SCRYER: A SCALABLE FRAMEWORK FOR FORECASTING NEURAL POPULATION ACTIVITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural forecasting requires identifying which co-recorded neurons provide useful information about a target's future activity, despite changes in neuronal identity, anatomical coverage, and population dynamics across recordings. We introduce SCRYER, a retrieval-conditioned forecasting architecture that gives each target direct access to individually represented source neurons. The model combines learned identity embeddings with current activity features to retrieve neuronal information, then uses that information to weight candidate forecasts generated from the target's history. We train the architecture using standard forecasting objectives and evaluate it on multi-region electrophysiology from the IBL Brain-Wide Map and calcium-imaging recordings from mice, *C. elegans*, and zebrafish. In addition to providing high-quality forecasts that outperform previous methods, we also show that the embeddings learned by our model have meaningful structure that reflects the underlying brain region the neuron comes from. These results support neuron-level retrieval as an effective mechanism for forecasting heterogeneous neural populations and learning anatomically informative representations.

1 INTRODUCTION

Neural activity emerges from interactions among neurons: each neuron's response reflects its own dynamics together with inputs from the surrounding network. A neuron's recent history therefore provides only a partial description of the processes shaping its future (Shenoy & Carmena, 2014; Vyas et al., 2020). Forecasting offers a way to learn these dependencies from recordings themselves, testing whether a model can use activity in other neurons to improve predictions of a target. Crucially, this requires identifying which recorded neurons provide useful information for that prediction. Learning these predictive relationships provides supervision without behavioral or anatomical labels and could support representations that capture the organization of neural activity.

Identifying these predictive relationships is challenging across heterogeneous recordings. Multi-region electrophysiology combines neurons from distinct anatomical structures and different probe locations (IBL et al., 2025), while calcium-imaging datasets span distinct Cre-lines that sample different neuronal populations (de Vries et al., 2020). Across sessions, changes in neuron identity and regional coverage alter which sources of information are available (Azabou et al., 2023; Dabagia et al., 2023). Moreover, a neuron that helps predict one target may provide little information about another, and its relevance may change with current activity. A shared forecasting model must thus learn where to obtain predictive information without assuming a fixed set of neurons or relationships between them.

Existing approaches typically combine neuronal signals before determining their influence on a target's forecast. For instance, NDT and the NDT-based implementations of MtM process tokens containing activity from multiple neurons (Ye & Pandarinath, 2021; Zhang et al., 2024). POCO aggregates activity into shared latent tokens, which each neuron queries to obtain feature-modulation parameters (Duan et al., 2025). Because of the mixing of neurons early before training, existing forecasting architectures do not allow for individual neurons to query each other's history. Consequently, they do not explicitly select which neurons to obtain information from for each forecast. This leaves a gap between learning a representation of the recorded population and directly retrieving predictive evidence from particular neurons. With such a capability, we would be able to potentially access neuron-to-neuron relationships that help to predict the future.

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

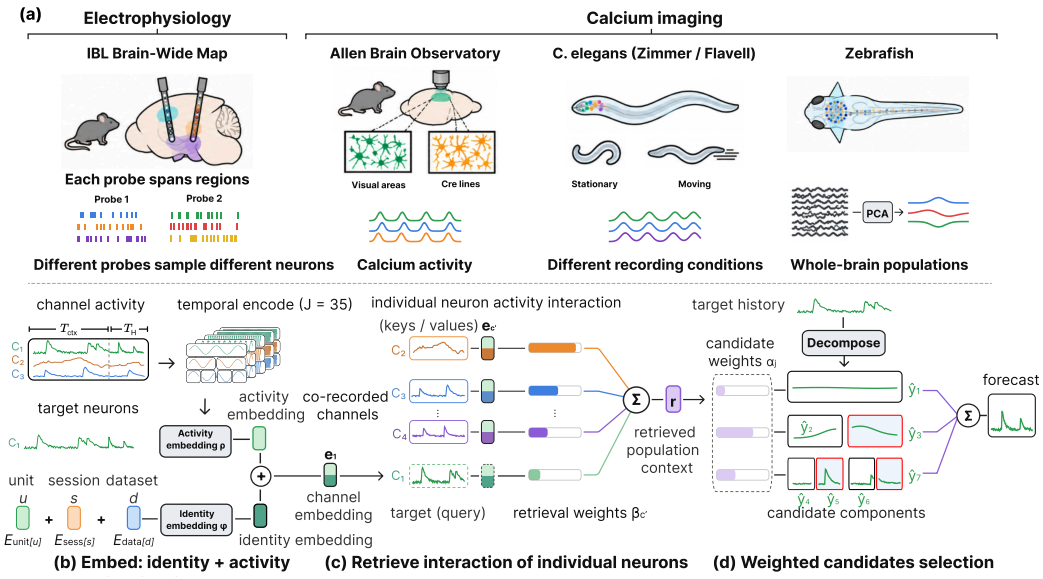


Figure 1: **Datasets and the SCRYER architecture. (a)** Training data span diverse species, modalities, and recording populations. **(b)** Each neuron is embedded from its current activity and learned unit, session, and dataset identity. **(c)** Population attention integrates information across co-recorded neurons. **(d)** Multiscale temporal components generate candidate forecasts, which are weighted using the population-informed context to produce the final prediction.

To address this limitation, we introduce SCRYER (**S**electing **C**andidate fo**R**ecasts b**Y** population-cont**E**xt **R**etrieval), an architecture that retrieves information from individual co-recorded neurons to guide each target's forecast. A hierarchical temporal encoder decomposes the target's history into localized signals and generates candidate forecasts from them. In parallel, SCRYER combines learned identity embeddings with current activity features to represent each neuron. The target uses these representations to retrieve a weighted summary of information from co-recorded neurons. The retrieved context determines how strongly each candidate forecast contributes to the final prediction (Figure 1). This connects the selection of informative source neurons with the combination of forecasts derived from different parts of the target's history.

We test SCRYER on multi-region electrophysiology from the IBL Brain-Wide Map using Brain-WideBench (IBL et al., 2025; Andre et al., 2026), calcium imaging from more than 1,300 Allen Brain Observatory sessions in mice, and recordings from *C. elegans* and zebrafish. Across these diverse datasets, SCRYER achieves state-of-the-art forecasting performance, with further improvements as the pretraining corpus grows. Because retrieval uses neuron-level representations to determine which recorded neurons provide useful predictive information, we ask whether these representations also capture distinctions associated with anatomical origin. The learned embeddings surpass all comparable transductive methods on brain-region prediction and support strong decoding even with linear probes, despite receiving no anatomical labels during forecasting training. These findings indicate that representations learned for predictive retrieval retain accessible anatomical structure. Finally, retrieval ablations assess its contribution to forecasting, while weight visualizations identify the source neurons emphasized for individual predictions.

Our contributions are:

- We introduce neuron-level retrieval for forecasting, allowing each target neuron to query its co-recorded neurons individually, using both identity and current activity.
- We develop SCRYER, which uses retrieved neuronal information to weight candidate forecasts derived from the target's history, and demonstrate forecasting improvements across electrophysiology and calcium-imaging datasets.
- We examine the contribution of retrieval and the information learned through forecasting using ablations, source-neuron visualizations, and transductive brain-region prediction, where SCRYER outperforms all compared methods.

- By selectively masking a neuron's own history or its retrieved population context, we find that neurons span a continuum from population-driven to history-driven prediction.

2 Related Work

Forecasting models of neural activity. Prediction has long served as an organizing principle for understanding neural computation. The idea that neural circuits anticipate future inputs or internal states motivates a range of theoretical and empirical frameworks (Shenoy & Carmena, 2014; Vyas et al., 2020). This has motivated the development of a number of state-space models for neural population data, including LFADS and piecewise-linear recurrent models, which describe population trajectories as arising from an evolving low-dimensional dynamical system (Pandarinath et al., 2018; Linderman et al., 2016; Durstewitz, 2017; Brenner et al., 2022). While instructive in many settings, these approaches mix population signals before any neuron-specific computation, and are typically applied to single sessions at a time.

More expressive architectures and learning objectives have therefore been explored: masked neural models (Ye et al., 2023) and multi-task masking (Zhang et al., 2024), which includes forward prediction among its objectives, support this regime at scale. These NDT-based architectures process population vectors or multi-neuron patches, again mixing signals early, which means individual neurons cannot directly query each other's history. POCO takes a different approach for calcium imaging, encoding population context with a POYO-style encoder to produce normalization parameters for a univariate forecaster via FiLM conditioning (Duan et al., 2025). This preserves a neuron-level interface while allowing population context to shape each target's forecast; however, the contribution of individual neurons is obscured. Instead, SCRYER makes neuron-level retrieval explicit: rather than conditioning through a shared population summary, each target neuron selects specific co-recorded sources using identity- and activity-dependent representations, and uses the retrieved information to score and combine candidate forecasts derived from its own history.

Learning shared structure across neural recordings. A central challenge in building models of neural activity at scale is that recordings from different individuals, sessions, and brain regions contain distinct neurons with no known correspondence (Azabou et al., 2023). Early neural population models were designed for single sessions, learning latent dynamical descriptions of shared population trajectories (Linderman et al., 2016; Pandarinath et al., 2018; Durstewitz, 2017; Brenner et al., 2022). Extending these models across recordings requires mechanisms that do not assume a fixed set of neurons or fixed relationships between them.

Unit identity embeddings and flexible tokenization provide one path forward, by enabling joint training across heterogeneous datasets (Ye et al., 2023; Azabou et al., 2023; Zhang et al., 2024). Representations can also be aligned across sessions through the latent space directly, mapping distinct recordings into a shared geometry without neuron correspondence (Dabagia et al., 2023; Karpowicz et al., 2025). Recent models have pushed further, demonstrating transfer across brain regions, cellular sub-types, and behavioral tasks by training on large-scale imaging datasets, where shared structure emerges even across heterogeneous circuits (Azabou et al., 2024). SCRYER builds on these developments but frames the cross-recording problem at the level of individual neurons rather than population summaries: learned unit, session, and dataset embeddings combine with current activity to compute target-specific retrieval weights over source neurons, so the model determines which recorded neurons provide useful predictive information for each target and observation window without assuming fixed inter-neuron relationships.

3 Methods

SCRYER identifies which portions of a neuron's history and which co-recorded neurons provide useful information for forecasting. To do this, each target retrieves information from the recorded population and uses it to combine its candidate predictions (Figure 1).

3.1 Problem Formulation

A recording session s contains C_s simultaneously recorded neurons (channels). Each channel corresponds to a spike-sorted unit in electrophysiology or a segmented cell in calcium imaging.

Given a context window $X \in \mathbb{R}^{C_s \times T_{\text{ctx}}}$, we predict activity over the subsequent forecast horizon $Y \in \mathbb{R}^{C_s \times T_{\text{H}}}$. The number and identity of recorded neurons vary across sessions, so the model must accommodate different populations without requiring correspondence between their neurons.

Each neuron has a unit identifier u, a session identifier s, and dataset identifier d. Learned embeddings of these identifiers provide recording-specific information; all other model parameters are shared across sessions. For calcium imaging, the model predicts continuous fluorescence activity. For electrophysiology, it predicts the log expected spike count in each forecast bin. Below, we describe one target neuron and one context window, suppressing these indices unless needed.

3.2 Encoding the Observed History

Different portions of a neuron's history may contribute different information about its future activity. We therefore decompose the context into nested temporal segments, giving the model access to both longer intervals and shorter, localized periods of activity. Within each segment, we then apply further frequency decomposition to separate slower variations from faster changes.

Let $v_1, \ldots, v_J$ denote the resulting signals, where each v_j corresponds to one temporal segment and frequency band. A learned scorer weights the signals within each segment:

$$w_j = \frac{\exp f(v_j)}{\sum_{j' \in \text{seg}(j)} \exp f(v_{j'})}, \qquad z_j = w_j v_j, \tag{1}$$

where $\text{seg}(j)$ denotes the frequency components belonging to the same temporal segment as j. Each z_j thus retains a weighted view of activity within a particular portion and frequency band of the observed history. This decomposition serves two roles. The weighted signals z_j provide inputs to candidate forecasting networks. In addition, summary statistics of the unweighted signals describe current activity for neuronal retrieval: $a(x) = \text{stats}(v_1, \ldots, v_J)$. These statistics summarize activity level, variability, peaks, and roughness. The temporal partition, frequency decomposition, and statistic definitions are specified in Appendix A.2.

3.3 Retrieving Context from Co-Recorded Neurons

A target's own history provides only part of the information available for prediction. Co-recorded neurons can supply additional context, but their relevance depends on the target and the current activity. We therefore retain an individual representation of each source neuron and learn target-specific retrieval over these representations. Each neuron receives an embedding that combines learned identity information with its activity descriptor:

$$e = \underbrace{\phi(E_{\text{unit}}[u] + E_{\text{sess}}[s] + E_{\text{data}}[d])}_{\text{identity}} + \underbrace{\rho(a(x))}_{\text{activity}}. \tag{2}$$

The unit lookup table assigns a learned vector to each recorded neuron, while the session vector is shared by neurons recorded together. These identifiers contain no anatomical annotations. The mappings ϕ and ρ project both terms into a common D-dimensional space. The identity term remains fixed across observation windows, while the activity term changes with the neuron's observed history. The target uses its embedding to query the neurons recorded in the same session:

$$\beta_{c'} = \text{softmax}_{c'}\left(\frac{\langle W_q e, W_k e_{c'} \rangle}{\sqrt{D}}\right), \qquad r = \sum_{c'=1}^{C_s} \beta_{c'} W_v e_{c'}. \tag{3}$$

Here, e represents the target, $e_{c'}$ represents source neuron c', and W_q, W_k, and W_v are shared linear projections. The source set includes the target itself. Attention operates over individual neurons, producing a retrieved context r for each target. Because queries and keys include both identity and activity, retrieval can vary across targets and observation windows. The weights $\beta_{c'}$ receive no direct supervision; the forecasting objective trains them to retrieve information useful for prediction. Figure 2 illustrates the source weights for one target.

3.4 Retrieval-Conditioned Forecast Selection

The model uses retrieved neuronal information to determine how different portions of the target's history contribute to its forecast. Each weighted signal z_j first generates a candidate prediction over

the full horizon:

$$\widehat{y}_j = h_j(z_j) \in \mathbb{R}^{T_H}, \qquad j = 1, \ldots, J. \tag{4}$$

The component-specific networks h_j share parameters across neurons and recordings. Although their inputs describe different temporal segments and frequency bands, all candidates predict the same future activity window.

The retrieved context r then scores these candidates to determine their mixture weights:

$$\alpha_j = \mathrm{softmax}_j \left(\frac{\langle W_a \widehat{y}_j, r \rangle}{\sqrt{D}} \right), \; \widehat{y} = \sum_{j=1}^{J} \alpha_j \widehat{y}_j + \delta. \tag{5}$$

The projection W_a maps candidate trajectories into the retrieval space. The additive correction δ is conditioned on retrieved context, identity, and horizon position. Network architectures are provided in Appendix A. The target's history determines the candidate predictions, while retrieved neuronal information determines their mixture weights and supplies the correction. The temporal weights w_j act within the target's history; the forecast weights α_j connect that history to information from the recorded population.

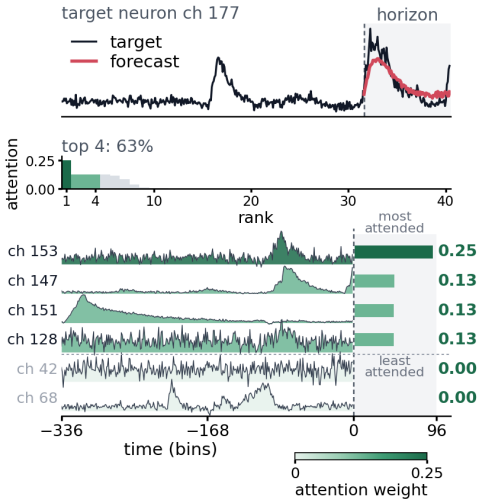


Figure 2: **Retrieval weights for one target neuron.** Top: the target and its forecast (red). Below: co-recorded neurons ordered by their attention weights in Eq. (3), computed from the context alone. In this example, sources with a transient in the context receive the largest weights.

3.5 Training Objectives

We use the same architecture across modalities and match the output parameterization and objective to the measured signal. For calcium imaging, $\widehat{y}_t$ predicts z-scored $\Delta F/F$, the change in fluorescence relative to baseline. For electrophysiology, $\widehat{y}_t$ represents the log expected spike count, so $\lambda_t = \exp(\widehat{y}_t)$ is the Poisson mean. The per-neuron forecasting loss is

$$\mathcal{L} = \sum_{t=1}^{T_H} \begin{cases} \exp(\widehat{y}_t) - y_t \widehat{y}_t, & \text{spike counts,} \\ (\widehat{y}_t - y_t)^2, & \text{calcium fluorescence.} \end{cases} \tag{6}$$

The spike-count objective is the Poisson negative log-likelihood. The calcium objective is squared error, corresponding to a Gaussian observation model with fixed variance. For spike counts, candidate combination operates in log-count space, and exponentiation gives the predicted expected counts. We train the temporal encoder, identity embeddings, retrieval mechanism, and forecasting networks jointly through this objective, without event labels or anatomical supervision.

4 Results

We evaluate whether retrieval-conditioned forecasting can accommodate heterogeneity both within neural populations and across recordings. The experiments address three questions: whether the same architecture supports forecasting in the two recording modalities considered here, whether cross-recording pretraining improves prediction, and whether forecasting representations retain information about neuronal anatomy.

4.1 Evaluation Setup

Spiking data are Neuropixels recordings from the IBL Brain-Wide Map (IBL et al., 2025), evaluated through BrainWideBench (Andre et al., 2026) with 20-ms bins, 900 ms of context, and a 100-ms horizon. Calcium data are the Allen Visual Coding dataset (de Vries et al., 2020), 1,334 sessions of mouse visual cortex across six areas and 14 Cre-line groups, of which we use a 200-session cohort stratified by area and Cre line; two whole-brain *C. elegans* datasets; and a larval zebrafish dataset.

We use benchmark-defined temporal splits for IBL and chronological splits for the calcium imaging datasets (Appendix B).

For spikes we report Poisson deviance explained (D^2) and bits per spike (bps), both relative to a constant-rate forecast. For calcium we report MSE and MAE; traces are z-scored per neuron, so an MSE of 1 corresponds to predicting the mean. Region decoding (Sec. 4.3) is scored by macro-F1. Firing regularity (Sec. 4.5) is measured by CV2 (Holt et al., 1996), the local coefficient of variation of interspike intervals, which is 0 for regular firing and near 1 for Poisson-like firing. Metrics are computed per recording and averaged. On IBL we compare with the published BrainWideBench baselines in both the single-session and the pretrained setting; on calcium with NetFormer (Lu et al., 2025), NLinear (Zeng et al., 2023), leaky PLRNN (Brenner et al., 2022), TexFilter (Yi et al., 2024), and POCO (Duan et al., 2025), all trained using the same data, splits, and parameters (Appendix C.2).

Pretraining and finetuning. For IBL, we follow the benchmark training and evaluation protocols. We pretrain SCRYER on 423 recordings from 126 mice, and evaluate on the 29 held out sessions from 13 other mice, so evaluation data are unseen at both the session and the animal level. As no evaluation unit or session exists in pretraining, the unit and session embedding tables are discarded and re-initialized from $\mathcal{N}(0, 0.02)$ for each evaluation recording; all other parameters are transferred from the pretrained model. For forecasting (TS2) all parameters are then finetuned on the training split; for region decoding (TS3) only the two new embedding tables are fit and the rest stays frozen (Sec. 4.3). Single-session and pretrained models are evaluated on the same data and splits (Appendix C).

4.2 FORECASTING ACROSS DIVERSE NEURAL RECORDINGS

Multi-region electrophysiology. Multi-region recordings combine neurons with different firing statistics and temporal dynamics, while variable regional coverage changes the population context available for prediction. This tests whether SCRYER can identify context relevant to each target across heterogeneous populations. We evaluate future spike-count prediction using BrainWideBench's forecasting task (TS2), comparing single-session training with cross-recording pretraining followed by target-recording finetuning (Table 1).

With single-session training, SCRYER achieves a D^2 of 0.166 ± 0.001 and 0.351 ± 0.002 bps, the highest among single-session methods on both metrics. It exceeds the strongest single-session baselines by 0.008 in D^2 (NDT and POCO) and 0.003 in bps (NDT), and also outperforms pretrained MtM. Pretrained NDT-Stitch and POCO score higher, indicating that other recordings supply information that a single session lacks. With cross-recording pretraining (Section 4.1), SCRYER achieves the highest D^2 (0.185 ± 0.001) and bps (0.389 ± 0.001) among all compared methods, exceeding NDT-Stitch by 0.008 and 0.005, respectively. Relative to single-session training, pretraining improves D^2 by 0.019 (11.4%) and bps by 0.038 (10.8%), with consistent gains on both metrics. SCRYER is thus the strongest model under both training regimes, and its population retrieval benefits from structure shared across recordings.

Calcium imaging recordings. We next test whether the same architecture extends to calcium fluorescence, where measurement dynamics and recorded populations differ from electrophysiology. We evaluate a 200-session Allen Brain Observatory cohort spanning six visual areas and 14 Cre-line groups (de Vries et al., 2020), alongside two *C. elegans* datasets and whole-brain zebrafish recordings.

SCRYER achieves the lowest mean MSE and MAE on all four datasets (Table 2). Relative to the strongest baseline on each dataset, MSE decreases by 1.6% on Allen, 2.6% on *C. elegans*–Zimmer, 1.4% on *C. elegans*–Flavell, and 3.4% on zebrafish. Within Allen, performance varies across genetically targeted populations (Table A3). Nr5a1-Cre and Vip-IRES-Cre have the lowest MSE (0.813 and 0.823), whereas Ntsr1-Cre_GN220 has the highest (0.977). For these z-scored traces, this corresponds approximately to 18–19% versus 2% variance explained, showing variation that the aggregate score obscures. Appendix D.2 examines this variation, and Appendix Figures A3 and A4 illustrate forecasts around calcium transients.

4.3 ANATOMICAL INFORMATION IN LEARNED REPRESENTATIONS

Neuron-level retrieval requires representations that distinguish sources of predictive information. We therefore ask whether the representations learned through forecasting also capture differences associated with anatomical origin. SCRYER receives no brain-region labels during forecasting

Table 1: **Forecasting performance on IBL BrainWideBench.** Pretrained models use 423 recordings from 126 mice before finetuning on new held-out animals. Forecasting performance is measured with Poisson D^2 and bits per spike (bps) on 29 recordings from 13 mice held out from pretraining. Baseline values are from BrainWideBench; SCRYER and POCO are mean ± SEM over five seeds.

	Method	D^2 (↑)	bps (↑)
Scratch	Autoencoder	0.037 ± 0.001	0.065 ± 0.001
	NDT	0.158 ± 0.000	0.348 ± 0.001
	LFADS	0.144 ± 0.000	0.325 ± 0.001
	POCO	0.158 ± 0.002	0.340 ± 0.002
	SCRYER (scratch)	$\mathbf{0.166 \pm 0.001}$	$\mathbf{0.351 \pm 0.002}$
Pretrain	NDT-Stitch	0.177 ± 0.000	0.384 ± 0.000
	MtM	0.131 ± 0.001	0.288 ± 0.001
	POCO	0.169 ± 0.002	0.360 ± 0.004
	SCRYER (pretrained + finetuned)	$\mathbf{0.185 \pm 0.001}$	$\mathbf{0.389 \pm 0.001}$

Table 2: **Forecasting performance across calcium-imaging datasets spanning different species and cell classes.** All methods use the same context and forecast lengths within each dataset: $(T_{\text{ctx}}, T_{\text{H}}) = (336, 96)$ for Allen Visual Coding, and $(T_{\text{ctx}}, T_{\text{H}}) = (48, 16)$ with multi-session training for *C. elegans*–Zimmer, *C. elegans*–Flavell, and Zebrafish–Ahrens. Results for $T_{\text{H}} = 48$ and for single-session training are reported in Appendix Table A2. Values are reported in their original units, without rescaling, as mean ± standard deviation across four training seeds. Lower is better. Bold and underline indicate the lowest and second-lowest mean errors, respectively.

	Allen Visual Coding		*C. elegans*–Zimmer		*C. elegans*–Flavell		Zebrafish–Ahrens (PCA)	
Method	**MSE ↓**	**MAE ↓**	**MSE ↓**	**MAE ↓**	**MSE ↓**	**MAE ↓**	**MSE ↓**	**MAE ↓**
NetFormer	0.998 ± 0.005	0.603 ± 0.006	0.460 ± 0.001	0.469 ± 0.001	0.590 ± 0.000	0.544 ± 0.000	55.35 ± 0.210	4.849 ± 0.004
NLinear	0.986 ± 0.000	0.614 ± 0.000	0.410 ± 0.000	0.437 ± 0.000	0.590 ± 0.000	0.545 ± 0.000	56.48 ± 0.006	4.868 ± 0.003
L-PLRNN	0.968 ± 0.000	0.593 ± 0.001	0.480 ± 0.005	0.488 ± 0.002	0.620 ± 0.004	0.561 ± 0.003	52.57 ± 0.435	4.966 ± 0.005
TexFilter	0.957 ± 0.023	0.601 ± 0.018	0.360 ± 0.000	0.404 ± 0.000	0.570 ± 0.001	0.526 ± 0.002	46.22 ± 0.055	4.444 ± 0.007
MLP	$\underline{0.946 \pm 0.012}$	0.596 ± 0.016	0.358 ± 0.001	0.407 ± 0.002	$\underline{0.554 \pm 0.003}$	$\underline{0.524 \pm 0.002}$	44.61 ± 0.069	4.320 ± 0.003
POCO	0.990 ± 0.0003	$\underline{0.590 \pm 0.0001}$	$\underline{0.350 \pm 0.003}$	$\underline{0.402 \pm 0.003}$	0.600 ± 0.023	0.556 ± 0.011	$\underline{39.68 \pm 0.198}$	$\underline{4.096 \pm 0.006}$
SCRYER	$\mathbf{0.931 \pm 0.001}$	$\mathbf{0.581 \pm 0.000}$	$\mathbf{0.341 \pm 0.002}$	$\mathbf{0.396 \pm 0.001}$	$\mathbf{0.546 \pm 0.008}$	$\mathbf{0.523 \pm 0.001}$	$\mathbf{38.32 \pm 0.274}$	$\mathbf{4.003 \pm 0.006}$

training; anatomical prediction provides a separate test of the information retained in its embeddings. We evaluate this using BrainWideBench TS3, which trains linear and MLP probes to predict Cosmos region labels at the single-unit and multi-unit levels. For recordings held out from pretraining, we adapt only the unit and session embeddings using the forecasting objective, keeping all shared model parameters fixed. Region labels are used only for the subsequent probes.

SCRYER outperforms all compared transductive methods (Fig. 3c). Its linear and MLP probes achieve macro-F1 scores of 0.240 and 0.240 at the single-unit level and 0.258 and 0.262 at the multi-unit level. NDT-Stitch achieves 0.136/0.167 and 0.155/0.187, respectively, while POCO achieves 0.075/0.084 and 0.065/0.083 under the same benchmark. The small difference between SCRYER's linear and MLP probes indicates that much of the anatomical information is accessible through a linear readout, without requiring a more expressive probe.

We further examine how this information is organized in the embedding space. A projection onto two discriminant axes groups pretraining units into three broad clusters: cortex and hippocampus, thalamus, and brainstem and cerebellum (Fig. 3a). A decoder evaluated on held-out recordings achieves 0.62 accuracy, compared with 0.19 for a majority-class baseline. After adapting their identity embeddings, units from the 13 held-out mice occupy similar portions of this space (Fig. 3b). Thalamus and hindbrain are recognized across all seeds, while finer distinctions within the broad clusters remain less reliable. Together, these results show that forecasting-trained representations retain anatomical information and that this organization extends to newly adapted recordings.

4.4 SCALING ANALYSIS

We next test whether pretraining on more neural recordings improves forecasting. We compare single-session training with pretraining on 40, 100, and all 423 BrainWideBench pretraining recordings, followed by finetuning on the same evaluation recordings. All settings use five random seeds. Figure 4 shows consistent improvements as the pretraining corpus grows. Pretraining on all 423 recordings improves D^2 by 11.4% and bits per spike by 10.8% relative to training from scratch, demonstrating

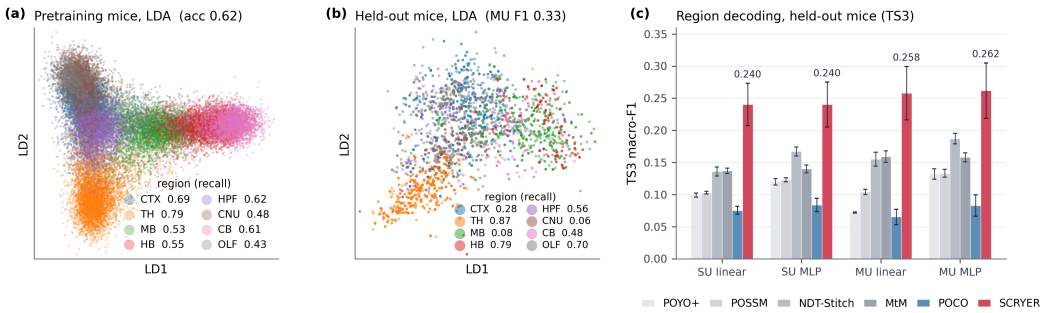


Figure 3: **Brain region is recoverable from SCRYER's learned representations. (a)** Pretraining mice: unit representations, learned without region labels, on two LDA axes fit with those labels (30,000 units, 8 Cosmos regions). **(b)** Held-out mice: identity embeddings of 3,209 units from the 13 TS3 mice, projected onto the axes of (a) without refitting. The lobes are preserved. Legend: per-region recall of the official probe. **(c)** TS3 macro-F1, single- and multi-unit, linear and MLP probes; five seeds per method.

that SCRYER benefits from learning across additional recorded populations. We report complementary scaling experiments on Allen calcium-imaging recordings in Appendix D.4.

4.5 POPULATION-DRIVEN NEURONS AND HISTORY-DRIVEN NEURONS

In our model, the representations learned through forecasting distinguish neurons from different brain regions. This raises the question of what differs between regions. One possibility is fixed properties of each neuron, stored in its identity embedding (Eq. (2)). The other possibility is that the representation captures how a neuron's activity is generated: whether its upcoming activity follows its history or the current activity of the surrounding population. To test if the model has learned this from activity, we mask one source of activity at a time, either the neuron's history or the retrieved population context, and measure how much of the forecast survives.

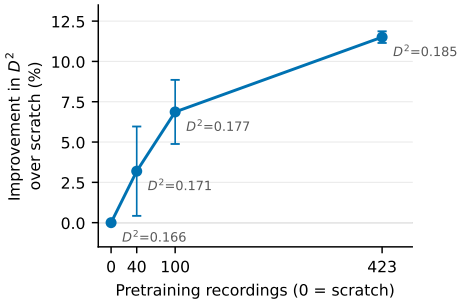


Figure 4: **Scaling with pretraining corpus size on IBL.** Improvement in Poisson D^2 over single-session training ($D^2 = 0.166$; Table 1) across 29 TS2 evaluation recordings after pretraining on 40, 100, or all 423 recordings and target-recording finetuning. Point labels report absolute D^2; error bars show standard deviations across five seeds.

Masking the neuron's own history, by replacing its spike column in the context with its mean rate, gives a *population-only* forecast; masking the population, by replacing the population context r of Eq. (3) with its average over windows, gives a *history-only* forecast; identity is kept in both, so any difference between the two comes from activity. For each neuron we compare the share of the full model's D^2 that each masked forecast retains, negative Δ means the neuron is predicted from its population, positive from its own past (2,539 held-out units, 29 recordings, 13 mice; Figure 5).

This analysis reveals that neurons span a continuum from population-driven to history-driven prediction (Fig. 5), where population-driven neurons retain much of the full model's accuracy when their own history is masked but lose accuracy when population context is removed while history-driven neurons show the reverse. Across units, Δ increases with firing rate (Spearman $\rho = 0.39$) and decreases with CV2 ($\rho = -0.34$). History-driven neurons in CB, MB, and HB tend to fire faster and more regularly, making their recent activity informative about subsequent firing, whereas population-driven HPF and OLF neurons tend to fire more sparsely, with examples showing population activity preceding changes in the target neuron. These differences partly align with anatomical decoding (Fig. 3): HPF and OLF, which favor population context, and HB and CB, which favor history, are among the better-decoded regions, whereas CTXsp, HY, and CNU show greater within-region variation in Δ and lower decoding performance. This correspondence suggests that learning how individual history and co-recorded activity support prediction helps organize the embeddings around region-associated dynamics, without anatomical supervision.

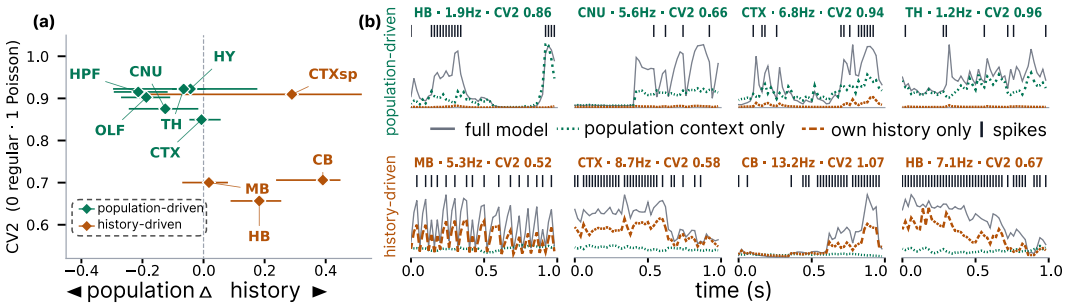


Figure 5: **Population-driven and history-driven neurons.** $\Delta = D^2_{\text{history}}/D^2_{\text{full}} - D^2_{\text{pop}}/D^2_{\text{full}}$, with D^2 as in Sec. 4.1; *full* is the trained model, *pop* the same model with the target's own history masked, *history* the same model with the population context masked (Sec. 4.5). Green, $\Delta < 0$: predicted from the population; orange, $\Delta > 0$: predicted from own history. CV2 measures the local variability of interspike intervals, from 0 for regular firing to about 1 for Poisson-like firing. **(a)** Regional median Δ against regional median CV2; error bars, 95% bootstrap intervals over units; dashed line, $\Delta = 0$. $n = 2{,}539$ units, 29 recordings, 13 mice. **(b)** Predicted firing rate over 1 s of held-out data from the full model (gray), population-only (green dotted), and history-only (orange dashed); ticks mark observed spikes.

4.6 Ablations

We remove or replace model components and retrain each variant from scratch to assess the contributions of temporal encoding, neuronal retrieval, and identity conditioning (Table A4; Appendix D.3). The results show a consistent benefit from combining temporal structure with neuronal context. Retaining only the frequency encoder produces the worst performance on every dataset: MSE increases from 0.341 to 0.391 on *C. elegans* and from 38.3 to 45.3 on zebrafish, while IBL D^2 drops from 0.1654 to 0.0857. Removing multiscale tokens or channel self-attention also degrades all three datasets, supporting both temporal organization and retrieval from co-recorded neurons. On IBL, removing retrieval reduces D^2 to 0.1409 and bps to 0.3032, while removing identity conditioning produces the largest individual decline ($D^2 = 0.0996$). The remaining ablations assess implementation choices: FiLM, compression, the horizon decoder, and the choice of frequency decomposition. These components improve calcium forecasting, while their removal or replacement changes IBL D^2 by at most 0.0010, within the reported seed-to-seed variation. Together, the ablations support the combination of temporal encoding and neuron-level retrieval, with additional gains from identity conditioning and dataset-specific refinements.

5 Discussion

We introduced SCRYER, a retrieval-conditioned architecture in which identity- and activity-dependent representations select which co-recorded neurons provide context, and that context weights candidate forecasts derived from each target's own history. Across electrophysiology and calcium imaging, this design improves forecasting under standard objectives, without auxiliary event supervision. The model is trained only to predict future activity and never sees anatomical labels, yet the representations it learns carry information about brain-region decoding.

These results have limits. Identity embeddings must still be learned for each new recording, and what a neuron can retrieve depends on which other neurons were recorded. Predicting a region is not explaining it: the embeddings may reflect firing rates or other properties that differ between regions, and retrieval weights do not show causal interactions. Finally, the architecture is shared across electrophysiology and calcium imaging, but we did not transfer one pretrained model between them.

A natural next step is to extend retrieval across many sessions at once, so that a neuron can draw context not only from the neurons recorded with it but also from similar neurons in other animals. This would reduce the dependence on which neurons are recorded, and with more neurons per region the model may resolve the balance between own history and population state more clearly. Relating this balance to cell type, layer and recording depth would then test whether it reflects biological differences, and combining these analyses with behavior and perturbation experiments could connect what the model learns to hypotheses about how neural populations are organized.

6 AI DISCLOSURE

In this work, we used generative AI tools to aid and polish writing, for retrieval and discovery of related work, and for writing and debugging experiment code, including baseline reproduction scripts, cluster job scripts, evaluation utilities, figure-generation scripts, and the scripts that aggregate per-seed results and compute the reported statistics and confidence intervals. We have not used generative AI tools for proving mathematical claims or for designing the SCRYER architecture; the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to audit consistency between the manuscript and the implementation (e.g., checking that equations, evaluation protocols, and hyperparameters stated in the text match the code and the run configurations) and to cross-check reported tables against raw per-seed logs and checkpoints. We have reviewed all AI-assisted work. Every reported number was produced by running the code on our own hardware and is traceable to raw logs; benchmark results follow the official evaluation scripts without modification. AI-assisted experiment code was reviewed by the author(s), and baseline reproductions were checked against the official implementations and their published hyperparameters. Statistics were recomputed directly from per-seed results, and AI-suggested text edits were checked against the implementation before inclusion. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

REFERENCES

Alexandre Andre, Shivashriganesh P. Mahato, Vinam Arora, Keshav Balaji, Divyansha Lachi, Nanda H. Krishna, Jingyun Xiao, Yizi Zhang, Ximeng Mao, Wenrui Ma, Han Yu, International Brain Laboratory, Daniel Birman, Niccolò Bonacchi, Gaelle A. Chapuis, Joana A. Catarino, Felicia Davatolhagh, Mayo Faulkner, Laura Freitas-Silva, Fei Hu, Julia M. Huntenburg, Anup Khanal, Inês Laranjeira, Petrina Lau, Guido T. Meijer, Nathaniel J. Miska, Jean-Paul Noel, Alejandro Pan-Vazquez, Georg Raiser, Cyrille Rossant, Karolina Z. Socha, Anne E. Urai, Miles J. Wells, Steven J. West, Olivier Winter, Blake Richards, Guillaume Lajoie, Cole Hurwitz, Mehdi Azabou, Matthew R. Whiteway, Liam Paninski, and Eva L. Dyer. Brainwidebench: Benchmarking large-scale pretraining and across-animal transfer in multi-region neural recordings, 2026. URL https://arxiv.org/abs/2609.22064.

Alexandru A. Atanas, Jiyoon Kim, Zhi Wang, Eric Bueno, Max Becker, Dae Kang, Joon Park, Taylor S. Kramer, Franklin K. Wan, Sibel Baskoylu, et al. Brain-wide representations of behavior spanning multiple timescales and states in *C. elegans*. *Cell*, 186(19):4134–4151, 2023.

Mehdi Azabou, Vinam Arora, Venkataramana Ganesh, Ximeng Mao, Santosh Nachimuthu, Michael Mendelson, Blake Richards, Matthew Perich, Guillaume Lajoie, and Eva Dyer. A unified, scalable framework for neural population decoding. *Advances in Neural Information Processing Systems*, 36:44937–44956, 2023.

Mehdi Azabou, Krystal Xuejing Pan, Vinam Arora, Ian Jarratt Knight, Eva L Dyer, and Blake Aaron Richards. Multi-session, multi-task neural decoding from distinct cell-types and brain regions. In *The Thirteenth International Conference on Learning Representations*, 2024.

Michael Brenner, Fabian Hess, John M. Mikhaeil, Lukas F. Bereska, Zohreh Monfared, Po-Chun Kuo, and Daniel Durstewitz. Tractable dendritic RNNs for reconstructing nonlinear dynamical systems. In *International Conference on Machine Learning*, pp. 2292–2320. PMLR, 2022.

Xiuye Chen, Yu Mu, Yu Hu, Aaron T Kuan, Maxim Nikitchenko, Owen Randlett, Alex B Chen, Jeffery P Gavornik, Haim Sompolinsky, Florian Engert, et al. Brain-wide organization of neuronal activity and convergent sensorimotor transformations in larval zebrafish. *Neuron*, 100(4):876–890, 2018.

Max Dabagia, Konrad P Kording, and Eva L Dyer. Aligning latent representations of neural activity. *Nature Biomedical Engineering*, 7(4):337–343, 2023.

Saskia EJ de Vries, Jerome A Lecoq, Michael A Buice, Peter A Groblewski, Gabriel K Ocker, Michael Oliver, David Feng, Nicholas Cain, Peter Ledochowitsch, Daniel Millman, et al. A large-scale standardized physiological survey reveals functional organization of the mouse visual cortex. *Nature neuroscience*, 23(1): 138–151, 2020.

Yu Duan, Hamza Tahir Chaudhry, Misha B Ahrens, Christopher D Harvey, Matthew G Perich, Karl Deisseroth, and Kanaka Rajan. Poco: Scalable neural forecasting through population conditioning. *arXiv preprint arXiv:2506.14957*, 2025.

Daniel Durstewitz. A state space approach for piecewise-linear recurrent neural networks for identifying computational dynamics from neural measurements. *PLoS computational biology*, 13(6):e1005542, 2017.

Yu Fu, Jason M Tucciarone, J Sebastian Espinosa, Nengyin Sheng, Daniel P Darcy, Roger A Nicoll, Z Josh Huang, and Michael P Stryker. A cortical circuit for gain control by behavioral state. *Cell*, 156(6):1139–1152, 2014.

Gary R Holt, William R Softky, Christof Koch, and Rodney J Douglas. Comparison of discharge variability in vitro and in vivo in cat visual cortex neurons. *Journal of neurophysiology*, 75(5):1806–1814, 1996.

IBL, Brandon Benson, Julius Benson, Daniel Birman, Niccolò Bonacchi, Matteo Carandini, Joana A Catarino, Gaelle A Chapuis, Anne K Churchland, Yang Dan, Peter Dayan, Eric EJ DeWitt, Tatiana A Engel, Michele Fabbri, Mayo Faulkner, Ila Rani Fiete, Charles Findling, Laura Freitas-Silva, Berk Gerçek, Kenneth D Harris, Michael Häusser, Sonja B Hofer, Fei Hu, Félix Hubert, Julia M Huntenburg, Anup Khanal, Christopher Krasniak, Christopher Langdon, Petrina Y P Lau, Zachary F Mainen, Guido T Meijer, Nathaniel J Miska, Thomas D Mrsic-Flogel, Jean-Paul Noel, Kai Nylund, Alejandro Pan-Vazquez, Alexandre Pouget, Cyrille Rossant, Noam Roth, Rylan Schaeffer, Michael Schartner, Yanliang Shi, Karolina Z Socha, Nicholas A Steinmetz, Karel Svoboda, Anne E Urai, Miles J Wells, Steven Jon West, Matthew R Whiteway, Olivier Winter, and Ilana B Witten. A brain-wide map of neural activity during complex behaviour. *Nature*, 645 (8079):177–191, 2025. URL https://www.nature.com/articles/s41586-025-09235-0.

Brianna M Karpowicz, Yahia H Ali, Lahiru N Wimalasena, Andrew R Sedler, Mohammad Reza Keshtkaran, Kevin Bodkin, Xuan Ma, Daniel B Rubin, Ziv M Williams, Sydney S Cash, et al. Stabilizing brain-computer interfaces through alignment of latent dynamics. *Nature Communications*, 16(1):4662, 2025.

Saul Kato, Harris S Kaplan, Tina Schrödel, Susanne Skora, Theodore H Lindsay, Eviatar Yemini, Shawn Lockery, and Manuel Zimmer. Global brain dynamics embed the motor command sequence of caenorhabditis elegans. *Cell*, 163(3):656–669, 2015.

Scott W Linderman, Andrew C Miller, Ryan P Adams, David M Blei, Liam Paninski, and Matthew J Johnson. Recurrent switching linear dynamical systems. *arXiv preprint arXiv:1610.08466*, 2016.

Z. Lu, W. Zhang, T. Le, H. Wang, U. Sümbül, E. T. SheaBrown, and L. Mi. Netformer: An interpretable model for recovering dynamical connectivity in neuronal population dynamics. OpenReview, 2025. URL: https://openreview.net/forum?id=bcTjW5kS4W.

Chethan Pandarinath, Daniel J. O'Shea, Joshua Collins, Rafal Jozefowicz, Sergey D. Stavisky, Jonathan C. Kao, Eric M. Trautmann, Matthew T. Kaufman, Stephen I. Ryu, Leigh R. Hochberg, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, 2018.

Krishna V Shenoy and Jose M Carmena. Combining decoder design and neural adaptation in brain-machine interfaces. *Neuron*, 84(4):665–680, 2014.

Saurabh Vyas, Matthew D Golub, David Sussillo, and Krishna V Shenoy. Computation through neural population dynamics. *Annual review of neuroscience*, 43(1):249–275, 2020.

Joel Ye and Chethan Pandarinath. Representation learning for neural population activity with neural data transformers. *arXiv preprint arXiv:2108.01210*, 2021.

Joel Ye, Jennifer Collinger, Leila Wehbe, and Robert Gaunt. Neural data transformer 2: multi-context pretraining for neural spiking activity. *Advances in Neural Information Processing Systems*, 36:80352–80374, 2023.

K. Yi, J. Fei, Q. Zhang, H. He, S. Hao, D. Lian, and W. Fan. Filternet: Harnessing frequency filters for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 37, pp. 55115–55140, 2024.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11121–11128, 2023.

Yizi Zhang, Yanchen Wang, Donato M Jiménez-Benetó, Zixuan Wang, Mehdi Azabou, Blake Richards, Renee Tung, Olivier Winter, Eva Dyer, Liam Paninski, et al. Towards a" universal translator" for neural dynamics at single-cell, single-spike resolution. *Advances in Neural Information Processing Systems*, 37:80495–80521, 2024.

APPENDIX

A MODEL DETAILS

A.1 TASK AND NOTATION

This appendix section follows the five stages of Sec. 3 in order and uses the same names. A training example is one context window from one session. Session s has C_s neurons; a batch stacks B such windows, so a batch of contexts is $X \in \mathbb{R}^{B\times C\times T_{\text{ctx}}}$ with $C = \max_s C_s$ after padding (Appendix A.6), and the corresponding horizons are $Y \in \mathbb{R}^{B\times C\times T_{\text{H}}}$. We index examples by b, neurons by c, segments of the temporal tree by n, and frequency bands by k; the main-text component index j enumerates all (n, k) pairs. Each neuron carries a unit index $u_{b,c}$, a session index s_b, and a dataset index d_b. Dataset-specific dimensions and optimization settings are in Appendix C.

The model must satisfy three constraints that the main text states in words. It must accept any number of neurons in any order, since C_s and the neuron identities change from session to session. It must produce, for every neuron, a forecast that can depend on the other neurons of the same session but never on neurons of another session, since only co-recorded activity is available at inference. And it must learn everything about a neuron from its three indices and its trace, since no anatomical or connectivity labels are given. The five stages below are designed around these constraints, and each subsection first states what the stage must achieve and then how it is implemented.

A.2 TEMPORAL ENCODING

Our purpose here is to represent each neuron's context as a set of components, each carrying one time scale over one part of the window, in a form whose size does not depend on the neuron or the session, so that shared networks can process every component the same way.

Temporal tree. The context is split recursively into a binary tree of depth $D = 2$: the root is the full window, its two children the left and right halves, and their four children the quarters, giving $1 + 2 + 4 = 7$ segments. A window of length L is split at $\lfloor L/2 \rfloor$; both halves are then extended outward by $\text{ov} = \min(\text{overlap}, \lfloor L/2 \rfloor, L - \lfloor L/2 \rfloor)$ steps and trimmed to equal length, so the two children of any node have the same length and T_{ctx} need not be divisible by four. The overlap defaults to 4 steps and is used unchanged in all reported runs; it keeps a transient that falls on a split point intact in at least one segment.

Frequency bands. Each segment is decomposed into K bands at the segment's own temporal resolution, so a component from a quarter-window segment is a quarter as long as one from the root. For the calcium datasets the decomposition is a Haar wavelet transform: $K - 1$ rounds of stride-2 averaging and differencing produce $K - 1$ detail bands from finest to coarsest plus one final low-pass approximation, and the K bands sum back to the original segment. Segment edges are reflect-padded before each round (replicate-padded on the left in the causal variant), and one time step is appended by reflection when a segment has odd length. The IBL models use a difference-of-Gaussians decomposition instead, chosen on validation ($+0.005$ Poisson D^2, $+0.013$ bits/spike over Haar in a paired comparison); the number and resolution of bands is unchanged. K is a per-dataset setting: $K = 5$ for Allen and IBL, giving $J = 7 \times 5 = 35$ components, and $K = 6$ for *C. elegans*–Flavell, giving $J = 42$ (Appendix C.1).

Component statistics. Components differ in length by up to a factor of four across the tree, so any network that reads them directly would need a separate input size per segment. We instead summarize every component z_j (a signal of length ℓ) by six scalars,

$$\text{stats}(z_j) = \Big[\mu,\ \sigma,\ \max|z_j|,\ \tfrac{1}{\ell-1}\|\Delta z_j\|_1,\ \tfrac{1}{\ell-2}\|\Delta^2 z_j\|_1,\ \tfrac{\max|z_j|}{\sigma+\epsilon}\Big], \tag{7}$$

where Δ is temporal differencing and ϵ a small constant: level, spread, peak amplitude, first- and second-order roughness, and peak-to-spread contrast. They are used twice: per component by the band scorer below, and averaged over all components by the activity embedding of Appendix A.3. In the main text they are the "mean, spread, peaks, roughness" of Eq. (2).

Band scorer f. A fixed frequency decomposition treats every band as equally informative, but which bands carry the predictive signal differs between neurons (a bursting cell lives in the slow bands, a sparsely spiking one in the fast bands of its most recent segment) and between windows of the same neuron. The scorer f of Eq. (1) therefore re-weights the bands per neuron and per window. It is a two-layer MLP with hidden width 32 and LeakyReLU activation, shared across all segments and neurons. Its input for component $j = (n, k)$ concatenates $\mathrm{stats}(z_j)$, the neuron's identity embedding e^{id}, a learned embedding of segment n's band statistics, and a positional encoding p_n of the segment's place in the tree. Before normalization the raw logit is multiplied by a sigmoid gate computed from e^{id} and shifted by a learned scalar multiple of a linear function of the segment embedding; both are one-dimensional modulations of the logit and are active in all reported models. The K scores within a segment are then normalized by a softmax at temperature $\tau_{\mathrm{sel}} = 0.7$, and the two children of every tree node compete through a further softmax between them at the same temperature, so that the model can favor the earlier or the later half of a window. Each band is scaled by the product of these weights, $\widetilde{z}_j = w_j z_j$, and the weighted bands are propagated separately down the tree rather than summed, so that the candidate heads of Appendix A.5 receive all J components.

A.3 Neuron identity and activity representations

Our goal is to give every neuron a vector e in one shared space, so that neurons from different sessions, which share no channel index, can be compared by the retrieval stage; the vector must say both which neuron this is and what it is doing in the current window, because retrieval should prefer different neighbors at different times.

Identity. Who a neuron is cannot be read from its trace alone, so identity is stored in three learned lookup tables indexed by unit, session, and dataset,

$$e^{\mathrm{id}} = \phi\big(E_{\mathrm{unit}}[u] + E_{\mathrm{sess}}[s] + E_{\mathrm{data}}[d]\big) \in \mathbb{R}^D, \tag{8}$$

where ϕ is a LayerNorm–linear–GEGLU block. The tables are initialized from $\mathcal{N}(0, 0.02)$ and trained only through the forecasting loss; they receive no anatomical, waveform, or cell-type information. Unit indices are offset per session so that each session occupies a disjoint range of the unit table: the embedding *space* is shared across sessions, individual entries are not, and there is no notion of "the same unit" appearing in two sessions. When the model is finetuned on a session unseen in pretraining, its new unit and session entries are initialized the same way and trained jointly with the rest of the network. In TS2 tasks, nothing is frozen. For TS3, we randomly initialize and finetune only the unit and session embedding tables, while keeping all other model parameters frozen.

Activity. What a neuron is doing right now is read from its components. The six statistics of Eq. (7) are computed for every component at every level of the tree, aggregated across all J components, and passed through a second LayerNorm–linear–GEGLU block ρ to give $e^{\mathrm{act}} \in \mathbb{R}^{256}$. Aggregating over the whole tree rather than the root alone lets the embedding reflect fast structure in the most recent segments as well as the slow level of the full window. (Separately from this, the band scorer of Appendix A.2 uses its own per-segment embeddings of the same statistics; those are not reused here.)

Combination. Identity is injected additively into the activity embedding after a linear map to the common width,

$$e = e^{\mathrm{act}} + W_{\mathrm{id}}\, e^{\mathrm{id}} \in \mathbb{R}^{256}, \tag{9}$$

which is Eq. (2) of the main text with W_{id} absorbed into ϕ. The identity term is constant for a neuron across windows; the activity term changes with every window. This e is the input to retrieval (Appendix A.4), and e^{id} on its own additionally conditions the band scorer, the candidate heads, and the correction term, so that identity reaches every stage of the model.

A.4 Retrieve context from the co-recorded population

To give each neuron a summary of what the other neurons in its session are doing, weighted by how relevant each of them is to this neuron in this window. The operation must accept any number of neurons in any order, since C_s and the identities change per session, and it must be able to re-weight neighbors from window to window, since a neighbour matters when it is active and not otherwise. Attention over the embeddings of Appendix A.3 satisfies both: it is permutation-invariant, and its weights depend on the activity term inside e.

Population attention. Eq. (3) of the main text is standard multi-head self-attention in which the sequence axis is the neuron axis. For a session with C_s neurons and embeddings $e_1, \dots, e_{C_s} \in \mathbb{R}^{256}$, each neuron c forms a query $W_q e_c$ and attends over the keys $W_k e_{c'}$ and values $W_v e_{c'}$ of all $c' \in \{1, \dots, C_s\}$, itself included. The attention sits inside a single pre-norm transformer block: a LayerNorm, 8-head attention with head dimension 32 and attention dropout 0.1, a residual connection, a second LayerNorm, a feed-forward layer of width 4×256 with squared-GELU activation and dropout, and a second residual connection. Scores are scaled by $\sqrt{d}$ only, with no additional temperature, and all parameters are shared across sessions and datasets. The block's output r_c *replaces* the neuron's embedding for every downstream module, so from this point on "the neuron's representation" means its representation after seeing its population. This is the only operation in the model in which neurons exchange information.

Batching. Sessions are stacked along the batch axis, one session per example, so attention never crosses sessions; within a batch, sessions with fewer neurons than the widest are zero-padded to a common width and given a reserved padding unit index.

A.5 Select the forecast with population context

Our purpose is to turn the J weighted components of a neuron into one forecast of the horizon, in a way that lets the population context decide which components to trust rather than averaging them.

Candidate heads h_j. Each weighted component $\widetilde{z}_j$ is mapped to a full-horizon candidate $\widehat{y}_j \in \mathbb{R}^{T_{\mathrm{H}}}$ by its own two-layer network h_j (linear to a hidden width, then linear to T_{H}), conditioned on e^{id}; the input width of h_j equals the length of component j, so the heads for shorter segments are smaller. The J heads share no parameters, which lets a head specialize to the time scale and window position of its component. With a tree of depth D and K bands per segment, $J = (2^{D+1} - 1)K$, i.e. $7K$ for $D = 2$: 35 candidates for $K = 5$ (Allen, IBL) and 42 for $K = 6$ (*C. elegans*–Flavell).

Scoring against the population context. The similarity in Eq. (5) is a scaled dot product between a projected candidate and the projected population context,

$$\mathrm{sim}(\widehat{y}_j, r) = \frac{\langle W_a \widehat{y}_j + p_j,\ W_r\, r \rangle}{\sqrt{d}\,\tau_{\mathrm{mix}}}, \tag{10}$$

where W_a maps a candidate trajectory to the $d = 256$-dimensional space of r, p_j is a learned embedding of component j's position in the tree (so that the scorer knows which segment and band a candidate came from), W_r is a linear projection, and τ_{mix} is a learned log-temperature initialized at 1. The softmax over j gives attention weights α_j^{attn}. These are mixed with a second set of weights α_j^{SE} produced by a Squeeze-and-Excitation block over the candidates, which favors components with higher signal energy independently of the population,

$$\alpha_j \propto (1 - \lambda)\, \alpha_j^{\mathrm{attn}} + \lambda\, \alpha_j^{\mathrm{SE}}, \qquad \lambda = \sigma(\lambda_{\mathrm{logit}}), \tag{11}$$

renormalized to one sum. λ_{logit} is learned and initialized at -4, so at initialization $\lambda \approx 0.018$ and the mixture is driven almost entirely by the population-conditioned weights; the energy prior is enabled in every reported configuration.

Correction term δ. A horizon decoder produces the per-step correction of Eq. (5) from the concatenation of a learned embedding of the horizon position, the identity e^{id}, and the population context r,

$$\delta = \mathrm{MLP}\big([\,p_{\mathrm{H}};\ e^{\mathrm{id}};\ r\,]\big) \in \mathbb{R}^{T_{\mathrm{H}}}, \qquad \widehat{y} = \kappa\Big(\textstyle\sum_j \alpha_j \widehat{y}_j + \sigma(\omega)\,\delta\Big), \tag{12}$$

where the MLP has three linear layers (widths $3d \to d \to d \to T_{\mathrm{H}}$) with squared-GELU activations and dropout 0.1 after each hidden layer, $\sigma(\omega)$ is a learned scalar initialized at $\sigma(0.8) \approx 0.69$, and $\kappa = \sigma(W_\kappa e^{\mathrm{id}})$ is a learned per-neuron output scale, bounded in $(0, 1)$ and equal to 0.5 at initialization. Together, κ and $\sigma(\omega)$ set the overall scale of the output at the start of training. Because r enters both α_j and δ, population information reaches the forecast along two paths: by choosing among the neuron's candidates, and by adjusting the chosen mixture step by step.

A.6 Objectives and training

Score the forecast against the observed horizon with a likelihood appropriate to the data type, and train every parameter from that score alone.

Losses. For spike counts the output $\widehat{y}_{b,c,t}$ is read as the log of the expected count, $\lambda_{b,c,t} = \exp(\widehat{y}_{b,c,t})$, and the loss is the Poisson negative log-likelihood averaged over examples, neurons, and horizon steps,

$$\mathcal{L}_{\text{Poisson}} = \frac{1}{BCT_{\text{H}}} \sum_{b,c,t} \left(e^{\widehat{y}_{b,c,t}} - y_{b,c,t}\, \widehat{y}_{b,c,t} + \log y_{b,c,t}! \right), \tag{13}$$

implemented as `PoissonNLLLoss(log_input=True, full=True)`; the link is applied to the final output $\widehat{y}$ of Eq. (12), after candidate mixing and correction, so candidates and correction live in log-rate space. For calcium traces the output is the predicted $\Delta F/F$ and the loss is the mean squared error over the same three axes. Each neuron's trace is z-scored over its full processed recording before temporal splitting (Appendix B.2). Allen and *C. elegans* errors are reported in these z-scored units. For zebrafish, the z-scored traces are projected onto the leading 512 principal components (Appendix B), and errors are reported in unscaled PCA-score units, whose magnitude is not comparable to that of the other datasets. The same preprocessing is applied to every method.

Optimization. AdamW jointly trains all parameters, including identity tables, band scorer, attention block, and mixture and correction weights ($\beta_1 = 0.9, \beta_2 = 0.999$, weight decay 10^{-4}) from forecast loss alone; there are no auxiliary losses, event labels or anatomical supervision. Dataset-specific learning rates, schedules, batch sizes, and early-stopping criteria are listed in Appendix C.1.

B Datasets and evaluation

B.1 Dataset details

Mouse - Spikes: **IBL BrainWideBench.** We evaluate on multi-region extracellular recordings from the IBL Brain-Wide Map IBL et al. (2025) using BrainWideBench Andre et al. (2026), which defines 423 pretraining sessions from 126 animals and 29 evaluation sessions from 13 held-out animals. TS2 and TS3 use selected units passing firing-rate and unit-level quality criteria, with probe-level quality control additionally applied to evaluation sessions. We evaluate SCRYER on the TS2 future spike-count forecasting task using Poisson deviance explained (D^2) and bits per spike (bps). Each forecasting example spans 1 s using 20 ms bins, with 900 ms of observed context and a 100 ms prediction horizon. For cross-recording experiments, SCRYER is pretrained on all 423 BrainWideBench pretraining sessions before recording-specific finetuning on the 29 evaluation sessions. Within each evaluation recording, we use the benchmark-provided TS2 train, validation, and test domains, defined using interleaved five-minute blocks in a 40/20/40 proportion, without constructing an additional temporal split. TS3 evaluates anatomical information through prediction of 10 Cosmos-level brain regions using linear and MLP probes on frozen unit embeddings. Following the TS3-specific anatomical and neural quality-control filters, probes are fitted on 41,834 pretraining units and evaluated on 3,209 units from held-out animals. We report macro-F1 for both single-unit and spatially pooled multi-unit predictions.

Mouse - Calcium: **Allen Brain Observatory.** We use two-photon calcium-imaging recordings from the Allen Brain Observatory de Vries et al. (2020), collected from awake mice during visual stimulation. The dataset spans six visual cortical areas (VISp, VISpm, VISam, VISrl, VISal, and VISl) and multiple imaging depths, sampling neuronal populations across cortical layers. Cre driver lines genetically target different neuronal populations for calcium-indicator expression, providing recordings from broad excitatory populations, more restricted excitatory subclasses, and inhibitory populations such as those targeted by Vip, Sst, and Pvalb lines. Consequently, recordings differ in both anatomical coverage and the neuronal populations observed. The visual stimulus battery includes gratings, natural images, and movies; our forecasting models receive neural activity without stimulus or behavioral labels.

We construct three cohorts to evaluate forecasting across this diversity. Mice-200 contains 200 recording sessions selected by stratified sampling over visual area and Cre line, increasing coverage of less frequent recording conditions. Mice-800 provides a larger stratified cohort, while Mice-1334

contains the full 1,334-session dataset (Figure A1). Across sessions, the number and identity of recorded neurons vary, and co-recorded activity provides context from different anatomical and genetically defined populations. These cohorts therefore test whether a shared model can adapt its temporal predictions and use of population context across heterogeneous recordings.

***C. elegans* - *Calcium*: Zimmer dataset.** The Zimmer dataset Kato et al. (2015) contains whole-brain calcium recordings from five animals expressing the pan-neuronal nuclear indicator GCaMP5K. Each animal is recorded for approximately 18 minutes at 2.85 volumes/s, with 107–131 neurons per session. The imaged volume covers the head ganglia, including sensory neurons, interneurons, head motor neurons, and anterior ventral cord motor neurons. Recordings include constant-oxygen and alternating-oxygen conditions. This dataset provides a compact population-level forecasting setting in which the model must use the recent activity of individual neurons together with the dynamics of their co-recorded population. Models receive calcium traces without oxygen-stimulus timing or biological neuron-class labels.

***C. elegans* - *Calcium*: Flavell dataset.** The Flavell dataset Atanas et al. (2023) contains brain-wide calcium recordings from freely moving animals expressing nuclear-localized GCaMP7f and mNeptune2.5. A tracking microscope maintains the animal's head within the imaging field, and segmentation and registration pipelines extract neuronal activity during movement. We retain 40 sessions with NeuroPAL neuron-identity annotations: 21 baseline sessions and 19 sessions involving a noxious heat stimulus associated with a persistent behavioral-state change.

These recordings extend evaluation to neural activity measured during ongoing behavior and across distinct experimental conditions. Models must forecast from the observed population activity without access to stimulus timing, behavioral measurements, or NeuroPAL-derived neuron-class labels. The biological annotations determine the included cohort but are not supplied as forecasting features.

Zebrafish-Ahrens. We use whole-brain calcium-imaging recordings from larval zebrafish Chen et al. (2018), acquired with light-sheet microscopy and a nuclear-localized GCaMP6f indicator. Recordings were acquired at approximately 2.1 Hz, with durations varying across fish. After excluding subjects 8, 9, and 11 due to file incompatibility, the retained cohort contains 15 fish with 1,650–6,424 time points per recording (mean 3,879; approximately 13–51 minutes).

To reduce the dimensionality of these large populations, we process each fish independently. Each neuron's activity is standardized over the full retained recording, after which PCA is fitted to the resulting time-by-neuron activity matrix. Preprocessing retains up to 2,048 principal components, while all forecasting experiments use the leading 512 components. The same precomputed PCA representation is provided to all methods. PCA is computed before constructing the temporal training, validation, and test splits. Stimulus and behavioral labels are excluded from the model inputs.

Table A1: **Calcium dataset overview.** Summary statistics for each calcium-imaging dataset, including the number of sessions (#Sess), approximate average number of recorded neurons per session (#Neur), average recording length in time steps (#Steps), sampling frequency (f_s), and calcium indicator (Ca Ind.).

Species	Source	#Sess	#Neur	#Steps	f_s	Ca Ind.
Zebrafish	Ahrens Chen et al. (2018)	15	77K	3.9K	2.1 Hz	GCaMP6f
Mice	Allen de Vries et al. (2020)	1334	120K	112.9K	30.2 Hz	GCaMP6
C. elegans	Zimmer Kato et al. (2015)	5	126	3.1K	2.8 Hz	GCaMP5K
C. elegans	Flavell Atanas et al. (2023)	40	136	1.6K	1.7 Hz	GCaMP7f

B.2 Splits and cohort construction

For the mouse calcium-imaging experiments, each processed recording is divided into contiguous patches of at most 5,000 frames. Within each patch, the first 60% of frames are used for training, the next 20% for validation, and the final 20% for testing. Forecasting windows are constructed only after this split, ensuring that their 336-frame contexts and 96-frame prediction horizons lie entirely within a single split. Windows may overlap within a split but never cross split boundaries. Eligible forecasting windows are generated with a stride of 16 frames and shuffled globally during training. Sessions therefore contribute in proportion to their numbers of eligible windows rather than through uniform session sampling.

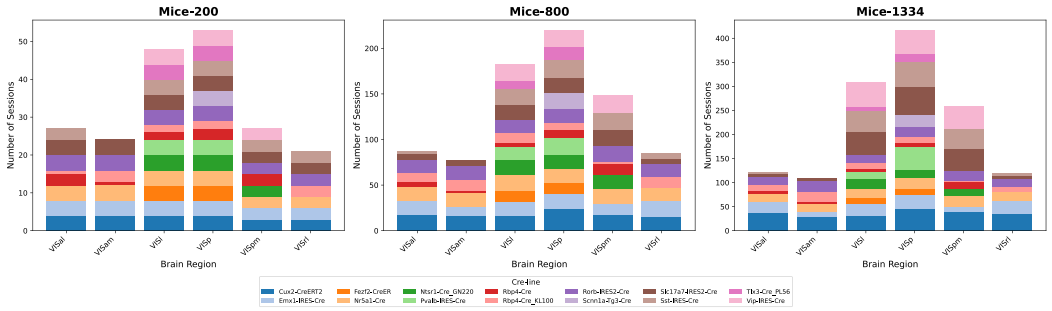


Figure A1: **Allen cohort composition.** Session counts by visual area and Cre line for Mice-200, Mice-800, and Mice-1334. The smallest cohort balances area–Cre-line coverage; larger cohorts approach the repository distribution.

Calcium traces are z-scored independently for each neuron over the full processed recording before temporal splitting. The reported runs apply no additional runtime or model-level normalization and use the original retained temporal resolution. Single-session models are trained independently for each recording, whereas the multi-session model shares a backbone and uses recording-specific identity embeddings. The normalized forecasting score is computed separately for each session and then averaged equally across sessions, whereas MSE and MAE are pooled over all evaluated prediction elements. A training batch may contain windows from multiple sessions, which are grouped by session during collation.

C Implementation Details

C.1 Model Configurations and Optimization

Shared configuration. Unless otherwise specified, calcium-imaging experiments use batch size 64, an overlapping binary temporal tree with depth 2 and overlap 8, Haar decomposition, and a band-selector hidden dimension of 32. Unit, session, and dataset embeddings jointly encode recording identity. We distinguish model width d_{model}, channel-context dimension d_{channel}, identity dimension d_{id}, and FiLM hidden dimension d_{FiLM} below.

Shared optimization. Unless otherwise specified, calcium-imaging benchmark models are optimized with AdamW using $\beta_1 = 0.9$, $\beta_2 = 0.999$, AMSGrad, learning rate 3×10^{-4}, and weight decay 10^{-4}. We clip the global gradient norm at 5. No learning-rate scheduler, early stopping, or neuron dropout is used. The `max_batch` parameter denotes a global cap on optimizer updates over the complete run rather than a per-epoch limit.

Allen Brain Observatory. For our Mice 200 configuration, we use $(T_{\text{ctx}}, T_{\text{H}}) = (336, 96)$, giving a total forecasting window of 432 frames. Forecasting windows are generated with a stride of 16 frames during both training and evaluation. For single session runs, the model uses $d_{\text{model}} = 256$, five Haar bands, $d_{\text{channel}} = 256$, $d_{\text{id}} = 128$, and $d_{\text{FiLM}} = 512$. Band statistics are embedded into the 256-dimensional channel-context representation, which is separately projected to a 128-dimensional selector feature used by the band-routing MLP.

Training is capped at 150,000 optimizer updates. Validation is performed every 5,000 updates. Checkpoints are selected using MAE or MSE. Separate scaling experiments use 200, 800, and 1334 processed sessions with the same window geometry and architecture.

***C. elegans*: Zimmer.** The benchmark configurations use a fixed 48-frame context with prediction horizons of 16 and 48 frames, $(T_{\text{ctx}}, T_{\text{H}}) \in \{(48, 16), (48, 48)\}$. Models use $d_{\text{model}} = 256$, five Haar bands, $d_{\text{channel}} = 256$, $d_{\text{id}} = 128$, and $d_{\text{FiLM}} = 128$. The channel-context representation is projected to a 16-dimensional selector feature. Training is capped at 3,000 optimizer updates. The currently available benchmark artifacts use four seeds $\{0, 1, 2, 3\}$. Single-session experiments train a separate model for each of the five sessions using the same base configuration.

***C. elegans*: Flavell.** The benchmark configurations use the same 48-frame context with prediction horizons of 16 and 48 frames. Models use $d_{\text{model}} = 256$, five Haar bands, $d_{\text{channel}} = 256$, $d_{\text{id}} = 128$,

and $d_{\mathrm{FiLM}} = 128$, with a 16-dimensional selector projection. Training is capped at 1,000 optimizer updates and uses MSE loss. The currently available benchmark artifacts use seeds $\{0, 1, 2, 3\}$.

Zebrafish-Ahrens. The benchmark configurations use a fixed 48-frame context with prediction horizons of 16 and 48 frames. Models use $d_{\mathrm{model}} = 256$, five Haar bands, $d_{\mathrm{channel}} = 256$, $d_{\mathrm{id}} = 128$, and $d_{\mathrm{FiLM}} = 128$, with a 16-dimensional selector projection. Training is capped at 5,000 optimizer updates and uses MSE loss. The currently available benchmark artifacts use seeds $\{0, 1, 2, 3\}$. The benchmark configurations operate on the PCA representation described in Sec. B.1.

IBL BrainWideBench. Each forecasting window spans 1 s at 20 ms resolution, with 45 observed bins (900 ms) and five forecast bins (100 ms). The model predicts the log expected spike count in each forecast bin and is trained with `PoissonNLLLoss(log_input=True, full=True)`. For single-session evaluation, a separate SCRYER model is trained from random initialization for each of the 29 evaluation recordings. The model uses a depth-2 temporal tree with five DoG bands per node, yielding 35 component-specific candidate forecasts, together with temporal tokenization at compression factors 3, 5, and 9. We use $d_{\mathrm{model}} = d_{\mathrm{channel}} = 256$, identity dimension 16, selector dimension 128, forecast hidden width 128, and dropout 0.2. The additive forecast head (`hdec_out`=1) uses population self-attention over co-recorded units. Single-session models are optimized with AdamW using batch size 32, weight decay 10^{-4}, global gradient-norm clipping at 1.0, and a OneCycle schedule with peak learning rate 5×10^{-4}, `pct_start`=0.5, and `div_factor`=1. Training runs for up to 300 epochs in fp32, with validation every 10 epochs using the same forecast masking as at test time and early-stopping patience 20. Single-session results are averaged over five seeds. For cross-recording experiments, SCRYER is pretrained on all 423 BrainWideBench pretraining sessions, which are disjoint from the evaluation animals. Source pretraining uses AdamW with batch size 32 and base learning rate 7.07×10^{-5}; the pretraining-specific $\sqrt{\text{batch size}}$ scaling gives a OneCycle peak learning rate of 4×10^{-4}, with `pct_start`=0.3 and `div_factor`=25. For each evaluation recording, new unit and session embeddings are initialized from $\mathcal{N}(0, 0.02)$, after which all model parameters are finetuned on that recording. We use the benchmark-provided TS2 train, validation, and test domains, defined by interleaved five-minute blocks in a 40/20/40 proportion, and construct forecasting windows entirely within individual domains so that no window crosses a split boundary.

Temporal encoder and forecast head. The band selector uses temperature 0.7. For each temporal component, the selector receives summary statistics comprising the mean, sample standard deviation, absolute maximum, mean absolute first- and second-order differences, and the ratio of absolute maximum to standard deviation. Identity and activity features are projected using LayerNorm–linear–GEGLU blocks. For K frequency bands and tree depth D, the temporal hierarchy produces $K + 2K(2^D - 1)$ component-specific candidate forecasts; the default $K = 5, D = 2$ therefore yields 35 candidates. The final conditioning path depends on the forecast-head configuration. In the IBL scoring configuration, population self-attention provides the context used to weight candidate forecasts and parameterize an additive forecast correction.

C.2 BASELINE CONFIGURATIONS

POCO. POCO Duan et al. (2025) combines population attention, learned unit/session embeddings, and population-conditioned decoding. For the Allen experiments, POCO tokenizes the 336-frame context with a compression factor of 16, yielding 21 temporal tokens. The decoder uses eight latent tokens with hidden dimension 128 and one latent self-attention processing block with 16 heads. The conditioning MLP has dimension 256, and the rotary temporal scale is set to 100. For the IBL experiments, the 45-bin context is tokenized with a compression factor of 5, yielding nine temporal tokens. The IBL configuration uses 16 latent tokens with hidden dimension 128, two latent self-attention processing blocks with eight heads, and single-head encoder and decoder cross-attention. Its conditioning MLP has dimension 512, with the rotary temporal scale again set to 100. Unit dropout is disabled in both settings.

MLP. The MLP baseline uses the temporal decoder infrastructure of POCO but removes population attention. Its exact width depends on the benchmark. For the Allen experiments, the 336-frame context is tokenized with a compression factor of 16, yielding 21 temporal tokens. The tokenized history is flattened and mapped through a single 128-dimensional hidden layer with ReLU activation to form the decoder representation. In parallel, the raw 336-frame history is projected to a 256-dimensional conditioning representation, which modulates the forecast decoder through FiLM before

projection to the 96-frame forecast horizon. For the external POCO calcium-imaging baselines used on Zimmer, Flavell, and Zebrafish, the corresponding MLP decoder uses a single hidden layer of width 1024. Thus, the MLP architecture is one-hidden-layer in both settings, but its hidden width and conditioning configuration are benchmark-specific.

NetFormer. NetFormer Lu et al. (2025) has a single attention layer. In the Allen configuration, the activity representation has dimension 336 and is concatenated with a 30-dimensional learned neuron embedding. The model uses a single attention layer rather than multi-head Transformer attention: queries and keys are obtained from linear projections of the concatenated representation, attention scores are scaled by $(336 + 30)^{-1/2}$, and the resulting weights are normalized with softmax. A single global embedding table assigns a 30-dimensional embedding to each neuron. Layer normalization is applied to the concatenated activity–identity representation and again after the attention-based update.

TexFilter. TexFilter Yi et al. (2024) processes each neuron's temporal representation with a learned complex-valued frequency-domain filter and maps the filtered representation back to the time domain. In the Allen configuration, the model uses a filter embedding dimension of 128, hidden dimension 512, and a single complex filtering block with dropout 0.3. The filter band width equals the 336-frame context length. Complex-valued filter parameters are initialized with scale 0.02, the sparsity threshold is 0.01, and RevIN normalization is applied independently to each neuronal channel.

Latent PLRNN (L-PLRNN). The latent dynamical baseline uses a single-layer piecewise-linear recurrent neural network (PLRNN) with latent dimension 512. Its latent transition combines a linear recurrent term and a ReLU nonlinearity through a leaky update with fixed coefficient $\alpha = 0.05$. The leak coefficient is used in the recurrent transition but is not learned (`learnable_alpha=false`). Training uses sparse teacher forcing with an interval of four steps. All calcium benchmark runs reported in Table A2 use `rnn_type=PLRNN` under the `LatentModel` wrapper.

DLinear and NLinear. DLinear Zeng et al. (2023) decomposes each context into trend and residual components using a moving-average filter and independently projects the two components to the forecast horizon before summing their predictions. For the Allen 336-frame context, the moving-average kernel is 169, obtained from $336//4 \times 2 + 1$. The trend and seasonal projections both map 336 context frames to 96 forecast frames, share their weights across neurons, and are initialized with constant weights $1/336$. NLinear uses a single shared linear projection from 336 context frames to 96 forecast frames: it subtracts the final context value before the projection and adds that value back to the forecast. Neither model uses neuron-specific projection weights in the reported Allen configuration.

D Additional experiments

D.1 Single-session and multi-session forecasting

We evaluate forecasting on three whole-brain calcium-imaging datasets: two *C. elegans* datasets (Zimmer and Flavell) and one larval zebrafish dataset. Each model is trained in two settings. In the *single-session* setting, a separate model is trained and evaluated on each recording. In the *multi-session* setting, one model is trained jointly on all recordings of a dataset and evaluated on each of them. Comparing the two settings tests whether the shared temporal and population representations learned by SCRYER improve prediction when pooling heterogeneous recordings, rather than only fitting individual sessions. Table A2 reports the results across forecast horizons. SCRYER's gains are more consistent under multi-session training, particularly at longer horizons, while lightweight baselines remain competitive when trained on individual sessions.

D.2 Performance across Cre lines

Table A3 reports Allen forecasting errors for each of the 14 Cre lines in the 200-session cohort. Rbp4-Cre and Rbp4-Cre_KL100 are distinct labels in the Allen metadata, with different session counts, and are therefore reported separately. Because traces are z-scored per neuron, $1 - \mathrm{MSE}$ approximates the fraction of variance explained.

Table A2: **Main forecasting results.** Lower MSE, lower MAE, and higher Score indicate better forecasting performance. $T_{\text{ctx}} = 48$.

		C. elegans–Zimmer			C. elegans–Flavell			Zebrafish–Ahrens-pc		
Method	T_{H}	**MSE** ↓	**MAE** ↓	**Score** ↑	**MSE** ↓	**MAE** ↓	**Score** ↑	**MSE** ↓	**MAE** ↓	**Score** ↑
Single-Session										
NetFormer	16	0.466 ±0.002	0.471 ±0.001	0.145 ±0.004	0.635 ±0.001	0.560 ±0.001	0.168 ±0.002	56.17 ±0.527	4.852 ±0.007	0.208 ±0.008
NLinear	16	0.411 ±0.000	0.438 ±0.000	0.250 ±0.000	0.597 ±0.000	0.545 ±0.000	0.217 ±0.001	55.74 ±0.016	4.872 ±0.001	0.211 ±0.000
L-PLRNN	16	0.470 ±0.007	0.484 ±0.004	0.143 ±0.015	0.633 ±0.005	0.559 ±0.002	0.170 ±0.005	55.05 ±0.189	5.092 ±0.008	0.212 ±0.002
TexFilter	16	0.368 ±0.003	0.409 ±0.002	0.333 ±0.005	0.588 ±0.001	0.537 ±0.001	0.230 ±0.002	44.04 ±0.215	4.416 ±0.012	0.378 ±0.003
MLP	16	0.365 ±0.001	0.413 ±0.001	0.336 ±0.001	0.582 ±0.001	0.540 ±0.001	0.236 ±0.002	43.34 ±0.024	4.322 ±0.001	0.388 ±0.001
POCO	16	0.369 ±0.005	0.418 ±0.004	0.329 ±0.009	0.696 ±0.011	0.604 ±0.007	0.079 ±0.017	40.12 ±0.473	4.116 ±0.006	0.433 ±0.008
SCRyER	16	**0.354** ±0.003	**0.406** ±0.002	**0.360** ±0.006	**0.565** ±0.002	**0.533** ±0.002	**0.258** ±0.002	**39.08** ±0.250	**4.106** ±0.004	**0.445** ±0.004
NetFormer	48	0.817 ±0.009	0.629 ±0.003	0.020 ±0.011	1.117 ±0.014	0.759 ±0.004	0.023 ±0.012	99.42 ±0.788	6.169 ±0.019	0.156 ±0.006
NLinear	48	0.644 ±0.000	0.533 ±0.000	0.233 ±0.000	0.817 ±0.001	0.654 ±0.001	0.286 ±0.001	83.39 ±0.041	5.696 ±0.003	0.288 ±0.000
L-PLRNN	48	0.659 ±0.010	0.589 ±0.001	0.205 ±0.013	0.878 ±0.023	0.687 ±0.010	0.227 ±0.020	74.54 ±0.638	5.744 ±0.013	0.366 ±0.005
TexFilter	48	0.599 ±0.004	0.538 ±0.001	0.290 ±0.004	0.800 ±0.003	0.648 ±0.002	0.300 ±0.003	73.43 ±0.465	5.433 ±0.012	0.369 ±0.004
MLP	48	0.570 ±0.007	0.537 ±0.009	0.322 ±0.008	0.794 ±0.006	0.651 ±0.003	0.302 ±0.006	65.97 ±0.310	5.186 ±0.006	0.435 ±0.003
POCO	48	0.602 ±0.015	0.557 ±0.002	0.283 ±0.013	0.794 ±0.014	0.656 ±0.006	0.302 ±0.012	59.70 ±0.636	4.962 ±0.013	0.487 ±0.006
SCRyER	48	**0.568** ±0.006	**0.532** ±0.002	**0.327** ±0.011	**0.786** ±0.002	**0.646** ±0.000	**0.307** ±0.002	**56.51** ±0.390	**4.786** ±0.008	**0.491** ±0.003
Multi-Session										
NetFormer	16	0.460 ±0.001	0.469 ±0.001	0.150 ±0.002	0.590 ±0.000	0.544 ±0.000	0.217 ±0.001	55.35 ±0.210	4.849 ±0.004	0.220 ±0.002
NLinear	16	0.410 ±0.000	0.437 ±0.000	0.253 ±0.000	0.590 ±0.000	0.545 ±0.000	0.221 ±0.000	56.48 ±0.006	4.868 ±0.003	0.202 ±0.000
L-PLRNN	16	0.480 ±0.005	0.488 ±0.002	0.118 ±0.011	0.620 ±0.004	0.561 ±0.003	0.183 ±0.006	52.57 ±0.435	4.966 ±0.005	0.248 ±0.007
TexFilter	16	0.360 ±0.000	0.404 ±0.000	0.346 ±0.000	0.570 ±0.001	0.526 ±0.002	0.256 ±0.002	46.22 ±0.055	4.444 ±0.007	0.349 ±0.000
MLP	16	0.358 ±0.001	0.407 ±0.002	0.348 ±0.002	0.554 ±0.003	0.524 ±0.002	0.274 ±0.004	44.61 ±0.069	4.320 ±0.003	0.370 ±0.001
POCO	16	0.350 ±0.003	0.402 ±0.003	0.364 ±0.005	0.600 ±0.023	0.556 ±0.011	0.213 ±0.030	39.68 ±0.198	4.096 ±0.006	0.440 ±0.003
SCRyER	16	**0.341** ±0.002	**0.396** ±0.001	**0.381** ±0.005	**0.546** ±0.008	**0.523** ±0.001	**0.284** ±0.011	**38.32** ±0.274	**4.003** ±0.006	**0.452** ±0.004
NetFormer	48	0.909 ±0.006	0.662 ±0.001	-0.084 ±0.007	1.120 ±0.002	0.761 ±0.000	0.019 ±0.002	97.14 ±0.298	5.985 ±0.011	0.133 ±0.003
NLinear	48	0.654 ±0.002	0.560 ±0.001	0.22 ±0.002	0.821 ±0.002	0.657 ±0.001	0.281 ±0.002	80.53 ±0.222	5.487 ±0.016	0.283 ±0.002
L-PLRNN	48	0.630 ±0.011	0.577 ±0.006	0.240 ±0.014	0.840 ±0.001	0.673 ±0.001	0.259 ±0.001	67.22 ±0.098	5.389 ±0.005	0.393 ±0.001
TexFilter	48	0.598 ±0.001	0.530 ±0.001	0.290 ±0.001	0.789 ±0.000	0.636 ±0.001	0.309 ±0.000	75.87 ±0.173	5.332 ±0.005	0.325 ±0.002
MLP	48	0.547 ±0.002	0.527 ±0.002	0.349 ±0.003	0.758 ±0.004	0.634 ±0.002	0.333 ±0.004	60.65 ±0.223	4.855 ±0.003	0.451 ±0.003
POCO	48	0.574 ±0.003	0.537 ±0.003	0.319 ±0.005	0.943 ±0.109	0.733 ±0.042	0.169 ±0.097	58.07 ±0.528	4.762 ±0.004	0.482 ±0.003
SCRyER	48	**0.543** ±0.004	**0.523** ±0.003	**0.357** ±0.007	**0.752** ±0.001	**0.632** ±0.001	**0.339** ±0.001	**57.88** ±0.498	**4.760** ±0.002	**0.483** ±0.002

Table A3: **Allen forecasting errors by Cre line.** Same runs and metric as Table 2. Traces are z-scored per neuron over the full recording, and errors are averaged over all neurons, forecast windows, and the 96 forecast steps. Predictions are counted as window × neuron × forecast step. Values are mean ± standard deviation across 4 training seeds; the final row is the prediction-weighted average over all 200 sessions and matches the aggregate result in Table 2.

Cre line	Sessions	Predictions	MSE ↓	MAE ↓
Cux2-CreERT2	22	214,441,248	0.9446 ±0.0009	0.5984 ±0.0001
Emx1-IRES-Cre	22	249,991,104	0.9580 ±0.0010	0.6344 ±0.0001
Nr5a1-Cre	22	120,661,632	0.8129 ±0.0020	0.4592 ±0.0002
Rorb-IRES2-Cre	22	232,062,240	0.9616 ±0.0008	0.5724 ±0.0001
Slc17a7-IRES2-Cre	22	191,745,024	0.9521 ±0.0011	0.5935 ±0.0000
Sst-IRES-Cre	17	16,892,256	0.8901 ±0.0017	0.6542 ±0.0002
Rbp4-Cre	12	65,227,008	0.8833 ±0.0016	0.4879 ±0.0001
Ntsr1-Cre_GN220	11	64,123,296	0.9767 ±0.0008	0.6762 ±0.0000
Rbp4-Cre_KL100	11	25,195,008	0.8963 ±0.0017	0.5021 ±0.0001
Vip-IRES-Cre	11	10,845,024	0.8233 ±0.0024	0.5652 ±0.0003
Fezf2-CreER	8	61,637,568	0.9098 ±0.0011	0.5842 ±0.0000
Pvalb-IRES-Cre	8	9,566,304	0.8623 ±0.0006	0.6877 ±0.0002
Tlx3-Cre_PL56	8	100,365,312	0.9389 ±0.0010	0.5752 ±0.0000
Scnn1a-Tg3-Cre	4	44,738,784	0.8601 ±0.0008	0.5381 ±0.0001
All (weighted)	200	1,407,491,808	0.9305	0.5814

Errors are similar in magnitude across lines, with MSE ranging from 0.813 (Nr5a1-Cre) to 0.977 (Ntsr1-Cre_GN220). The two metrics rank lines differently: Pvalb-IRES-Cre and Sst-IRES-Cre have below-average MSE (0.862 and 0.890) but among the highest MAE (0.688 and 0.654), indicating fewer large errors but larger typical errors than in most excitatory lines. Nr5a1-Cre has the lowest error on both metrics.

Table A4: **Ablations across datasets.** Each variant removes or replaces one component of the full model. For calcium recordings, lower MSE and MAE and higher normalized forecasting Score are better. IBL results use single-session models trained from scratch with seed 42 on the 29 TS2 evaluation recordings; all variants use the same training protocol as the full model. The identity-embedding ablation is evaluated on 28 recordings because one recording diverged. Differences below 0.002 on IBL are comparable to single-seed variation. – indicates that the variant was not evaluated for that dataset.

	C. elegans			Zebrafish–Ahrens			IBL (TS2, scratch)	
Model Variant	**MSE** ↓	**MAE** ↓	**Score** ↑	**MSE** ↓	**MAE** ↓	**Score** ↑	D^2 ↑	**bps** ↑
Only freq encoder	0.391	0.445	0.297	45.3	4.48	0.389	0.0857	0.2333
w/o multi-scale tokens	0.366	0.418	0.336	42.1	4.20	0.405	0.1474	0.3175
w/o channel self-attention	0.346	0.403	0.371	40.4	4.12	0.430	0.1409	0.3032
w/o identity embedding	0.341	0.396	0.381	41.2	4.15	0.419	0.0996	0.2524
w/o FiLM	0.349	0.402	0.368	39.6	4.11	0.442	0.1662	0.3531
w/o compress factor	0.345	0.400	0.373	41.0	4.13	0.422	0.1664	0.3492
w/o horizon decoder	0.344	0.402	0.374	40.3	4.12	0.429	0.1659	0.3512
w/o Haar (use Binom)	0.344	0.399	0.374	40.2	4.13	0.431	0.1662	**0.3486**
SCRyER	**0.341**	**0.396**	**0.381**	**38.3**	**4.00**	**0.452**	**0.1654**	0.3496

Several factors may contribute to this variation, and the present results do not distinguish among them. Imaging depth may limit the poorest case: Ntsr1-Cre_GN220 labels layer 6 corticothalamic neurons, among the deepest populations imaged, where two-photon signal-to-noise is lowest. Depth alone does not explain the ordering, however, because Nr5a1-Cre and Rorb-IRES2-Cre both label layer 4 neurons yet lie at opposite ends. For Vip-IRES-Cre, low error is consistent with the strong modulation of VIP interneurons by locomotion and arousal Fu et al. (2014), which produces slow fluctuations that are predictable from recent history. Cre lines also differ in visual area, signal statistics, and sample size, from 4 to 22 sessions and from 9.6 to 250 million predictions per line. Relating per-session errors to imaging depth, signal-to-noise, and activity timescale would be needed to test these explanations.

Standard deviations across training seeds are small (at most 0.0024 in MSE), so the per-line errors are stable across runs. They do not reflect variability across sessions, so differences between lines should not be read as statistically significant.

D.3 Architectural ablations

Every variant in Table A4 removes or replaces exactly one component of the full model and is retrained from scratch under the configuration of the full model; nothing is transferred from a pretrained checkpoint. On *C. elegans* (Zimmer, $T_{\mathrm{H}} = 16$) and zebrafish, values are means over 4 seeds. On IBL, every variant is trained on each of the 29 TS2 evaluation recordings separately (seed 42, 300 epochs, base learning rate 5×10^{-4}) and scored on that recording's test split; because these are single-session models, their absolute D^2 and bits/spike are below the pretrained results of Table 1, and differences below 0.002 are within single-seed variation. Rows are grouped by the stage of Sec. 3 they test.

For each variant, *Only freq encoder* keeps the Haar decomposition and an MLP readout and removes everything else: no identity, no population attention, no correction term. *w/o multi-scale tokens* keeps a single full-window segment, so the decomposition has frequency bands but no temporal tree. *w/o Haar (use Binom)* replaces the Haar filters with binomial smoothing filters of matched scale. *w/o compress factor* disables the temporal compression applied before the candidate heads. *w/o identity embedding* removes the unit, session, and dataset tables, so e is the activity embedding alone. *w/o FiLM* removes the identity-conditioned modulation of the candidate heads. *w/o channel self-attention* removes the population attention block, so $r = e$ and the forecast depends on the neuron alone. *w/o horizon decoder* removes the correction term δ.

Keeping only the decomposition is the worst configuration everywhere (*C. elegans* MSE $0.341 \rightarrow 0.391$, zebrafish $38.3 \rightarrow 45.3$, IBL D^2 $0.165 \rightarrow 0.086$), and nearly all of the gap is recovered by the three components that represent activity structure, whose individual losses sum to more than the joint loss. Multiscale components are the largest single factor on calcium (zebrafish $\rightarrow 42.1$; IBL $\rightarrow 0.147$). Population attention costs 0.005, 2.1, and 0.025 on the three datasets, growing with

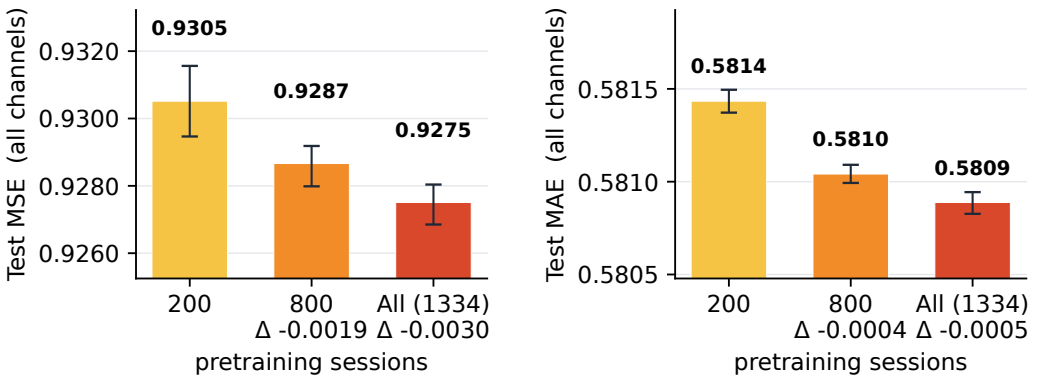


Figure A2: **Allen scaling results.** The scaling results with session counts for 200, 800, and 1334 on Allen Mice dataset. Lower is better.

population size. Identity costs nothing on *C. elegans*, 2.9 on zebrafish, and 0.066 D^2 on IBL, where it is the largest single loss and one recording diverged without it; identity is worth little where there are few units to distinguish and most where hundreds of spiking units share one model. FiLM, the compression factor, the correction term, and the wavelet choice change IBL D^2 by less than 0.002 and *C. elegans* MSE by at most 0.008, within or near seed variation; on zebrafish they cost 1.3, 2.7, 2.0, and 1.9 MSE respectively, against a seed SD of about 0.5, so there they are real contributors of the same order as population attention (2.1) though below multiscale components (3.8) and identity (2.9).

D.4 Scaling with pretraining data on Allen

We ask whether adding calcium sessions helps as much as adding IBL recordings does (Sec. 4.4). Models are trained on 200, 800, and all 1,334 Allen sessions under one shared configuration and evaluated on the same 200 test sessions (Fig. A2). MSE falls from 0.9305 to 0.9287 to 0.9275 and MAE from 0.5814 to 0.5810 to 0.5809; the paired difference between 200 and all sessions is -0.0030 in MSE (95% paired-bootstrap CI $[-0.0037, -0.0025]$ over the 200 sessions) and -0.0005 in MAE (CI $[-0.0006, -0.0005]$); between 200 and 800 sessions it is -0.0019 ($[-0.0025, -0.0013]$) and -0.0004 ($[-0.0004, -0.0003]$). The gain is an order of magnitude smaller than on IBL; the interval excludes zero and exceeds the seed-to-seed spread of a single model (SD 0.0009 in MSE over 4 200-session seeds), so it is a consistent but small effect.

Table A5: Per-region summary of Δ (own-history minus population-state retained performance) over the 2,539 held-out neurons of the 29 TS3 recordings (13 mice). Regions ordered by median Δ; positive values = history-driven.

Region	n	median Δ	IQR	$\Delta > 0$ (%)	median rate (Hz)	median CV2
HPF	244	-0.21	[-0.62, +0.08]	31	5.3	0.92
OLF	188	-0.19	[-0.51, +0.27]	37	3.9	0.90
CNU	119	-0.13	[-0.61, +0.25]	41	4.7	0.88
TH	532	-0.07	[-0.36, +0.32]	46	8.0	0.92
HY	33	-0.04	[-0.44, +0.31]	48	5.3	0.92
CTX	501	-0.01	[-0.37, +0.29]	49	4.1	0.85
MB	586	+0.02	[-0.37, +0.38]	51	9.1	0.70
HB	201	+0.18	[-0.21, +0.43]	62	10.0	0.66
CTXsp	21	+0.29	[-0.26, +0.61]	62	2.9	0.91
CB	114	+0.39	[+0.09, +0.59]	82	23.5	0.71
All	2539	-0.03	[-0.38, +0.34]	48	6.7	0.84

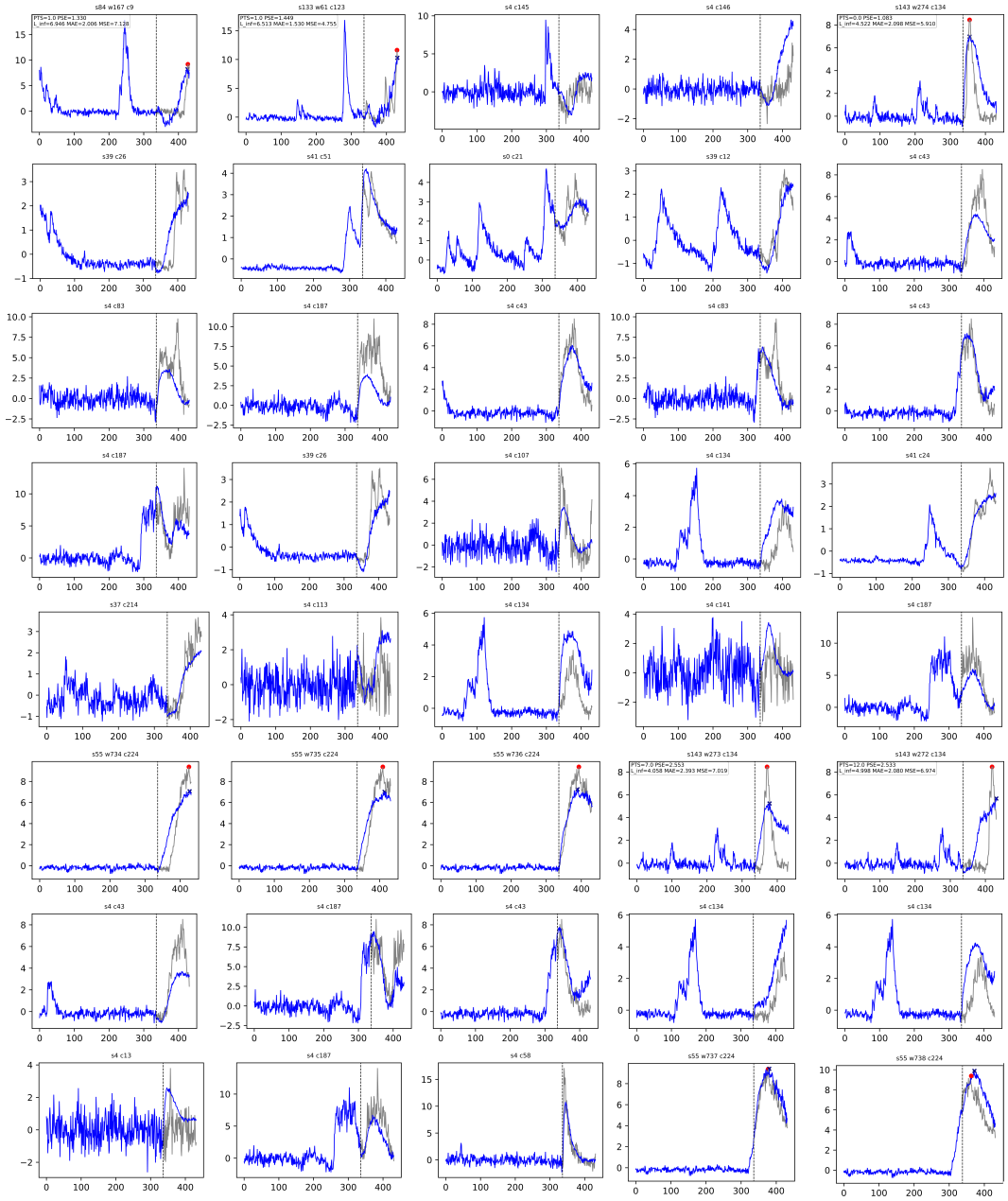


Figure A3: Calcium forecasts across 40 neuron–window examples from Allen.

D.5 Qualitative examples

Forecast galleries compare predicted and observed calcium activity across neurons (Figures A3–A4). Retrieval visualizations show how the model distributes weight across the recorded population (Figure A5). These weights describe model usage of population context, not causal interactions.

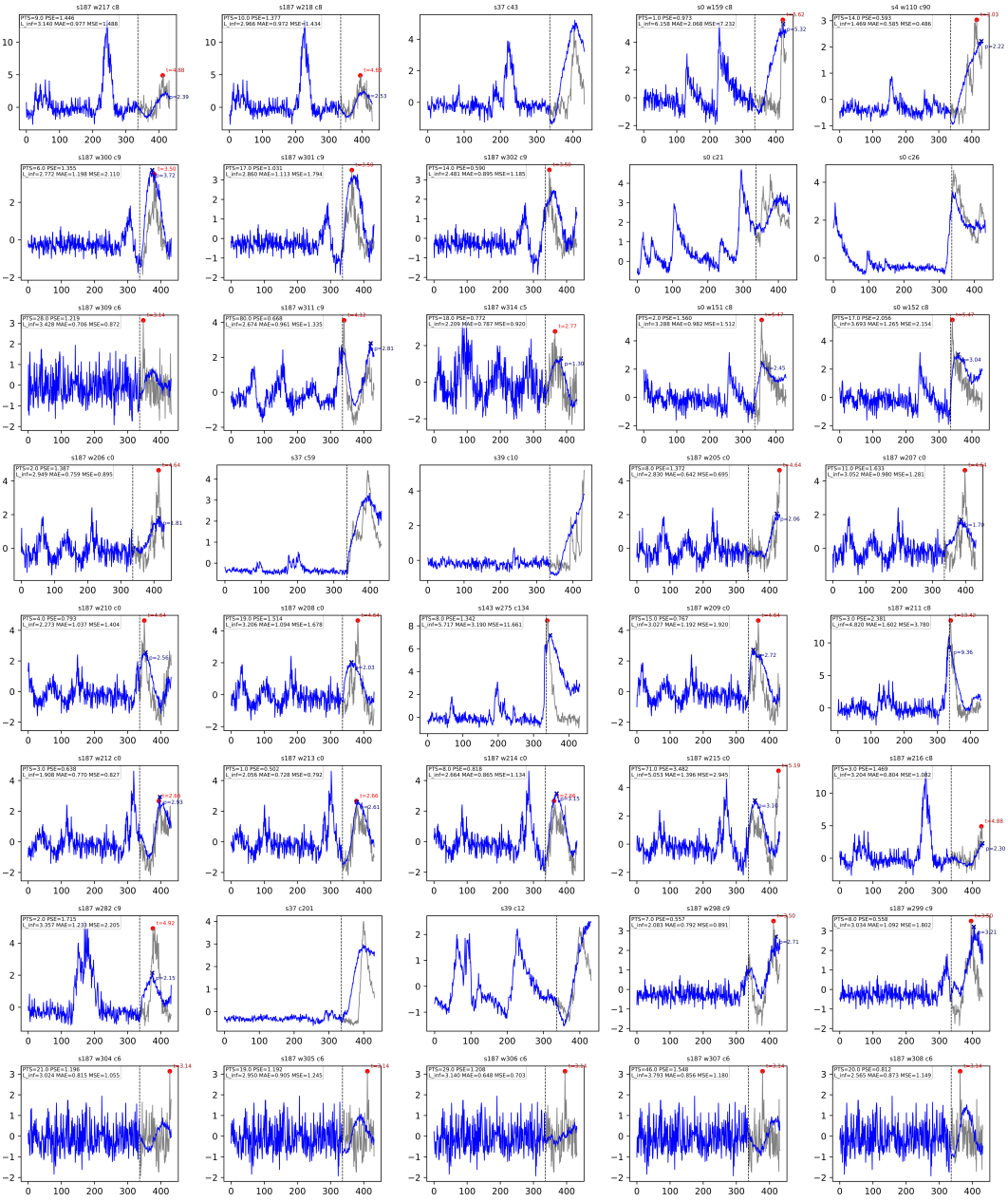


Figure A4: Additional Allen calcium-forecast examples.



Figure A5: Retrieved population context for example Allen target neurons.