Cassandra T2 Design Doc

Cassandra T2: A Pre-Registered Architecture and Training Plan for Retrieval-Native Masked-Diffusion Language Models

Thomas Garren · SOPHIA XT LLC · `thomas@sophiaxt.com` **Version 0.1** · *2026-05-10*

> **Status: design doc, no results yet.** This document specifies what we propose to build and how we will measure whether it works. Sections 6.x ("Results") are intentionally blank — they will be filled in as training runs complete. Claims are calibrated by the founder at approximately 30% probability of producing the full headline result. Failure modes are enumerated in Section 7.

Abstract

We propose **Cassandra T2**, a masked-diffusion language model whose attention layer is *structurally aware of its retrieval substrate*. The design composes four ideas: (i) **Anchor-Aware Triple Attention** with content, timestep, and anchor paths (shipped as a tested module; not yet trained); (ii) **LTMi-XT priors** in which the anchor path consumes per-anchor retrieval relevance scores and 3D lattice coordinates from the LTMi-XT v0.1 retrieval format; (iii) **Lattice-Compressed Context (LCC)**, a long-context mechanism in which older context is auto- crystallized into LTMi-XT loci and addressed via the same lattice-anchor path as user-supplied retrieval anchors; (iv) a **preconditioning training stack** combining Multi-Head Latent Attention (MLA), Sophia optimizer, lens-xt-aware curriculum, selective int4 quantization, and fused kernels, targeting a wall-clock parity between an RTX 3090 Ti and H200-days of conventional training.

The Cassandra T1.5 baseline produced a measured 59× corpus-overlap lift over T1 base via anchor-token-masked continued pretraining. T2 extends this baseline architecturally for the

first time. The training plan includes pre-registered ablations and success criteria. Results sections are marked TBD and will be revised as runs complete.

We commit to releasing the Anchor-Aware Triple Attention module and the LCC reference implementation under Apache 2.0 regardless of the outcome. Specifications, training scripts, and bundle formats remain CC BY 4.0. Negative results will be published.

---

1. Introduction

1.1 Position

SOPHIA XT has shipped two validated artifacts in 2026:

1. **The anchor-token-masking training methodology** (Garren 2026a), a controlled empirical study comparing three masking schedules on retrieval-grounded MDLM fine-tuning. Anchor-token masking (Q never masked, A always subject to schedule) produces a **1.67× pooled out-of-distribution generalisation advantage** over standard random- position masking, with mechanism causally tested via V/O-only LoRA ablation. Bootstrap CIs are non-overlapping with both controls.

2. **Cassandra T1.5**, the same 1.3B-parameter masked-diffusion model as T1 with 500 steps of anchor-token-masked continued pretraining on LTMi-XT bundles. Held-out diagnostic pooled corpus-overlap rises from 0.008 (T1 base) to 0.481 (T1.5 step-500), a 59× lift on the same eval the training methodology was validated against.

The *training* methodology is validated. The *architecture* is unchanged from the original Cassandra T1 design — a standard masked-diffusion transformer with sliding-window GQA attention, RoPE, RMSNorm, SwiGLU. T2 is the first proposed architectural break.

1.2 Why now

Three reasons:

- **The architecture treats retrieval as if it were content.** Forced- anchor decoding works (`anchor_preservation_rate=1.000` measured), but the model has no structural mechanism to *trust-weight* anchors by retrieval confidence or to leverage lattice-region structure. The attention layer is blind to its own retrieval substrate.

- **Context scaling is a wall everyone hits the same way.** Compressive Transformers (Rae et al. 2020) compress with learned summarisation; Memorizing Transformers (Wu et al. 2022) use kNN lookup; Titans (Behrouz et al. 2024) learn a separate memory module. None *address* memory in an inspectable, externally manipulable space. We have such a space — the LTMi-XT lattice — and we are not using it for memory yet.
- **Compute economics.** Cassandra T1.5 was trained on a single RTX 3090 Ti. Most current MDLM work targets H100/H200 clusters. Closing that gap is necessary, not optional, for sole-founder research velocity.

1.3 What this document is

A pre-registered plan. It specifies:

1. The exact architectural changes (Section 3)
2. The exact training pipeline changes (Section 4)
3. The experiments that will test each (Section 5)
4. The success criteria *before* any results are seen (Section 5.5)
5. The failure modes and mitigation strategy (Section 7)
6. The open-source release plan regardless of outcome (Section 8)

Sections 6.1–6.5 are placeholders. They are kept in the document so that updates can be tracked in-place.

---

2. Background — what is already real

2.1 Cassandra T1 / T1.5

- 28-layer transformer, 1.3B parameters, hidden 2048, 16 query heads with 4 KV heads (GQA 4:1), head dim 128, RoPE $\theta = 5\times10^5$, RMSNorm, SwiGLU, 128K context with 4096 sliding window. Vocabulary 32,768 BPE. Mask token at id 32,766.
- Open-released under Apache 2.0 at `github.com/Chorozion/Casandra-t1-diffusion-edge-model`.
- T1.5 = T1 epoch-5 checkpoint + 500 steps of anchor-token-masked continued pretraining on LTMi-XT bundles C1–C7.
- Held-out pooled `corpus_overlap`: T1 = 0.008, T1.5-step500 = 0.481, T1.5-step1000 = 0.440 (mild overfitting past 500). Step 500 selected as the empirical winner.

2.2 Anchor-token masking

Asymmetric masking schedule. In each (Q, A) training pair, Q tokens are never masked; A tokens are subject to a random or curriculum masking schedule. Validated to produce:

- 1.67× pooled OOD corpus_overlap advantage over random masking
- Mechanism: V/O projections carry the inductive bias (V-only LoRA outperforms full LoRA at half the trainable parameters; Q-only LoRA underperforms a random-mask control with P = 0.006).
- Replicates with P = 1.000 across astronomy, medical, and legal held- out corpora.

2.3 LTMi-XT v0.1 format

Apache 2.0 retrieval format with three load-bearing properties:

- **4-level breadcrumb hierarchy**: `[topic, subtopic, concept, slot]`
- **3D lattice coordinates** derived via BLAKE2b over breadcrumb prefixes: `lattice = (h(b₀)%64, h(b₀b₁)%64, h(b₀b₁b₂)%64)`. Loci sharing k breadcrumb levels share k lattice coordinates.
- **Decay and horizon** fields for temporal management.

2.4 lens-xt and forced-anchor decoding

`lens-xt 0.1.0b1` (live on PyPI) is a declarative YAML spec language for token-level deterministic generation. A `.lensx` document specifies position-locked content; the runtime resolves the spec into a forced-anchor decode where locked positions are excluded from the unmasking loop. On masked-diffusion backends, anchor preservation is guaranteed by construction. Verified end-to-end against Cassandra T1.5: `anchor_preservation_rate = 1.000`, `achieved_guarantee = DETERMINISTIC`.

2.5 What is shipped, not yet trained

The Anchor-Aware Triple Attention module (Section 3.1) is implemented, unit-tested (7/7 smoke tests passing), and gated by config flag. It has not been used in any training run. **No claims rest on it yet.**

---

3. Proposed architectural changes

3.1 Anchor-Aware Triple Attention

The standard sliding-window GQA attention is replaced with a three-path block. All three paths share the input hidden state `x`; their outputs are mixed by a per-position softmax gate over `[P₁, P₂, P₃]`:

Path 1 — Content. Bit-for-bit identical to the existing SlidingWindowAttention. Q, K, V from `x`; standard scaled-dot-product attention with optional sliding-window mask. Reproduces T1.5 behaviour when isolated.

Path 2 — Timestep. Q from `x`; K and V from the diffusion timestep embedding `t_emb`, broadcast across all sequence positions. Lets the model pull diffusion-stage information as a routing signal distinct from token content, rather than only through AdaLN normalisation.

Path 3 — Anchor. Q from `x`; K and V from `x` but with attention scores masked so each query position can only attend *to* anchor positions (positions where `anchor_mask` is True). During training, anchors are the always-visible Q tokens (per the anchor-token-masking methodology). During inference under lens-xt forced-anchor decoding, anchors are the lock positions specified in the `.lensx` spec.

Gate. A per-position linear layer maps `[x, anchor_present]` → 3 softmax logits. Bias initialised to `[2, 0, 0]`, so at step 0 the gate weights are approximately `[88%, 6%, 6%]` — the new paths are small perturbations the optimiser learns into.

Warm-start from T1.5. Path 1 weights are initialised from the single-path T1.5 checkpoint (via `init_triple_from_single`). Paths 2 and 3 start at random init but contribute only 6% each via the gate. This means the very first training step of T2 behaves nearly identically to T1.5, then drifts as the optimiser turns the new paths on.

Parameter overhead. Approximately +48% over the single-path baseline at 1.3B scale, mostly in the duplicate Q/K/V/O projections. This is significant but enables the structural prior.

3.2 LTMi-XT priors

Path 3 optionally consumes two additional tensors at the boundary between the lens-xt runtime and the model:

- `anchor_scores` [B, S]: per-anchor retrieval relevance in [0, 1], zero at non-anchor positions. Passed through a `Linear(1, num_heads)` projection (zero-initialised) and added as additive bias to the Path 3 attention scores. Scaled by a scalar `relevance_gate` (initialised zero) so the contribution is exactly zero at training step 0.

- `anchor_lattice` [B, S, 3]: 3D lattice coordinates of the source locus for each anchor position, zero at non-anchors. Three learned embedding tables (one per axis, each `Embedding(64, kv_heads × d)`) produce a positional vector added into the Path 3 K projection. Scaled by a separate `ltmi_gate` scalar.

Both gates are scalar Parameters initialised to zero. At init the LTMi-aware variant is functionally identical to plain Triple Attention. The optimiser turns the gates up as it discovers value.

Conceptual claim. This is the first MDLM design (to our knowledge) where the attention layer's bias is structurally calibrated by the retrieval system's confidence and lattice topology. The model can learn:

- "Anchors with high retrieval relevance deserve more attention."
- "Anchors from the same lattice region tend to cluster — give them coherent positional encoding."

3.3 Lattice-Compressed Context (LCC) — the new mechanism

The novel composition. Older context tokens are *auto-crystallised* into LTMi-XT loci every N tokens by the same crystalliser used in Sophia Harness. Crystallised loci are placed at their canonical lattice coordinates and fed to Path 3 of Triple Attention via the same `anchor_mask` / `anchor_scores` / `anchor_lattice` interface used for user-supplied retrieval anchors.

The model does not distinguish user-supplied anchors from self- crystallised memory. Both are lattice-indexed addressable content.

What LCC composes from existing work:

- **Compressive Transformers** (Rae et al. 2020): compress older context into a smaller bank. We agree, but compress *deterministically* via the crystalliser rather than via learned summarisation.
- **Memorizing Transformers** (Wu et al. 2022): retrieve from past activations via kNN. We agree on the retrieval mechanism, but use hash-based lattice lookup (O(1)) instead of learned kNN (O(K log N)).
- **Titans** (Behrouz et al. 2024): a neural memory module that learns at test time. We don't learn at test time; the crystalliser is deterministic and the model only learns *how to use* the lattice memory, not how to construct it.
- **In-Context Memory variants** (various 2023–2024): memory tokens placed in prompt sequence. We place memory at *lattice* positions, not sequence positions — RoPE doesn't

have to extrapolate.

Claim novelty. The combination — deterministic BLAKE2b compression into a fixed 64^3 = 262,144-cell lattice, addressed by the same anchor-aware attention path that consumes user-supplied retrievals — is, to our knowledge, not in the published literature.

Realistic capacity argument. With 64^3 cells and a per-cell capacity of ~1 locus statement (~400 chars ≈ 100 tokens), the addressable memory ceiling is ~26M tokens per context. Beyond that, older crystallisations evict newer ones at the same lattice cell (natural LRU via the locus `last_referenced` timestamp). This is a hard upper bound, not a soft one — important to state.

3.4 Multi-Head Latent Attention (MLA) — efficiency layer

For T2 we propose replacing GQA with MLA per DeepSeek-V2/V3. K and V are factored through a per-layer low-rank latent (rank ≈ 128), then decompressed on the fly. This reduces attention parameter count and KV-cache memory at inference by ~4–5×.

MLA composes cleanly with Triple Attention: the latent K, V cache is shared across Paths 1, 2, and 3 (each path has its own decompression matrices). The Path 3 anchor mask is applied after decompression as before.

---

4. Preconditioning training stack

A stack of techniques targeting wall-clock parity between RTX 3090 Ti and H200-days of conventional training. The aggregate target is **10–15× wall-clock speedup over the conventional pipeline that produced T1.5**.

4.1 Optimiser: Sophia (Liu et al. 2024)

Drop-in replacement for AdamW. Uses a diagonal Hessian estimate to precondition the gradient. Published ~2× speedup on LLM pretraining at matched perplexity. We adopt with identical hyperparameter template; risk is low.

4.2 Curriculum from lens-xt specs

The training data is `.lensx` specs at various complexity tiers:

- **Tier 1**: single-literal-lock spec, no retrieval (easy)
- **Tier 2**: single-locus-retrieval spec, top-k = 1

- **Tier 3**: multi-locus-retrieval spec, top-k = 3
- **Tier 4**: reasoning-scaffold spec with multi-stage locks (hard)

Training batches are sampled with a curriculum that linearly shifts from Tier 1 → Tier 4 over the first 30% of training, then samples uniformly. Expected convergence speedup: ~1.5× to a given loss target (based on prior curriculum-learning literature; not yet measured for this specific setup).

4.3 Quantisation

Selective int4 in feed-forward layers (where MoE-style gating tolerates quantisation well), bf16 in attention and embeddings (where stability matters). Expected memory reduction: ~2.5×; expected wall-clock speedup: ~1.3–1.5× on Ampere/Hopper.

We explicitly *do not* go full BitNet b1.58 in v1 of T2 — that's a larger risk to take in the same change.

4.4 Kernel fusion: Liger + Flash Attention 3

RMSNorm + RoPE + SwiGLU fused via Liger Kernels (~1.4×); Flash Attention 3 on supported hardware (~1.2× on Ampere, ~1.5× on Hopper).

4.5 Honest speedup analysis

```
 Stacked theoretical:   2 × 1.5 × 1.4 × 1.3 × 1.2 ≈ 6.6×

Empirical decay:       MLA + Triple Attention add overhead that partially
                       offsets the optimiser/curriculum/kernel gains.

Conservative estimate: 4–6× wall-clock over the v1.5 baseline pipeline
                       on identical hardware.

Hardware ratio:        RTX 3090 Ti ≈ 80 TFLOPS FP16 sparse
                       H200        ≈ 1000 TFLOPS FP16 sparse
                       Ratio:      ~12.5×

To reach parity:       Pipeline efficiency needs to wash out the 12.5×
                       hardware gap. 4–6× is ~half of that. Realistic
                       claim is "1–2 days on 3090 Ti for what would
                       take ~1 day on H200", not "matches H200-days
                       exactly".
```

The aspirational claim becomes: **wall-clock parity within 2× on a 3090 Ti compared to a single H200 conventional pipeline.** That is still significant for a sole-founder lab.

---

5. Experimental plan

5.1 Baseline

Cassandra T1.5 step-500, the empirical winner from the validated continued-pretraining run. Held-out pooled corpus_overlap = 0.481. Frozen checkpoint at SHA-256 `d75d404cec050931ca75362e50229a668b69a7bf1cf4dd84c90ec3e7babd8618`.

5.2 Ablation grid

Each variant is trained from the T1.5 step-500 checkpoint (warm-start for Triple variants) for the same number of additional steps (target: 2,000) on the same data corpus (target: 50K-locus Wikipedia LTMi-XT bundle).

ID	Architecture	LTMi priors	LCC	MLA	Sophia
V0	T1.5 (single-path GQA)	n/a	n/a	n/a	AdamW
V1	+ Triple Attention (no priors)	off	off	off	AdamW
V2	+ Triple Attention	**on** (gates active)	off	off	AdamW
V3	+ LCC self-crystalliser	on	**on**	off	AdamW
V4	+ MLA replacement	on	on	**on**	AdamW
V5	+ Sophia optimiser	on	on	on	**Sophia**
V6	full T2	on	on	on	Sophia

The grid is full-stack incremental. Each ablation adds one variable. V0 = baseline. V6 = full T2 proposal.

5.3 Metrics

Quality metrics (preregistered, same as anchor-token-masking paper):

- `pooled_corpus_overlap` — primary; on held-out C8 + C9 (to be built from a different Wikipedia subset than the training data)
- `pooled_english_ratio` — sanity (must stay above 0.5)
- `anchor_preservation_rate` — must stay at 1.0 for DETERMINISTIC guarantee
- `bootstrap_95_ci` — 1000-sample resample, must be non-overlapping with V0 to claim improvement

Efficiency metrics:

- `seconds_per_step` (wall clock) on RTX 3090 Ti
- `peak_memory_mb`
- `attention_flops` (theoretical, from profiler)
- `convergence_steps_to_loss_X` (for loss target X = 2.0)

5.4 Statistical methodology

Same as the anchor-token-masking paper: bootstrap confidence intervals with 1,000 resamples on a held-out eval set of ≥ 50 queries. Improvement claims require non-overlapping 95% CIs from baseline. P-values reported via permutation test.

5.5 Success criteria (preregistered)

Before any results are seen, we commit to the following thresholds. Falling below means the variant is *not better than baseline*, regardless of how it looks.

For T2 to be claimed as an improvement over T1.5:

1. **Quality:** V6's pooled corpus_overlap > V0's, with non-overlapping bootstrap 95% CIs.
2. **Anchor preservation:** V6 maintains `anchor_preservation_rate ≥ 0.99` on a 100-query forced-anchor decode test set.
3. **Efficiency:** V6 reaches V0's final loss in ≤ 50% of V0's wall- clock time on the same hardware.

For the LCC mechanism specifically (V3 - V2):

4. On a long-context eval (sequences > 8K tokens, requiring recall of information from > 4K tokens back), V3 outperforms V2 with non- overlapping bootstrap 95% CIs on a "long-context recall accuracy" metric to be specified before the eval set is built.

If any of the four fails, we report it as such. **There is no shame in a negative result; there is shame in claiming a positive one without earning it.**

5.6 Compute budget

Estimated GPU-hours on RTX 3090 Ti:

- V0 baseline (already in hand from T1.5 work): 0
- V1 — V5 ablations, 2,000 steps each: ~10–15 GPU-hours each = 50–75 total
- V6 full T2, 5,000 steps: ~25–40 GPU-hours
- Long-context eval (V3 vs V2): ~5 GPU-hours

Total estimated: 80–120 GPU-hours = 3–5 days of continuous run-time on a single RTX 3090 Ti.

If the preconditioning stack delivers the predicted 4–6× speedup, the total compresses to ~1–1.5 days.

---

6. Results

6.1 V1 — Triple Attention ablation

To be filled in after V1 run completes. Will include: training loss curve, held-out corpus_overlap (T1.5 vs V1), bootstrap CIs, anchor_preservation_rate, wall-clock per step.

6.2 V2 — LTMi-XT priors ablation

To be filled in after V2 run completes. Will include: gate value trajectories (relevance_gate and ltmi_gate over training steps; if they remain near zero, the priors aren't being used).

6.3 V3 — Lattice-Compressed Context ablation

To be filled in after V3 run completes. Includes: long-context recall benchmark against V2.

6.4 V4 + V5 — MLA and Sophia stack

To be filled in after V4 and V5 runs complete.

6.5 V6 — Full T2 efficiency analysis

To be filled in after V6 run completes. Includes: H200-equivalent analysis based on measured TFLOPS utilisation.

7. Limitations and failure modes

7.1 Compute

Single RTX 3090 Ti. We cannot run a true scratch T2 in any reasonable timeframe. All T2 work proceeds via warm-start from T1.5 step-500. This is a real limitation: warm-start from a single-path arch into a triple-path arch may not be a stable starting point. **If V1 (Triple Attention with frozen priors) degrades versus V0, the warm-start strategy is broken and we revisit before continuing.**

7.2 Data scaling

The LTMi-XT corpus we have today is ~10K loci across C1–C7. Real training needs 100K+. We commit to building a 50K-locus Wikipedia- derived LTMi-XT bundle (script written, not yet executed at full scale) before any T2 training run starts.

Risk: Wikipedia-derived loci may not match the (Q, A) shape required by the anchor-token-masking pipeline. Mitigation: synthetic (Q, A) generation via a Mercury-class API (cost: ~$50 for 50K pairs at current Inception pricing — acceptable).

7.3 Architecture risk

Triple Attention with LTMi priors and LCC is three architectural extensions stacked. Each independently has known precedent (Triple Attention via Compressive Transformers' multi-head extensions; LTMi priors via cross-attention with metadata; LCC via memory mechanisms generally). The *composition* is novel and has no precedent we can cite for stability. **Each ablation in Section 5.2 must individually clear the success threshold** — if V1 wins but V2 loses, we ship V1 without the priors.

7.4 Open questions before training

- Should `lattice_x_emb`, `lattice_y_emb`, `lattice_z_emb` be added or concatenated? Currently added (head-dim-preserving). Concatenation would need a downstream projection.
- Should the LCC crystalliser fire every N tokens or every N seconds of attention compute?
- For long-context eval, what's the right held-out eval set? Need to build one that's independent of training data.

7.5 What would falsify the design

Pre-registered: any of the following falsifies the T2 thesis as written.

- V6's pooled corpus_overlap is within 5% of V0's baseline (no meaningful quality improvement).
- The Triple Attention gates (Sections 3.1, 3.2) remain near zero throughout training (model never learns to use the new paths).
- V6's anchor_preservation_rate drops below 0.95 (we broke the DETERMINISTIC guarantee in exchange for noise).
- V6's wall-clock per step is greater than V0's despite all the preconditioning techniques (the architectural overhead outweighed the efficiency gains).

---

8. Open-source strategy

8.1 Apache 2.0 (community)

Released to the public regardless of T2 outcome:

- Anchor-Aware Triple Attention module (`triple_attention.py`)
- LTMi priors implementation
- LCC self-crystalliser interface
- All training scripts and ablation code
- `lens-xt` runtime updates required to pass priors and lattice info

8.2 CC BY 4.0 (specifications)

- This document
- Updates to the LTMi-XT specification covering the LCC mechanism
- Spec for the `.lensx` extension covering self-crystallised memory

8.3 Internal / commercial

- Trained Cassandra T2 weights (initially Pro/Scale tier on the LENS API; potentially open-released later in the lineage)
- The 50K+ locus Wikipedia LTMi-XT bundle (initially internal; the builder script is open)
- Customer-specific V/O LoRA adapters trained on Sophia Custom

8.4 Reproducibility commitments

- All experiments use deterministic seeds (preregistered: 1234, 5678, 9012 for three runs each).
- All eval sets used for V6's headline number are released as Apache 2.0 LTMi-XT bundles after the result is announced.
- All hyperparameters are documented in the released training scripts.
- Negative results are published with the same effort as positive ones.

---

9. Conclusion

This document specifies a six-variant ablation testing whether LTMi-XT-aware Triple Attention with Lattice-Compressed Context, atop a preconditioning training stack, produces a measurable quality and efficiency improvement over Cassandra T1.5 on retrieval-grounded masked-diffusion language modelling.

The architecture has zero results yet. The training code is partial. The eval corpora are not yet built. **Sections 6.1–6.5 are intentionally empty.**

The founder estimates 30% probability that V6 clears all four preregistered success criteria. The bet rests on:

- a validated training methodology (anchor-token masking, 1.67× OOD proven)
- a validated decoding paradigm (forced-anchor, anchor_preservation_rate = 1.000 proven)
- a deterministic retrieval substrate with structural properties no other MDLM has access to (LTMi-XT v0.1 spec, BLAKE2b lattice)
- a sole founder with limited compute committed to publishing negative results

The architecture and training code referenced in this document are shipped as of v0.1 of this report. Results will be reported in the v0.2 revision after V1–V3 runs complete.

References

Behrouz, A., et al. (2024). *Titans: Learning to Memorize at Test Time*.

DeepSeek-AI. (2024). *DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model*. arXiv:2405.04434.

DeepSeek-AI. (2024). *DeepSeek-V3 Technical Report*. arXiv:2412.19437.

Garren, T. (2026a). *Anchor-Token Masking Produces an Out-of-Distribution Generalisation Advantage in Retrieval-Grounded Masked-Diffusion Language Models*. SOPHIA XT LLC technical report.

Garren, T. (2026b). *LTMi-XT: A Topological Retrieval Format for Masked-Diffusion Language Models, v0.1*. SOPHIA XT LLC specification.

Liu, H., et al. (2024). *Sophia: A Scalable Stochastic Second-order Optimizer for Language Model Pre-training*. ICLR 2024.

Ma, S., et al. (2024). *The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits*. arXiv:2402.17764.

Peng, B., et al. (2023). *RWKV: Reinventing RNNs for the Transformer Era*. arXiv:2305.13048.

Rae, J. W., et al. (2020). *Compressive Transformers for Long-Range Sequence Modelling*. ICLR 2020.

Shah, J., et al. (2024). *FlashAttention-3: Fast and Accurate Attention with Asynchrony and Low-precision*. arXiv:2407.08608.

Wu, Y., et al. (2022). *Memorizing Transformers*. ICLR 2022.

Yang, G., & Hu, E. J. (2022). *Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer*. arXiv:2203.03466.

Revision history

- **v0.1** (2026-05-10): initial pre-registered design. No results. Founder confidence ~30%. Ablation grid frozen. Success criteria frozen. Sections 6.1–6.5 reserved for results.