Feature Geometry of Language Models Transfer Across Modalities to Time Series

Prateek Humane [1 2 *] **Alexis Roger** [3 2 *] **Zhenghan Tai** [4 *]
Gwen Legate [5 2] **Andrei Mircea** [1 2] **Vasilii Feofanov** [6] **Irina Rish** [1 2]

Abstract

Language models transfer to time-series forecasting, but it is unclear whether this reflects reusable internal structure or rapid relearning under a familiar architecture. We study this transfer directly by comparing pretrained and randomly initialized versions of the same model on a forecasting objective whose inputs have little semantic overlap with text but still require autoregressive sequential structure. Across Qwen3-0.6B finetuning experiments, language initialization gives coherent per-example gradients from the first update, while random initialization first passes through a low-alignment warmup phase. Effective-rank and hidden-state analyses show that finetuning selectively reshapes an existing representation geometry rather than constructing the simpler temporal geometry found by models trained from scratch. Cross-domain sparse features and causal ablations then expose candidate transferred primitives, including a Layer 1 head–MLP circuit whose ablation selectively increases loss on periodic forecasting and repetitive language passages. These results support an account of cross-modal transfer in which autoregressive pretraining creates temporal feature geometry that can be selected and specialized outside language.

1. Introduction

Pretrained language models can be adapted to time-series forecasting, but the source of this cross-modal transfer remains unclear. Does transfer come from reusable internal structure learned during language pretraining, or does the transformer architecture simply relearn forecasting quickly once given time-series supervision? Because time series have little semantic overlap with text, successful transfer is unlikely to be explained by ordinary linguistic knowledge. Instead, it would suggest that autoregressive pretraining can produce abstract sequential geometry: directions, sparse features, and circuits for repetition, phase, magnitude change, missingness, and other temporal motifs that are useful beyond the modality in which they were learned.

Time series provide a useful probe because the downstream task is behaviorally clear but internally nontrivial. Forecasting requires the model to organize hidden states around temporal continuation, yet the inputs are discretized numerical values rather than words. Prior work on LLMs for time series has debated whether gains come from language pretraining, architecture, or tokenization effects (Ansari et al., 2024; Tan et al., 2024; Zheng et al., 2025; Zhang et al., 2025; Qiu et al., 2026). We take a complementary view: rather than asking only whether transfer improves forecasting or not, we ask what representation geometry would make the transfer work.

Our central claim is that language pretraining creates reusable temporal feature geometry that time-series finetuning can select and specialize. When a pretrained LLM is finetuned on time series, the model receives coherent gradient signals immediately, selectively reshapes its representational rank, and reuses features that fire on structurally related motifs in text and time series. Randomly initialized models can eventually learn the task, but they do so through a different geometric path: first collapsing broad random representations and then building simpler phase-ordered trajectories from scratch.

We contribute four pieces of evidence. First, per-example gradients for the time-series objective are aligned from the first finetuning step in language-initialized models but not in random initializations. Second, effective-rank dynamics show selective compression and middle-layer redistribution rather than uniform collapse. Third, hidden-state trajectories on synthetic temporal inputs reveal reuse of richer inherited geometry, while random models learn simpler low-dimensional phase curves. Fourth, cross-domain sparse features and zero-ablation expose candidate primitives shared between language and time-series prediction, including a

[1]Université de Montréal [2]Mila - Quebec AI Institute [3]McGill University [4]University of Toronto [5]Concordia University [6]42.com. Correspondence to: <alexis.roger@mila.quebec>.

Proceedings of the Mechanistic Interpretability Workshop at the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. 2026.

Layer 1 circuit whose ablation selectively increases loss on periodic time series and structurally repetitive WikiText passages.

Related work. This paper connects work on LLM adaptation for forecasting (Jin et al., 2023; Gruver et al., 2023; Zhou et al., 2023; Wolff et al., 2025; Roger et al., 2025), cross-modal representation geometry (Huh et al., 2024; Jha et al., 2025; Chen et al., 2024; Roschmann et al., 2025), and mechanistic tools for feature and circuit discovery (Elhage et al., 2021; Conmy et al., 2023). Our focus is not a new forecaster, but an internal account of what high-level temporal structure is present and how finetuning uses it.

2. Setup: Language-to-Time-Series Transfer

We use Qwen3-0.6B (Yang et al., 2025) as a pretrained autoregressive transformer and compare it to the same architecture with randomly initialized weights. Continuous univariate time series are converted into next-token prediction problems: each window has context length $C = 512$ and prediction horizon $L = 64$, is normalized using context-window statistics, clipped to $[-5, 5]$, and discretized into $V = 1024$ uniform bins. Bin indices are mapped directly to the first positions of the model vocabulary, while the original Qwen special tokens are preserved. The model is trained with quantile loss over $\mathcal{Q} = \{0.1, \ldots, 0.9\}$ by interpreting logits over ordered bins as a categorical distribution and extracting quantiles from the induced CDF.

We finetune on sliding windows from GiftEval (Aksu et al., 2024) and evaluate on held-out windows with probabilistic forecasting metrics following GluonTS conventions (Alexandrov et al., 2020). Forecasting performance establishes that the transfer is behaviorally meaningful; the internal analyses below ask what pretrained geometry makes that transfer possible. We compare four adaptation regimes: full finetuning; input/output-only tuning, where the transformer trunk is frozen; LoRA on attention projections; and LoRA on attention plus trainable input/output layers. Full hyperparameters, metrics, and benchmark curves are in Appendix B.

Throughout the paper, we use *LangInit* for models initialized from pretrained Qwen3 weights and *RandInit* for identical architectures trained from scratch on the same time-series objective. This isolates the effect of pretrained internal structure from architecture, tokenizer, loss, and data pipeline.

3. Finetuning Reveals Alignment to Pretrained Geometry

If language pretraining creates reusable temporal geometry, then the time-series objective should already point in coherent directions before any time-series specialization has occurred. We test this by measuring per-example gradient alignment during finetuning. For a fixed batch of $N = 32$ held-out time-series sequences, we compute each per-sample gradient $\mathbf{g}_i = \nabla_\theta \mathcal{L}(x_i; \theta)$ and report the mean off-diagonal cosine similarity,

$$\text{align}(\theta) = \frac{1}{N(N-1)} \sum_{i \neq j} \cos(\mathbf{g}_i, \mathbf{g}_j).$$

High alignment means that different time-series examples agree on how the model should change; low alignment indicates destructive interference similar to the zero-sum learning dynamics observed in language-model training (Mircea et al., 2025).

Figure 1 shows a consistent split across all adaptation regimes. LangInit models have aligned gradients from the first update and their loss decreases immediately. RandInit models begin with nearly unaligned gradients and only start learning once alignment emerges after a warmup phase. Figure 2 shows the same transition geometrically on a sine-wave probe: RandInit develops phase structure only when gradient alignment rises, whereas LangInit begins with phase-coherent hidden states. The effect is visible even in IO-only tuning, where the transformer trunk is frozen and only the embedding and output layers can adapt. This suggests that the pretrained residual stream already organizes numerical-token inputs into directions where many examples induce mutually reinforcing updates.

These results motivate the rest of the paper. Gradient coherence tells us that the pretrained model offers usable directions for the new objective, but not what those directions are. Therefore, we will now inspect how the hidden-state geometry changes during finetuning and which features appear to populate the transferred subspace.

4. Representation Geometry Shows Reuse, Not Reinvention

4.1. Rank Dynamics Reveal Selective Reshaping

We next ask what changes inside the model when the coherent gradients from Section 3 are applied. At every saved checkpoint and layer, we collect hidden states from 10,000 time-series windows and compute the effective rank of the centered covariance matrix. Effective rank tracks how many

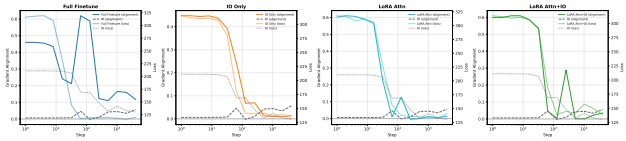


Figure 1. **Pretraining gives the time-series objective coherent gradients.** Each panel shows mean pairwise per-sample gradient alignment (left axis) and CRPS evaluation loss (right axis) across training steps for each adaptation regime. Solid lines are LangInit; dashed lines are RandInit. LangInit models begin with high alignment and decreasing loss, while RandInit models remain near zero alignment until roughly steps 64–256, after which loss begins to fall.

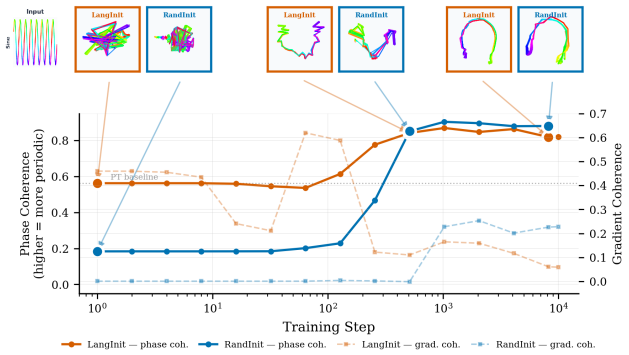


Figure 2. **Phase and gradient coherence across training.** Higher phase coherence means the model better captures the periodic structure of a sine-wave input. Using t-SNE, we visualize hidden states at training steps 1, 512, and 8192. LangInit begins with phase-coherent hidden states, while RandInit starts near zero and undergoes an abrupt transition around step 300, where phase coherence and gradient alignment rise together.

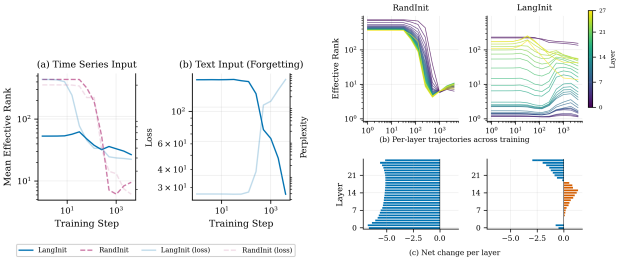


Figure 3. **Finetuning reshapes pretrained geometry selectively.** Left: mean effective rank across layers during training. RandInit starts near-isotropic and rapidly collapses before learning, while LangInit begins compressed and changes more gradually. Right: layerwise trajectories show that LangInit redistributes capacity, with middle layers increasing rank while early and late layers compress.

high-variance directions the representation uses and is computed as follows:

$$\text{erank}(\Sigma) = \exp(-\sum_i p_i \log p_i) \qquad \text{where} \quad p_i = \lambda_i / \sum_j \lambda_j$$

Figure 3 shows that RandInit and LangInit reach forecasting behavior through different representational trajectories. RandInit begins near-isotropic and undergoes a rapid, nearly uniform rank collapse across the network, suggesting that it must first carve a usable low-dimensional temporal structure out of random features. LangInit starts from an already compressed representation and changes selectively: time-series effective rank decreases only moderately on average, while middle layers expand and peripheral layers contract. This pattern is consistent with reuse of pretrained sequence-processing directions rather than wholesale reconstruction of temporal structure.

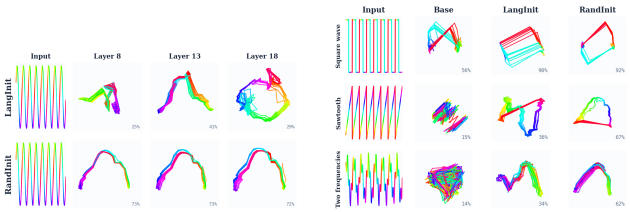


Figure 4. **LangInit and RandInit solve temporal structure with different geometry.** Left: for a sine wave, RandInit learns clean low-dimensional arcs across layers, while LangInit retains richer layer-specific loops inherited from pretraining. Right: across synthetic waveforms, RandInit forms simple PCA-readable trajectories; LangInit forms more heterogeneous but structured trajectories.

4.2. Synthetic Inputs Expose Different Internal Solutions

To inspect the learned geometry directly, we pass synthetic waveforms through LangInit, RandInit, and the base language model, then visualize hidden states with PCA. These inputs isolate simple temporal properties such as phase, periodicity, slope, and discontinuities.

Figure 4 shows that both models encode phase, but not in the same way or number of dimensions. RandInit discovers clean arcs with most variance captured in two principal components, a simple solution learned from scratch. LangInit instead represents the same temporal structure through more complex, layer-specific loops with lower two-PC variance. Despite this visual difference, trained LangInit and RandInit representations are highly aligned across synthetic inputs (CKA $= 0.855$–0.988; Appendix C), indicating that they recover similar relational structure through different coordinate systems. The pretrained model is therefore not merely faster; it brings an inherited geometry that finetuning reshapes into a task-usable form.

4.3. Shared Features and a Candidate Periodicity Circuit

Finally, we ask what high-level properties populate the reused geometry. 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 windows and WikiText passages.

Table 1 gives examples of candidate transferred high-level properties. They are not semantic labels copied from text into forecasting; rather, they are structural motifs that can be realized in both domains. For example, a feature that fires on time-series level shifts also fires on measurement-heavy text, while a feature for missing time-series values fires on `<unk>` and incomplete references.

The strongest causal evidence is for one such candidate primitive. In an IO-only finetuned model whose transformer layers are identical to the pretrained model, zero-ablation over 476 components identifies a superadditive Layer 1 pair, head L1H4 with MLP_{L1}. Ablating this pair selectively increases loss on periodic time series ($\Delta\mathcal{L} = 13.34$ versus 2.59 on controls; Cohen's $d = 0.61$) and preferentially degrades structurally repetitive WikiText passages (Spearman $\rho = 0.161$, $p = 4.9\times10^{-13}$). This is not a complete circuit analysis, but it suggests that a transferred repetition/periodicity primitive is causally involved rather than merely correlated.

Table 1. **Candidate shared pretrained–finetuned crosscoder features.** Features concentrate in layers 7–10 and link concrete time-series motifs to coherent text motifs. Full examples are in Appendix D.

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

5. Discussion and Limitations

Our results address why language-to-time-series transfer can work: high-level temporal geometry can be present in a language model's internal states and reused when the model is adapted to a domain with little semantic overlap. Gradient alignment shows that pretrained representations make the new objective locally coherent. Rank dynamics and synthetic inputs show that finetuning reshapes inherited geometry rather than following the same representational path as random initialization. Crosscoder features and causal ablations suggest that the reused structure includes candidate interpretable primitives such as transitions, spikes, missingness, and repetition.

Forecasting performance establishes that the transfer is meaningful, but the main finding is internal: autoregressive pretraining can learn feature geometry that is more general than the surface modality. Under this view, finetuning acts partly as selection and specialization over existing directions, while randomly initialized training must first construct a simpler temporal coordinate system.

Our study also has several limitations. Experiments are conducted on a single backbone (Qwen3-0.6B), dataset (GiftEval), and discretization pipeline, and broader validation across architectures and forecasting settings would strengthen the generality of the conclusions. While the crosscoder analysis reveals shared temporal features, it provides only a partial characterization of the transferred representation subspace, and feature interpretations remain qualitative. Similarly, the circuit analysis should be viewed as suggestive evidence rather than a complete mechanistic account. Finally, although additional experiments support the compatibility of pretrained hidden states with time-series trajectories, further work is needed to disentangle the respective roles of representation geometry, initialization scale, and optimization dynamics in producing the observed transfer behavior.

6. 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

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.

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.

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.

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.

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/.

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.

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.

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.

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.

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.

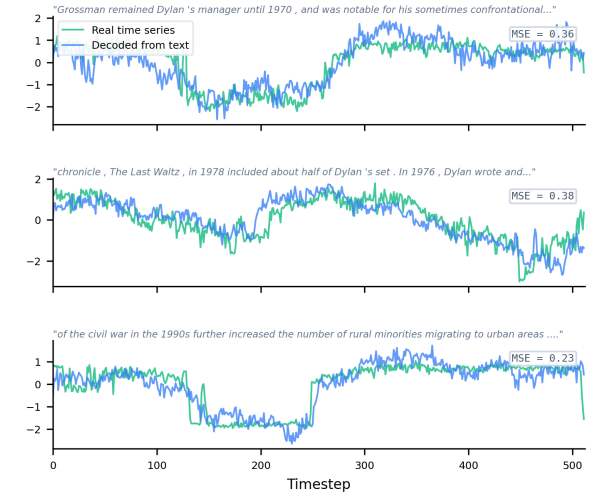
L. N. Zheng, C. Dong, W. E. Zhang, L. Yue, M. Xu, O. Maennel, and W. Chen. Understanding why large language models can be ineffective in time series analysis: The impact of modality alignment. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 4026–4037.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

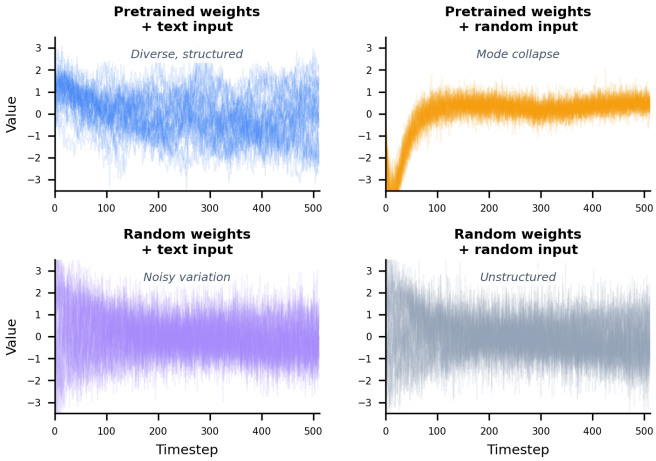
A. Frozen-State Linear Mapping and Retrieval

This appendix contains the linear mapping experiment omitted from the short-paper main text. It asks whether frozen language-model hidden states are geometrically compatible with realistic time-series trajectories before any time-series supervision.

A.1. Linear Decoding from Text Hidden States



(a) Decoded time series from text



(b) Ablation: what creates the structure?

Figure 5. **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 with no paired text–time-series data.

We pass $N = 1{,}920$ WikiText-103 sequences of $T = 512$ tokens through frozen Qwen3-0.6B. For each token, we concatenate hidden states from all 28 layers:

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

A single linear layer is applied independently at each timestep,

$$\hat{y}_{i,t} = \mathbf{w}^\top \mathbf{h}_{i,t} + b, \quad \mathbf{w} \in \mathbb{R}^D,\ b \in \mathbb{R}. \tag{2}$$

The output sequence $\hat{\mathbf{y}}_i$ is z-score normalized to match the target time-series normalization.

We train the map with an EM-style nearest-neighbor objective. At each step, each prediction is matched to its nearest z-scored window from a bank of $M = 10{,}000$ GiftEval time series (Aksu et al., 2024). 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

$$\mathcal{L} = \frac{1}{B}\sum_{i=1}^{B}\frac{1}{T}\|\hat{\mathbf{y}}_i - \mathbf{Y}_{j_i^*}\|^2 + \lambda\frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}}\cos\left(|\mathrm{FFT}(\hat{\mathbf{y}}_i)|^2, |\mathrm{FFT}(\hat{\mathbf{y}}_j)|^2\right). \tag{3}$$

The second term discourages mode collapse through a PSD diversity penalty. We use $\lambda = 0.5$ and train with Adam at learning rate 10^{-3} for 100 epochs with batch size 32.

Across the 1,920 training predictions, nearest neighbors span 686 distinct real time series, so 36% of predictions map to unique real windows rather than collapsing to repeated patterns. On 1,000 held-out WikiText sequences, the map achieves comparable quality (mean nearest-neighbor MSE 0.72 versus 0.67 on training text), suggesting that the compatibility is a property of the pretrained representation distribution rather than memorized text inputs.

We also run a 2×2 ablation crossing model weights (pretrained versus randomly initialized) and input (WikiText versus random tokens). 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). Only pretrained weights and meaningful text together produce diverse temporal signals.

Table 2. Fair top-K comparison across ablation conditions. For each condition, we take the best-K unique matches and report mean nearest-neighbor MSE. This controls for diversity: conditions with fewer unique matches are only compared at K values they can support.

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**	—	—	—

A.2. Retrieval Forecasting from Projected Dynamics

The linear decoding result shows shape-level similarity between projected hidden states and real time series. We also test whether this extends to predictive structure. Given 500 GiftEval windows, we observe only the first 256 of 512 timesteps. Among 10,000 WikiText projections, we retrieve the projection with lowest MSE on the observed first half and use its second half as the forecast. The baseline repeats the final observed value.

Across all 500 queries, retrieval forecasting achieves MSE = 1.91 versus last-value's 2.27, a 16% reduction. Retrieval wins in only 37% of cases, but when it wins, the advantage is large (median $4\times$ lower MSE), typically because the retrieved projection captures trends and level shifts that a last-value baseline misses. This supports the weaker claim that pretrained hidden-state trajectories are

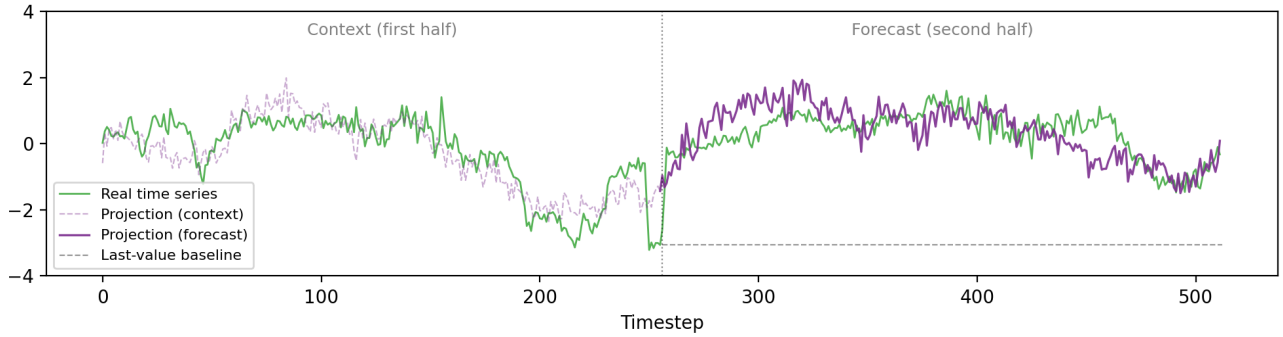


Figure 6. Forecasting example with largest gap compared to baseline from 500 evaluation queries. The first 256 timesteps are used to retrieve the closest WikiText projection; the projection's second half (purple) forecasts the continuation.

compatible with time-series-like dynamics, not that they contain an optimal forecasting subspace.

B. Experimental Setup

B.1. Hyperparameters and Reproducibility

Table 3. 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

B.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.

B.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:

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

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

B.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 $\text{QL}_q(y, \hat{y}) = (q - \mathbf{1}_{y \leq \hat{y}}) \cdot (y - \hat{y})$ 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:

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

B.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.

B.3. Experimental Results

Here we provide the complete set of evaluation results across all training regimes and metrics. Figure 7 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 4 and 5 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.

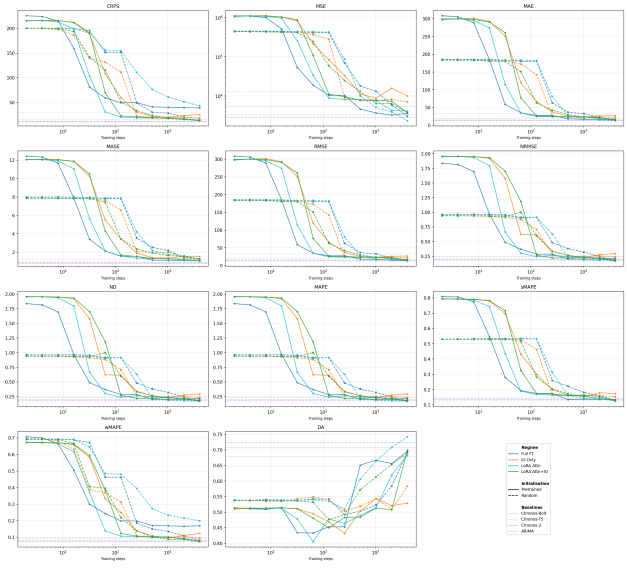


Figure 7. Training progression of all $h = 1$ forecasting metrics across training steps for four training regimes. Solid lines denote language-pretrained initialization (Qwen3-0.6B); dashed lines denote random initialization. Horizontal dotted lines indicate baseline performance (Chronos-T5, Chronos-Bolt, 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 4. 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↓
Pretrained	Full Finetune	49.51	10461	27.60	**1.561**	27.60	0.286
	IO Only	59.56	84846	66.53	3.430	66.53	0.614
	LoRA Attn	**20.10**	**8818**	**24.68**	1.607	**24.68**	**0.243**
	LoRA Attn+IO	22.54	11220	26.11	1.702	26.11	0.270
Random Init	Full Finetune	151.8	415339	180.4	7.819	180.4	0.914
	IO Only	111.6	280932	142.3	6.565	142.3	0.706
	LoRA Attn	154.5	430387	182.2	7.872	182.2	0.917
	LoRA Attn+IO	**50.68**	**59864**	**62.90**	**3.438**	**62.90**	**0.599**
Baselines	Chronos-Bolt	14.94	5409	18.91	0.934	18.91	0.240
	Chronos-T5	29.88	12764	32.99	1.212	32.99	**0.173**
	Chronos-2	**10.89**	**2886**	**13.38**	0.830	**13.38**	0.192
	ARIMA	11.94	3327	14.86	**0.805**	14.86	0.201
	Qwen3 Text	213.5	2.6M	213.5	376667	213.5	39508

Figure 8. 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 5. 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

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

$$\text{coherence} = \frac{\text{mean}(d_{ij} \mid i \bmod P = j \bmod P)}{\text{mean}(d_{ij} \mid \text{all pairs})}. \quad (4)$$

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.

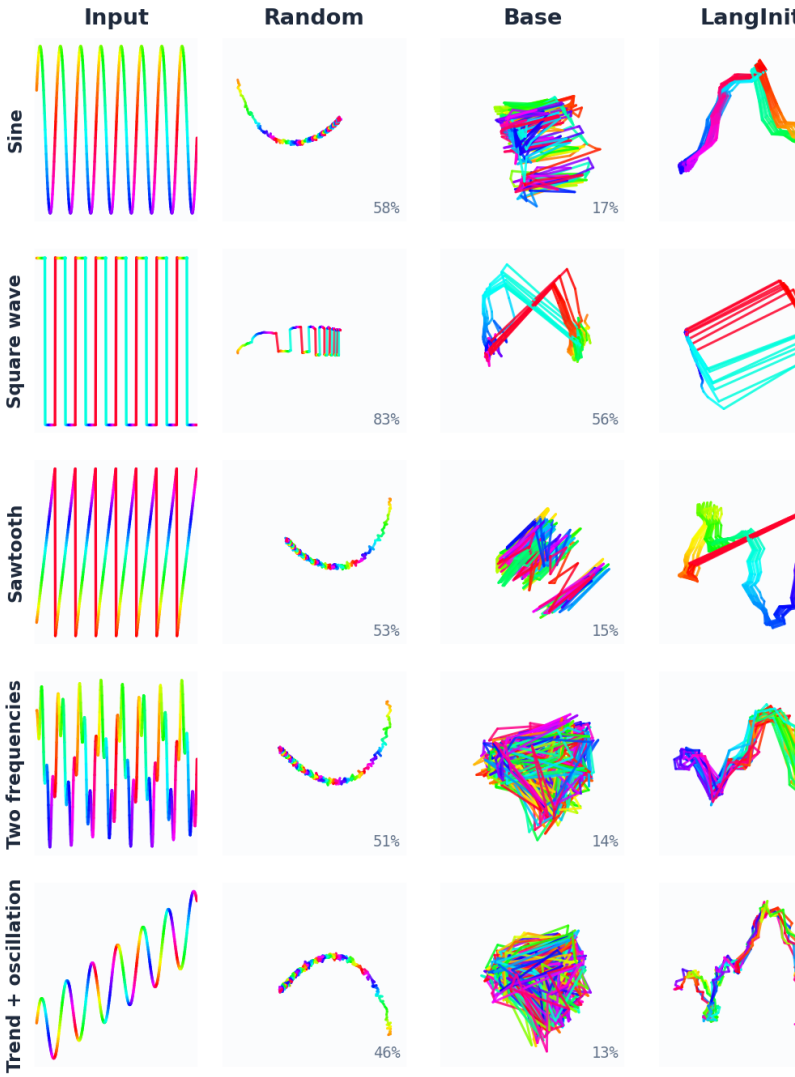


Figure 9. **Hidden-state trajectories for synthetic inputs at Layer 13 (2 PCA).** Trajectories are 2D PCA projections of hidden states, colored by input phase. Percentages show variance captured by 2 PCs. Random and Base produce unstructured trajectories. RandInit discovers clean, low-dimensional representations (62–92% PCA variance) while LangInit creates geometrically complex but structured trajectories (34–98%) that vary by input type.

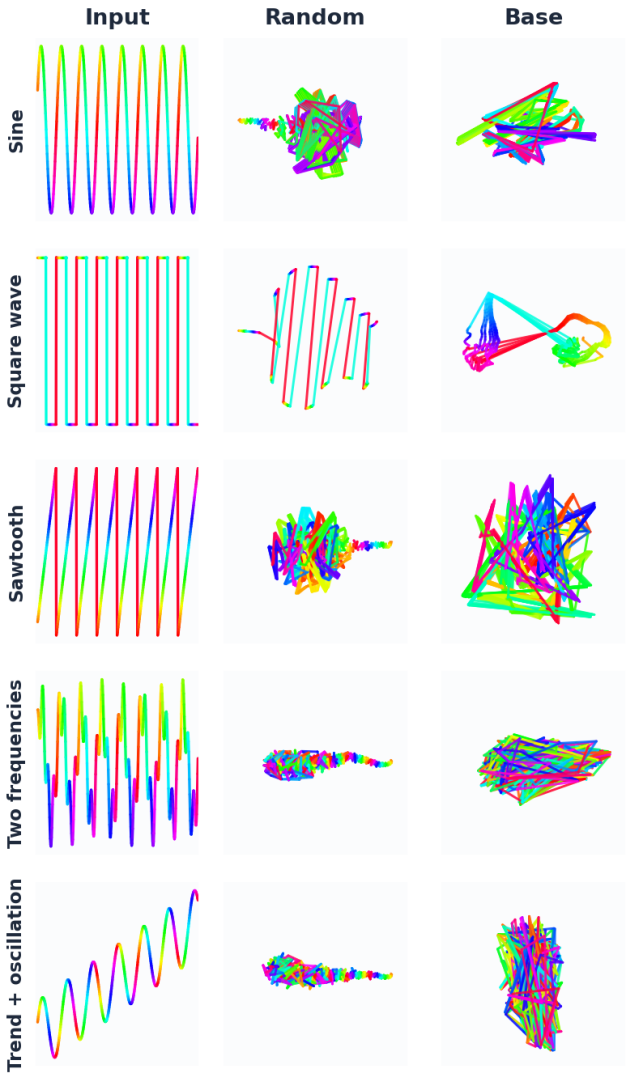


Figure 10. **Hidden-state trajectories for synthetic inputs at Layer 13 (t-SNE).** Same setup as Figure 9 but using t-SNE (perplexity 30) instead of PCA. Despite the different global geometries revealed by PCA, t-SNE shows that LangInit and RandInit develop similar local neighborhood structure: periodic inputs form smooth, phase-ordered curves in both models, confirming that both arrive at functionally equivalent representations through different geometric paths.

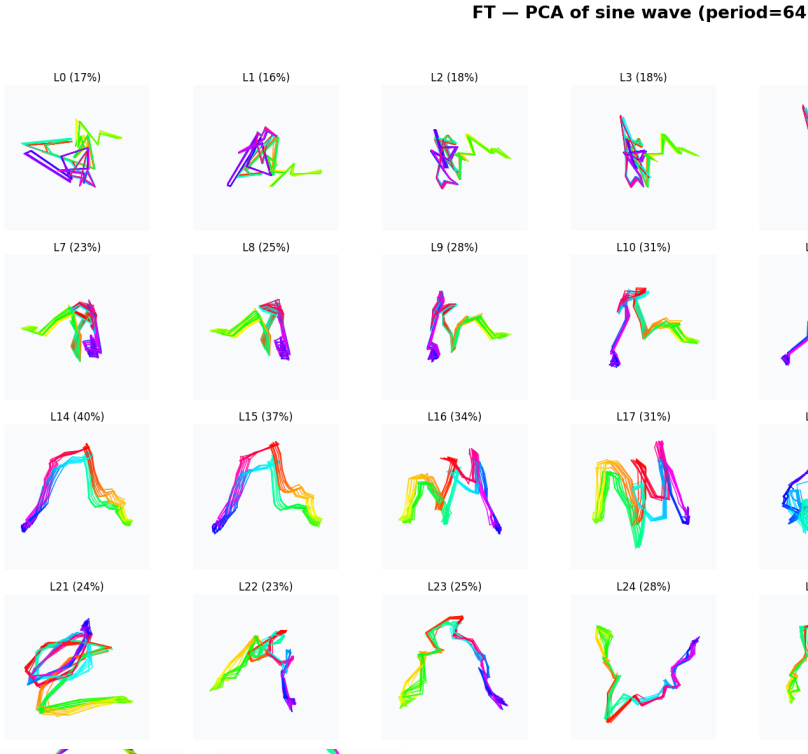


Figure 11. **LangInit: sine wave PCA trajectories across all 28 layers.** Each subplot shows the 2D PCA projection of hidden states from a sine wave (period 64) at one transformer layer, colored by input phase. Percentages show variance explained by 2 PCs. LangInit exhibits layer-specific geometry with varying complexity and moderate PCA variance (17–49%), reflecting the rich, heterogeneous representations inherited from language pretraining.

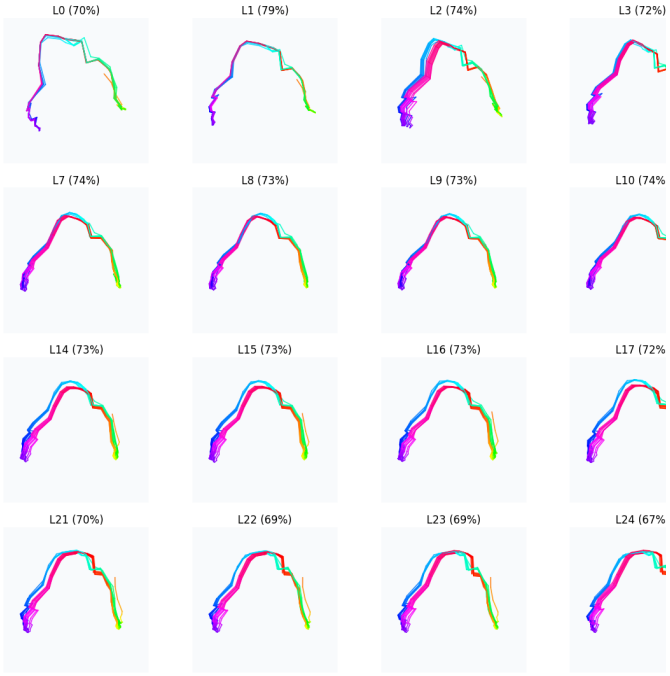


Figure 12. **RandInit: sine wave PCA trajectories across all 28 layers.** Same setup as Figure 11. RandInit produces nearly identical clean arcs at every layer with uniformly high PCA variance (67–79%), confirming that its learned representation is geometrically homogeneous across the network.

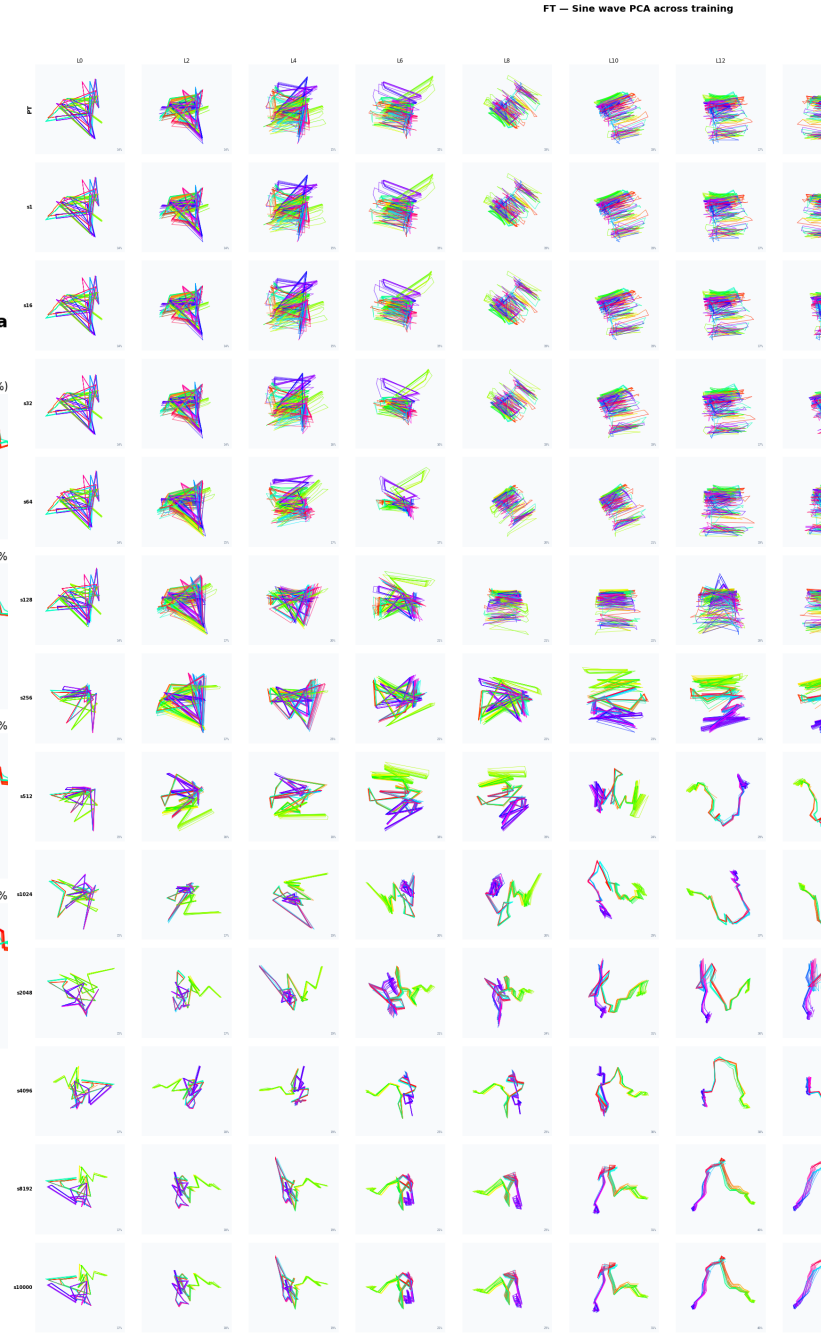


Figure 13. **LangInit: sine wave representations across training checkpoints.** Rows are training steps (top: PT baseline; bottom: final checkpoint at step 10,000), columns are selected layers. LangInit begins from the pretrained model's complex trajectories and gradually reshapes them into structured loops, with the transition occurring around steps 256–1024.

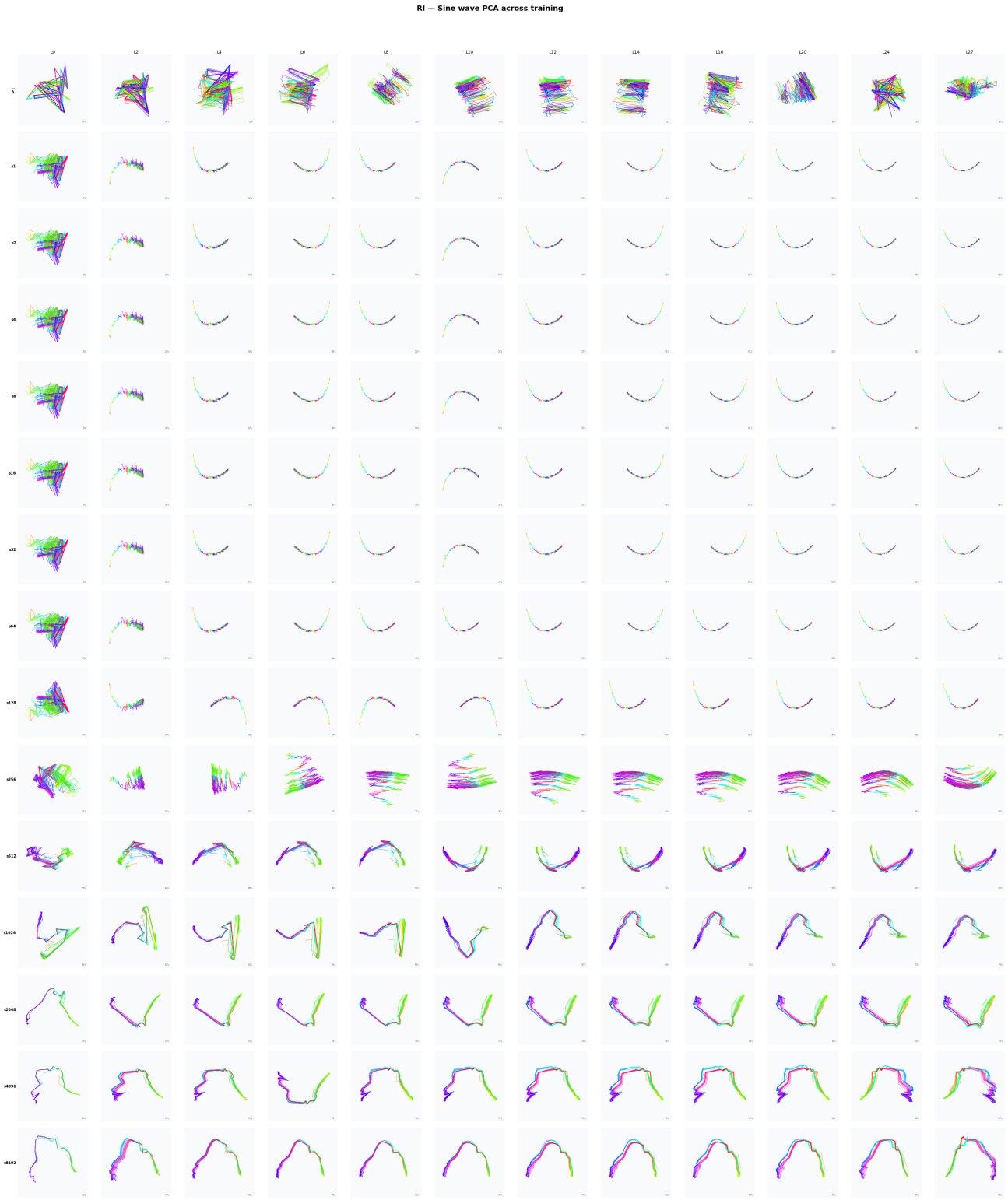


Figure 14. **RandInit: sine wave representations across training checkpoints.** Same setup as Figure 13. RandInit starts from near-empty representations (random weights) and converges to uniform clean arcs by step 2048, with all layers collapsing to the same geometry simultaneously.

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 $\geq$1% 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 15).

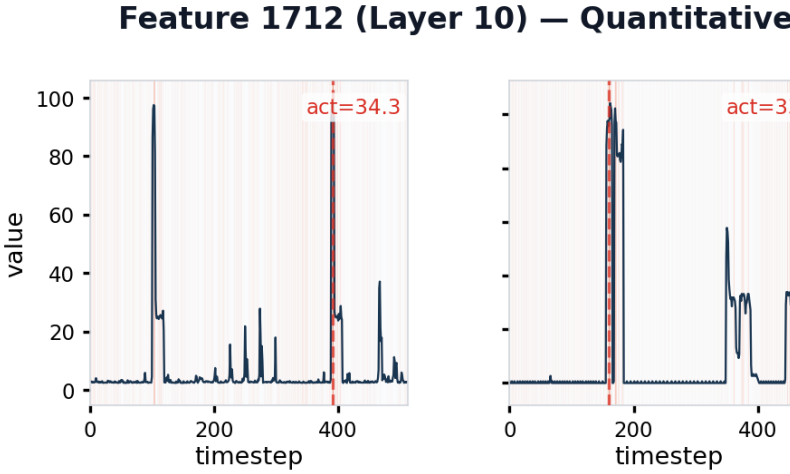


Figure 15. **Feature 1712 (Layer 10): Quantitative magnitude transitions.** Top-3 activating time-series windows. Blue: raw signal; red dashed: peak-activation timestep; orange shading: activation intensity. All three windows share a sudden jump from a low baseline to a high-magnitude plateau.

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 16).

Top-5 WikiText passages:

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..."

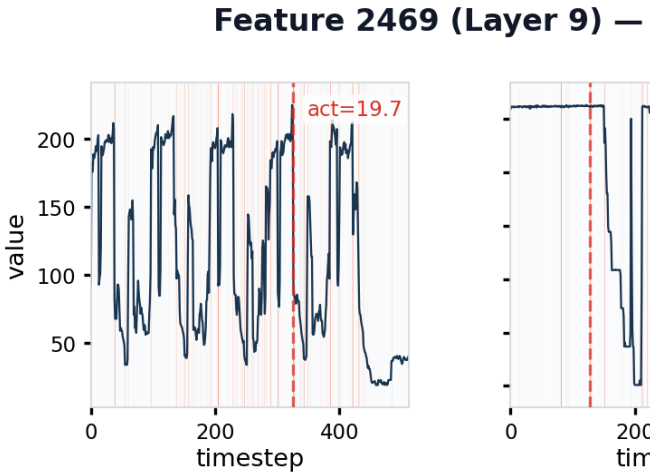


Figure 16. **Feature 2469 (Layer 9): Tropical weather systems.** Top-3 activating windows. The left window shows high-variance oscillations with abrupt drops; the middle and right show diverse volatile patterns with regime switching.

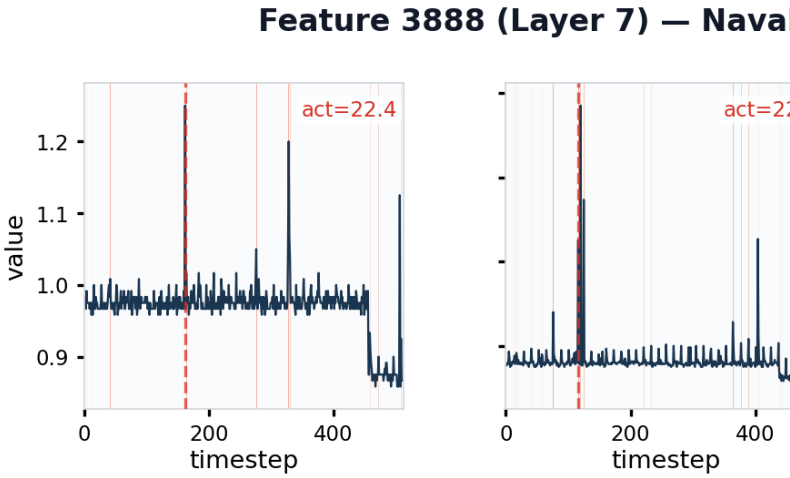


Figure 17. **Feature 3888 (Layer 7): Naval battle events.** Top-3 activating windows. Each shows a low-variance baseline punctuated by sharp spikes at the peak-activation timestep (red dashed line).

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 17).

Top-5 WikiText passages:

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 4 show that finetuning reuses pretrained directions and that cross-domain features exist. Here we present a 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 100 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 900 evaluation windows total (500 periodic, 400 control).

Selectivity metric. For each ablated component we compute:

$$\Delta\mathcal{L}_{\text{periodic}} = \bar{\mathcal{L}}^{\text{periodic}}_{\text{ablated}} - \bar{\mathcal{L}}^{\text{periodic}}_{\text{baseline}}, \tag{5}$$

$$\Delta\mathcal{L}_{\text{control}} = \bar{\mathcal{L}}^{\text{control}}_{\text{ablated}} - \bar{\mathcal{L}}^{\text{control}}_{\text{baseline}}, \tag{6}$$

$$\text{selectivity} = \Delta\mathcal{L}_{\text{periodic}} - \Delta\mathcal{L}_{\text{control}}. \tag{7}$$

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 Layer 1 pair and the top-5 circuit components 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. We score each passage by structural repetition using bigram/trigram repetition, consecutive identical words, and repeated phrases; the 50 most-degraded and 50 least-degraded passages are also 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.

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

Expanded validation of the Layer 1 pair. We rerun the Layer 1 pair ablation on 100 windows per synthetic type and estimate 95% confidence intervals with 1,000 bootstrap resamples. The effect is selective for periodic structure: periodic inputs average $\Delta\mathcal{L} = 13.34$ versus 2.59 on controls (Cohen's $d = 0.61$; Table 8). The largest effect is on square waves, but the result does not depend entirely on that case: removing square waves, the remaining periodic types average $\Delta\mathcal{L} = 4.91$, still above the control mean.

The periodicity circuit in language. For the Layer 1 pair, the mean WikiText loss increase is $\Delta\mathcal{L} = 4.07$ nats. The effect is larger on structurally repetitive passages: Spearman $\rho = 0.161$ (95% CI $[0.117, 0.206]$, $p = 4.9\times10^{-13}$), and the most repetitive quintile loses 4.37 nats versus 3.83 for the least repetitive quintile (Cohen's $d = 0.48$; Table 9). We also ablate the top-5 circuit components simultaneously on the same 2,000 passages. This

Table 8. Expanded Layer 1 pair ablation by synthetic time-series type. The pair is head L1H4 + MLP_{L1}. Confidence intervals are bootstrapped over 100 windows per type.

Category	Type	$\Delta\mathcal{L}$	95% CI
Periodic	square wave	**47.06**	[40.19, 53.77]
Periodic	sawtooth	11.43	[10.41, 12.37]
Periodic	sine	3.79	[3.04, 4.54]
Periodic	damped sine	2.59	[2.14, 3.01]
Periodic	seasonal	1.82	[1.03, 2.51]
Control	linear trend	4.00	[3.38, 4.62]
Control	white noise	2.45	[2.13, 2.80]
Control	constant	2.24	[1.77, 2.68]
Control	random walk	1.65	[0.83, 2.45]
Periodic mean		**13.34**	**[11.24, 15.42]**
Control mean		**2.59**	**[2.26, 2.88]**

broader 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 9. Layer 1 pair ablation on WikiText-103 by structural-repetition quintile. The monotonic trend indicates that the same pair used for periodic time-series prediction is especially important for repetitive language passages.

Quintile	Repetitiveness range	Mean $\Delta\mathcal{L}$
Q1 (least)	[14, 62]	3.83
Q2	[63, 87]	4.02
Q3	[88, 113]	4.05
Q4	[114, 161]	4.08
Q5 (most)	[162, 1597]	**4.37**

The most-degraded passages ($\Delta\mathcal{L} = 8$–9; Table 10) 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, disentangling the circuit's role in general sequence modeling from periodicity-specific computation requires further investigation.

Table 10. 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
8.88	"… Preston winger Will Hayhurst, a Republic of Ireland under-21 internatio
8.88	"… with the NHK Symphony Orchestra, but cancelled both deals upon Mw quit…"
8.88	"… his/her final score on the song, with money being awarded in Guitar He
8.75	"… Viscount, sets a task for each stage. This task must be completed before
8.50	"… The player character is Michel Ardan, an eccentric and intrepid French
8.38	"… Best Music Video, Long Form. In 1998, the categories were retitled Bes
8.31	"… on the Billboard Hot 100 twenty-nine, the highest U.S. entry among all
8.31	"… long run, with The A.V. Club attributing much of the show's early succe
8.25	"… Yankovic recorded 'Here's Johnny', a parody of 'Who's Johnny' by El
8.19	"… The music video of 'Crazy in Love', released in May 2003, was directe
8.19	"… Finishing with the worst record in the NHL, Columbus had the best cha
8.13	"… while one in the Pyramid Texts says the name is based on words shoute
8.06	"… the album Daydream most resembles in its emphasis on R&B grooves. Day'…"
8.00	"… similar to Konami's Guitar Freaks and to a lesser extent Harmonix's pre
8.00	"… he 'hit the wall with play-along music games', and challenged the game
7.94	"… saying that he 'was upset. But when you see the talent that was there, it
7.94	"… Flags indicate national team as defined under FIFA eligibility rules. Pla
7.94	"… at the Television Critics Association summer media tour in Beverly Hill
7.94	"… co-wrote and produced a song with Kenneth 'Babyface' Edmonds, with
Top 10 least degraded	
0.83	"… Porto e Santa Rufina; Sub-dean of the Sacred College of Cardinals; pref
1.48	"… Cardinal-Priest of SS. Giovanni e Paolo; Grand penitentiary; prefect of
1.59	"… From the Jewish Question to the Jewish State: An Essay on the Theory
1.77	"… called Saprang's transfer a 'demotion' and a 'punishment.' However, Sa
1.82	"… the Society of Jesus, provided he observed the canon law; and that it wa
1.89	"… the abnormal excess of white blood cells in people with the clinical syn
1.89	"… The protagonist sounds like a 'colonial administrator', and his reference
1.94	"… Mbaruk's nephew, Mbaruk bin Rashid, refused to acknowledge the appo
1.98	"… 00 works were published underground over the course of the war. Litera
2.02	"… Although the poem was defended by a few critics, E.C. Pettet returned t