LLM Pretraining Shapes a Generalizable Manifold: Insights into Cross-Modal Transfer to Time Series

Alexis Roger*
McGill University
Mila - Quebec AI Institute

Prateek Humane*
Université de Montréal
Mila - Quebec AI Institute

Zhenghan Tai
University of Toronto

Gwen Legate
Concordia University
Mila - Quebec AI Institute

Andrei Mircea
Université de Montréal
Mila - Quebec AI Institute

Vasilii Feofanov
42.com

Irina Rish
Université de Montréal
Mila - Quebec AI Institute

Abstract

Can language-pretrained transformers become effective time-series forecasters, and why? In this paper, we show that cross-modal transfer arises because language pretraining preconditions time series training with a reusable manifold. A linear probe on frozen LLM states decodes realistic time-series trajectories without paired supervision, and retrieval in this projected space yields competitive forecasts, showing that structure and dynamics exist before finetuning. Pretrained initialization also improves optimization, producing coherent gradients and a highly anisotropic loss landscape unlike random initialization. Finetuning then acts as low-dimensional alignment, reusing existing directions rather than learning temporal primitives from scratch, as evidenced by low-rank updates, subspace alignment, and shared features for periodicity, trend, and repetition. Together, these results support a geometric account of LLM-to-time-series transfer: language pretraining builds the manifold, and finetuning projects numerical dynamics onto task-relevant directions.

1 Introduction

Large Language Models (LLMs) transfer effectively across many downstream tasks, but it remains unclear whether their pretrained representations generalize beyond language itself. Time series forecasting provides a compelling test case: like language modeling, it is a sequential prediction problem, yet it operates on continuous numerical signals rather than discrete semantic tokens. If language pretraining improves forecasting, this would suggest that transformers learn reusable sequential structure rather than purely linguistic knowledge.

Recent work has explored routes from LLMs to time series, including input reprogramming [Jin et al., 2023], zero-shot digitized prompting [Gruver et al., 2023, Williams et al., 2024], parameter-efficient finetuning [Zhou et al., 2023, Wolff et al., 2025], and tokenized forecasting approaches such as Chronos [Ansari et al., 2024, Talukder et al., 2025, Roger et al., 2025]. However, empirical evidence remains contradictory. Some studies report little benefit from language pretraining and attribute gains primarily to transformer architectures or tokenization effects [Ansari et al., 2024, Tan et al., 2024, Zheng et al., 2025, Zhang et al., 2025], while others find strong improvements in low-data, cross-domain, or distribution-shift settings [Riachi et al., 2025, Bayazi et al., 2024, Qiu et al., 2026].

Preprint.

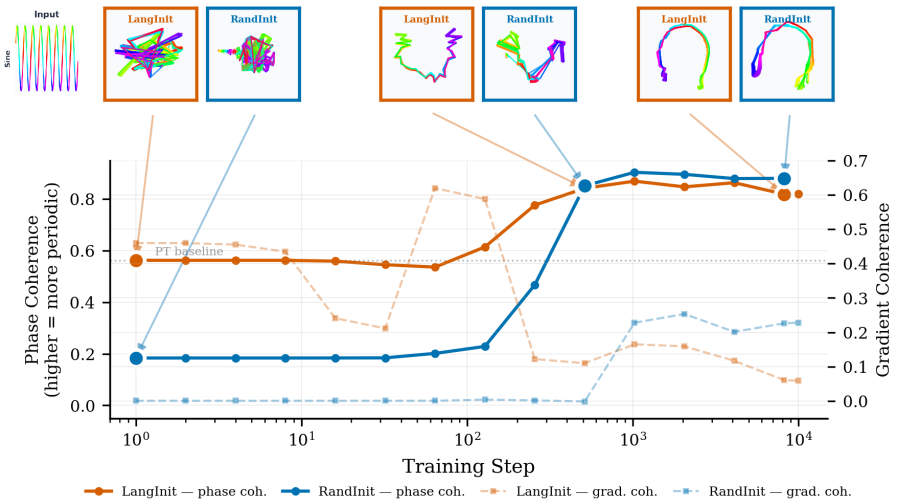


Figure 1: **Phase and gradient coherence across training.** Higher the coherence, better the model captures the periodic nature of the input (a sine wave here). Using t-SNE, we visualize hidden states at training steps 1, 512, and 8192 and see that pretrained LLM (LangInit) inherits periodic structure from pretraining, starting high. Random initialization (RandInit) starts near zero and undergoes an abrupt phase transition around step 300 where gradient coherence spikes, indicating that the model discovers periodic geometry. This is followed by aligning its gradients across the time series.

We believe that the transferred object is not semantic knowledge, but representational geometry. Language contains rich temporal structure (e.g., repetition, trends, periodicity, and long-range dependencies) and autoregressive pretraining organizes transformers into a sequential manifold encoding such dynamics. Under this view, time-series finetuning becomes a low-dimensional alignment problem: adapting existing temporal directions rather than learning forecasting structure from scratch.

To test this hypothesis, we repurpose Qwen3-0.6B [Yang et al., 2025], using next-token prediction paradigm for probabilistic time-series forecasting. We compare pretrained and randomly initialized models across full finetuning and parameter-efficient adaptation strategies on GiftEval benchmark [Aksu et al., 2024]. Our results show that pretrained models converge faster than identical architectures trained from scratch (Figure 1). We further find that pretrained hidden states linearly decode realistic temporal trajectories, produce coherent optimization gradients, and adapt through low-rank updates, supporting a geometric account of cross-modal transfer. Together, these findings suggest that LLMs help forecasting not because they understand the semantics of a signal, but rather because language pretraining already organizes the model into a reusable temporal representation space.

2 Related Work

LLMs and time-series forecasting. Recent time-series foundation models either pretrain directly on temporal data [Rasul et al., 2023, Goswami et al., 2024, Woo et al., 2024] or adapt pretrained LLMs through reprogramming, frozen-backbone alignment, or parameter-efficient finetuning [Jin et al., 2023, Zhou et al., 2023, Gruver et al., 2023, Wolff et al., 2025]. Tokenization-based approaches discretize continuous signals into vocabularies and apply language-model objectives [Ansari et al., 2024, Talukder et al., 2025, Roger et al., 2025]. While some studies argue that forecasting gains stem mainly from architectural biases or tokenization effects [Tan et al., 2024, Zheng et al., 2025, Zhang et al., 2025], more recent large-scale evaluations show that language pretraining becomes particularly beneficial under distribution shift and cross-domain generalization [Qiu et al., 2026]. Our work complements this literature by studying not only *when* transfer occurs, but *why* it occurs.

Cross-modal representation geometry. The Platonic Representation Hypothesis [Huh et al., 2024] posits that sufficiently scaled models converge toward a shared statistical structure of reality, regardless of modality. Supporting this, embeddings exhibit universal cross-modal geometry [Jha et al., 2025], and frozen vision backbones can forecast time series [Chen et al., 2024, Roschmann et al., 2025].

These findings motivate our hypothesis that inherent temporal structures already exist within language-pretrained transformers.

Loss geometry and low-dimensional adaptation. Prior work shows that optimization geometry plays a central role in transfer learning with analysis of loss landscapes [Li et al., 2018, Ghorbani et al., 2019, Fort and Jastrzebski, 2019] and efficient finetuning of pretrained models within low-dimensional subspaces [Aghajanyan et al., 2021] such as LoRA [Hu et al., 2021]. We extend these ideas to cross-modal transfer, showing that language-pretrained transformers adapt to time series via coherent gradients and low-rank specialization rather than full representational rewriting.

3 Training Methodology

We repurpose Qwen3-0.6B [Yang et al., 2025] as a probabilistic time-series forecaster by casting forecasting as next-token prediction over discretized values.

3.1 Tokenization and Forecasting

We extract sliding windows of length $C + L$, with context length $C = 512$ and prediction horizon $L = 64$. Each series is normalized using statistics from the context window and discretized into $V = 1024$ uniform bins over $[-5, 5]$. Following Roger et al. [2025], bin indices are mapped directly to the first vocabulary positions of the pretrained model, while standard Qwen3 special tokens are preserved. The model is trained using a weighted quantile loss over quantile levels $\mathcal{Q} = \{0.1, \ldots, 0.9\}$, following Chronos Bolt [Ansari et al., 2024]:

$$\mathcal{L} = \frac{1}{T \cdot |\mathcal{Q}|} \sum_{t=1}^{T} \sum_{\tau \in \mathcal{Q}} \rho_\tau(y_t - \hat{q}_{t,\tau}), \qquad \rho_\tau(u) = u \cdot (\tau - \mathbf{1}[u < 0]), \tag{1}$$

where ρ_τ is the standard quantile regression loss. Output logits are transformed into a categorical distribution over ordered bins, defining a discrete CDF from which arbitrary quantiles are extracted by inversion. This produces coherent non-crossing probabilistic forecasts from a single forward pass, without requiring quantile-specific training or repeated inference.

3.2 Experimental Setup

Models are trained on sliding windows from the GiftEval corpus [Aksu et al., 2024] using AdamW with linear warmup and decay. Training uses an effective batch size of 128 on NVIDIA A100 GPUs with bf16 precision. Evaluation is performed on 1,000 held-out windows using CRPS, MASE, and MSE amongst others error metrics following GluonTS conventions (see Appendix A.2).

For a pretrained LLM (Qwen3-0.6B), we compare four adaptation strategies that differ only in trainable parameters:

- **Full finetuning:** all model parameters are updated.
- **IO Only:** transformer backbone is frozen, only the input embeddings and output head are trained.
- **LoRA Attn:** LoRA adapters are applied only to attention projections, the rest is frozen.
- **LoRA Attn+IO:** LoRA attention adaptation with trainable input embeddings and output head.

3.3 Results: Transfer from Language to Time Series

Figure 2 (as well as Figure 6 in section 5) shows that language-pretrained models consistently converge faster and reach lower forecasting error than randomly initialized counterparts across all adaptation regimes. The advantage is strongest during early optimization, particularly for CRPS and MASE, indicating that language pretraining provides a favorable sequential inductive bias. LoRA-based adaptation achieves performance comparable to full finetuning, supporting the hypothesis that forecasting transfer operates through low-dimensional specialization of pretrained representations rather than complete relearning. These observations hold on longer horizons, including h=64 which can be found in Appendix A, along with the complete set of metrics.

4 The Shared Manifold Hypothesis

We now present a geometric explanation for the observed phenomena. Due to the recursive structure and statistical patterns exhibited by language [Ou et al., 2025], we theorize that LLM representations already contain native temporal structures well suited for adaptation to time series data. We present several experiments demonstrating that LLMs do possess a manifold that is already compatible with time series forecasting, and that finetuning does not need to create new directions, it simply identifies and collapses time series representations onto the best pre-existing directions.

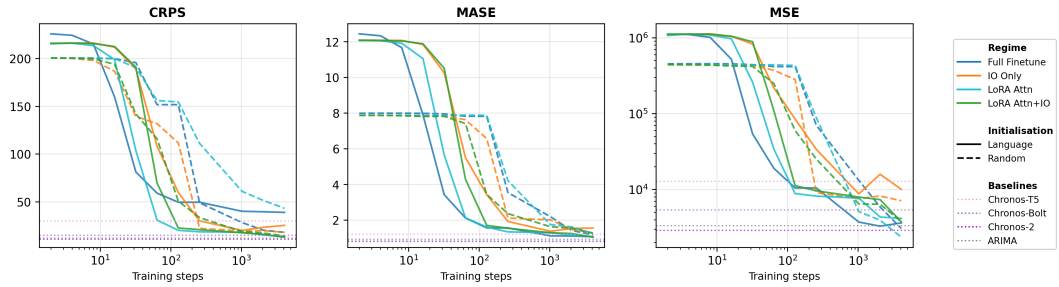
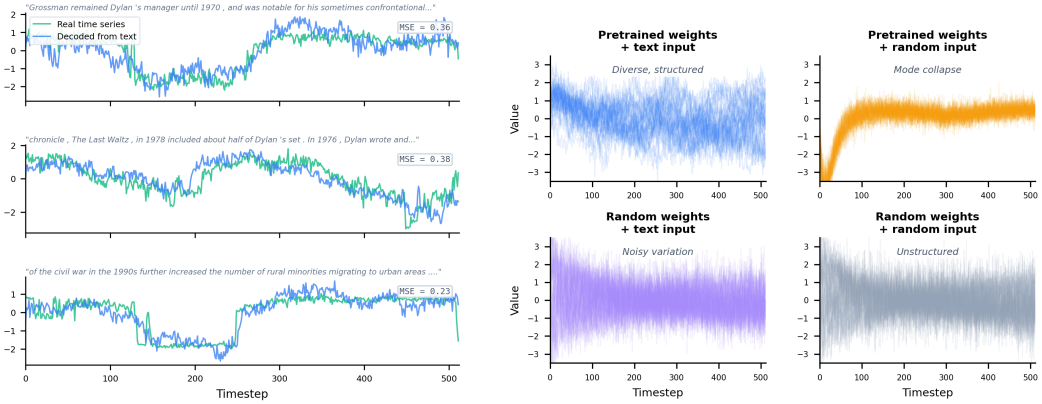


Figure 2: Training progression of h=1 forecasting metrics across training steps for four training regimes. Solid lines denote language-pretrained Qwen3-0.6B initialization; dashed lines denote random initialization. Horizontal dotted lines indicate baseline model performance. Language-pretrained models converge faster and to lower error across all regimes, with LoRA-based methods matching full finetuning performance.



(a) Decoded time series from text

(b) Ablation: what creates the structure?

Figure 3: **Pretrained hidden states contain time-series-compatible structure.** A single linear map $\hat{y}_t = \mathbf{w}^\top \mathbf{h}_t + b$ is trained on frozen LLM hidden states via EM-style nearest-neighbor matching to a bank of 10,000 real time series (no paired data). **(a)** Three decoded outputs (blue) overlaid with their nearest real time series (green). Input text shown above each plot (gray). MSE $= 0.23$–0.38; full distribution in Appendix. **(b)** Ablation crossing learned weights vs. random init with text vs. random input. *Top-left*: pretrained + text—diverse, structured outputs. *Top-right*: pretrained + random tokens—mode collapse to one shape (manifold exists but is not traversed). *Bottom-left*: random weights + text—noisy variation without structure. *Bottom-right*: random weights + random tokens—unstructured. Only pretrained weights *and* meaningful text together produce diverse temporal signals.

4.1 A Linear Probe Reveals Latent Temporal Structure

We test if the distribution of pretrained language model representations is already geometrically compatible with the distribution of real time series: can a simple linear map, trained without paired text–time-series supervision, project frozen LLM hidden states onto realistic time-series windows?

Method. We pass $N = 1{,}920$ WikiText-103 sequences of $T = 512$ tokens through the frozen Qwen3-0.6B model. For each token, we represent the LLM state by concatenating the hidden states from all 28 layers:

$$\mathbf{h}_{i,t} = \left[\mathbf{h}_{i,t}^{(0)};\ \mathbf{h}_{i,t}^{(1)};\ \ldots\ ;\ \mathbf{h}_{i,t}^{(27)}\right] \in \mathbb{R}^D, \quad D = 28 \times 1024 = 28{,}672. \tag{2}$$

A single linear layer is then applied independently at each timestep to produce a scalar prediction:

$$\hat{y}_{i,t} = \mathbf{w}^\top \mathbf{h}_{i,t} + b, \quad \mathbf{w} \in \mathbb{R}^D,\ b \in \mathbb{R}. \tag{3}$$

This produces a predicted sequence $\hat{\mathbf{y}}_i \in \mathbb{R}^T$ for each input text. The output is z-score normalized to zero mean and unit variance, matching the normalization of the target time series.

Training objective. We use an EM-style algorithm to match text and time-series targets. We match each prediction to its nearest z-scored window from a bank of $M = 10{,}000$ GiftEval time series [Aksu et al., 2024]. At each step, we sample $K = 128$ candidate windows per prediction, choose $j_i^* = \arg\min_j \frac{1}{T}\|\hat{\mathbf{y}}_i - \mathbf{Y}_j\|^2$, and optimize:

$$\mathcal{L} = \underbrace{\frac{1}{B}\sum_{i=1}^{B}\frac{1}{T}\|\hat{\mathbf{y}}_i - \mathbf{Y}_{j_i^*}\|^2}_{\text{MSE to nearest real TS}} \quad + \quad \lambda \underbrace{\frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} \cos\left(|\text{FFT}(\hat{\mathbf{y}}_i)|^2,\, |\text{FFT}(\hat{\mathbf{y}}_j)|^2\right)}_{\text{PSD diversity penalty}}. \tag{4}$$

The nearest-neighbor assignments are treated as fixed targets for the gradient step. The PSD diversity penalty discourages mode collapse; we set $\lambda = 0.5$ and train with Adam ($\eta = 10^{-3}$) for 100 epochs with batch size $B = 32$.

Results. We evaluate each decoded output by finding its nearest neighbor (by per-timestep MSE) among the 10,000 real time series in the bank. Across the 1,920 training predictions, the nearest neighbors span 686 distinct real time series—36% of predictions map to a unique real time series rather than collapsing to repeated patterns (Fig. 3a). When predictions are ranked by match quality and compared at equal diversity levels, the pretrained model consistently achieves the lowest MSE across all conditions (Appendix 5). On 1,000 held-out WikiText sequences never seen during training, the map achieves comparable quality (mean nearest-neighbor MSE 0.72 vs. 0.67 on training text), confirming that the temporal structure is a general property of the pretrained representation space.

What creates this structure? We disentangle the contributions of the learned weights, the input, and the architecture with a 2×2 ablation crossing model weights (pretrained vs. randomly initialized, same architecture) with input (WikiText vs. random tokens from the full vocabulary). The ablation is decisive: with random initialization, unique coverage drops to 2.8% (54 distinct matches); with random tokens through pretrained weights, it collapses to 0.2% (4 distinct matches). Neither the architecture alone nor the input alone is sufficient—only the combination of pretrained weights and meaningful text produces diverse, realistic temporal signals (Fig. 3b).

4.2 The Projected Dynamics Forecast Beyond the Context

The linear decoding result shows shape-level similarity between projected hidden states and real time series. We test whether this extends to *predictive* structure: given the first half of a time series, can a retrieved text projection forecast the second half?

Method. We take 500 real time series from GiftEval and observe only the first 256 of 512 timesteps. Among 10,000 WikiText projections, we retrieve the one with lowest MSE on the observed first half and use that projection's second half as the forecast. The baseline is last-value carry-forward, *i.e.* repeating the final observed value.

Result. Across all 500 queries, the retrieval-based forecast achieves MSE $= 1.91$ versus last-value's 2.27 (16% lower). The retrieval wins in only 37% of cases; however, when it wins, the advantage is large (median $4\times$ lower MSE), because the retrieved projection captures trends and level shifts that last-value misses entirely. Figure 4 shows the best-case example of this phenomena from the evaluation set.

4.3 Implications for Transfer

The linear probing and forecasting results suggest a possible explanation for why finetuning can succeed with limited adaptation. A pretrained autoregressive language model can be viewed as inducing a deterministic update over its prefix representations:

$$\mathbf{h}_{t+1} = F(\mathbf{h}_t, x_{t+1}), \tag{5}$$

where F denotes the transformer's forward computation and x_{t+1} is the next input token. Next-token prediction encourages these hidden states to preserve information relevant to the continuation of the sequence.

Our experiments identify a direction $\mathbf{w}$ whose projected trajectory $\{y_t = \mathbf{w}^\top \mathbf{h}_t\}_{t=1}^T$ resembles real time series (Sec. 4.1), and show that these projections contain useful continuation signal in a retrieval setting (Sec. 4.2). Taken together, this suggests that parts of the pretrained representation space are already compatible with time-series-like trajectories, even before any time-series supervision.

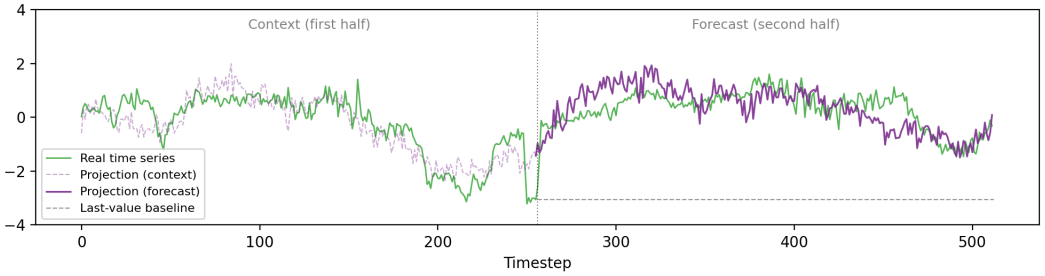


Figure 4: Forecasting example with largest gap compared to baseline from 500 evaluation queries. The first 256 timesteps are used to retrieve the closest WikiText projection; the projection's second half (purple) forecasts the continuation. The retrieved projection captures the level shift and subsequent trend (MSE $= 0.42$), while last-value carry-forward (gray dashed) is stranded at the final context value (MSE $= 11.7$).

Finetuning for time series may therefore be *less about learning temporal behavior from scratch and more about alignment: identifying directions in the existing representation space that are useful for temporal prediction, and adjusting the model to use them more reliably.*

4.4 Pretraining Creates Coherent Gradient Signal for the TS Objective

Mircea et al. [2025] attribute slow training progress in language models to *zero-sum learning*: per-example gradients that conflict, so reducing loss on one subset of examples raises it on another. Here we show that language pretraining resolves this opposition for the time-series objective: pretrained models exhibit coherent, mutually reinforcing gradients across diverse time-series inputs from the outset of finetuning, which bypasses the destructive interference phase that randomly initialized models must first overcome.

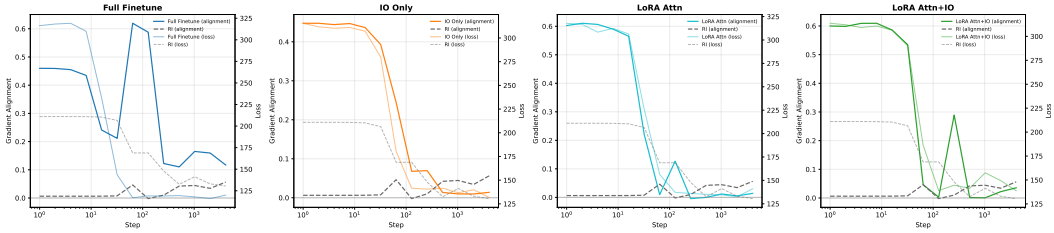


Figure 5: **Gradient alignment and evaluation loss across training for four adaptation regimes.** Each panel shows per-sample gradient alignment (left axis; mean pairwise cosine similarity of individual time-series gradients over 32 held-out examples) and CRPS evaluation loss (right axis) as a function of training steps (log scale). Solid lines denote language-pretrained (LangInit) models; dashed lines denote randomly initialized (RandInit) model. Across all regimes, language-pretrained models exhibit high gradient alignment from step one and loss decreases immediately, whereas randomly initialized models remain near-zero alignment until step 64–256, at which point loss begins to fall, indicating that coherent gradient signal is a prerequisite for effective optimization, and that language pretraining provides it from the outset.

Method. For a fixed evaluation batch of $N{=}32$ individual time-series sequences, we compute the per-sample gradient $\mathbf{g}_i = \nabla_\theta \mathcal{L}(x_i; \theta)$ over all model parameters at each checkpoint. We measure gradient alignment as the mean off-diagonal pairwise cosine similarity: $\frac{1}{N(N-1)} \sum_{i \neq j} \cos(\mathbf{g}_i, \mathbf{g}_j)$.

Interpretation. In Figure 5, we see that across all finetuning regimes, gradient alignment is quite high, especially when compared to gradient alignment in the randomly initialized model. The randomly initialized model takes a warmup phase before it is able to learn a representation such that gradients are aligned and there is a clear learning signal. By this point, the loss has already dropped for the language initialized model. The initial alignment indicates that there is a clear gradient signal to repurpose in pretrained representations and transfer to time series forecasting.

5 Finetuning Projects the Pretrained Manifold onto Time Series

5.1 The Weight Changes are Low-Rank

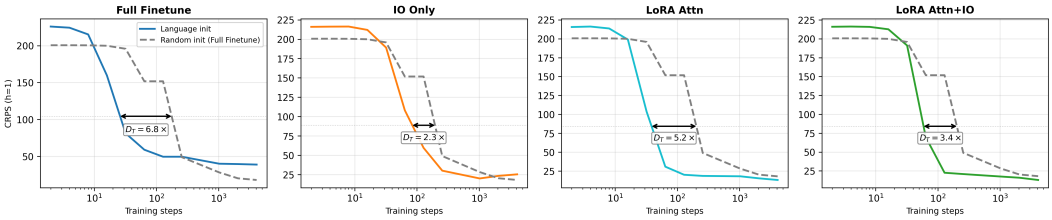


Figure 6: Effective data transfer (see Appendix A.4) from language pretraining, following the framework of Hernandez et al. [2021]. Each panel shows CRPS (h=1) over training steps for a single training regime, comparing language-pretrained initialization (solid) against the full finetuned random initialization (dashed). D_T indicates the multiplicative factor in training steps that the randomly initialized model requires to match the pretrained model at a given performance level.

To test whether adaptation from language to time series requires broad changes to pretrained parameters, we compare full finetuning against the parameter-efficient methods from Section 3.2. If low-rank updates recover most of the transfer benefit, then adaptation can be understood as a small perturbation around the pretrained solution rather than a full reconfiguration of the network.

As shown in Figure 6, we find that a low-rank adaptation preserves most of the optimization advantage of language pretraining: following the framework of Hernandez et al. [2021] (see Appendix A.4), we define the *effective data transfer* at a given loss level, and see that full finetuning achieves $D_T = 6.8$, attention-only LoRA achieves $D_T = 5.2$, and the constrained LoRA+IO setting still reaches $D_T = 3.4$, outperforming IO-only tuning ($D_T = 2.3$). Across all settings, pretrained initialization reaches a target CRPS in substantially fewer optimization steps than random initialization, indicating that only a low-dimensional modification of the pretrained weights is needed for time-series forecasting.

5.2 Effective Rank Dynamics Reveal Selective Compression

Method. To track how representational geometry evolves during training, we compute the effective rank of hidden-state covariance matrices at every saved checkpoint. For each checkpoint and each of the 28 transformer layers, we pass 10,000 time-series windows (length 512, drawn from the GiftEval validation split) through the model, collecting hidden states at every position except position 0 (excluded to avoid the attention-sink artifact). For FT and IO-only FT we additionally pass 10,000 WikiText-103 sequences to track the residual text representation quality.

At each layer we accumulate the centered covariance matrix $\Sigma = \frac{1}{N}\sum_i \mathbf{h}_i\mathbf{h}_i^\top - \bar{\mathbf{h}}\bar{\mathbf{h}}^\top$ in float64 over all N hidden-state vectors ($N \approx 130{,}000$ per batch), then compute its eigendecomposition. The effective rank is $\mathrm{erank}(\Sigma) = \exp\left(-\sum_i p_i \log p_i\right)$, where $p_i = \lambda_i / \sum_j \lambda_j$ are the normalized eigenvalues [Roy and Vetterli, 2007]. This equals 1 when all variance concentrates on a single direction and equals the ambient dimension d when variance is uniformly spread.

Results. Figure 7 shows that language-pretrained and randomly initialized models reach time-series representations through different trajectories. RandInit begins with near-isotropic hidden states and rapidly collapses to a low effective rank across nearly all layers, suggesting that it must first compress random representations into a usable low-dimensional structure before specialization can occur. LangInit, in contrast, starts from an already compressed representation and changes more selectively: the mean effective rank decreases only moderately on time-series inputs, while text-input representations collapse sharply, indicating catestrophic forgetting of language-specific directions.

The layerwise dynamics indicate that RandInit undergoes a uniform rank collapse across the network, whereas LangInit redistributes representational capacity: early and late layers lose rank, while middle layers increase effective rank during finetuning. This pattern is consistent with the view that finetuning does not build temporal structure from scratch, but repurposes pretrained middle-layer circuitry and contracts directions less relevant to forecasting.

The observed dynamics align with the representation phases described in Li et al. [2025]. The randomly initialized model undergoes the warmup phase marked by rapid representational collapse, followed by an entropy-seeking phase that expands useful directions. The language initialized model seemingly skips the warmup collapse, and move directly toward selective entropy redistribution and

compression, preserving variance along directions useful for forecasting while contracting directions less relevant to the new modality.

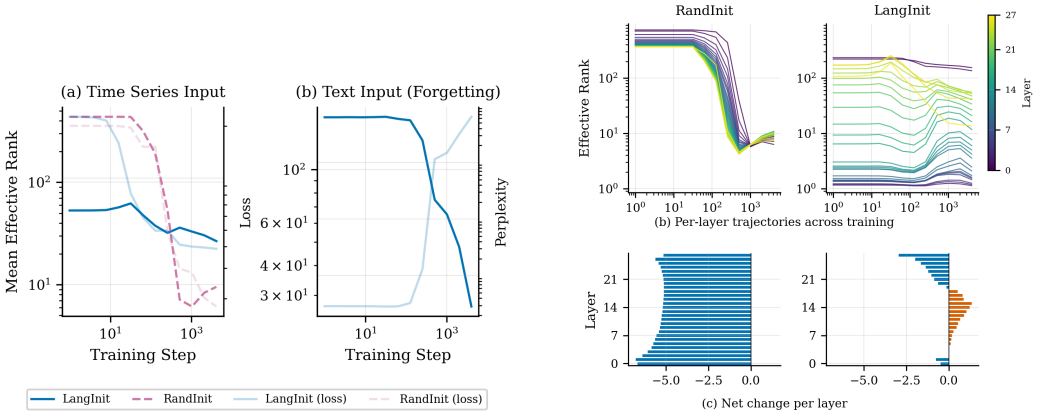


Figure 7: **Effective rank dynamics during training. (a)** Mean effective rank across all 28 layers. Left: on time-series input, RandInit starts near-isotropic ($\sim$440) and collapses to $\sim$10 within 500 steps; LangInit declines more gradually from the pretrained baseline ($\sim$50) to $\sim$27. Transparent lines show CRPS loss on a secondary axis, where decreasing loss correlates with rank. Right: on text input, LangInit's effective rank drops from $\sim$165 to $\sim$25 (85% reduction) while perplexity rises, confirming catastrophic forgetting. **(b)** Per-layer trajectories (each line = one of 28 layers, colored by depth). RandInit layers collapse as a tight bundle; LangInit middle layers ($\sim$8–17) *increase* their effective rank while early and late layers decrease. **(c)** $\log_2$ change (final/initial) per layer. RandInit is uniformly negative while middle layers of LangInit show positive change indicating finetuning redistributes capacity from the network's periphery to its core, reusing abstract language circuitry from middle layers.

5.3 Reuse vs. Reinvention: FT and RI Find Different Solutions

Method. To compare how pretrained and randomly initialized models represent simple temporal structure, we pass synthetic waveforms through our models. We visualize hidden-state geometry with 2D PCA, using the middle layer (13) whenever no layer is specified. For training dynamics, we additionally measure phase coherence in the full hidden-state space. Full preprocessing, PCA, and phase-coherence details, along with all-layer visualizations, are provided in Appendix C.

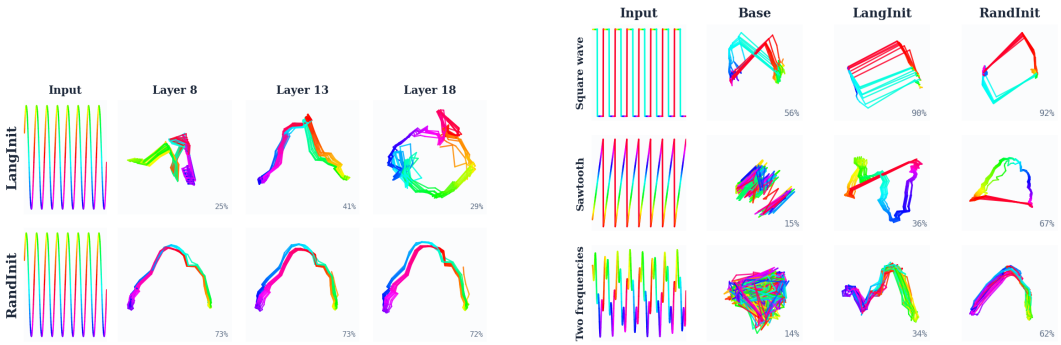


Figure 8: **Left: Sine wave representations across layers.** A sine wave (period 64) passed through LangInit (top) and RandInit (bottom) at layers 8, 13, and 18, shown as 2D PCA colored by input phase. RandInit produces clean arcs at every layer (72–73% PCA variance); LangInit produces complex, layer-varying loops (25–41%). **Right: Hidden-state trajectories for synthetic inputs at Layer 13 (PCA).** Each row is a different input waveform; columns show the pretrained LLM (Base), the finetuned LLM (LangInit), and the randomly-initialized model trained on time series (RandInit). Trajectories are 2D PCA projections of hidden states, colored by input phase. Percentages show variance captured by 2 PCs. Random and Base produce unstructured trajectories. RandInit discovers clean, low-dimensional representations (62–92% PCA variance) while LangInit creates geometrically complex but structured trajectories (34–98%) that vary by input type.

Result. Figure 8 (left) shows this contrast for a sine wave across three representative layers. RandInit converges to nearly uniform arcs, with 72–73% of variance explained by two principal components. LangInit instead produces layer-specific loops with only 25–41% PCA variance. Thus, both models encode phase, but RandInit represents it in a simple low-dimensional geometry, whereas LangInit distributes the same periodic structure across a richer inherited representation. Notice on the (right) plot, the square wave representation from BASE is already sufficiently similar to the learned representation.

Figure 1 extends the comparison to five synthetic inputs at Layer 13, and across all sine-wave layers in the Appendix (Figures 15–16). Here, RandInit learns geometries that are easy to recover with linear projections while LangInit retains more heterogeneous, nonlinear structure inherited from pretraining. This is inline with the observation that the eigenspectrum is more spread out in LangInit than RandInit. The key difference is therefore not whether the models learn temporal structure, but how they learn it. Quantitatively, the trained LangInit and RandInit layer-13 representations are highly aligned across synthetic inputs (CKA $= 0.855$–0.988), confirming that they recover nearly the same relational geometry up to rotation and scaling (this difference is clear in t-SNE Figure 12).

5.4 Interpreting the Transferred Features

Method. To test whether the reused directions correspond to interpretable features, we train one sparse crosscoder per layer with a shared encoder and separate pretrained/finetuned decoders. The shared encoder maps pretrained activations on decimal-string time series and finetuned activations on binned time series into a common 4,096-dimensional Top-K latent space; features are ranked by cross-domain firing between time series and WikiText.

Table 1: Examples of shared PT–FT crosscoder features. Shared features concentrate in layers 7–10; the examples below link concrete time-series motifs to coherent WikiText genres. (See Appendix D)

Layer	Feature	Time-series motif	WikiText motif
10	1712	magnitude jumps / plateaus	measurements and unit conversions
9	2469	volatile regime changes	tropical cyclone narratives
7	3888	isolated spikes	timestamped naval battles
8	2567	missing / NaN windows	`<unk>` and incomplete references

A few top activated features are shown in Table 1. These features suggest that the aligned LangInit subspace is populated by reusable temporal primitives that may correlate to text semantics, not just arbitrary low-rank directions. Appendix E provides preliminary causal evidence for the same picture: zero-ablation identifies a superadditive Layer 1 circuit (head L1H4 $\leftrightarrow$ MLP_{L1}) that is critical for periodic time-series prediction in IO-only FT and most affects pretrained text with sequential repetitive structure.

6 Limitations

All experiments use a single backbone (Qwen3-0.6B), one training corpus (GiftEval), and one tokenization scheme (1024 uniform bins); the geometric account has not been verified at larger scale or on alternative architectures. The linear probe establishes that the pretrained manifold is *compatible* with real time series, not that it contains the optimal forecasting subspace. The crosscoder analysis surfaces shared primitives in middle layers but does not exhaustively characterize the transferred subspace, and the causal circuit evidence (Appendix) is preliminary. We do not fully isolate whether gradient coherence (Sec. 4.4) is attributable to manifold geometry versus initialization statistics. Finally, "geometric rather than semantic" describes the dominant mechanism we observe.

7 Discussion and Conclusion

We investigated why language-pretrained transformers transfer to time-series forecasting and found that the mechanism is geometric rather than semantic. Pretrained representations already contain directions aligned with realistic temporal trajectories, enabling optimization to descend immediately through coherent gradients and favorable curvature. Finetuning then acts primarily as low-rank specialization over a pretrained sequential manifold rather than learning forecasting structure from scratch. The reused subspace captures structural primitives shared across modalities, including periodicity, regime changes, magnitude shifts, and missingness. In contrast, randomly initialized

models eventually recover competitive forecasting performance only after constructing a new and geometrically simpler representation space. Our results suggest that this mechanism extends beyond language models: any sufficiently rich autoregressive sequence model with latent temporal structure may support similar transfer. This perspective reconciles prior conflicting findings by explaining why transfer is strongest in low-data and distribution-shift regimes, where pretrained directions provide structure that limited forecasting data alone cannot establish. It also provides a principled explanation for the effectiveness of LoRA and related parameter-efficient methods, since if transfer is fundamentally direction selection, low-rank adaptation is the natural mechanism for exploiting it.

8 Acknowledgment

We thank 42.com for providing computational resources that supported this work. We also acknowledge the AMD University Program AI & HPC Cluster for additional compute resources used in this research.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. Gift-eval: A benchmark for general time series forecasting model evaluation, 2024. URL https://arxiv.org/abs/2410.10393.

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, et al. Gluonts: Probabilistic and neural time series modeling in python. *Journal of Machine Learning Research*, 21(116):1–6, 2020.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series, 2024. URL https://arxiv.org/abs/2403.07815.

Mohammad Javad Darvishi Bayazi, Hena Ghonia, Roland Riachi, Bruno Aristimunha, Arian Khorasani, Md Rifat Arefin, Amin Darabi, Guillaume Dumas, and Irina Rish. General-purpose brain foundation models for time-series neuroimaging data. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024.

Mouxiang Chen, Lefei Shen, Zhuo Li, Xiaoyun Joy Wang, Jianling Sun, and Chenghao Liu. Visionts: Visual masked autoencoders are free-lunch zero-shot time series forecasters, 2024. URL https://arxiv.org/abs/2408.17253.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability, 2023. URL https://arxiv.org/abs/2304.14997.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Stanislav Fort and Stanislaw Jastrzebski. Large scale structure of neural network loss landscapes, 2019. URL https://arxiv.org/abs/1906.04724.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density, 2019. URL https://arxiv.org/abs/1901.10159.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models, 2024. URL https://arxiv.org/abs/2402.03885.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters, 2023. URL https://arxiv.org/abs/2310.07820.

Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer, 2021. URL https://arxiv.org/abs/2102.01293.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis, 2024. URL https://arxiv.org/abs/2405.07987.

Rishi Jha, Collin Zhang, Vitaly Shmatikov, and John X. Morris. Harnessing the universal geometry of embeddings, 2025. URL https://arxiv.org/abs/2505.12540.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-llm: Time series forecasting by reprogramming large language models, 2023. URL https://arxiv.org/abs/2310.01728.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets, 2018. URL https://arxiv.org/abs/1712.09913.

Melody Zixuan Li, Kumar Krishna Agrawal, Arna Ghosh, Komal Kumar Teru, Adam Santoro, Guillaume Lajoie, and Blake A. Richards. Tracing the representation geometry of language models from pretraining to post-training, 2025. URL https://arxiv.org/abs/2509.23024.

Andrei Mircea, Supriyo Chakraborty, Nima Chitsazan, Irina Rish, and Ekaterina Lobacheva. Training dynamics underlying language model scaling laws: Loss deceleration and zero-sum learning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28154–28188, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1366. URL https://aclanthology.org/2025.acl-long.1366/.

Yulin Ou, Yu Wang, Yang Xu, and Hendrik Buschmeier. Identifying the periodicity of information in natural language. *arXiv preprint arXiv:2510.27241*, 2025.

Xin Qiu, Junlong Tong, Yirong Sun, Yunpu Ma, Wei Zhang, and Xiaoyu Shen. Rethinking the role of llms in time series forecasting, 2026. URL https://arxiv.org/abs/2602.14744.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. Lag-llama: Towards foundation models for probabilistic time series forecasting, 2023. URL https://arxiv.org/abs/2310.08278.

Roland Riachi, Kashif Rasul, Arjun Ashok, Prateek Humane, Alexis Roger, Andrew R. Williams, Yuriy Nevmyvaka, and Irina Rish. Random initialization can't catch up: The advantage of language model transfer for time series forecasting, 2025. URL https://arxiv.org/abs/2506.21570.

Alexis Roger, Gwen Legate, Kashif Rasul, Yuriy Nevmyvaka, and Irina Rish. Small vocabularies, big gains: Pretraining and tokenization in time series models, 2025. URL https://arxiv.org/abs/2511.11622.

Simon Roschmann, Quentin Bouniot, Vasilii Feofanov, Ievgen Redko, and Zeynep Akata. Time series representations for classification lie hidden in pretrained vision transformers. *arXiv preprint arXiv:2506.08641*, 2025.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European Signal Processing Conference*, pages 606–610, Sep. 2007.

Sabera Talukder, Yisong Yue, and Georgia Gkioxari. Totem: Tokenized time series embeddings for general time series analysis, 2025. URL https://arxiv.org/abs/2402.16412.

Mingtian Tan, Mike A. Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are language models actually useful for time series forecasting? In *Advances in Neural Information Processing Systems*, volume 37, pages 60162–60191, 2024.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. Easyedit: An easy-to-use knowledge editing framework for large language models, 2024. URL https://arxiv.org/abs/2308.07269.

Andrew Robert Williams, Arjun Ashok, Étienne Marcotte, Valentina Zantedeschi, Jithendaraa Subramanian, Roland Riachi, James Requeima, Alexandre Lacoste, Irina Rish, Nicolas Chapados, and Alexandre Drouin. Context is key: A benchmark for forecasting with essential textual information, 2024. URL https://arxiv.org/abs/2410.18959.

Malcolm L. Wolff, Shenghao Yang, Kari Torkkola, and Michael W. Mahoney. Using pre-trained llms for multivariate time series forecasting, 2025. URL https://arxiv.org/abs/2501.06386.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers, 2024. URL https://arxiv.org/abs/2402.02592.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

X. Zhang, S. Feng, and X. Li. From text to time? rethinking the effectiveness of the large language model for time series forecasting. *arXiv preprint arXiv:2504.08818*, 2025.

L. N. Zheng, C. Dong, W. E. Zhang, L. Yue, M. Xu, O. Maennel, and W. Chen. Understanding why large language models can be ineffective in time series analysis: The impact of modality alignment. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 4026–4037. Association for Computing Machinery, 2025.

Tian Zhou, PeiSong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm, 2023. URL https://arxiv.org/abs/2302.11939.

A Experimental Setup

A.1 Hyperparameters and Reproducibility

Table 2: Complete hyperparameter configuration for all experiments. All runs share the same configuration except where noted.

Category	Parameter	Value
Model	Base model	Qwen3-0.6B
	Architecture	Qwen3ForCausalLM and Qwen defaults
Tokenizer	Scaling	Z-score (mean/std from context)
	Binning	Uniform
	Vocab size (V)	1024
	Bin range	$[-5, 5]$
	Use EOS token	No
Training	Optimizer	AdamW
	Learning rate	1×10^{-4}, 3×10^{-4}, 1×10^{-3}, or 3×10^{-3}
	LR schedule	Linear warmup + cosine decay
	Warmup ratio	3%
	LR end factor	1×10^{-4}
	Precision	bf16 (mixed)
Batching	Effective batch size	128
	Epoch length	5,000 steps
Data	Dataset	GiftEval Pretrain (152 sub-datasets)
	Context length	512
	Target length	64
	Window stride	600
Loss	Loss function	Quantile (pinball) loss
	Quantiles	$\{0.1, 0.2, \ldots, 0.9\}$
	Softmax temperature	10^{-2}
Evaluation	Eval samples	1,000 extracted from the dataset (see script)
	Eval horizons	$h \in \{1, 64\}$
	Eval strategy	Autoregressive
LoRA	Rank (r)	4, 8
	Alpha (α)	16, 32
	Dropout	0.05
	Bias	None
	Targets (Attn)	`q_proj, k_proj, v_proj, o_proj`
Reproducibility	Random seed	420
	Seeded modules	Python, NumPy, PyTorch (CPU + CUDA)
	Compute Usage	About 12 hours on 8 Nvidia A100 per model

A.2 Evaluation Metrics

This section details the evaluation metrics implemented for assessing the forecasting performance of the models. For all fixed-point metrics, the 0.5 quantile (median) is extracted from the probabilistic forecasts and used as the deterministic prediction.

Aggregation Strategy and Implementation: In our evaluation methodology, we follow closely the GluonTS library [Alexandrov et al., 2020], particularly, the aggregation approach. All metrics described below are computed *per-sequence first*, and then averaged across all sequences in the dataset. Specifically, the error or score is aggregated over the forecast horizon H for each individual time series, and the final reported metric is the unweighted mean of these per-sequence scores. This macro-averaging ensures that every sequence contributes equally to the final evaluation, regardless of its absolute magnitude or scale.

To ensure robust evaluation, any time steps containing NaN values in the ground truth are dynamically excluded from computation by masking. Additionally, a small utility constant (ϵ = 1e-8) is added to denominators in percentage and scale-normalized metrics to prevent division by zero.

A.2.1 Point Forecast Metrics

Basic Error Metrics Mean Squared Error (MSE) quantifies prediction accuracy by heavily penalizing larger errors, making it sensitive to outliers. Root Mean Squared Error (RMSE) provides the error in the original data units while retaining MSE's sensitivity.

$$\text{MSE} = \frac{1}{H}\sum_{t=1}^{H}(y_t - \hat{y}_t)^2$$

$$\text{RMSE} = \sqrt{\text{MSE}}$$

where H is the forecast horizon, y_t is the true value, and $\hat{y}_t$ is the predicted median.

Mean Absolute Error (MAE) applies a linear penalty to measure the average absolute difference:

$$\text{MAE} = \frac{1}{H}\sum_{t=1}^{H}|y_t - \hat{y}_t|$$

Normalized Metrics Mean Absolute Scaled Error (MASE) measures forecast accuracy relative to a naive baseline, allowing for cross-series comparison across data with vastly different scales.

$$\text{MASE} = \frac{\text{MAE}}{\frac{1}{H-m}\sum_{t=m+1}^{H}|y_t - y_{t-m}|}$$

The seasonality parameter m is selected based on the expected periodic patterns of the timeframe ($m = 1$ for a general naive baseline, $m = 60$ for 1-minute data, and $m = 24$ for 1-hour data).

To further achieve scale-invariance, RMSE and absolute deviations are normalized by the scale of the data (mean or sum of absolute values) to compute the Normalized Root Mean Squared Error (NRMSE) and Normalized Deviation (ND):

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{H}\sum_{t=1}^{H}(y_t - \hat{y}_t)^2}}{\frac{1}{H}\sum_{t=1}^{H}|y_t|}$$

$$\text{ND} = \frac{\sum_{t=1}^{H}|y_t - \hat{y}_t|}{\sum_{t=1}^{H}|y_t|}$$

Percentage-Based Metrics Mean Absolute Percentage Error (MAPE) provides a scale-independent metric as a percentage. To address MAPE's asymmetry problem and instability near zero, the Symmetric Mean Absolute Percentage Error (sMAPE) equally penalizes over-predictions and under-predictions:

$$\text{MAPE} = \frac{100}{H}\sum_{t=1}^{H}\frac{|y_t - \hat{y}_t|}{|y_t| + \epsilon}$$

$$\text{sMAPE} = \frac{200}{H}\sum_{t=1}^{H}\frac{|y_t - \hat{y}_t|}{|y_t| + |\hat{y}_t| + \epsilon}$$

A.2.2 Probabilistic Metrics

Continuous Ranked Probability Score (CRPS) The approximated CRPS measures the accuracy of probabilistic forecasts by evaluating the entire predictive distribution using quantile loss (QL).

$$\text{CRPS}_{\text{approx}} = 2\sum_{q} w_q \cdot \text{QL}_q(y_{\text{true}}, y_{\text{pred}})$$

where $\text{QL}_q(y, \hat{y}) = (q - \mathbf{1}_{y \le \hat{y}}) \cdot (y - \hat{y})$ and w_q represents the discrete weight calculated based on the distance between quantile levels.

Weighted Mean Absolute Percentage Quantile Loss (wMAPE) To compare probabilistic performance across heterogeneous time series, we implement a scale-invariant version of quantile loss weighted by the sum of absolute true values:

$$\text{wMAPE} = \frac{1}{Q}\sum_{q=1}^{Q}\frac{\sum_{t=1}^{H}\text{QL}_q(y_t, \hat{y}_{q,t})}{\sum_{t=1}^{H}|y_t|}$$

A.2.3 Directional and Correlation Metrics

Directional Accuracy (DA) DA evaluates how often the model correctly predicts the direction of movement relative to a historical anchor point y_0 (the last observed value).

$$\text{DA} = \frac{1}{H} \sum_{t=1}^{H} \mathbf{1}_{\text{sign}_\tau(\hat{y}_t - y_0) = \text{sign}_\tau(y_t - y_0)}$$

Correlation Metrics To evaluate the model's ability to capture trend dynamics independently of absolute magnitude errors, we utilize Pearson correlation coefficients. Pearson evaluates the linear relationship between predictions and ground truth.

A.3 Experimental Results

Here we provide the complete set of evaluation results across all training regimes and metrics. Figure 9 shows the training progression of all eleven h=1 metrics over 4096 training steps, revealing a consistent pattern across metrics: language-pretrained models begin converging within the first 10–100 steps, while randomly initialized models require half an order of magnitude more training to reach comparable performance. By step 4096, both initialization strategies converge to similar error levels across all regimes, confirming that the pretrained weights accelerate optimization rather than alter the final solution. Tables 3 and 4 report the full numerical results at step 128 for single-step (h=1) and multi-step (h=64) forecasting respectively, alongside four established baselines and a zero-shot Qwen3 text baseline. At this early checkpoint, the transfer advantage of language pretraining is clearly visible: pretrained LoRA Attn achieves a CRPS of 20.10 compared to 154.5 for its randomly initialized counterpart, while all pretrained regimes outperform the random initializations that have not yet begun to converge. The Qwen3 text baseline performs orders of magnitude worse than all trained models, confirming that the language model's pretrained representations require domain-specific finetuning to be useful for time series forecasting.

A.4 Effective Transfer

Let $\mathcal{L}_R(d)$ and $\mathcal{L}_P(d)$ denote the validation losses achieved after training for d time series tokens using random and pre-trained initializations, respectively. For a given validation loss ℓ, $\mathcal{L}_\cdot^{-1}(\ell)$ therefore denotes the amount of data required to train a model to reach the loss level ℓ begining from a given initialization. We define transfer as "effective" or "positive" when starting the timeseries model training with the language model's weights allows us to achieve the same validation loss with less data than a model initialised randomly. This data quantity difference is what we refer to as the amount of data we "saved". Concretely, the *effective data transferred* $D_T(\ell)$ for a target loss ℓ is defined as the difference of these data amounts:

$$D_T(\ell) := \mathcal{L}_R^{-1}(\ell) - L_P^{-1}(\ell).$$

A positive $D_T(\ell)$ indicates that initializing the model from pre-trained language weights requires fewer training examples to reach the loss ℓ compared to random initialization, signifying an effective transfer from the upstream task to time series forecasting.

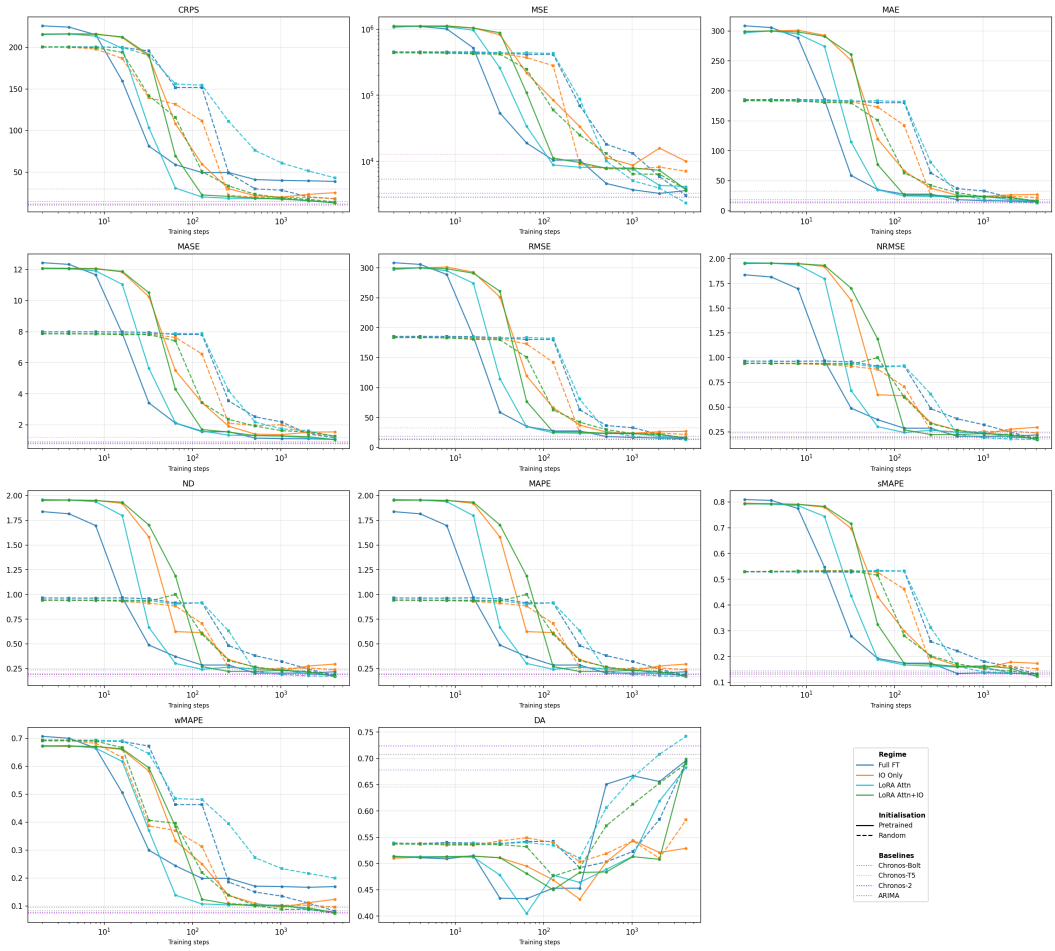


Figure 9: Training progression of all $h = 1$ forecasting metrics across training steps for four training regimes. Solid lines denote language-pretrained initialization (Qwen3-0.6B); dashed lines denote random initialization. Horizontal dotted lines indicate baseline performance (Chronos-T5, Chronos-Bolt, Chronos-2, ARIMA). Language-initialized models consistently converge earlier than their randomly initialized counterparts across all metrics, with the gap most pronounced in the first 100 steps. Both initializations converge in performance by 4096 steps, reaching levels competitive with established forecasting baselines.

Table 3: Single-step (h=1) forecasting performance on the held-out evaluation set. All NanoTS variants use Qwen3-0.6B at training step 128. Best value per metric within each section is **bolded**. ↑: higher is better; ↓: lower is better.

	Model	CRPS↓	MSE↓	MAE↓	MASE↓	RMSE↓	NRMSE↓	ND↓	MAPE↓	sMAPE↓	wMAPE↓	DA↑
Pretrained	Full Finetune	49.51	10461	27.60	**1.561**	27.60	0.286	0.286	0.286	0.175	0.199	0.453
	IO Only	59.56	84846	66.53	3.430	66.53	0.614	0.614	0.614	0.300	0.250	0.469
	LoRA Attn	**20.10**	**8818**	**24.68**	1.607	**24.68**	**0.243**	**0.243**	**0.243**	**0.168**	**0.107**	**0.478**
	LoRA Attn+IO	22.54	11220	26.11	1.702	26.11	0.270	0.270	0.270	0.174	0.124	0.450
Random Init	Full Finetune	151.8	415339	180.4	7.819	180.4	0.914	0.914	0.914	0.532	0.463	**0.542**
	IO Only	111.6	280932	142.3	6.565	142.3	0.706	0.706	0.706	0.462	0.313	0.540
	LoRA Attn	154.5	430387	182.2	7.872	182.2	0.917	0.917	0.917	0.532	0.481	0.535
	LoRA Attn+IO	**50.68**	**59864**	**62.90**	**3.438**	**62.90**	**0.599**	**0.599**	**0.599**	**0.281**	**0.220**	0.476
Baselines	Chronos-Bolt	14.94	5409	18.91	0.934	18.91	0.240	0.240	0.240	0.146	0.096	0.678
	Chronos-T5	29.88	12764	32.99	1.212	32.99	**0.173**	**0.173**	**0.173**	**0.124**	**0.073**	0.646
	Chronos-2	**10.89**	**2886**	**13.38**	0.830	**13.38**	0.192	0.192	0.192	0.133	0.077	**0.724**
	ARIMA	11.94	3327	14.86	**0.805**	14.86	0.201	0.201	0.201	0.138	0.083	0.708
	Qwen3 Text	213.5	2.6M	213.5	376667	213.5	39508	39508	39508	0.563	19754	0.503

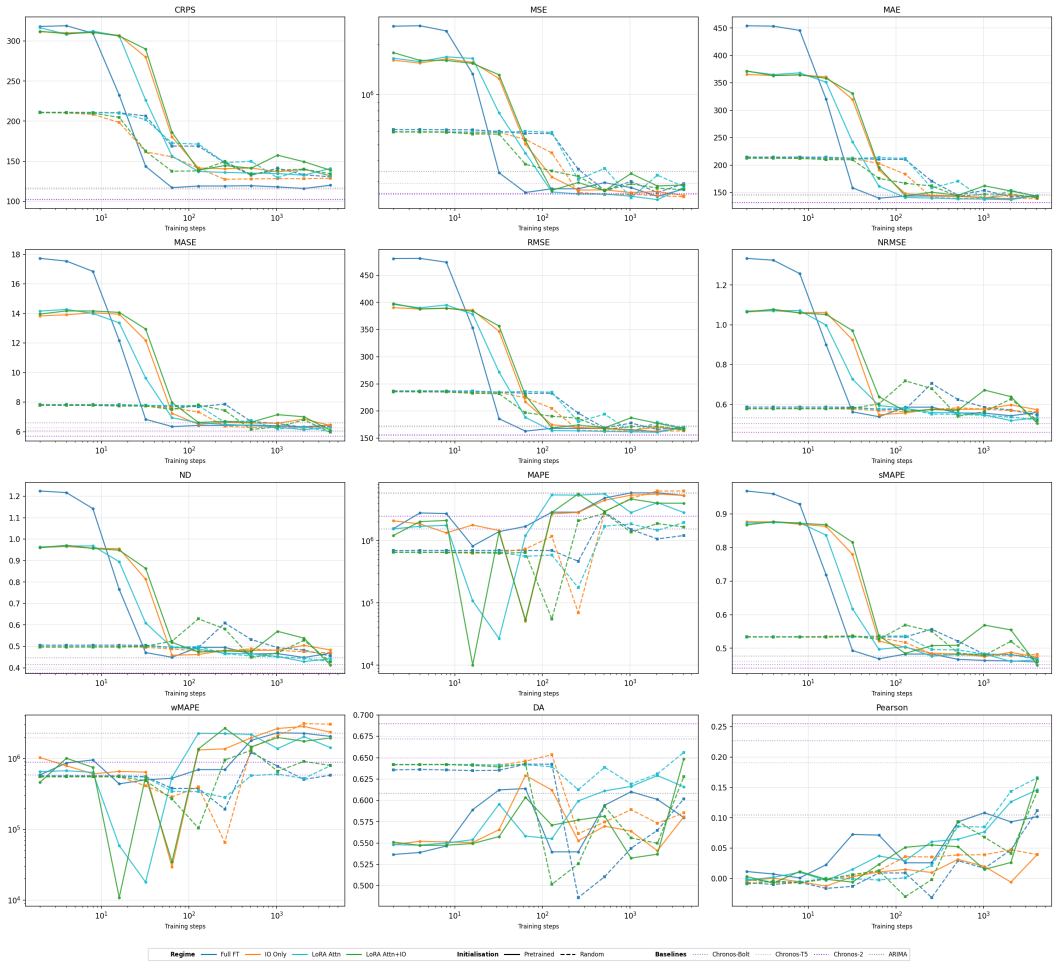


Figure 10: Training progression of all $h = 64$ forecasting metrics across training steps for four training regimes. Solid lines denote language-pretrained initialization (Qwen3-0.6B); dashed lines denote random initialization. Horizontal dotted lines indicate baseline performance (Chronos-T5, Chronos-Bolt, Chronos-2, ARIMA). Language-initialized models consistently converge earlier than their randomly initialized counterparts across all metrics, with the gap most pronounced in the first 100 steps. Both initializations converge in performance by 4096 steps, reaching levels competitive with established forecasting baselines.

Table 4: Multi-step (h=64) forecasting performance on the held-out evaluation set. All NanoTS variants use Qwen3-0.6B at training step 128. Best value per metric within each section is **bolded**. ↑: higher is better; ↓: lower is better.

	Model	CRPS↓	MSE↓	MAE↓	MASE↓	RMSE↓	NRMSE↓	ND↓	MAPE↓	sMAPE↓	wMAPE↓	DA↑	Pearson↑
Pretrained	Full Finetune	**119.0**	271341	143.7	**6.424**	168.2	0.586	0.495	**2.85M**	**0.483**	**696208**	0.540	0.026
	IO Only	141.5	319226	147.9	6.502	174.9	**0.557**	**0.462**	2.68M	0.503	1.32M	**0.612**	0.015
	LoRA Attn	137.2	**259652**	**140.8**	6.602	**164.0**	0.581	0.487	5.39M	0.505	2.27M	0.555	0.029
	LoRA Attn+IO	138.8	265546	144.0	6.628	169.0	0.566	0.473	2.82M	0.485	1.36M	0.571	**0.052**
Random Init	Full Finetune	168.9	583696	210.5	7.700	232.7	0.575	0.494	693452	0.534	377358	0.642	0.009
	IO Only	141.3	445540	183.5	**7.341**	205.0	**0.563**	**0.478**	1.17M	**0.517**	394850	**0.653**	**0.036**
	LoRA Attn	171.4	592435	212.3	7.770	234.7	0.579	0.499	583786	0.536	339884	0.639	0.001
	LoRA Attn+IO	**137.9**	**346961**	**166.8**	7.812	**190.5**	0.719	0.628	**55012**	0.569	**104026**	0.502	−0.030
Baselines	Chronos-Bolt	**101.9**	**251450**	**131.7**	6.068	156.8	0.504	0.416	**1.55M**	0.452	**587680**	0.672	0.227
	Chronos-T5	115.2	292023	147.4	6.291	173.7	0.484	0.393	5.84M	0.462	1.98M	0.650	0.191
	Chronos-2	102.4	255127	132.0	**5.709**	**156.2**	**0.460**	**0.376**	2.48M	**0.441**	888025	**0.690**	**0.255**
	ARIMA	116.6	345041	145.4	6.609	171.8	0.534	0.446	5.80M	0.478	2.27M	0.608	0.105
	Qwen3 Text	489.1	213.1M	489.1	9911.0	720.8	5672.0	824.6	3.23M	0.568	1.62M	0.589	−0.048

B Linear Mapping Experiment

Table 5: Fair top-K comparison across ablation conditions. For each condition, we take the best-K unique matches (deduplicated) and report mean nearest-neighbor MSE. This controls for diversity: conditions with fewer unique matches are only compared at K values they can support. "—" indicates fewer than K unique matches available. Text + PT achieves the lowest MSE at every K.

K	text + PT	text + RandInit	rand + PT	rand + RandInit
4	**0.254**	0.383	0.330	0.412
54	**0.350**	0.721	—	0.755
99	**0.383**	0.825	—	0.862
200	**0.441**	0.971	—	1.004
439	**0.535**	—	—	—
686	**0.639**	—	—	—

C Reuse vs. Reinvention: Additional Figures

Method details. We generate five length-512 synthetic inputs: a sine wave (period 64), a square wave (period 64), a sawtooth wave (period 64), a two-frequency signal $\sin(2\pi t/64) + 0.5\sin(2\pi t/17)$, and a linear trend with oscillation $t/T + 0.3\sin(2\pi t/80)$. Each input is independently z-score normalized, clipped to $[-5, 5]$, and uniformly binned into 1024 tokens using the same discretization as in the forecasting experiments. After passing the tokenized sequence through each model, we extract hidden states and fit PCA independently for each model–input pair; therefore, the plots compare within-trajectory structure and variance explained rather than absolute PCA axes across models. We exclude the first five positions to reduce attention-sink effects and color points by position modulo the dominant period of the input.

For the phase-coherence analysis, we compute Euclidean distances d_{ij} between hidden states at positions i and j in the full 1024-dimensional hidden-state space, again excluding the first five positions. For an input with period P, positions are treated as same-phase when $i \bmod P = j \bmod P$. We define

$$\text{coherence} = \frac{\text{mean}(d_{ij} \mid i \bmod P = j \bmod P)}{\text{mean}(d_{ij} \mid \text{all pairs})}. \tag{6}$$

If the model encodes periodic structure, same-phase hidden states should be close relative to arbitrary pairs, so lower values indicate stronger phase structure. In the training-dynamics figure we plot $1 - \text{coherence}$ so that higher values correspond to stronger periodic encoding.

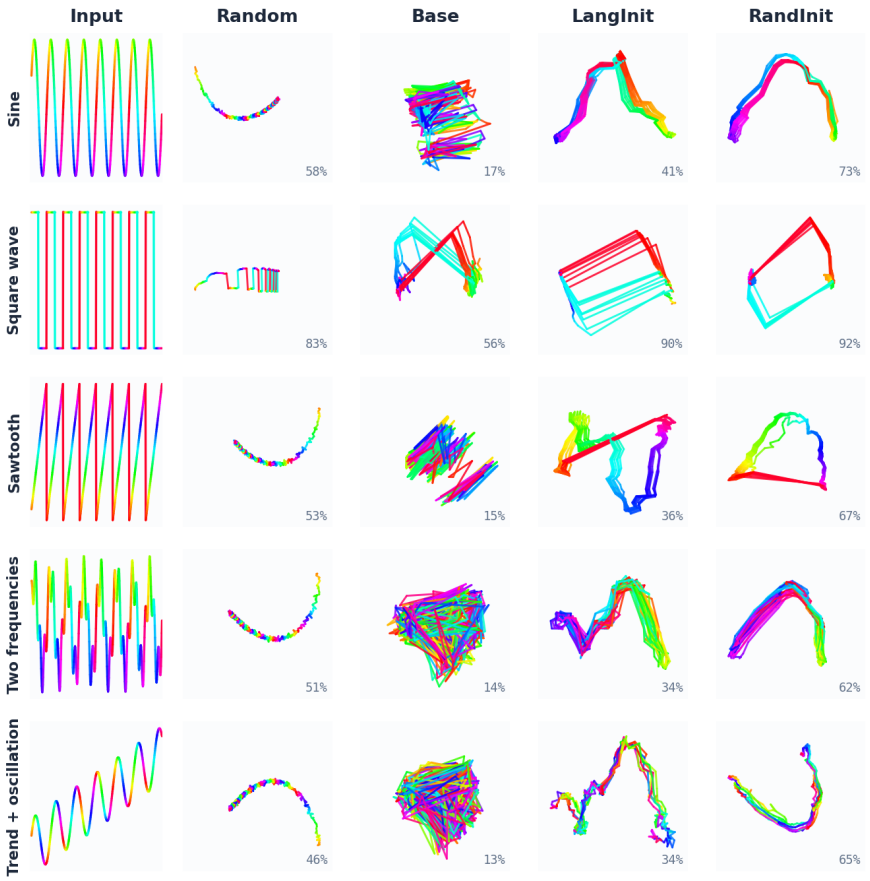


Figure 11: **Hidden-state trajectories for synthetic inputs at Layer 13 (2 PCA).** Trajectories are 2D PCA projections of hidden states, colored by input phase. Percentages show variance captured by 2 PCs. Random and Base produce unstructured trajectories. RandInit discovers clean, low-dimensional representations (62–92% PCA variance) while LangInit creates geometrically complex but structured trajectories (34–98%) that vary by input type.

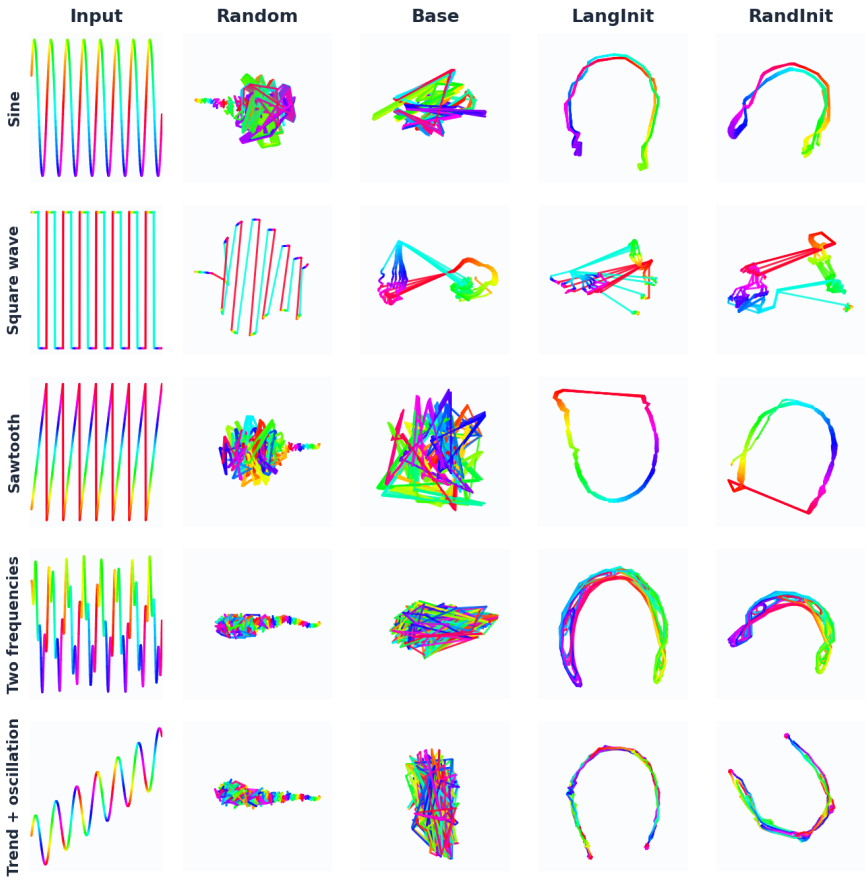


Figure 12: **Hidden-state trajectories for synthetic inputs at Layer 13 (t-SNE).** Same setup as Figure 11 but using t-SNE (perplexity 30) instead of PCA. Despite the different global geometries revealed by PCA, t-SNE shows that LangInit and RandInit develop similar local neighborhood structure: periodic inputs form smooth, phase-ordered curves in both models, confirming that both arrive at functionally equivalent representations through different geometric paths.

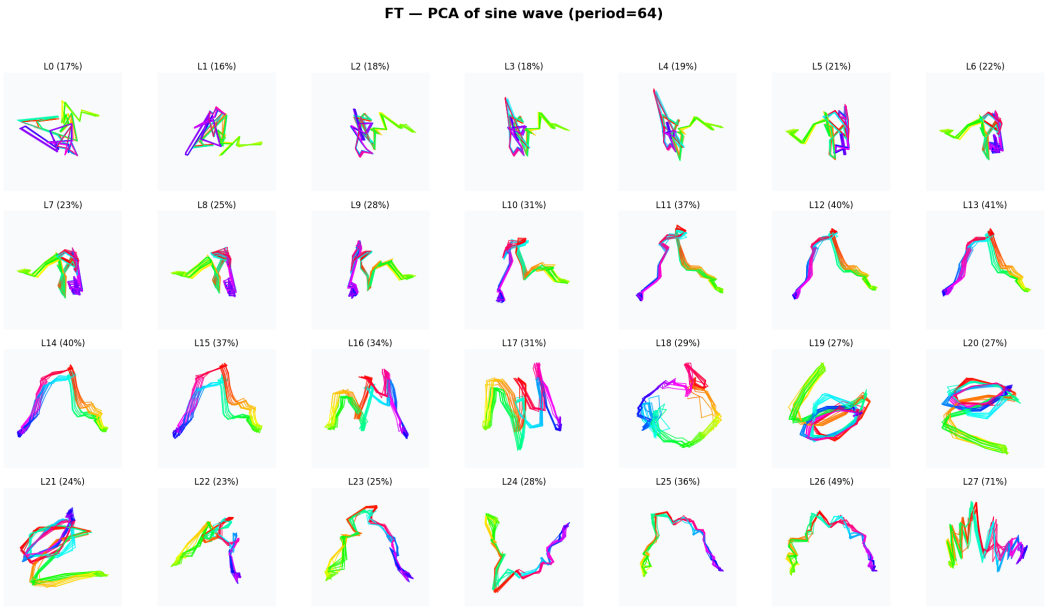


Figure 13: **LangInit: sine wave PCA trajectories across all 28 layers.** Each subplot shows the 2D PCA projection of hidden states from a sine wave (period 64) at one transformer layer, colored by input phase. Percentages show variance explained by 2 PCs. LangInit exhibits layer-specific geometry with varying complexity and moderate PCA variance (17–49%), reflecting the rich, heterogeneous representations inherited from language pretraining.

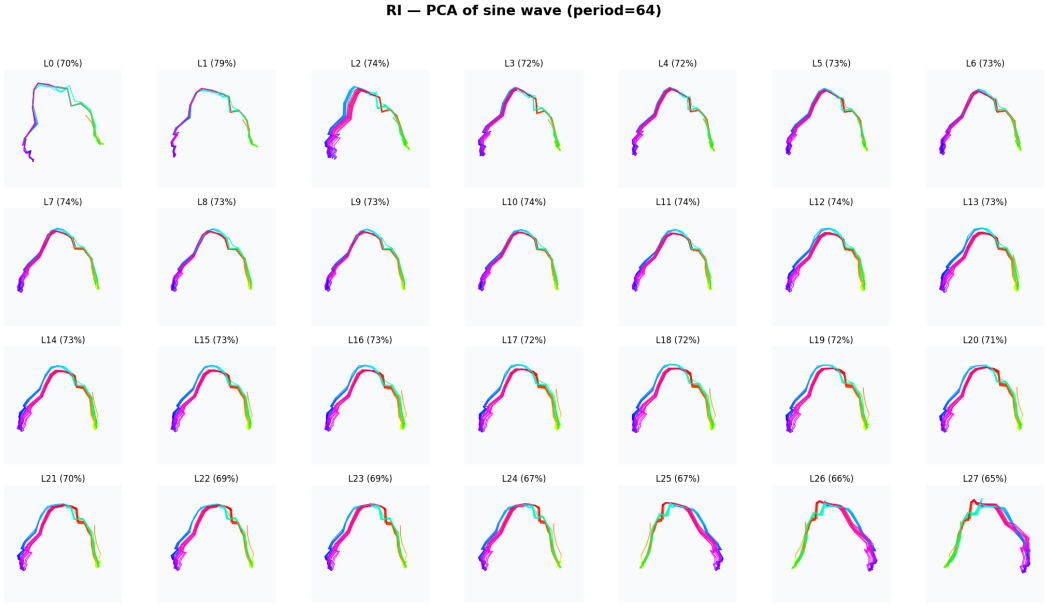


Figure 14: **RandInit: sine wave PCA trajectories across all 28 layers.** Same setup as Figure 13. RandInit produces nearly identical clean arcs at every layer with uniformly high PCA variance (67–79%), confirming that its learned representation is geometrically homogeneous across the network.

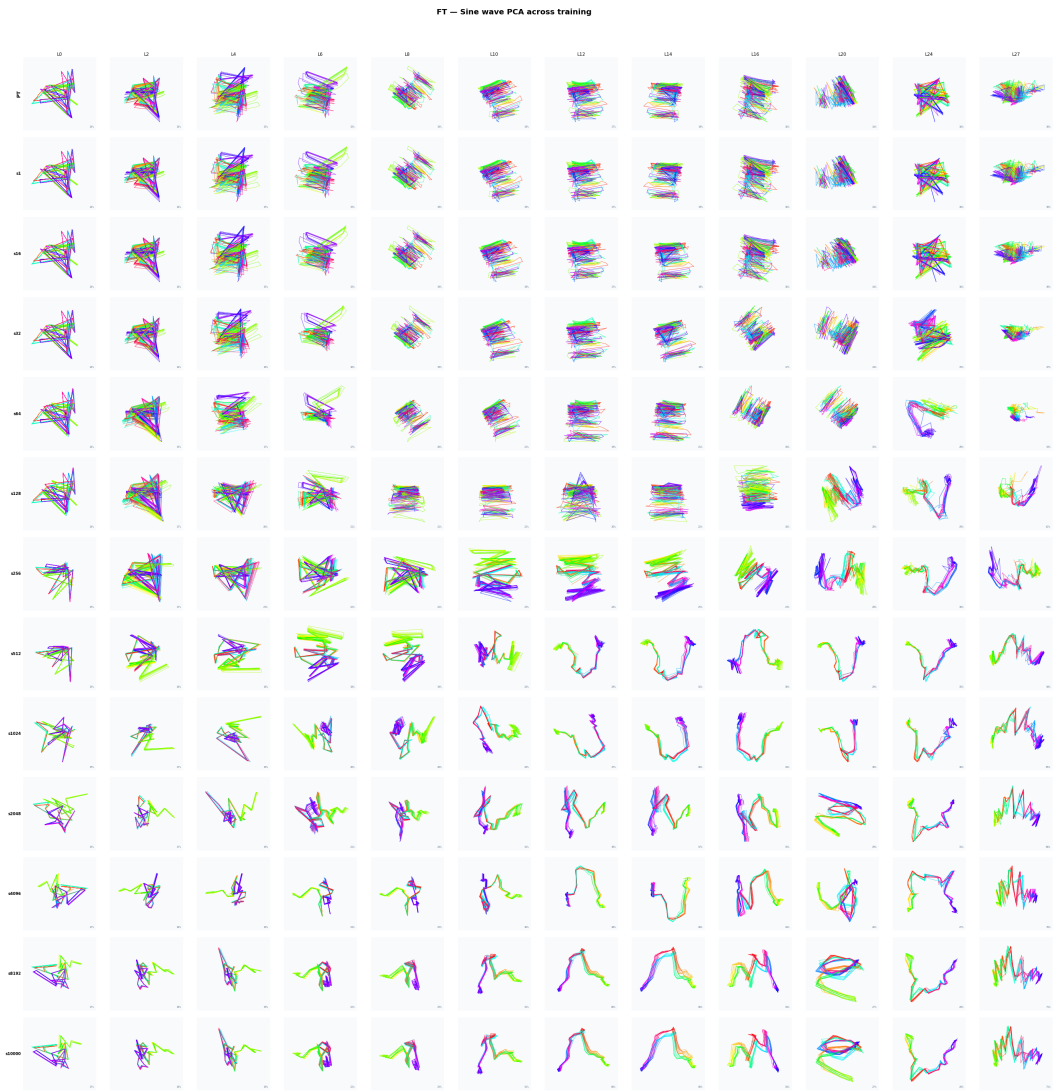


Figure 15: **LangInit: sine wave representations across training checkpoints.** Rows are training steps (top: PT baseline; bottom: final checkpoint at step 10,000), columns are selected layers. LangInit begins from the pretrained model's complex trajectories and gradually reshapes them into structured loops, with the transition occurring around steps 256–1024.

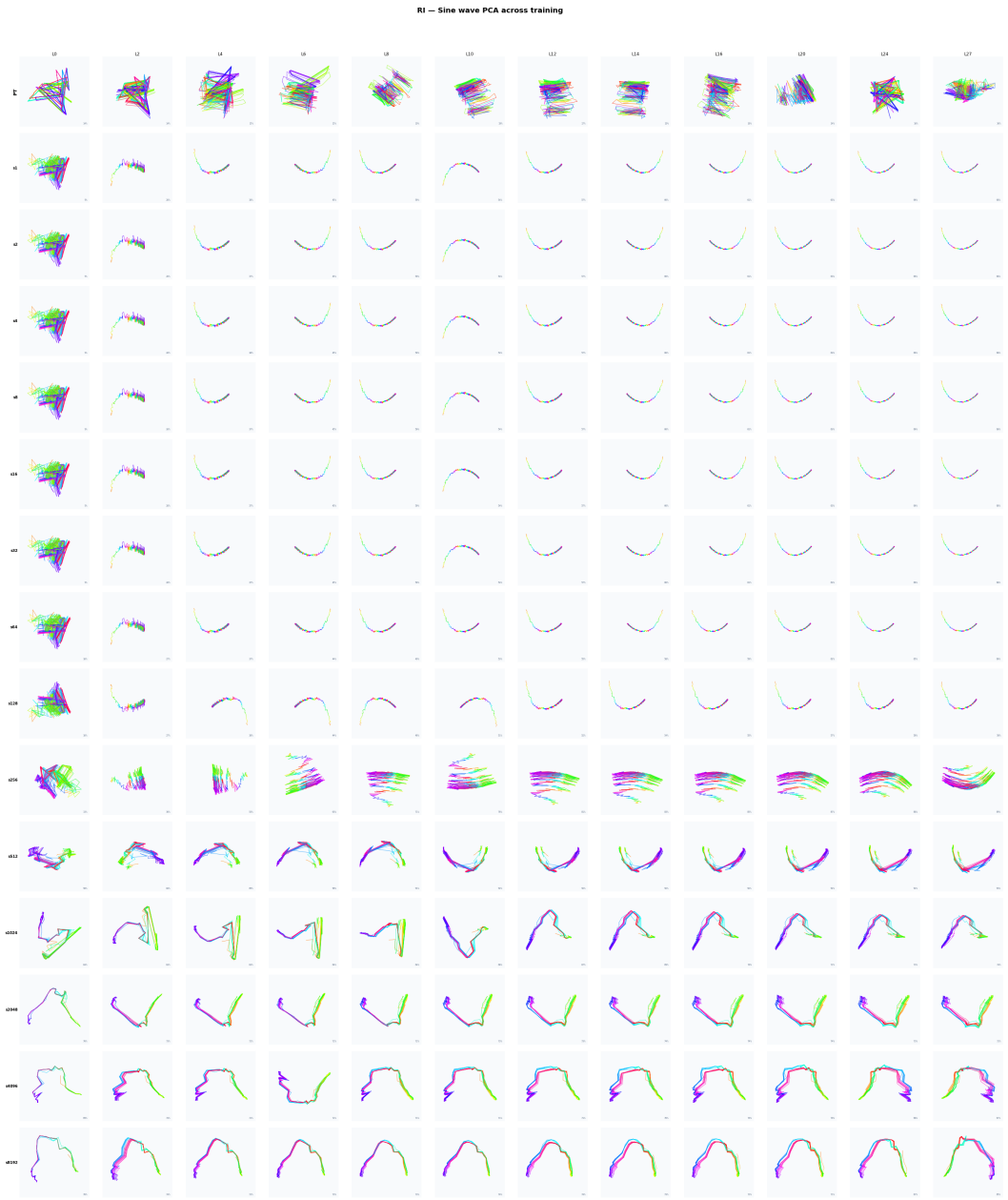


Figure 16: **RandInit: sine wave representations across training checkpoints.** Same setup as Figure 15. RandInit starts from near-empty representations (random weights) and converges to uniform clean arcs by step 2048, with all layers collapsing to the same geometry simultaneously.

D Cross-Domain Feature Analysis via Crosscoders

The circuit-level analysis in Appendix E shows that specific components are shared between time-series periodicity prediction and repetitive language modeling. Here we complement that *component-level* analysis with a *feature-level* analysis using crosscoders—sparse autoencoders with a shared encoder applied across domains—to discover individual latent features that fire on both time-series inputs and semantically coherent natural-language passages.

D.1 Method

Setup. We train one *linear crosscoder* per transformer layer of Qwen3-0.6B. Each crosscoder has a shared encoder $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{1024\times4096}$ followed by Top-K sparsity (K=64), and two per-domain decoders $\mathbf{W}_{\text{dec}}^{(\text{PT})}, \mathbf{W}_{\text{dec}}^{(\text{FT})} \in \mathbb{R}^{4096\times1024}$ (12.6 M parameters total, float32). The encoder is applied independently to each domain's hidden states; because the weights are shared, the same latent feature can fire on both pretrained (PT) and finetuned (FT) inputs.

Domains. *PT*: Time series formatted as space-separated decimal strings (e.g., `0.123 -0.456 1.789 ...`), tokenized by the Qwen3-0.6B tokenizer. Sub-tokens corresponding to each timestep value are mean-pooled to produce one 1024-dim vector per timestep. *FT*: The same time series normalized per-window (z-score), clipped to $[-5, 5]$, and uniformly binned into 1024 tokens for the finetuned model.

Training. Input: 3,000 windows (T=512) from GiftEval [Aksu et al., 2024], precomputed as memory-mapped hidden-state arrays. Loss: per-domain MSE between normalized inputs and decoder reconstructions, summed over PT and FT. Dead-feature recovery (AuxK; top-64 among features firing $<1\%$ over the last 1,000 steps, weight $\frac{1}{32}$, active after step 1,000). AdamW, lr=3×10^{-4}, 500 warmup steps, cosine decay over 10,000 steps. Early stopping after 1,500 steps without validation improvement (minimum step 2,000).

Feature analysis pipeline. After training, 50,000 time-series windows are encoded; for each of the 4,096 features we record the PT and FT firing rates (fraction of windows with non-zero activation). Features firing $\geq1\%$ in both domains are labeled PT_FT and ranked by balance $= \min(\text{rate}_{\text{PT}}, \text{rate}_{\text{FT}})$. For the top-30 balanced features, we extract the 10 highest-activating time-series windows and 10 highest-activating WikiText-103 passages (30,000 sequences, 512 tokens each, encoded through the PT model and the shared crosscoder encoder with separate normalization statistics).

D.2 Results

Below we present four features with the clearest cross-domain links: each fires on a specific time-series pattern *and* on thematically coherent WikiText passages.

Feature 1712 (Layer 10) — Quantitative magnitude transitions. PT firing rate: 50.4%, FT: 41.8%, WikiText: 14.1%. This feature detects step changes and regime shifts in time series—values that jump from a near-zero baseline to a sustained high plateau—and fires on text passages dense with numerical measurements and unit conversions (Figure 17).

Top-5 WikiText passages at the peak-activation token:

1. *(act=14.1)* "...becoming later in the year by about two days every 243-year cycle. Transits usually occur in pairs, on nearly the same date eight years apart."
2. *(act=13.8)* "In practice, forward premiums and discounts are quoted as annualized percentage deviations from the spot exchange rate..."
3. *(act=13.7)* "Wages are reflective of the type of jobs available locally, including higher than average employment in manufacturing and the public sector. The working age population of the town in 2011..."
4. *(act=13.7)* "So for americium-241, the resistivity at 4.2 K increases with time from about 2 µOhm·cm to 10 µOhm·cm after 40 hours, and saturates at about 16 µOhm·cm..."
5. *(act=13.7)* "Falcon's Fury can theoretically accommodate 800 riders per hour. Carbon-fiber wings buttress each end of a group of seats..."

The shared representation encodes *quantitative magnitude and transition*: literal level shifts in time series, and passages dense with measurements, unit conversions, and numerical comparisons in text.

Feature 2469 (Layer 9) — Tropical weather systems. PT: 29.9%, FT: 20.7%, WikiText: 12.6%. Fires on volatile, regime-switching time series and exclusively on tropical cyclone and hurricane narratives in text (Figure 18).

Top-5 WikiText passages:

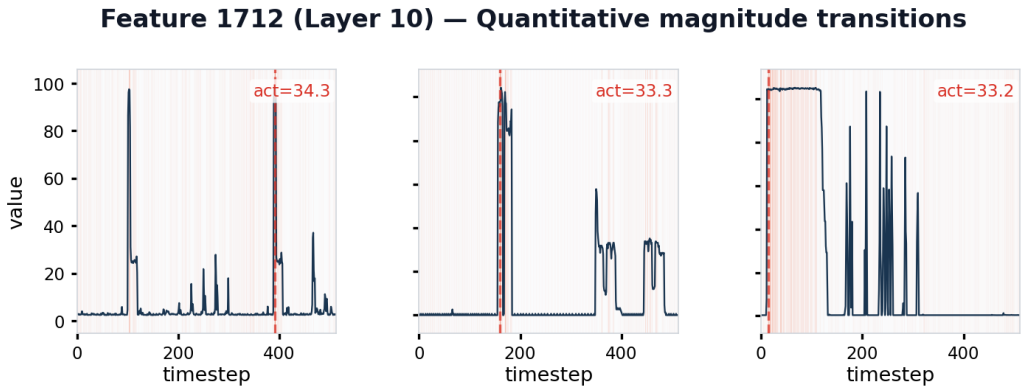


Figure 17: **Feature 1712 (Layer 10): Quantitative magnitude transitions.** Top-3 activating time-series windows. Blue: raw signal; red dashed: peak-activation timestep; orange shading: activation intensity. All three windows share a sudden jump from a low baseline to a high-magnitude plateau.

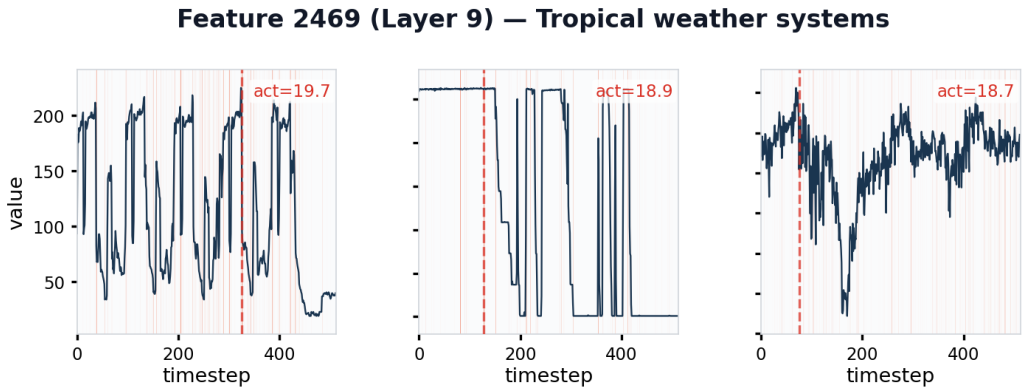


Figure 18: **Feature 2469 (Layer 9): Tropical weather systems.** Top-3 activating windows. The left window shows high-variance oscillations with abrupt drops; the middle and right show diverse volatile patterns with regime switching.

1. *(act=8.7)* "...the mean locus of formation shifts westward to the Caribbean and Gulf of Mexico, reversing the eastward progression of June through August. Wind shear from westerlies increases substantially through November..."
2. *(act=8.3)* "...due to a combination of very high wind shear and dry air. By October 17, most of the deep convection associated with the system dissipated; however, a brief decrease in wind shear allowed Omar to re-strengthen..."
3. *(act=8.1)* "The wave continued westward and related thunderstorm activity increased during the following week. The convective system organized into Tropical Depression Twenty-E on September 28..."
4. *(act=8.1)* "A tropical wave moved across the northeast Pacific Ocean and formed a tropical depression south of Mexico on October 16. It strengthened at a moderate pace and reached hurricane intensity on October 18."
5. *(act=7.9)* "...formation of Typhoon Chanchu in the western Pacific enhanced convective activity over the Bay of Bengal. By April 22, a trough developed along an axis from the southern Bay of Bengal eastward to the Andaman Sea."

The time-series patterns—volatile signals with sudden regime changes—mirror the physical phenomena described in the text: tropical storms that intensify, weaken under wind shear, and shift track.

Feature 3888 (Layer 7) — Naval battle events. PT: 18.5%, FT: 26.3%, WikiText: 10.5%. Fires on sharp isolated spikes in otherwise stable time series and on naval/military battle narratives with precise timestamps (Figure 19).

Top-5 WikiText passages:

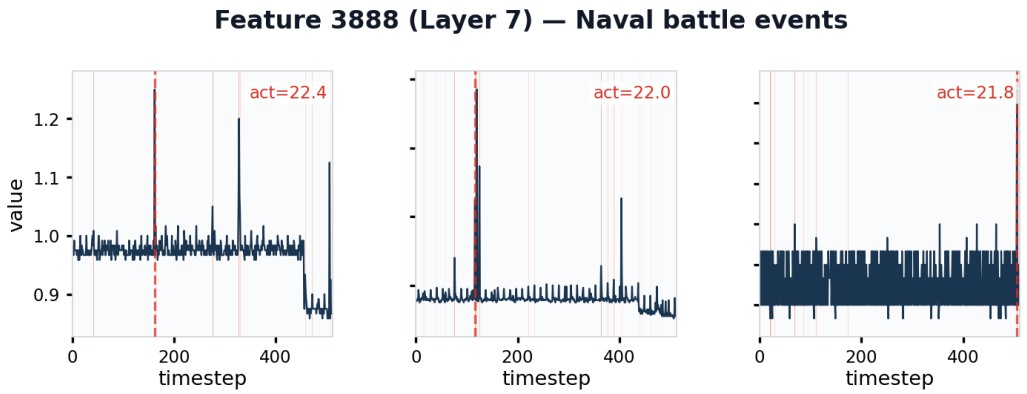


Figure 19: **Feature 3888 (Layer 7): Naval battle events.** Top-3 activating windows. Each shows a low-variance baseline punctuated by sharp spikes at the peak-activation timestep (red dashed line).

1. *(act=10.1)* "King George V had only 32 percent of her fuel left while Rodney had only enough fuel to continue the chase at high speed until 8:00 the following day. Admiral Tovey signalled his battlegroup…"
2. *(act=9.3)* "At 7:20 on 19 July, the destroyer force spotted and was spotted by a pair of Italian light cruisers; Giovanni dalle Bande Nere and Bartolomeo Colleoni, which opened fire seven minutes later."
3. *(act=9.2)* "Shortly before 16:00 the battlecruisers of I Scouting Group encountered the British 1st Battlecruiser Squadron under the command of Vice Admiral David Beatty. The opposing ships began an artillery…"
4. *(act=9.2)* "The eastern wind was not communicated to the aircraft, but was 270°, varying from 20 to 40 knots (37 to 74 km/h). The take-off started at 14:42:43…"
5. *(act=9.1)* "…torpedo boat attacks and at 07:30, Burrough sent Eskimo and Somali back to help Manchester but they arrived too late, took on survivors…"

Both modalities encode *sudden, precisely-located events*: an anomalous spike at a single timestep in time series, and a precisely-timestamped combat event in text.

Feature 2567 (Layer 8) — Missing / null data. PT: 35.9%, FT: 36.8%, WikiText: 9.4%. This feature fires exclusively on NaN/missing time-series data (all 10 top-activating windows are entirely NaN, with peak activation 40.3) and on <unk> tokens and incomplete references in text.

Top-5 WikiText passages:

1. *(act=19.8)* "…at the Royal Navy School of Flight Deck Operations at RNAS Culdrose. The following is an incomplete list of some of the surviving aircraft."
2. *(act=18.5)* "<unk>, <unk>, <unk>, <unk>, <unk>, Ulaid. Slightly later major groups included the Connachta, <unk>, <unk>. Smaller groups included the <unk>…"
3. *(act=18.0)* "…he encountered bad weather, forcing him to return to Japan with heavy damage. Without waiting for Vizcaino, another ship—built in Izu by the Tokugawa shogunate…"
4. *(act=17.8)* "Luke 9: <unk>-<unk> — καὶ <unk>, <unk> <unk> <unk> πνεύματος <unk> <unk>…"
5. *(act=17.7)* "…whom he married in the late 250s when she was 17 or 18 years old. The number of children Odaenathus had with his first wife is unknown and only one is attested."

The model represents *absent information* identically across modalities: NaN values in time series and <unk> tokens in text both occupy the same region of representation space.

E Causal Circuit Identification

The correlational analyses in Section 5 show that finetuning reuses pretrained directions and that cross-domain features exist. Here we present a preliminary causal analysis: we identify specific model components responsible for periodic time-series prediction and test what role those components play in language modeling.

E.1 Method

Since IO-only finetuning only trains embedding and LM-head layers, its 28 transformer layers are identical to PT. We zero-ablate [Wang et al., 2024] each of the 476 components (448 attention heads + 28 MLPs) on IO-only finetuning and measure the loss increase ($\Delta\mathcal{L}$) on periodic time series versus non-periodic controls.

Zero-ablation implementation. For each attention head, we register a `forward_pre_hook` on the output projection (`o_proj`) that zeros the head's 128-dimensional slice of the concatenated 2048-dimensional head output before projection. For each MLP layer, we register a `forward_hook` that replaces the MLP output with zeros, so the residual stream passes through unchanged. All ablations are performed under `torch.no_grad()`.

Synthetic time series. We generate 20 windows (length 512) for each of nine synthetic types. *Periodic*: sine, square wave, sawtooth, seasonal (trend + oscillation), damped sine. *Control*: white noise, constant, linear trend, random walk. Each window is independently z-score normalized, clipped to $[-5, 5]$, and uniformly binned into 1024 tokens (matching the IO-only finetuning tokenizer). This yields 180 evaluation windows total (100 periodic, 80 control).

Selectivity metric. For each ablated component we compute:

$$\Delta\mathcal{L}_{\text{periodic}} = \bar{\mathcal{L}}^{\text{periodic}}_{\text{ablated}} - \bar{\mathcal{L}}^{\text{periodic}}_{\text{baseline}}, \tag{7}$$

$$\Delta\mathcal{L}_{\text{control}} = \bar{\mathcal{L}}^{\text{control}}_{\text{ablated}} - \bar{\mathcal{L}}^{\text{control}}_{\text{baseline}}, \tag{8}$$

$$\text{selectivity} = \Delta\mathcal{L}_{\text{periodic}} - \Delta\mathcal{L}_{\text{control}}. \tag{9}$$

Components with high $\Delta\mathcal{L}_{\text{periodic}}$ and high selectivity are specifically critical for periodic prediction rather than general-purpose.

Cumulative ablation. We compose multiple ablation hooks simultaneously using nested context managers. For the pairwise sweep, we test all $\binom{8}{2} = 28$ pairs of the top-8 selective components. For the growing cumulative, we add components one at a time in order of individual $\Delta\mathcal{L}_{\text{periodic}}$. The superadditivity ratio is $\rho = \Delta\mathcal{L}_{\text{combined}} / \sum_i \Delta\mathcal{L}_i^{\text{individual}}$ [Conmy et al., 2023, Elhage et al., 2021]; we use a threshold of $\rho > 1.2$ to account for noise.

WikiText transfer. We ablate the top-5 circuit components simultaneously on 2,000 WikiText-103 passages (length 512) through the pretrained model (which shares identical transformer layers with IO-only finetuning). For each passage we record the per-sequence cross-entropy loss before and after ablation. The 50 most-degraded and 50 least-degraded passages are extracted for qualitative analysis.

E.2 Results

Individual component sweep. Table 6 shows the top-8 components ranked by $\Delta\mathcal{L}_{\text{periodic}}$. The two most impactful are both in Layer 1: MLP_{L1} ($\Delta\mathcal{L}_p = 5.75$, selectivity $= 3.44$) and head L1H4 ($\Delta\mathcal{L}_p = 4.50$, selectivity $= 3.81$). Head L20H0 is the third most impactful with the highest selectivity (3.28).

Table 6: Top-8 components by $\Delta\mathcal{L}$ on periodic TS under zero-ablation. Selectivity $= \Delta\mathcal{L}_{\text{periodic}} - \Delta\mathcal{L}_{\text{control}}$ isolates periodicity-specific impact. Full sweep: 476 components.

Component	$\Delta\mathcal{L}_{\text{periodic}}$	$\Delta\mathcal{L}_{\text{control}}$	Selectivity
MLP_{L1}	5.75	2.31	3.44
Head L1H4	4.50	0.69	3.81
Head L20H0	3.40	0.13	3.28
Head L13H6	3.35	2.56	0.79
Head L23H6	2.00	0.06	1.94
Head L21H8	1.85	-0.25	2.10
Head L15H3	1.55	0.38	1.18
Head L11H8	1.25	0.56	0.69

Cumulative ablation reveals a composed circuit. The Layer 1 pair (head L1H4 + MLP_{L1}) is strongly superadditive: ablating them together produces $\Delta\mathcal{L}_p = 15.50$, which is 51% larger than the sum of their individual effects ($\rho = 1.51$; Table 7). No other pair among the 28 tested exceeds the $\rho > 1.2$ threshold. Adding head L20H0 maintains superadditivity ($\rho = 1.36$); beyond three components, returns become subadditive ($\rho < 1$), indicating compensation rather than composition.

The periodicity circuit in language. We ablate the top-5 circuit components simultaneously on 2,000 WikiText-103 passages through the pretrained model. The circuit ablation causes a mean loss increase of $\Delta\mathcal{L} = 4.53$ nats—two orders of magnitude larger than any individual head's WikiText effect (≤ 0.04), confirming circuit-level interaction.

Table 7: Cumulative ablation. The Layer 1 pair is strongly superadditive ($\rho = 1.51$); the core circuit saturates at 3 components.

Components	$\Delta\mathcal{L}_{\text{periodic}}$	$\sum \Delta\mathcal{L}_{\text{indiv.}}$	ρ	$\Delta\mathcal{L}_{\text{control}}$
Head L1H4 + MLP_{L1}	15.50	10.25	**1.51**	2.38
+ Head L20H0 (top-3)	18.55	13.65	**1.36**	3.69
+ Head L21H8 (top-4)	18.40	15.50	1.19	3.06
+ Head L23H6 (top-5)	18.55	17.50	1.06	3.00
All top-8	18.30	23.65	0.77	3.13

The most-degraded passages ($\Delta\mathcal{L} = 8$–9; Table 8) are dominated by sequential enumerations and repeating grammatical templates: biographical sequences ("married to… she gave birth to…"), game mechanics with parallel clause structure ("sets a task for each stage… this task must be completed… the player with the most"), Billboard chart progressions, award category listings, and structured Wikipedia sections with repeating headers. In contrast, the least-degraded passages ($\Delta\mathcal{L} < 2$) are dense, non-repetitive prose: ecclesiastical titles, academic citations, canonical law, and literary criticism—text where predicting the next token depends on semantic content rather than structural repetition.

This suggests the circuit tracks *sequential repetitive structure*—the same abstract property shared by periodic time series and repetitive text. However, the evidence is preliminary: the WikiText passages most degraded by the circuit do not all show an obvious repetitive pattern, and disentangling the circuit's role in general sequence modeling from periodicity-specific computation requires further investigation.

Table 8: WikiText-103 passages most and least degraded by ablation of the periodicity circuit (top-5 components). Mean $\Delta\mathcal{L} = 4.53$ across 2,000 passages.

$\Delta\mathcal{L}$	Passage excerpt
Top 20 most degraded	
9.19	"…Townsend has been married to Tracy Turner, his girlfriend since he was 19. She gave birth to thei[r]…"
8.88	"…Preston winger Will Hayhurst, a Republic of Ireland under-21 international, was signed on a one-month loan…"
8.88	"…with the NHK Symphony Orchestra, but cancelled both deals upon Mwanga's return from Japan. Mwanga immediately quit…"
8.88	"…his/her final score on the song, with money being awarded in Guitar Hero World Tour. The games have also added…"
8.75	"…Viscount, sets a task for each stage. This task must be completed before the player can continue to another map…"
8.50	"…The player character is Michel Ardan, an eccentric and intrepid French scientist who is enthusiastic, daring…"
8.38	"…Best Music Video, Long Form. In 1998, the categories were retitled Best Short Form Music Video, and Best Long…"
8.31	"…on the Billboard Hot 100 twenty-nine, the highest U.S. entry among all singles released from the album…"
8.31	"…long run, with The A.V. Club attributing much of the show's early success to the character…"
8.25	"…Yankovic recorded 'Here's Johnny', a parody of 'Who's Johnny' by El DeBarge. The song, a loving ode to…"
8.19	"…The music video of 'Crazy in Love', released in May 2003, was directed by Jake Nava and filmed…"
8.19	"…Finishing with the worst record in the NHL, Columbus had the best chance of receiving the first overall pick…"
8.13	"…while one in the Pyramid Texts says the name is based on words shouted by Osiris…"
8.06	"…the album Daydream most resembles in its emphasis on R&B grooves. Tucker specifically complimented 'One Sweet Day'…"
8.00	"…similar to Konami's Guitar Freaks and to a lesser extent Harmonix's previous music games such as Frequency…"
8.00	"…he 'hit the wall with play-along music games', and challenged the game makers to explore other ways…"
7.94	"…saying that he 'was upset. But when you see the talent that was there, it was an honour just to be in the final'…"
7.94	"…Flags indicate national team as defined under FIFA eligibility rules. Players may hold more than one non-FIFA…"
7.94	"…at the Television Critics Association summer media tour in Beverly Hills, California…"
7.94	"…co-wrote and produced a song with Kenneth 'Babyface' Edmonds, with whom she had collaborated on Music Box…"
Top 10 least degraded	
0.83	"…Porto e Santa Rufina; Sub-dean of the Sacred College of Cardinals; prefect of the S.C. of the Good Government…"
1.48	"…Cardinal-Priest of SS. Giovanni e Paolo; Grand penitentiary; prefect of the Congregation for the correction…"
1.59	"…From the Jewish Question to the Jewish State: An Essay on the Theory of Zionism (thesis), Princeton Univ.…"
1.77	"…called Saprang's transfer a 'demotion' and a 'punishment.' However, Saprang himself claimed that he did not…"
1.82	"…the Society of Jesus, provided he observed the canon law; and that it was desirable that the pope should…"
1.89	"…the abnormal excess of white blood cells in people with the clinical syndrome described by Velpeau and Bennett…"
1.89	"…The protagonist sounds like a 'colonial administrator', and his reference to seeking a newer world echoes…"
1.94	"…Mbaruk's nephew, Mbaruk bin Rashid, refused to acknowledge the appointment of a new leader…"
1.98	"…00 works were published underground over the course of the war. Literary discussions were held…"
2.02	"…Although the poem was defended by a few critics, E.C. Pettet returned to the argument that the poem lacked…"