricts the residual update subspace, and QK compression restricts the conditioning geometry (Lu & Yang, 2026). HyperMLP's rank asymmetry supports protecting the second-stage VO working rank in FW-MLP. The same perspective explains why VO-side gates are natural: a diagonal gate placed between W_v and W_o modulates the readout/action map while preserving the addressing geometry. Recent gated-attention experiments likewise find that adding a gate after SDPA improves performance and stability by adding nonlinearity and sparse output modulation (Qiu et al., 2025), and AFT-style models use a query-side gate multiplying a pooled key-value context (Zhai et al., 2021). Within each budget setting, the comparisons of this paper match the mixer budget and share the feed-forward block, and the solver of Appendix O.1 gives the second-stage VO rank the largest allocation weight. Appendix R.4 reports the variant ordering at mixer budgets $6d^2$ and $4d^2$.

Gated Slot Attention and two-stage mixing. Gated Slot Attention (GSA) (Zhang et al., 2024) also uses a two-pass mixing structure. The GSA paper builds on Attention with Bounded-Memory Control (ABC) (Peng et al., 2022) and adds a gating mechanism inspired by Gated Linear Attention (Yang et al., 2024a). It summarizes GSA as a two-layer GLA linked by a softmax operation, with context-aware memory reading and adaptive forgetting over a compact recurrent state. GSA's stages are recurrent GLA-like memory passes over a bounded slot state. FW-MLP starts from the attention-as-dynamic-MLP equivalence and explicitly parameterizes a fast-MLP score over three causal regions: a query-prefix endpoint MLP over $X_{1:t}$, a key-prefix endpoint MLP over $X_{1:i}$, and a cancellative interval fast MLP over $X_{i+1:t}$. As a recurrent-update backbone, GSA can serve as the second-stage mixer of an FW lift (Proposition P.1).

Summary of the relation to this literature. The mathematical results above connect these literatures to the prefix and interval results for FW-MLP. In the prefix-sum setting, softmax quotients out a shared prefix term and exposes a sequence-conditioned linear suffix score. A raw bilinear score equals a fixed difference function $g(s-p)$ only when g is the zero function (Proposition D.2). Two-tower endpoint MLPs enlarge the score class through finite separable-rank features, and the cancellative prefix embeddings are exactly the affine-additive ones. FW-MLP therefore introduces an explicit additive interval potential for $X_{i+1:t}$ and combines it with asymmetric endpoint fast MLPs for $X_{1:t}$ and $X_{1:i}$. Among efficient-attention and online-update models, FW-MLP contributes a neural-network-native tri-region score parameterization whose learned interval transport is computed exactly in linear time.

R EXPERIMENTS: PROTOCOL AND FULL RESULTS

This appendix gives the protocol and the complete result tables of each setting in Section 5, and the code is at `https://anonymous.4open.science/r/FasterWeight-MLP-FC32/`.

R.1 SHARED PROTOCOL AND MATCHED ATTENTION BUDGET

Every comparison in the paper varies one component of the model: the sequence-mixing module inside an otherwise fixed scaffold. A common variant interface performs the substitution, so the surrounding architecture, the data pipeline, the optimizer settings, the task splits, and the evaluation code are shared by all variants of a given setting.

The attention budget B sets a target attention-side capacity of approximately Bd^2 trainable parameters per layer at model width d. Each variant meets that target through its own attention-side capacity parameter, such as attention width, state width, or expansion width, with the classifier, the data pipeline, and the feed-forward configuration held at the shared setting. The FW-MLP ranks that satisfy the target come from the exact-count solver of Appendix O.1, which accounts for projections, gates, shortcuts, biases, convolutional terms, and interval-generator overheads. The realized parameter count is verified after construction. Budget 8 is the main setting, and Appendix R.4 reports budgets 6 and 4.

The VariantBench tasks outside the Time-Series Library (TSLib) share a compact backbone: hidden width $d = 128$, four pre-norm residual blocks, learned absolute positional embeddings, MLP ratio 2.0, dropout 0, and bias-free linear layers. Head-parameterized attention baselines use four heads. FW-MLP and FW-MLP-NoΔ use two heads, whose first-stage outputs are routed into the two second-stage fast-MLP prefix branches of Section 3. All other backbone and training settings are shared with the other variants, and the budget solver matches every variant's attention-side parameter count to the same target.

Supervision follows the task type. Language-model and completion tasks use causal cross-entropy on supervised positions, with prompt and padding positions ignored. Text datasets are represented as UTF-8 byte sequences with a 257-token byte vocabulary, which keeps the comparison independent of tokenizer choice. Classification and feature-sequence tasks read the final valid sequence position through a linear classifier. SASRec-style recommendation predicts the next item from the final valid prefix state and reports Hit@10 by ranking the full item vocabulary.

Every training run in the paper uses $8\times$NVIDIA A100 (80 GB) GPUs. These runs are VariantBench at budgets 8, 6, and 4, the interval-gated retrieval diagnostic, the nanoGPT study, and the 340M/15B and 1.3B/100B language-model runs. The compiled throughput measurement of Appendix R.8 uses one A100 of the same type, and the long-context sweep uses one NVIDIA L40S.

R.2 BASELINE IMPLEMENTATIONS

The comparison includes 12 sequence mixers. Softmax is the canonical Transformer attention reference (Vaswani et al., 2017) with learned absolute positional embeddings, and Softmax RoPE adds rotary embeddings (Su et al., 2024) under the same budget rule. Gated Softmax adds a learned output gate to softmax attention. Differential Transformer (Ye et al., 2025) and Differential Transformer-v2 (Ye et al., 2026) implement two differential-attention variants. Linear Attention is causal feature-map attention (Katharopoulos et al., 2020). Gated DeltaNet, GLA, and Kimi Delta Attention keep the gating and recurrent state updates of the corresponding linear-attention methods (Yang et al., 2025; 2024a; Kimi Team et al., 2025), and Mamba-3 enters through the same block-level interface as a selective state-space mixer (Lahoti et al., 2026). FW-MLP and FW-MLP-NoΔ are the two configurations of Section 3, the second with $F^{\Delta} \equiv 1$.

Where a baseline has a widely used public implementation, that implementation supplies the attention layer and a thin wrapper adapts it to the shared interface. Gated DeltaNet, GLA, and Kimi Delta Attention use Flash Linear Attention (Yang & Zhang, 2024), and Mamba-3 uses the official Mamba implementation (Gu & Dao, 2024; Lahoti et al., 2026). The softmax-family baselines use FlashAttention (Dao et al., 2022) or PyTorch scaled-dot-product attention (Paszke et al., 2019). FW-MLP is implemented in the accompanying code, with linear-attention and causal-convolution kernels used as computational accelerators where available. The time-series experiments are built on the

Table 4: TSLib raw task coverage behind the family-level aggregation.

Family task	Raw task groups	Raw metric	Seeds	Attention budget
TSLib forecasting	6	MSE	3	8
TSLib classification	30	accuracy	3	8
TSLib anomaly	15	F-score	3	8
TSLib forecasting	6	MSE	1	6
TSLib classification	30	accuracy	1	6
TSLib anomaly	15	F-score	1	6

Time-Series Library (Wu et al., 2023; Wang et al., 2026), which supplies the data loaders, task heads, training loops, and metrics, with the self-attention module substituted. Every variant is sized by the same attention-budget rule, evaluated under the same task protocol, and kept at one architectural configuration across the tasks of a setting.

R.3 VariantBench: tasks, aggregation, and full results

VariantBench is the controlled cross-domain benchmark used for the main attention-variant comparison. It collects widely used tasks from the sequence-modeling and linear-attention literature into one protocol with a matched attention budget, covering synthetic reasoning, associative retrieval, compositional generalization, time series, recommendation, language, audio, and vision sequences.

At the raw level the suite holds 74 task groups: 23 outside the Time-Series Library and 51 within it, the latter comprising 6 forecasting, 30 classification, and 15 anomaly-detection groups (Table 4). Twelve variants and three seeds (1337–1339) give 828 non-TSLib and 1836 TSLib variant-task-seed evaluations, for 2664 evaluations in the main budget setting. The paper-level ranking aggregates the 51 TSLib groups into their three families, which gives 26 paper-level tasks and keeps the time-series portion proportionate to the other domains in the cross-domain average.

Non-TSLib task configurations. The upper block of Table 5 lists the 23 non-TSLib tasks with their input specification, dataset sizes, and optimization settings. All non-TSLib runs use AdamW with fixed task-specific epoch counts and no early stopping, and the reported metric is computed once, after training, on the held-out validation or test split. Weight decay is 0.01 at the default learning rate of 3×10^{-4}, and tasks trained at 10^{-3} or 5×10^{-4} use no weight decay. The primary metric of each task appears in Table 8.

Synthetic and procedural tasks. The synthetic suite is generated procedurally, which fixes sequence length, vocabulary size, training and evaluation lengths, and the number of examples exactly. The MQAR tasks follow the Zoology-style multi-query associative-recall protocol (Arora et al., 2024) and report token accuracy. The packed copy, prefix-3-gram, and suffix-3-gram tasks use a fixed length with a shorter payload placed inside it. The selective-copy task follows the selective-copying setup of the MAD-Lab benchmark suite (Poli et al., 2024), and the induction-head task tests repeated-pattern completion. The modular-sum task classifies the sum of the input tokens modulo 10 and trains on shorter sequences than it is evaluated on, which measures length generalization. The ListOps task follows the ListOps diagnostic setup (Nangia & Bowman, 2018). Four vision-as-sequence tasks read images as token sequences: sMNIST and pMNIST flatten MNIST digits (LeCun et al., 1998), with pMNIST applying a fixed random pixel permutation; sequential CIFAR-10 flattens CIFAR-10 images (Krizhevsky, 2009); and dSprites patchifies the images of the DeepMind dSprites release (Matthey et al., 2017) and predicts the shape latent of each image.

Text, recommendation, audio, and medical-image tasks. SCAN (Lake & Baroni, 2018) is evaluated on the `simple`, `length`, `addprim_jump`, and `addprim_turn_left` splits, which the tables name SCAN simple, SCAN length, SCAN jump, and SCAN turn-left. Each split is converted into a byte-level `IN:`/`OUT:` completion problem scored by exact match over the supervised output bytes. For bAbI (Weston et al., 2015), the public English release supplies tasks 1, 2, 3, 16, and 19, concatenated into a byte-level question-answering benchmark scored by answer exact match. The SASRec-style recommendation task (Kang & McAuley, 2018) uses MovieLens-1M (Harper &

Table 5: Configuration of the VariantBench tasks. The upper block lists the 23 non-TSLib tasks: example counts are train / validation or train / test according to the split the dataset provides, tasks marked † resample their training set every epoch, and text tasks use a 257-token byte vocabulary. The lower block lists the three TSLib families. Layers gives the encoder and decoder depths, and FF the feed-forward width. The anomaly row gives FF first for Transformer and iTransformer and then for PatchTST. A dash marks a family without further settings. Ep. is the number of training epochs and LR the learning rate.

Task	Input specification	Examples	Ep.	Batch	LR
MQAR	length 64, 4 key–value pairs, vocabulary 8192	100,000 / 1000	32	256	10^{-3}
MQAR-hard†	length 128, 16 pairs, 8 queries	50,000 / 1000	32	128	10^{-3}
MQAR-long†	length 256, 32 pairs, 16 queries	32,768 / 1000	24	64	10^{-3}
copy†	length 220, vocabulary 30, payload 5–20	10,000 / 1000	20	32	3×10^{-4}
prefix 3-gram†	length 220, vocabulary 30, payload 5–20	10,000 / 1000	20	32	3×10^{-4}
suffix 3-gram†	length 220, vocabulary 30, payload 5–20	10,000 / 1000	20	32	3×10^{-4}
selective copy	length 256, vocabulary 16, 16 copied tokens	12,800 / 1280	50	128	5×10^{-4}
induction heads†	length 256, repeated-pattern completion	10,000 / 1000	20	32	3×10^{-4}
modular sum†	train lengths 32–64, evaluation lengths 96–128	50,000 / 2000	20	128	5×10^{-4}
sMNIST	length 784, grayscale pixel sequence	4096 / 512	5	16	3×10^{-4}
pMNIST	length 784, fixed pixel permutation	4096 / 512	5	16	3×10^{-4}
sequential CIFAR-10	length 1024, RGB pixel sequence	4096 / 512	5	8	3×10^{-4}
dSprites	64×64 binary image, patch size 4	4096 / 512	5	16	3×10^{-4}
ListOps	maximum length 128, depth 4, branching 4	4096 / 512	5	32	3×10^{-4}
SCAN simple	byte-level completion, maximum length 256	16,728 / 4182	5	32	3×10^{-4}
SCAN length	byte-level completion, maximum length 256	16,990 / 3920	5	32	3×10^{-4}
SCAN jump	byte-level completion, maximum length 256	14,670 / 7706	5	32	3×10^{-4}
SCAN turn-left	byte-level completion, maximum length 256	21,890 / 1208	5	32	3×10^{-4}
bAbI	tasks 1, 2, 3, 16, 19, maximum length 512	3716 / 3701	5	16	3×10^{-4}
SASRec ML-1M	MovieLens-1M, prefix length 200, 3706 items	988,129 / 6040	5	128	3×10^{-4}
TinyStories	byte windows of length 256, stride 128	10,000 / 1000	5	32	3×10^{-4}
PathMNIST	image size 64, patch size 8, RGB patches	89,996 / 10,004	5	32	3×10^{-4}
Speech Commands	1 s at 16 kHz, 64-bin log-mel features	36,923 / 4443	5	32	3×10^{-4}

TSLib family	Backbones	Ep.	Patience	Batch	LR	Layers	FF	Dropout	Other settings
Forecasting	iTransformer	10	3	32	10^{-4}	2 / 1	128	0.1	input length 720, label length 48, prediction length 96
Classification	Transformer, iTransformer, PatchTST	100	10	16	10^{-3}	3 / 1	256	0.1	–
Anomaly	Transformer, iTransformer, PatchTST	10	3	128	10^{-4}	3 / 1	128 / 256	0.1	window length 100, anomaly ratio 1.0

Konstan, 2015) with user histories sorted by timestamp: all non-final positions form training prefixes, each user's final item is the evaluation target, and the metric is Hit@10. TinyStories (Eldan & Li, 2023) cuts the V2 GPT-4 text files into byte-level autoregressive windows. PathMNIST comes from the MedMNIST release (Yang et al., 2023) and is scored by 9-way accuracy. Speech Commands v2 (Warden, 2018) uses the standard 12-class setting: ten keywords, an unknown class, and a silence class, the last two each sampled at 10% of the target-keyword count. Each waveform is padded or truncated to one second at 16 kHz and converted to 64-bin log-mel features with a 30 ms window and a 10 ms hop.

TSLib time-series tasks. The time-series experiments use the Time-Series Library (Wu et al., 2023; Wang et al., 2026) with its native task-specific data loaders, training loops, early stopping, and metrics. In each run, the self-attention layers of the selected backbone are replaced by one attention variant, implemented as in the rest of the suite. The datasets are the public preprocessed collections linked by the library: the ETT, Weather, and Electricity forecasting datasets, the UEA classification datasets (Bagnall et al., 2018), and the PSM, MSL, SMAP, SMD, and SWAT anomaly-detection datasets. The forecasting family uses the iTransformer backbone (Liu et al., 2024) on ETTh1, ETTh2, ETTm1, ETTm2, Weather, and Electricity in a multivariate-to-multivariate setting and is scored by MSE, where lower is better. The classification family uses Transformer, iTransformer, and PatchTST (Nie et al., 2023) on ten UEA datasets (EthanolConcentration, FaceDetection, Handwriting, Heartbeat,

Table 6: Main tie-aware variant ranking at attention budget 8 over the 26 paper-level tasks.

Rank	Variant	Avg tie-aware rank	Tie-aware wins	Tasks
1	FW-MLP	4.538	14	26
2	FW-MLP-NoΔ	5.019	11	26
3	Softmax RoPE	5.481	11	26
4	Differential Transformer-v2	5.731	12	26
5	Differential Transformer	5.885	9	26
6	Mamba-3	5.923	8	26
7	GLA	6.808	8	26
8	Gated Softmax	6.904	9	26
9	Softmax	7.077	9	26
10	Gated DeltaNet	7.654	8	26
11	Linear Attention	8.115	9	26
12	Kimi Delta Attention	8.865	8	26

Table 7: Tie-aware variant ranking at attention budget 8 over the 23 non-TSLib tasks.

Rank	Variant	Avg tie-aware rank	Tie-aware wins	Tasks
1	FW-MLP	4.913	12	23
2	Softmax RoPE	5.304	11	23
3	FW-MLP-NoΔ	5.370	10	23
4	Differential Transformer-v2	5.391	12	23
5	Differential Transformer	5.717	9	23
6	Mamba-3	5.913	8	23
7	Softmax	6.609	9	23
8	GLA	6.978	8	23
9	Gated Softmax	7.152	9	23
10	Gated DeltaNet	7.609	8	23
11	Linear Attention	8.457	9	23
12	Kimi Delta Attention	8.587	8	23

JapaneseVowels, PEMS-SF, SelfRegulationSCP1, SelfRegulationSCP2, SpokenArabicDigits, and UWaveGestureLibrary) and is scored by accuracy. The anomaly-detection family runs the same three backbones on PSM, MSL, SMAP, SMD, and SWAT, with the native TSLib F-score as its metric. PatchTST runs use patch length 16 with stride 8, and the lower block of Table 5 gives the rest of the training configuration of each family.

Aggregation protocol. For non-TSLib tasks, the budget-8 tables report the mean over seeds 1337–1339. Every paper-level task ranks the variants by its primary metric, with higher values better for accuracy, token accuracy, exact match, Hit@10, and F-score, and lower values better for forecasting MSE. Scores are rounded to six decimals for tie detection, and tied variants receive the average rank of the tied positions. TSLib results are aggregated in two stages. Each raw family/backbone/dataset run is ranked across variants by the same tie-aware rule, and the raw ranks are then averaged within forecasting, classification, and anomaly detection, so each family contributes one paper-level task to the 26-task ranking.

Full results at budget 8. Table 6 gives the 26-task aggregate and Table 7 the 23-task non-TSLib aggregate. Figure 4 shows every task-level rank. The per-task scores of FW-MLP, FW-MLP-NoΔ, and Softmax RoPE, set against the best variant on each task, are in Table 8, and the family-level time-series results are in Table 9. FW-MLP holds the best mean tie-aware rank, 4.538, with 14 task wins, and FW-MLP-NoΔ is second. Over the 23 non-TSLib tasks FW-MLP stays first, with Softmax RoPE second and FW-MLP-NoΔ third. Task wins spread across domains: FW-MLP ranks first outright on modular sum, pMNIST, the recommendation and audio tasks, and the time-series forecasting and classification aggregates; FW-MLP-NoΔ ranks first outright on MQAR-long, selective copy, and the time-series anomaly aggregate.

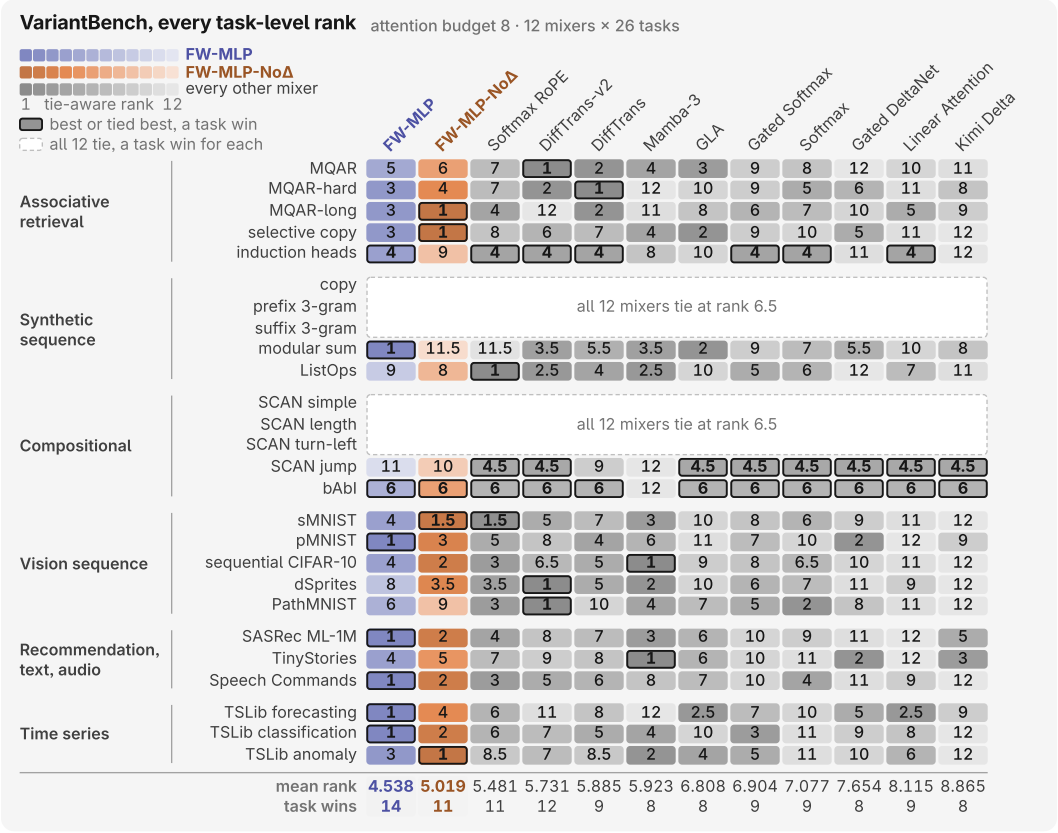

Figure 4: Task-level tie-aware ranks of all 12 mixers on the 26 paper-level tasks of VariantBench at attention budget 8. Each cell gives the rank of one mixer on one task, with lower values better and tied entries at their average rank; the shading runs from dark (rank 1) to light (rank 12), in blue for FW-MLP, orange for FW-MLP-NoΔ, and gray for the other ten mixers. An outlined cell is the best or tied-best rank on its task, and the two dashed bars cover the six tasks on which all 12 mixers tie at rank 6.5. Each outline and each of those six tasks is one task win, so the wins of a column are its outlines plus six. Tasks are grouped by domain along the vertical axis, and mixers are ordered by mean tie-aware rank along the horizontal axis; the two rows under the matrix repeat the mean rank and the task wins of Table 6. The column labels write DiffTrans and DiffTrans-v2 for the two Differential Transformer variants and Kimi Delta for Kimi Delta Attention.

Table 8: Compact per-task results at attention budget 8. Each cell gives the score followed by the tie-aware rank in parentheses. Where several variants hold the best score, the Best column names them when two tie and otherwise gives how many of the 12 variants reach it. On the non-TSLib tasks the Difference column gives the FW-MLP primary score minus the Softmax RoPE primary score; on the TSLib families it gives the Softmax RoPE average raw rank minus the FW-MLP average raw rank, so positive values favor FW-MLP in both cases.

Task	Metric	Best	FW-MLP	FW-MLP-NoΔ	Softmax RoPE	Difference
MQAR	token accuracy	Differential Transformer-v2: 0.9998	0.9970 (r=5)	0.9969 (r=6)	0.9962 (r=7)	+0.0008
MQAR-hard	token accuracy	Differential Transformer: 0.9996	0.9960 (r=3)	0.9925 (r=4)	0.3663 (r=7)	+0.6297
MQAR-long	token accuracy	FW-MLP-NoΔ: 0.9348	0.3243 (r=3)	0.9348 (r=1)	0.1109 (r=4)	+0.2135
copy	token accuracy	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
prefix 3-gram	token accuracy	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
suffix 3-gram	token accuracy	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
selective copy	token accuracy	FW-MLP-NoΔ: 0.9983	0.9931 (r=3)	0.9983 (r=1)	0.9728 (r=8)	+0.0203
induction heads	token accuracy	7 of 12 variants: 1.0000	1.0000 (r=4)	0.9922 (r=9)	1.0000 (r=4)	0.0000
modular sum	accuracy	FW-MLP: 0.1097	0.1097 (r=1)	0.1032 (r=11.5)	0.1032 (r=11.5)	+0.0065
sMNIST	accuracy	FW-MLP-NoΔ, Softmax RoPE: 0.9043	0.8737 (r=4)	0.9043 (r=1.5)	0.9043 (r=1.5)	-0.0306
pMNIST	accuracy	FW-MLP: 0.8743	0.8743 (r=1)	0.8568 (r=3)	0.8385 (r=5)	+0.0358
sequential CIFAR-10	accuracy	Mamba-3: 0.3646	0.3561 (r=4)	0.3620 (r=2)	0.3594 (r=3)	-0.0033
dSprites	accuracy	Differential Transformer-v2: 0.4512	0.4349 (r=8)	0.4479 (r=3.5)	0.4479 (r=3.5)	-0.0130
ListOps	accuracy	Softmax RoPE: 0.5534	0.5117 (r=9)	0.5234 (r=8)	0.5534 (r=1)	-0.0417
SCAN simple	exact match	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
SCAN length	exact match	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
SCAN jump	exact match	8 of 12 variants: 1.0000	0.9397 (r=11)	0.9679 (r=10)	1.0000 (r=4.5)	-0.0603
SCAN turn-left	exact match	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
bAbI	exact match	11 of 12 variants: 1.0000	1.0000 (r=6)	1.0000 (r=6)	1.0000 (r=6)	0.0000
SASRec ML-1M	Hit@10	FW-MLP: 0.2570	0.2570 (r=1)	0.2549 (r=2)	0.2468 (r=4)	+0.0102
TinyStories	token accuracy	Mamba-3: 0.6192	0.5643 (r=4)	0.5323 (r=5)	0.4832 (r=7)	+0.0811
PathMNIST	accuracy	Differential Transformer-v2: 0.8539	0.8190 (r=6)	0.8104 (r=9)	0.8417 (r=3)	-0.0227
Speech Commands	accuracy	FW-MLP: 0.9305	0.9305 (r=1)	0.9260 (r=2)	0.9224 (r=3)	+0.0081
TSLib forecasting	avg raw rank $\downarrow$	FW-MLP: 2.3333	2.3333 (r=1)	5.1667 (r=4)	5.6667 (r=6)	+3.3333
TSLib classification	avg raw rank $\downarrow$	FW-MLP: 5.0833	5.0833 (r=1)	5.5833 (r=2)	6.4833 (r=6)	+1.4000
TSLib anomaly	avg raw rank $\downarrow$	FW-MLP-NoΔ: 3.5333	5.4000 (r=3)	3.5333 (r=1)	7.1667 (r=8.5)	+1.7667

Table 9: Detailed TSLib family results at attention budget 8. Family ranks come from tie-aware ranks over the raw dataset and backbone groups and enter the 26-task aggregation, and the mean raw metric is shown for reference.

Variant	Family rank	Avg raw task rank	Raw wins	Raw tasks	Mean raw metric
TSLib forecasting (mean MSE)					
FW-MLP	1	2.333	2	6	0.247243
GLA	2.5	5.000	0	6	0.249539
Linear Attention	2.5	5.000	1	6	0.250675
FW-MLP-NoΔ	4	5.167	1	6	0.247624
Gated DeltaNet	5	5.500	0	6	0.249539
Softmax RoPE	6	5.667	0	6	0.252456
Gated Softmax	7	6.333	0	6	0.252020
Differential Transformer	8	6.833	1	6	0.251059
Kimi Delta Attention	9	7.833	1	6	0.252889
Softmax	10	8.500	0	6	0.255884
Differential Transformer-v2	11	9.000	0	6	0.254513
Mamba-3	12	10.833	0	6	0.260417
TSLib classification (mean accuracy)					
FW-MLP	1	5.083	4	30	0.715227
FW-MLP-NoΔ	2	5.583	3	30	0.713455
Gated Softmax	3	6.250	3	30	0.709892
Mamba-3	4	6.350	5	30	0.712896
Differential Transformer	5	6.417	4	30	0.710321
Softmax RoPE	6	6.483	1	30	0.709556
Differential Transformer-v2	7	6.567	3	30	0.710313
Linear Attention	8	6.733	1	30	0.707615
Gated DeltaNet	9	6.883	4	30	0.707802
GLA	10	7.100	1	30	0.711188
Softmax	11	7.217	1	30	0.706500
Kimi Delta Attention	12	7.333	2	30	0.705180
TSLib anomaly (mean F-score)					
FW-MLP-NoΔ	1	3.533	2	15	0.820800
Mamba-3	2	5.167	4	15	0.817544
FW-MLP	3	5.400	1	15	0.815433
GLA	4	5.633	3	15	0.815660
Gated Softmax	5	6.633	0	15	0.814953
Linear Attention	6	6.767	1	15	0.814827
Differential Transformer-v2	7	7.067	1	15	0.814433
Differential Transformer	8.5	7.167	0	15	0.814240
Softmax RoPE	8.5	7.167	1	15	0.815000
Gated DeltaNet	10	7.367	1	15	0.813960
Softmax	11	7.700	1	15	0.814653
Kimi Delta Attention	12	8.400	1	15	0.544547

Task-level uncertainty and aggregation rules. The ranking tables report one ordering, and Table 10 measures how much of it depends on the choice of tasks. A percentile bootstrap over the 26 tasks puts the mean rank of FW-MLP at 4.54 with a 95% interval of $[3.52, 5.60]$, and FW-MLP has the lowest mean rank in 71% of the resamples. The interval for the difference from FW-MLP-NoΔ is $[-1.62, 0.50]$, and FW-MLP ranks above the ablation on 11 tasks and below it on 8. Against Softmax RoPE, both Differential Transformers, and Mamba-3, FW-MLP ranks higher on 12 tasks and lower on 6 to 8. Against Gated Softmax, GLA, Softmax, Gated DeltaNet, Linear Attention, and Kimi Delta Attention, FW-MLP ranks higher on 14 to 17 of the 26 tasks and lower on at most 4, with sign tests between $p = 0.031$ and $p < 0.001$. Removing any single task leaves FW-MLP first. On six tasks all variants tie, which adds the same rank to each variant and one win to every win count.

The 26-task grouping collapses the 51 TSLib groups into three families. Table 11 recomputes the ordering under four alternatives: the median task rank, an equal weight on each of the 74 raw groups, the 23 non-TSLib tasks alone, and the count of first-place finishes. FW-MLP is first under every rule.

Table 10: Task-level uncertainty for the VariantBench ranking at attention budget 8. The bracketed interval is a 95% percentile-bootstrap interval over the 26 tasks (20,000 resamples), and $P(\text{first})$ is the share of resamples in which the variant has the lowest mean rank. W/L/T counts the tasks on which FW-MLP ranks above, below, or level with the variant; the sign test uses the untied tasks and the Wilcoxon test uses the nonzero rank differences.

Variant	Mean rank [95% CI]	$P(\text{first})$	FW-MLP W/L/T	Sign p	Wilcoxon p
FW-MLP	4.54 [3.52, 5.60]	0.71	–	–	–
FW-MLP-NoΔ	5.02 [3.88, 6.15]	0.15	11/8/7	0.648	0.655
Softmax RoPE	5.48 [4.62, 6.35]	0.06	12/6/8	0.238	0.222
Differential Transformer-v2	5.73 [4.67, 6.79]	0.06	12/6/8	0.238	0.230
Differential Transformer	5.88 [5.06, 6.67]	0.00	12/6/8	0.238	0.031
Mamba-3	5.92 [4.60, 7.29]	0.02	12/8/6	0.503	0.144
GLA	6.81 [5.79, 7.79]	0.00	16/3/7	0.004	0.006
Gated Softmax	6.90 [6.19, 7.63]	0.00	14/4/8	0.031	0.010
Softmax	7.08 [6.19, 7.96]	0.00	14/4/8	0.031	0.010
Gated DeltaNet	7.65 [6.58, 8.73]	0.00	17/2/7	$<$0.001	0.001
Linear Attention	8.12 [7.06, 9.17]	0.00	16/2/8	0.001	0.001
Kimi Delta Attention	8.87 [7.77, 9.94]	0.00	17/2/7	$<$0.001	$<$0.001

Table 11: The VariantBench ordering under five aggregation rules at attention budget 8. The 74-group rule weights every raw dataset and backbone group equally, so the 51 TSLib groups carry 51 of the 74 weights; each TSLib group enters at its family's average raw rank (Table 9). Lower is better except for first-place finishes. The fourth column names the best variant other than the two FW-MLP configurations. FW-MLP also has the lowest 26-task mean rank in all 26 leave-one-task-out rankings.

Rule	FW-MLP	FW-MLP-NoΔ	Best other variant	FW-MLP place
Mean over 26 tasks	4.54	5.02	Softmax RoPE (5.48)	1
Median over 26 tasks	4.00	5.50	Softmax RoPE (6.00)	1
Mean over 74 raw groups	4.87	5.07	Softmax RoPE (6.19)	1
Mean over 23 non-TSLib tasks	4.91	5.37	Softmax RoPE (5.30)	1
First places (ties count)	14	11	Differential Transformer-v2 (12)	1

The runner-up changes: FW-MLP-NoΔ is second under the 26-task mean, the 74-group mean, and the median; Softmax RoPE is second under the non-TSLib mean; and Differential Transformer-v2 is second by first-place finishes.

R.4 Attention-budget robustness

Recent mixers differ in how much capacity they place on the attention side: Gated DeltaNet, Kimi Delta Attention, and Gated Softmax add gates, projections, or state-update paths, and Mamba-3 moves capacity that would otherwise sit in the feed-forward path into the mixing layer. The budget rule of Appendix R.1 fixes this attention-side allocation, so every comparison measures the mixing mechanism at a common allocation. A sweep over $B \in \{8, 6, 4\}$ then shows how each variant responds to the allocation. Under the matched budget, rotary embeddings improve the mean rank of softmax attention in every reported aggregate. A learned output gate improves it over the 26 tasks and worsens it over the 23 non-TSLib tasks at budgets 8 and 6 (Tables 6, 7, 14, and 15).

Table 12 summarizes the sweep. At budget 6 the 26-task ordering of the two FW-MLP configurations holds, with FW-MLP first at 4.962 and FW-MLP-NoΔ second (Table 14), and FW-MLP is also first over the 23 non-TSLib tasks (Table 15). Table 16 gives every task-level rank at budget 6, and Tables 17 and 18 give the per-task scores and the family-level time-series results. At budget 4 Softmax RoPE leads the 26-task aggregate and FW-MLP places sixth of twelve, with the full ordering in Table 13. At this budget the fixed structural costs of FW-MLP (shortcuts, convolutional mixing, and, in the full model, the interval generator) take a larger share of the module budget, and the solver shrinks the learnable low-rank transport branches. The second-stage value/output rank VO_2 at $d = 128$ falls from 100 at budget 8 to 64 at budget 6 and 28 at budget 4 (Table 12).

Table 12: Attention-budget sweep on the 26 paper-level tasks. The budget-8 row averages seeds 1337–1339 and the budget-6 and budget-4 rows use seed 1337. VO_2 is the second-stage value/output rank the budget solver assigns to FW-MLP at model width $d = 128$.

Budget	FW-MLP VO_2 rank	FW-MLP position	Leading variant
8	100	1 of 12	FW-MLP
6	64	1 of 12	FW-MLP
4	28	6 of 12	Softmax RoPE

Table 13: Aggregate ordering of the 12 mixers at budget 4 over the 26 paper-level tasks with seed 1337, from the best mean tie-aware rank to the worst.

Position	Variant	Position	Variant
1	Softmax RoPE	7	FW-MLP-NoΔ
2	Gated DeltaNet	8	Softmax
3	Gated Softmax	9	Differential Transformer-v2
4	Differential Transformer	10	GLA
5	Mamba-3	11	Linear Attention
6	FW-MLP	12	Kimi Delta Attention

Table 14: Tie-aware variant ranking at attention budget 6 over the 26 paper-level tasks, with seed 1337.

Rank	Variant	Avg tie-aware rank	Tie-aware wins	Tasks
1	FW-MLP	4.962	12	26
2	FW-MLP-NoΔ	5.750	10	26
3	Differential Transformer	5.923	12	26
4	Softmax RoPE	6.115	11	26
5	Differential Transformer-v2	6.173	11	26
6	GLA	6.288	9	26
7	Gated Softmax	6.308	9	26
8	Softmax	6.404	10	26
9	Gated DeltaNet	6.904	10	26
10	Kimi Delta Attention	7.615	8	26
11	Linear Attention	7.692	9	26
12	Mamba-3	7.865	8	26

Table 15: Tie-aware variant ranking at attention budget 6 over the 23 non-TSLib tasks, with seed 1337.

Rank	Variant	Avg tie-aware rank	Tie-aware wins	Tasks
1	FW-MLP	5.348	10	23
2	Softmax RoPE	5.804	11	23
3	Differential Transformer	5.826	12	23
4	Differential Transformer-v2	5.978	11	23
5	FW-MLP-NoΔ	6.022	9	23
6	Softmax	6.087	10	23
7.5	Gated DeltaNet	6.500	10	23
7.5	GLA	6.500	9	23
9	Gated Softmax	6.565	9	23
10	Kimi Delta Attention	7.674	8	23
11.5	Linear Attention	7.848	9	23
11.5	Mamba-3	7.848	8	23

Table 16: Task-level tie-aware rank matrix at attention budget 6 over the twelve mixers. Lower rank is better, and tied entries receive their average rank. Column abbreviations are Diff/Diff-v2 for the two Differential Transformer variants, GDN for Gated DeltaNet, GSoft for Gated Softmax, FW for FW-MLP, Kimi for Kimi Delta Attention, Lin for Linear Attention, M3 for Mamba-3, NoΔ for FW-MLP-NoΔ, SM for Softmax, and SM-RoPE for Softmax RoPE.

Task	Diff	Diff-v2	GDN	GSoft	GLA	FW	Kimi	Lin	M3	NoΔ	SM	SM-RoPE
MQAR	1	2	8	3	11	7	4	5	12	9	10	6
MQAR-hard	1	12	2	7	10	3	4	9	11	5	6	8
MQAR-long	1	9	2	8	4	12	11	6	10	3	7	5
copy	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
prefix 3-gram	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
suffix 3-gram	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
selective copy	9	12	4	6	3	2	8	11	1	5	10	7
induction heads	6	6	6	6	6	6	12	6	6	6	6	6
modular sum	3	5.5	8	5.5	11	9	3	3	12	10	1	7
sMNIST	9	5	6	4	11	3	12	10	2	7	8	1
pMNIST	3	6	1	8	5	2	12	11	10	4	9	7
sequential CIFAR-10	9	1	10	7	8	2	12	11	3	6	5	4
dSprites	10	4.5	9	6	3	1	12	8	11	4.5	2	7
ListOps	4.5	2	10	3	11	7.5	12	9	6	7.5	4.5	1
SCAN simple	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
SCAN length	6	6	6	6	6	6	6	6	12	6	6	6
SCAN jump	5	5	5	5	5	11	5	5	12	10	5	5
SCAN turn-left	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5
bAbI	6	6	6	6	6	6	6	6	12	6	6	6
SASRec ML-1M	6	9	12	11	4	1	2	10	8	3	7	5
TinyStories	8	9	3	11	6	4	2	12	1	5	10	7
PathMNIST	4	1	11	10	2	5	12	9	7	8	3	6
Speech Commands	10	4	8	6	5	3	9	11	12	1	2	7
TSLib forecasting	11	10	9	4	5	1	7	2	12	8	6	3
TSLib classification	3	5	9	4	6	1	7.5	7.5	10	2	11.5	11.5
TSLib anomaly	6	8	12	5	3	4	7	10	2	1	9	11

Table 17: Compact per-task results at attention budget 6. Each cell gives the score followed by the tie-aware rank in parentheses. Where several variants hold the best score, the Best column gives how many of the 12 variants reach it. On the non-TSLib tasks the Difference column gives the FW-MLP primary score minus the Softmax RoPE primary score; on the TSLib families it gives the Softmax RoPE average raw rank minus the FW-MLP average raw rank, so positive values favor FW-MLP in both cases.

Task	Metric	Best	FW-MLP	FW-MLP-NoΔ	Softmax RoPE	Difference
MQAR	token accuracy	Differential Transformer: 1.0000	0.9960 (r=7)	0.9935 (r=9)	0.9962 (r=6)	-0.0002
MQAR-hard	token accuracy	Differential Transformer: 0.9995	0.9850 (r=3)	0.9656 (r=5)	0.1204 (r=8)	+0.8646
MQAR-long	token accuracy	Differential Transformer: 0.9998	0.00025 (r=12)	0.6048 (r=3)	0.0229 (r=5)	-0.0226
copy	token accuracy	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
prefix 3-gram	token accuracy	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
suffix 3-gram	token accuracy	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
selective copy	token accuracy	Mamba-3: 1.0000	0.9989 (r=2)	0.9931 (r=5)	0.9692 (r=7)	+0.0297
induction heads	token accuracy	11 of 12 variants: 1.0000	1.0000 (r=6)	1.0000 (r=6)	1.0000 (r=6)	0.0000
modular sum	accuracy	Softmax: 0.1070	0.1030 (r=9)	0.0980 (r=10)	0.1040 (r=7)	-0.0010
sMNIST	accuracy	Softmax RoPE: 0.9043	0.8750 (r=3)	0.8574 (r=7)	0.9043 (r=1)	-0.0293
pMNIST	accuracy	Gated DeltaNet: 0.9004	0.8867 (r=2)	0.8652 (r=4)	0.8164 (r=7)	+0.0703
sequential CIFAR-10	accuracy	Differential Transformer-v2: 0.3652	0.3555 (r=2)	0.3418 (r=6)	0.3477 (r=4)	+0.0078
dSprites	accuracy	FW-MLP: 0.4551	0.4551 (r=1)	0.4355 (r=4.5)	0.4316 (r=7)	+0.0234
ListOps	accuracy	Softmax RoPE: 0.5430	0.5117 (r=7.5)	0.5117 (r=7.5)	0.5430 (r=1)	-0.0312
SCAN simple	exact match	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
SCAN length	exact match	11 of 12 variants: 1.0000	1.0000 (r=6)	1.0000 (r=6)	1.0000 (r=6)	0.0000
SCAN jump	exact match	9 of 12 variants: 1.0000	0.9977 (r=11)	0.9984 (r=10)	1.0000 (r=5)	-0.0023
SCAN turn-left	exact match	all 12 variants: 1.0000	1.0000 (r=6.5)	1.0000 (r=6.5)	1.0000 (r=6.5)	0.0000
bAbI	exact match	11 of 12 variants: 1.0000	1.0000 (r=6)	1.0000 (r=6)	1.0000 (r=6)	0.0000
SASRec ML-1M	Hit@10	FW-MLP: 0.2526	0.2526 (r=1)	0.2480 (r=3)	0.2455 (r=5)	+0.0071
TinyStories	token accuracy	Mamba-3: 0.6037	0.5532 (r=4)	0.5306 (r=5)	0.4673 (r=7)	+0.0859
PathMNIST	accuracy	Differential Transformer-v2: 0.8579	0.8370 (r=5)	0.8310 (r=8)	0.8347 (r=6)	+0.0023
Speech Commands	accuracy	FW-MLP-NoΔ: 0.9309	0.9221 (r=3)	0.9309 (r=1)	0.9181 (r=7)	+0.0041
TSLib forecasting	avg raw rank $\downarrow$	FW-MLP: 3.1667	3.1667 (r=1)	7.3333 (r=8)	4.8333 (r=3)	+1.6667
TSLib classification	avg raw rank $\downarrow$	FW-MLP: 5.1333	5.1333 (r=1)	5.3500 (r=2)	7.2167 (r=11.5)	+2.0833
TSLib anomaly	avg raw rank $\downarrow$	FW-MLP-NoΔ: 4.3333	5.6667 (r=4)	4.3333 (r=1)	8.0667 (r=11)	+2.4000

Table 18: Detailed TSLib family results at attention budget 6. Family ranks come from tie-aware ranks over the raw dataset and backbone groups and enter the 26-task aggregation, and the mean raw metric is shown for reference.

Variant	Family rank	Avg raw task rank	Raw wins	Raw tasks	Mean raw metric
TSLib forecasting (mean MSE)					
FW-MLP	1	3.167	2	6	0.247348
Linear Attention	2	3.500	2	6	0.246292
Softmax RoPE	3	4.833	0	6	0.249669
Gated Softmax	4	5.000	1	6	0.249711
GLA	5	6.000	0	6	0.249768
Softmax	6	6.167	0	6	0.251423
Kimi Delta Attention	7	7.000	0	6	0.250592
FW-MLP-NoΔ	8	7.333	0	6	0.252329
Gated DeltaNet	9	8.167	1	6	0.252643
Differential Transformer-v2	10	8.333	0	6	0.253124
Differential Transformer	11	9.000	0	6	0.252138
Mamba-3	12	9.500	0	6	0.257376
TSLib classification (mean accuracy)					
FW-MLP	1	5.133	3	30	0.714082
FW-MLP-NoΔ	2	5.350	7	30	0.715104
Differential Transformer	3	6.017	7	30	0.708542
Gated Softmax	4	6.067	2	30	0.709295
Differential Transformer-v2	5	6.317	2	30	0.709863
GLA	6	6.633	3	30	0.710524
Kimi Delta Attention	7.5	6.917	1	30	0.709777
Linear Attention	7.5	6.917	3	30	0.705995
Gated DeltaNet	9	7.033	2	30	0.708975
Mamba-3	10	7.183	3	30	0.707382
Softmax RoPE	11.5	7.217	3	30	0.707116
Softmax	11.5	7.217	2	30	0.705670
TSLib anomaly (mean F-score)					
FW-MLP-NoΔ	1	4.333	4	15	0.819593
Mamba-3	2	4.533	4	15	0.818193
GLA	3	4.867	3	15	0.815453
FW-MLP	4	5.667	1	15	0.818240
Gated Softmax	5	5.867	1	15	0.815727
Differential Transformer	6	6.700	1	15	0.814800
Kimi Delta Attention	7	6.867	0	15	0.759227
Differential Transformer-v2	8	7.033	0	15	0.813473
Softmax	9	7.767	0	15	0.814467
Linear Attention	10	7.900	1	15	0.813480
Softmax RoPE	11	8.067	0	15	0.813300
Gated DeltaNet	12	8.400	0	15	0.813333

R.5 Interval-gated retrieval diagnostic

This diagnostic builds a retrieval task whose target depends on all three causal regions, the configuration that Proposition N.2 isolates. Each length-56 sequence holds six key–value records, each written as adjacent key and value tokens. The records are followed by a query token, a query key, and a query interval class. The key vocabulary has four entries and the value vocabulary twelve labels. Three records share the query key, so key matching alone identifies a set of three candidates. Positions outside the records carry neutral tokens or A/B marker tokens, with marker probability 0.45. Each record whose value sits at position i has the interval class

$$c_{t,i} = (\#\mathrm{A}_{i+1:t} + 2\#\mathrm{B}_{i+1:t}) \bmod 4,$$

computed from the marker tokens between i and the query position t. The target is the value of the record whose key matches the query key and whose interval class matches the query class. Sampling requires the three same-key records to carry distinct interval classes, and every sequence

Table 19: Interval-gated retrieval diagnostic. All runs use a two-layer causal backbone with hidden size 64, 4096 resampled training examples per epoch, 1024 validation examples, 12 epochs, attention budget 8, and seeds 1337–1339. Entries are the mean ± standard deviation over the three seeds, and chance accuracy is $1/12 = 8.33\%$.

Variant	Accuracy (%) ↑	Loss ↓
FW-MLP	34.34 ± 0.56	1.265 ± 0.015
FW-MLP-NoΔ	20.44 ± 6.74	1.840 ± 0.211
Gated DeltaNet	17.71 ± 1.66	2.199 ± 0.014
Linear Attention	17.19 ± 2.14	1.917 ± 0.005
Gated Softmax	16.96 ± 1.04	1.945 ± 0.033
Softmax RoPE	16.05 ± 1.11	1.931 ± 0.022

also contains a different-key record with the query class. A model therefore reaches the target through the conjunction of the query endpoint, the candidate key/value endpoint, and the marker statistic of $X_{i+1:t}$.

The diagnostic is reported separately from VariantBench and measures the value of an explicit interval factor on a task whose target depends on an additive statistic over $X_{i+1:t}$. The full FW-MLP adds the cancellative interval branch of Equation 9 to the endpoint towers of FW-MLP-NoΔ.

All variants use the same two-layer causal backbone with hidden size 64, MLP ratio 2, batch size 128, AdamW with learning rate 7×10^{-4} and no weight decay, and the attention-budget rule at budget 8; the head-parameterized baselines use four attention heads. Each run trains for 12 epochs with 4096 freshly resampled examples per epoch and evaluates on 1024 validation examples; reported values are the mean and standard deviation over seeds 1337–1339. FW-MLP and FW-MLP-NoΔ use the two-branch head layout of Appendix R.1, and the budget solver matches the attention-side parameter target.

Table 19 gives accuracy and loss for every variant, together with the chance rate of 8.33%. FW-MLP is first and the endpoint-only ablation second. Endpoint fast MLPs recover interval structure through their separable rank (Theorem D.2), and the interval branch supplies the additive statistic to the score directly. The larger margin and lower seed spread of the full model are consistent with this difference. The four baselines fall in a narrow band below both FW-MLP configurations.

R.6 nanoGPT validation

The nanoGPT setting trains each variant from scratch on OpenWebText (Gokaslan & Cohen, 2019; Radford et al., 2019; Karpathy, 2022) with a shared GPT-2-style scaffold: 12 pre-norm decoder blocks, hidden size 768, context length 1024, MLP ratio 4.0, no dropout, and tied input and output embeddings. Each run replaces the sequence-mixing module inside the attention block and keeps the residual path and the feed-forward block fixed. Training uses AdamW with peak learning rate 6×10^{-4}, cosine decay, 2000 warmup updates, minimum learning rate 6×10^{-5}, weight decay 0.1, gradient clipping at 1.0, and an effective global batch of approximately 0.5M tokens. The data are tokenized with the GPT-2 BPE vocabulary, and each validation point averages the loss over 200 random validation blocks. Each configuration is trained once. Checkpoints are numbered in the order of the logged validation evaluations. Perplexity ratios are computed as $\exp(\Delta\text{loss})$.

Table 20 gives the best validation loss of every run as a difference from FW-MLP, which reaches 3.0014. The ordering places FW-MLP ahead of Gated DeltaNet, the endpoint-only ablation FW-MLP-NoΔ, DeltaNet, Softmax RoPE, and Linear Attention RoPE at a matched budget. Two configurations that replace one or both FW-MLP stages with quadratic computation reach lower loss still and measure the headroom of the design once the linear-time budget is released.

Within the FW-MLP family the ladder locates the gain inside the block. Starting from FW-MLP-NoΔ, removing the local convolution, the gates, or the rotary embedding raises the best loss by at most 0.0076. Removing the endpoint split or replacing the hidden activation raises it further, and of these changes a linear hidden activation costs the most. A single-stage FW-MLP sits 0.0570 above the full model, the largest change of any single ablation. This gap is consistent with Proposition O.1: the

Table 20: nanoGPT best validation loss of every run, reported as a difference from FW-MLP, which reaches 3.0014. Negative values sit below FW-MLP. The two quadratic configurations release the linear-time budget; all other rows are matched-budget runs.

Run	Best loss − 3.0014
FW-MLP, quadratic in both stages	−0.0169
FW-MLP, quadratic in the second stage	−0.0052
FW-MLP	0.0000
Gated DeltaNet	+0.0048
FW-MLP-NoΔ	+0.0231
FW-MLP-NoΔ, no local convolution	+0.0265
FW-MLP-NoΔ, no gates	+0.0271
FW-MLP-NoΔ, no RoPE	+0.0307
DeltaNet	+0.0340
FW-MLP-NoΔ, no endpoint split	+0.0383
FW-MLP-NoΔ, ReLU hidden activation	+0.0383
FW-MLP, single stage	+0.0570
Softmax RoPE	+0.0625
FW-MLP-NoΔ, linear hidden activation	+0.0658
Linear Attention RoPE	+0.1419

Table 21: Late-training statistics of five matched-budget nanoGPT runs. L5 and L30 are the means over the last five and the last 30 logged validation checkpoints, and the lowest-of-five column counts the checkpoints among those 30 at which the run has the lowest validation loss of the five. The last column is the first checkpoint at which the validation loss reaches 3.03, and a dash marks a run whose logged curve stays above that value.

Run	Best loss	L5 loss	L30 loss	Lowest of five	Checkpoint at 3.03
FW-MLP	3.0014	3.0056	3.0166	29	54
Gated DeltaNet	3.0062	3.0154	3.0278	1	65
FW-MLP-NoΔ	3.0245	3.0303	3.0388	0	76
Softmax RoPE	3.0639	3.0685	3.0798	0	—
Linear Attention RoPE	3.1433	3.1554	3.1648	0	—

rank of the second-stage value/output branch bounds the subspace of every residual update. Table 21 reports the late-training statistics of FW-MLP, Gated DeltaNet, FW-MLP-NoΔ, Softmax RoPE, and Linear Attention RoPE. The best-loss ordering holds in every column: FW-MLP has the lowest mean loss over the last five and the last 30 checkpoints and the lowest loss of the five at 29 of those 30, and it reaches the 3.03 threshold first, at checkpoint 54.

R.7 Downstream language-model evaluation

The language-model setting trains each variant in a unified autoregressive configuration at two scales: 340M parameters on 15B tokens and 1.3B parameters on 100B tokens of FineWeb-Edu (Penedo et al., 2024). Within a scale, all models share the tokenizer, context length, optimizer recipe, training-token budget, and evaluation harness, and the comparison replaces the sequence-mixing block while the surrounding language-model architecture and the downstream protocol stay fixed.

The 16 evaluation tasks form two blocks. The Open LLM Leaderboard block (Fourrier et al., 2024; Gao et al., 2024) holds MMLU-Pro (Wang et al., 2024), GPQA (Rein et al., 2023), BBH (Suzgun et al., 2023), MATH level 5 (Hendrycks et al., 2021), MuSR (Sprague et al., 2024), and IFEval (Zhou et al., 2023), and follows the leaderboard protocol: 5-shot multiple choice for MMLU-Pro, 0-shot multiple choice for GPQA and MuSR, 3-shot multiple choice for BBH, 0-shot generation for IFEval, and 4-shot generation for MATH level 5. The general-ability block holds ARC-Challenge and ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), BoolQ (Clark et al., 2019), WinoGrande (Sakaguchi et al., 2020), COPA (Roemmele et al., 2011), OpenBookQA (Mihaylov et al., 2018), and SciQ (Welbl et al., 2017). Multiple-choice tasks report

Table 22: Unified language-model evaluation across model families and scales. Task scores are percentages, and higher is better. They are normalized accuracy where the task implementation provides it and accuracy otherwise, exact match for MATH level 5, and strict instruction-level (IFE-I) and prompt-level (IFE-P) success rates for IFEval. Rank is the mean rank over the 16 tasks within a scale, where lower is better, and Top1 counts the tasks on which a variant reaches the best score. Bold and underline mark the best and second-best entries in each column within each scale. The Variant column writes DiffTrans for Differential Transformer and Transformer for the softmax Transformer baseline, and the column headers abbreviate MMLU-Pro (MMLU-P), MATH level 5 (MATH), ARC-Challenge (ARC-C), ARC-Easy (ARC-E), HellaSwag (HS), WinoGrande (WinoG), and OpenBookQA (OBQA).

	Open LLM Leaderboard							General Ability									Ranking	
Variant	MMLU-P	GPQA	BBH	MATH	MuSR	IFE-I	IFE-P	ARC-C	ARC-E	HS	PIQA	BoolQ	WinoG	COPA	OBQA	SciQ	Rank	Top1
1.3B Params – 100B Tokens																		
DeltaNet	10.9	26.3	30.8	1.1	41.7	28.8	16.5	26.6	52.2	50.2	70.4	61.1	54.1	74.0	31.8	76.1	4.12	0
GSA	11.0	**27.0**	29.4	**1.3**	**43.8**	30.0	17.9	28.7	52.9	**51.0**	**71.2**	54.1	53.6	76.0	33.0	77.3	2.81	5
RetNet	11.0	25.2	29.3	0.1	38.4	5.6	2.4	27.1	48.9	48.0	70.1	58.3	53.3	71.0	32.4	73.6	6.69	0
HGRN	11.4	26.9	29.7	0.8	40.9	25.3	12.2	27.1	51.8	48.1	70.7	58.4	51.5	70.0	32.6	69.5	4.88	0
HGRN2	**11.5**	25.4	29.5	0.2	35.0	22.3	12.9	28.2	50.4	31.7	67.1	41.6	52.2	**77.0**	32.8	37.8	5.75	2
GLA	11.4	25.9	29.5	0.6	42.7	27.2	15.7	27.7	48.2	48.8	70.2	57.4	54.1	69.0	32.6	72.1	4.94	0
Transformer	11.4	25.9	29.6	1.1	36.5	27.0	14.1	28.0	49.2	49.2	70.5	**62.1**	**55.2**	76.0	31.8	76.9	3.94	2
FW-MLP	11.3	26.9	**31.6**	**1.3**	43.1	**31.3**	**18.6**	**29.7**	**56.9**	50.7	70.3	61.5	53.9	**77.0**	**36.1**	**78.4**	**1.94**	**9**
340M Params – 15B Tokens																		
DiffTrans	10.9	25.9	29.9	0.8	39.0	26.6	13.3	28.9	53.1	**40.8**	**66.8**	**60.3**	**53.4**	69.0	33.0	73.4	2.94	4
Gated DeltaNet	**11.3**	26.0	29.6	**1.0**	42.1	25.8	13.3	27.6	52.7	39.6	66.2	58.8	52.7	71.0	33.8	**73.5**	2.69	3
DeltaNet	11.2	26.0	30.0	0.9	**45.2**	27.7	15.0	26.9	50.2	40.5	65.3	51.9	50.4	69.0	31.6	71.7	3.50	1
GLA	11.0	25.8	28.9	0.7	41.5	22.8	10.9	24.7	47.8	36.6	63.7	54.7	48.9	64.0	29.4	64.9	5.69	0
Transformer	10.6	**26.7**	29.2	**1.0**	38.6	26.9	12.6	27.3	50.6	39.6	65.0	56.9	49.9	72.0	32.4	72.7	3.94	2
FW-MLP	11.2	26.4	**30.7**	**1.0**	44.7	**27.9**	**15.1**	**29.3**	**53.4**	40.5	66.0	59.7	52.2	**73.0**	**34.1**	73.2	**1.69**	**8**

accuracy or normalized accuracy according to the task implementation, MATH reports exact match, and IFEval reports strict instruction-level and prompt-level success rates, which count as two of the 16 tasks. The baselines are the softmax Transformer, Differential Transformer (Ye et al., 2025), DeltaNet (Yang et al., 2024b), Gated DeltaNet and GLA (Yang et al., 2025; 2024a), GSA (Zhang et al., 2024), RetNet (Sun et al., 2023), and HGRN and HGRN2 (Qin et al., 2023; 2024).

Average rank is computed within a scale by ranking the variants on each task, giving tied scores the best shared rank, and averaging over the 16 tasks; the Top1 column counts the tasks on which a model reaches the best score, including ties.

FW-MLP holds the best average rank at both scales, 1.69 at the smaller scale and 1.94 at the larger, with 8 and 9 task wins. The closest variant is Gated DeltaNet at the smaller scale and GSA at the larger. Table 22 gives every per-task score together with the full ordering at each scale. At 1.3B, FW-MLP's largest per-task margins over the best other model are on ARC-Easy and OpenBookQA. At 340M FW-MLP leads BBH, IFEval, ARC-Challenge, ARC-Easy, COPA, and OpenBookQA, and Differential Transformer leads HellaSwag, PIQA, BoolQ, and WinoGrande. FW-MLP also has the best average rank within each task block at both scales, shared with Differential Transformer on the 340M general-ability block.

Table 23 pairs FW-MLP with every other model task by task, with the 16 tasks as the unit of replication. At both scales, FW-MLP scores above the closest model on 11 tasks and below it on 4 (GSA at 1.3B, Gated DeltaNet at 340M), and it scores above the 340M Differential Transformer on 11 of 16. Every other model scores below FW-MLP on at least 12 of the 16 tasks, with sign tests of $p = 0.077$ against the 1.3B Transformer and $p \leq 0.007$ in the other eight comparisons.

R.8 Efficiency measurements

Fixed-shape throughput. Table 24 reports compiled forward-pass throughput for FW-MLP and FW-MLP-NoΔ against four recent mixers: Gated DeltaNet, GLA, Kimi Delta Attention, and Mamba-3. All variants use the VariantBench block and differ only in the sequence-mixing module. Runs use one NVIDIA A100-SXM4-80GB GPU with PyTorch 2.10, BF16 autocast, and `torch.compile` in reduce-overhead mode, and each measurement averages 10 timed iterations after 3 warmups. Dividing each variant's throughput by the median of the four baselines at the same shape and taking

Table 23: Paired comparison of FW-MLP with every other model of Table 22, over its 16 tasks. W/L/T counts the tasks on which FW-MLP scores above, below, or level with the model; the sign test uses the untied tasks. Gap is the mean score difference in points. In a bootstrap over tasks (20,000 resamples) FW-MLP has the lowest mean rank in 94% of resamples at 1.3B and 98% at 340M. DiffTrans is Differential Transformer.

Scale	Model	Mean rank [95% CI]	W/L/T	Sign p	Gap
1.3B	FW-MLP	1.94 [1.31, 2.69]	–	–	–
	GSA	2.81 [1.88, 3.88]	11/4/1	0.118	+1.28
	Transformer	3.94 [3.06, 4.81]	12/4/0	0.077	+2.13
	DeltaNet	4.12 [3.25, 5.06]	14/2/0	0.004	+1.62
	HGRN	4.88 [3.94, 5.81]	13/2/1	0.007	+3.23
	GLA	4.94 [4.12, 5.75]	14/2/0	0.004	+2.84
	HGRN2	5.75 [4.50, 6.88]	14/1/1	<0.001	+7.69
	RetNet	6.69 [6.19, 7.19]	16/0/0	<0.001	+5.24
340M	FW-MLP	1.69 [1.31, 2.06]	–	–	–
	Gated DeltaNet	2.69 [2.19, 3.25]	11/4/1	0.118	+0.84
	DiffTrans	2.94 [2.19, 3.69]	11/5/0	0.210	+0.83
	DeltaNet	3.50 [2.75, 4.19]	13/1/2	0.002	+1.56
	Transformer	3.94 [3.19, 4.62]	14/1/1	<0.001	+1.65
	GLA	5.69 [5.31, 6.00]	16/0/0	<0.001	+3.88

Table 24: Compiled forward-pass throughput (tokens/second) on one NVIDIA A100-SXM4-80GB for FW-MLP, FW-MLP-NoΔ, and four recent mixers, at three representative shapes (B, L, D, H) of batch size, sequence length, model width, and head count. All variants use the same block and differ only in the sequence-mixing module. Throughput is averaged over 10 timed iterations after 3 warmups. The last column reports the ratio of FW-MLP's throughput to the median throughput of the four recent baselines. The column header writes Kimi Delta for Kimi Delta Attention.

Setting	Shape (B, L, D, H)	FW-MLP	FW-MLP-NoΔ	Gated DeltaNet	GLA	Kimi Delta	Mamba-3	FW-MLP / median
VariantBench proxy	$(4, 1024, 256, 8)$	5.71M	5.79M	3.39M	4.88M	3.22M	6.49M	1.38×
nanoGPT proxy	$(2, 1024, 768, 12)$	1.82M	2.73M	1.70M	2.30M	1.52M	2.70M	0.91×
LM proxy	$(1, 2048, 1024, 16)$	1.53M	1.46M	1.63M	2.25M	0.67M	1.89M	0.87×

the geometric mean over the three shapes gives $1.03\times$ for FW-MLP and $1.17\times$ for FW-MLP-NoΔ, where higher is faster. FW-MLP therefore computes its larger score class at a throughput comparable to that of recent linear mixers. Panel (d) of Figure 3 draws the ratio of every mixer at each shape.

Long-context scaling and numerical equivalence. The second measurement varies the sequence length. Two implementations of the FW-MLP mixing core run on identical inputs: the materialized tri-region score of Equation 10, which builds the full (t, i) score matrix, and the interval recurrence of Equation 11, which carries the state of Theorem G.1. Both are written in plain PyTorch, and the recurrence uses a chunk size of 128. The sweep varies L at a fixed $(B{=}1, H{=}16, d_k{=}d_v{=}64, R{=}4)$ shape on one NVIDIA L40S (46,068 MiB, BF16), timing 10 iterations after 3 warmups and recording peak allocated memory. Panels (a) and (b) of Figure 3 draw the result, and Table 25 lists every measured value. The memory of the materialized form grows quadratically and reaches the device limit past $L{=}8{,}192$. The memory of the recurrence grows linearly, and the recurrence evaluates $L{=}131{,}072$ in under 3 GiB at a roughly constant 0.085M tokens per second.

Table 26 checks Theorem G.1 numerically. On three shapes in float32, the maximum absolute difference between the two implementations is at most 1.6×10^{-4} against outputs of mean magnitude 3.47 to 5.09; this difference is at the float32 accumulation floor for a sum of this length. The recurrence used for the memory and latency curves therefore computes the same function as the explicit tri-region score up to float32 accumulation error.

Tracks and heads. Table 27 repeats the recurrence measurement at $L{=}65{,}536$ for $R \in \{1, 2, 4, 8\}$ interval tracks and $H \in \{8, 16, 32\}$ heads. The reference implementation evaluates the tracks one after another, so latency scales with R: going from one track to eight multiplies it by 7.8 to 8.2 at

Table 25: Peak memory, forward latency, and throughput of the two forms of the FW-MLP mixing core at every sequence length L of the sweep drawn in panels (a) and (b) of Figure 3. All measurements use one NVIDIA L40S in BF16 with batch 1, $H = 16$, $d_k = d_v = 64$, $R = 4$, and chunk 128, and time 10 iterations after 3 warmups with both forms on identical inputs (PyTorch 2.12.1). The materialized score runs out of device memory from $L = 16{,}384$ on.

	Materialized score (Eq. 10)			Recurrence (Eq. 11)		
L	MiB	ms	M tokens/s	MiB	ms	M tokens/s
512	129.8	0.84	0.612	27.7	6.36	0.080
1,024	475.9	3.99	0.257	38.9	12.02	0.085
2,048	1,841.6	18.76	0.109	61.4	23.85	0.086
4,096	7,267.1	74.36	0.055	106.4	48.27	0.085
8,192	28,894.1	294.52	0.028	196.4	97.53	0.084
16,384	out of memory			376.4	193.44	0.085
32,768	out of memory			736.4	386.81	0.085
65,536	out of memory			1,456.4	759.72	0.086
131,072	out of memory			2,896.4	1,542.90	0.085

Table 26: Numerical check of the exact equivalence in Theorem G.1. For each shape we run the materialized tri-region score (Eq. 10) and the linear-time interval recurrence (Eq. 11) on identical float32 inputs and report the mean output magnitude and the maximum and mean absolute differences of their outputs. The differences are at the float32 accumulation floor. All runs use one NVIDIA L40S.

Shape (N, L, d_k, d_v, R)	Output mean $\lvert y\rvert$	Max abs. diff.	Mean abs. diff.
$(4, 128, 32, 32, 4)$	3.47	3.8×10^{-5}	8.9×10^{-7}
$(2, 256, 64, 64, 4)$	5.09	3.4×10^{-5}	2.1×10^{-6}
$(8, 512, 64, 64, 4)$	4.94	1.6×10^{-4}	3.6×10^{-6}

every head count. Heads are batched, so at batch 1 latency varies by at most 6% across H at fixed R. Peak memory doubles with each doubling of H and rises by at most 4% from R=1 to R=8. The (R=4, H=16) cell reproduces the corresponding point of panels (a) and (b) of Figure 3 to within 3%. Panel (c) draws the same sweep at $L \in \{4{,}096, 16{,}384, 65{,}536\}$.

Table 27: Latency and peak memory of the linear-time interval recurrence at sequence length 65,536 as the number of interval tracks R and heads H vary. All measurements use one NVIDIA L40S in BF16 with batch 1, $d_k = d_v = 64$, and chunk 128, time 10 iterations after 3 warmups, and run the mixing core alone with random factors (PyTorch 2.10.0). The reference implementation processes tracks one after another and batches heads, so latency grows linearly in R and stays nearly flat in H at batch 1, and memory grows with H.

	Latency (ms)			Peak memory (MiB)		
R	$H = 8$	$H = 16$	$H = 32$	$H = 8$	$H = 16$	$H = 32$
1	194	191	184	720	1,432	2,857
2	371	374	380	724	1,440	2,873
4	772	779	748	732	1,456	2,905
8	1,533	1,498	1,512	748	1,488	2,969

S FORWARD PASS, NOTATION, AND RELATION TO FAST-WEIGHT PROGRAMMERS

S.1 A CHUNKED FORWARD PASS

Algorithm 1 evaluates one FW-MLP mixer as defined in Section 3. The first stage keeps the low-rank projected prefix statistic

$$M^{(1)}_{t,a} = W^{(1)\top}_{k,a} C_t W^{(1)}_{v,a} = \sum_{i \le t} (x_i W^{(1)}_{k,a})^\top (x_i W^{(1)}_{v,a}),$$

so $x_t B_a C_t U_a = (x_t W^{(1)}_{q,a}) M^{(1)}_{t,a} W^{(1)\top}_{o,a}$ and the endpoint map $\mathcal{H}_a(x_t, C_t)$ of Equation 6 is evaluated without forming C_t. The second stage carries the scaled state $T_{t,h,r} = \gamma_{t,h,r} S_{t,h,r}$, for which Equation 11 becomes

$$T_{t,h,r} = e^{-d_{t,h,r}} T_{t-1,h,r} + F^{k\top}_{t,h} v^{(2)}_{t,h}, \qquad y^{(2)}_{t,h,r} = F^q_{t,h} T_{t,h,r}. \tag{153}$$

The write at t enters after the old state decays, so the pair (t,t) has an empty interval. Each step multiplies the old state by a decay $e^{-d} \le 1$. Within a chunk $J = (b, e]$ the recurrence is an inclusive scan under the associative composition $(\delta_2, E_2) \circ (\delta_1, E_1) = (\delta_2 \delta_1,\ \delta_2 E_1 + E_2)$ with identity $(1, 0)$: its prefix $(\bar\delta_t, \bar E_t)$ at $t \in J$ gives $T_t = \bar\delta_t T_b + \bar E_t$, and only T_e carries to the next chunk. Each chunk lies inside one sequence, and every state restarts at a sequence boundary (Appendix L).

Algorithm 1 One FW-MLP mixer forward pass, with chunked prefix scans

Require: Row tokens $x_{1:L}$, slow parameters, head routing a^q_h, a^k_h, chunk length c, configured gates, normalizers, and optional shortcut
Ensure: Mixer outputs $o_{1:L}$

for each sequence segment **do**
  $M^{(1)}_a \leftarrow 0$ for every a; $T_{h,r} \leftarrow 0$ for every h, r
  for each consecutive chunk $J = (b, e]$ of at most c tokens **do**
    for each first-stage head a **do**
      $Q^{(1)}_{t,a} \leftarrow x_t W^{(1)}_{q,a}$, $K^{(1)}_{t,a} \leftarrow x_t W^{(1)}_{k,a}$, $V^{(1)}_{t,a} \leftarrow x_t W^{(1)}_{v,a}$ for $t \in J$
      $M^{(1)}_{t,a} \leftarrow M^{(1)}_a + \mathrm{PrefixSum}_{j \in J}[(K^{(1)}_{j,a})^\top V^{(1)}_{j,a}]_t$
      $\mathcal{H}_{t,a} \leftarrow \sigma\Big(Q^{(1)}_{t,a} M^{(1)}_{t,a} W^{(1)\top}_{o,a} + x_t W^{(1)}_{\mathrm{static},a} + b^{(1)}_a\Big)$ for $t \in J$
      $M^{(1)}_a \leftarrow M^{(1)}_{e,a}$
    end for
    for each second-stage head h **do**
      $F^q_{t,h} \leftarrow \mathcal{H}_{t,a^q_h} W^{(2)}_{q,h}$, $F^k_{t,h} \leftarrow \mathcal{H}_{t,a^k_h} W^{(2)}_{k,h}$ for $t \in J$
      $v^{(2)}_{t,h} \leftarrow (x_t W^{(2)}_{v,h}) \odot G^{(2)}_{L,h}(x_t)$ for $t \in J$
      $d_{t,h} \leftarrow \lambda \phi_+(x_t W_{\Delta,h})$, $\omega_{t,h} \leftarrow [1 + \tanh(x_t W_{\omega,h})]/R$ for $t \in J$
      $E_{t,h} \leftarrow F^{k\top}_{t,h} v^{(2)}_{t,h}$ for $t \in J$
      for $r = 1, \dots, R$ **do**
        $(D_{t,h,r}, U_{t,h,r})_{t \in J} \leftarrow \mathrm{Scan}_\circ[(e^{-d_{t,h,r}}, E_{t,h})_{t \in J}]$
        $T_{t,h,r} \leftarrow D_{t,h,r} T_{h,r} + U_{t,h,r}$ for $t \in J$
        $y^{(2)}_{t,h,r} \leftarrow F^q_{t,h} T_{t,h,r}$ for $t \in J$
        $T_{h,r} \leftarrow T_{e,h,r}$
      end for
      $y^{(2)}_{t,h} \leftarrow \sum_{r=1}^R \omega_{t,h,r} y^{(2)}_{t,h,r}$ for $t \in J$
      $\widehat{y}_{t,h} \leftarrow \mathcal{N}_h(y^{(2)}_{t,h}) \odot G^{(2)}_{R,h}(x_t)$ for $t \in J$
    end for
    $o_t \leftarrow \mathrm{Concat}_h(\widehat{y}_{t,h}) W^{(2)}_o + \mathcal{S}^{(1)}_t$ for $t \in J$
  end for
end for
return $o_{1:L}$

In Algorithm 1, $\mathcal{N}_h$ is the per-head normalization, $G^{(2)}_{L,h}$ and $G^{(2)}_{R,h}$ are the write-side and output-side gates of Appendix O.2, and $\mathcal{S}^{(1)}_t$ is the optional first-stage shortcut, zero when disabled. The algorithm

omits the residual and feed-forward scaffold around the mixer, together with the local convolution and the rotary embedding that some configurations enable (Table 20 ablates both). FW-MLP-NoΔ fixes the interval factor at one ($R = 1$, $d_{t,h,1} = 0$, $\omega_{t,h,1} = 1$) and has no interval-generator parameters. The budget solver meets its parameter target through the ranks. Per sequence, the second-stage state holds $O(HRd_kd_v)$ numbers and the recurrence costs $O(LHRd_kd_v)$ operations.

The scalar example, end to end. Take scalar tokens $x_u = 1$, a ReLU activation, $B_a = U_a = 1$ with no static weight or bias, $W^{(2)}_{q,h} = 1$, $W^{(2)}_{k,h} = 2$, and values $v^{(2)}_{u,h} = 1$. Then $C_u = u$, so $F^q_u = u$ and $F^k_u = 2u$. One track with $\lambda = \log 2$, $W_{\Delta,h} = 1$, and $W_{\omega,h} = 0$ gives $d_u = \log 2$, $\omega = 1$, and $\gamma_u = 2^{-u}$, so $A^{\text{FW-MLP}}_{t,i} = 2ti\,2^{-(t-i)}$; for instance $A^{\text{FW-MLP}}_{4,2} = 4$. The recurrence of Equation 11 builds $S_1, \ldots, S_4 = 4, 20, 68, 196$ and returns $y_4 = 2^{-4} \cdot 4 \cdot 196 = 49$, the materialized sum $1{+}4{+}12{+}32$ over $i = 1, \ldots, 4$. The scaled state of Equation 153 gives $T_1, \ldots, T_4 = 2, 5, 8.5, 12.25$ and the same $y_4 = 4 \cdot 12.25 = 49$. Both values of y_4 are the mixing output before normalization, gating, and the output projection.

S.2 Notation

Table 28 collects the symbols of Section 3. The labels d, d_r, and d_{pre} denote the residual width, the first-stage activation width, and the pre-activation width, with $d_{\text{pre}} = d_r$ for ReLU and $d_{\text{pre}} = 2d_r$ for a split GLU; $r^{(1)}_{qk,a}$ and $r^{(1)}_{vo,a}$ are per-head factor dimensions.

Table 28: Notation for one mixer. Shapes suppress batch and sequence axes.

Symbol	Meaning and shape	Defined in
t, i, u	Query, key ($i \le t$), and generic token positions	Section 3
a, h, r; A, H, R	Head/track indices; first-stage head, second-stage head, and track counts	Section 3; this table
$x_u, X_{1:u}$	Row token $1 \times d$ and stacked prefix $u \times d$	Section 2
C_u	$X^\top_{1:u} X_{1:u}$; additive $d \times d$ prefix statistic	Section 3.1
$W^{(1)}_{q,a}, W^{(1)}_{k,a}$	Slow routing factors, each $d \times r^{(1)}_{qk,a}$	Eq. 6
$W^{(1)}_{v,a}, W^{(1)}_{o,a}$	Slow write/readout factors; $d \times r^{(1)}_{vo,a}$ and $d_{\text{pre}} \times r^{(1)}_{vo,a}$	Eq. 6
B_a, U_a	Products of slow factors; $d \times d$ and $d \times d_{\text{pre}}$	Eq. 6
$W^{(1)}_{\text{static},a}, b^{(1)}_a$	Trained context-independent map and bias; $d \times d_{\text{pre}}$ and $1 \times d_{\text{pre}}$	Eq. 6
$\mathcal{H}_a(x, C)$	Activated prefix-conditioned row, $1 \times d_r$	Eq. 6
a^q_h, a^k_h	First-stage heads selected for the second-stage query and key branches	Eq. 7
$W^{(2)}_{q,h}, W^{(2)}_{k,h}$	Slow endpoint projections, $d_r \times d_k$	Eq. 7
$F^q_{t,h}, F^k_{i,h}$	Query/key endpoint features, $1 \times d_k$	Eq. 7
$W_{\Delta,h}, W_{\omega,h}$	Slow increment and readout projections, $d \times R$	Eqs. 8–9
$d_{u,h}, p^\Delta_{t,h}, z^\Delta_{t,i,h}$	Nonnegative increments, prefix sums, and interval differences, each with R entries	Eq. 8
λ, ϕ_+	Increment scale and nonnegative activation	Section 3.2
$\omega_{t,h}, F^\Delta_{t,i,h}$	Track weights (R entries) and scalar interval factor conditional on x_t	Eq. 9
$A^{\text{FW-MLP}}_{t,i,h}$	Product of the scalar endpoint score and the interval factor	Eq. 10
$W^{(2)}_{v,h}, v^{(2)}_{i,h}$	Slow value projection $d \times d_v$ and gated value row $1 \times d_v$	Eq. 10
$\gamma_{t,h,r}, S_{t,h,r}$	Exponential prefix factor and unscaled $d_k \times d_v$ mixer state	Eq. 11
$T_{t,h,r}$	Scaled state $\gamma_{t,h,r} S_{t,h,r}$, $d_k \times d_v$	Eq. 153
$M^{(1)}_{t,a}$	Low-rank projected prefix state, $r^{(1)}_{qk,a} \times r^{(1)}_{vo,a}$	Algorithm 1

Symbol	Meaning and shape	Defined in
$G^{(2)}_{L,h}, G^{(2)}_{R,h}, \mathcal{N}_h$	Write gate, output gate, and per-head normalization; each acts in value coordinates	Appendix O.2; Algorithm 1
$W^{(2)}_o, \mathcal{S}^{(1)}_t$	Output projection $Hd_v \times d$ and optional $1 \times d$ first-stage shortcut	Algorithm 1

S.3 Score-side and value-side fast weights

Fast-weight programmers use one network to write the weights of another (Schmidhuber, 1992), and fast-weight memories and hypernetworks develop the same idea (Ba et al., 2016; Ha et al., 2017). In the value-side form the prefix writes a matrix that the query reads to produce the output value: linear attention is this form (Katharopoulos et al., 2020; Schlag et al., 2021), delta rules make the write error-driven (Yang et al., 2024b; 2025), and test-time-training layers replace the matrix with a network updated by gradient steps (Sun et al., 2025). For fixed features, an additive value-side memory induces a bilinear score between the query and each stored key. Gates and delta updates make the transport of that score depend on the history, in a way set by the form of each update rule.

FW-MLP places the fast network on the score side. The prefix instantiates the endpoint MLPs whose product is the score for (t, i), and the interval enters that score through a separately parameterized factor built from learned exponential tracks over an additive statistic. The second stage of FW-MLP is itself a value-side linear memory per track (Equation 11), so the two placements compose within one layer.

S.4 Cauchy's equation and the interval factor

Proposition E.1 reduces to the continuous Cauchy functional equation: subtracting $G(0)$ turns $G(a+b) - G(a) = g(b)$ into $\tilde{G}(a+b) = \tilde{G}(a) + \tilde{G}(b)$, whose continuous solutions are linear (Aczél, 1966). The equation thus fixes which prefix maps extract an interval exactly by subtraction, and it also fixes the exponential transport. A positive continuous D on $\mathbb{R}^m_+$ with $D(0) = 1$ and $D(z+z') = D(z)D(z')$ has additive $\log D$, hence $D(z) = e^{\beta^\top z}$, and $D \le 1$ on $\mathbb{R}^m_+$ forces $\beta = -\alpha$ with $\alpha \ge 0$.