# GENERATIVE NEURAL SYSTEM IDENTIFICATION AT SPIKE-TRAIN RESOLUTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

In populations of biological neurons, repeated presentations of the same stimulus evoke different spike trains, making neural responses fundamentally a conditional distribution. Yet, neural encoding models typically predict only the stimulus-conditioned mean response, either ignoring trial-to-trial variability or imposing it by an assumed noise distribution. Here, we introduce SpikeDiT: a stimulus-conditioned generative model of population activity that combines a frozen spike tokenizer with joint flow matching to generate spike trains at a native 1 ms resolution. Conditional mean responses, marginal firing statistics, fine temporal structure, and shared noise correlations all emerge from training and sampling from a single model. The framework is modular with respect to the underlying stimulus encoder and makes no *a priori* distributional assumptions about the form of neural variability. We evaluate the approach on two datasets that differ in sensory modalities, species, and recording technologies. Existing models capture either the conditional mean or trial-to-trial variability well whereas SpikeDiT matches or exceeds the strongest baselines on both and, without retraining, uses observed neurons to improve single-trial prediction of unobserved ones. By modeling population responses directly in spike space with a flexible generative framework, our approach provides a unified and scalable foundation for predicting and generating complete neural population responses.

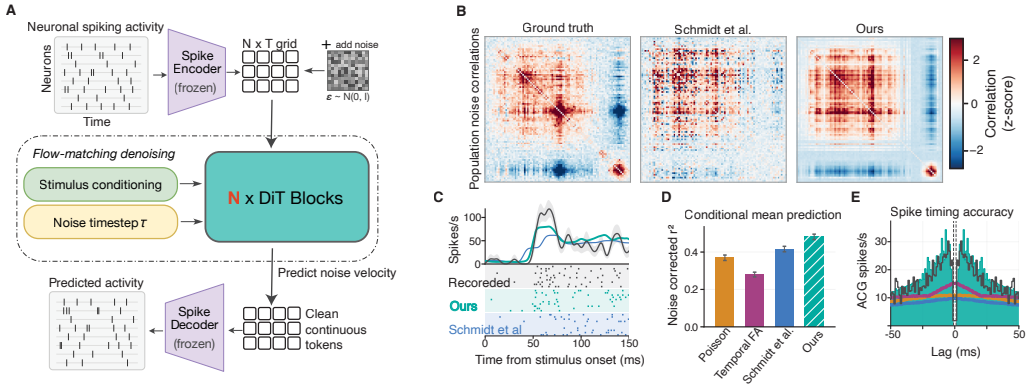


Figure 1: We present SpikeDiT (A), a latent flow transformer that generates population spiking activity (C) using neural data tokenizers. Our model reproduces population noise correlations more faithfully than latent-variable models (B), is competitive in predicting mean stimulus response (D), and generates individual and population spike times at millisecond resolution (C) with accurate spike-timing statistics (E).

## 1 INTRODUCTION

Most neural data analysis focuses on the mean response in repeated stimulus presentations, yet even in early sensory areas, this mean captures only a fraction of the variance from a single-trial (Arieli et al., 1996; Stringer et al., 2019). The remaining variability is structured: its magnitude is not fixed by the firing rate as in a Poisson process (Goris et al., 2014), it depends on internal brain



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.

state (Ecker et al., 2014), and it is organized across neurons (*noise correlations*; Cohen & Kohn, 2011). Under probabilistic accounts of perception, it reflects the uncertainty and priors with which the brain interprets each stimulus (Hoyer & Hyvärinen, 2002; Berkes et al., 2011; Orbán et al., 2016). Understanding how the brain represents a stimulus therefore requires the full stimulus-conditioned distribution of population spike trains, $p(Y \mid x)$, with $Y \in \{0,1\}^{N\times T}$ the spikes of $N$ neurons in $T$ bins of 1 ms.

Deep encoding models increasingly serve as *in silico* stand-ins for studying how sensory stimuli drive neural responses (Yamins et al., 2014; Klindt et al., 2017; Sinz et al., 2018; Cadena et al., 2019; Lurz et al., 2021; Cowley et al., 2026; Wingert et al., 2026). Most are trained with a Poisson loss on conditionally independent outputs, so the predicted mean fixes the variance and the model captures neither noise correlations nor spike timing within a bin. Latent-variable models explain shared variability through a low-dimensional state common to all neurons (Yu et al., 2009; Macke et al., 2011; Keeley et al., 2020), recently combined with an image-computable mean (Bashiri et al., 2021) or made time-varying (Schmidt et al., 2025). Both families fix the form of variability in advance: through a fixed count distribution, a low-dimensional state, or coarse binning. No current model combines the prediction accuracy of deep encoders with realistic trial-to-trial variability.

Learning $p(Y \mid x)$ without fixing its form is a conditional generation problem like text-to-video synthesis, with neurons and time bins in place of pixels. Generative modeling solves such problems with a common recipe, in which a transformer trained by diffusion or flow matching generates a grid of tokens in the latent space of a frozen autoencoder (Rombach et al., 2022; Peebles & Xie, 2023; Lipman et al., 2023; Yang et al., 2025). For neural data, spike tokenizers now provide such a latent space by encoding each neuron's spike train into continuous tokens (Anonymous et al., 2026).

Here, we treat neural encoding as conditional generation in this latent space (Figure 1): a stimulus-conditioned flow transformer jointly models the tokens of all recorded neurons, retaining their millisecond response structure. Our contributions are fourfold. **(1) Generative neural system identification at spike-train resolution.** Our model generates binary population spike trains at 1 ms without Poisson count laws, low-rank covariance assumptions, or binning; mean, dispersion, noise correlations, and timing arise from a single sampling pass. **(2) No cost in stimulus-driven accuracy.** Averaged samples match or exceed state-of-the-art encoding models on mean stimulus response prediction. **(3) Realistic trial-to-trial variability.** Without any explicit covariance formulations, the model reproduces empirical noise-correlation and dispersion patterns more accurately than specialized latent-variable models, while recovering spike times at millisecond resolution. **(4) General framework with training-free completion.** The identical architecture and objective generalize across species, sensory modalities, and number of recorded neurons. Our model also generalizes to inference-time completion of partially observed responses based on the other neurons in the population without any explicit training.

## 2 Related work

**Transformer-based latent-space generative models.** Most state-of-the-art diffusion models are latent diffusion models, which operate in the continuous latent space of a pretrained autoencoder rather than in data space. Prominent examples include the Latent Diffusion/FLUX model lineages for images (Rombach et al., 2022; Black Forest Labs et al., 2025), Wan (Team Wan et al., 2025) for video generation, and Stable Audio (Evans et al., 2024) for audio. Latent diffusion approaches have converged on transformer-based backbones (Peebles & Xie, 2023) operating on 2D or 3D latent patches for images or video, respectively (Rombach et al., 2022; Yang et al., 2025). These models have shown consistent scaling with parameter size while also allowing flexible conditioning on other modalities (Esser et al., 2024). We apply this recipe to neural data, replacing the image autoencoder with a frozen spike tokenizer and the space-time grid with a grid of neuron-time tokens.

**Deep encoding models and latent-state models of neural responses.** Neural encoding models have become the standard for predicting stimulus-driven responses, using a shared feature extractor followed by neuron-specific readouts (McIntosh et al., 2016; Batty et al., 2017; Antolík et al., 2016; Cadena et al., 2019; Sinz et al., 2018; Lurz et al., 2021; Cowley et al., 2026; Wingert et al., 2026; Wang et al., 2025; Willeke et al., 2026). These models typically admit a factorized form $p(\mathbf{Y} \mid x) = \prod_{i,t} p_{\text{Poisson}}(y_{i,t} \mid f_{i,t}(x))$, where responses are conditionally independent across neurons and time

given the stimulus $x$, with conditional mean $f(x)$. Latent-state models relax this factorization by introducing a shared latent state $z$, which captures trial-specific population variability and thereby induces correlations across neurons and, in temporal models, across time. Gaussian-process or Poisson-based latent models recover single-trial population states but do not model the stimulus (Yu et al., 2009; Macke et al., 2011). Meanwhile, signal-and-noise factor models separate stimulus-driven and shared variability but require repeated stimuli (Keeley et al., 2020). FlowFA, the method closest to our setting, combines an image-computable mean with a population state (Bashiri et al., 2021). Separately, Schmidt et al. (2025) introduce a shared state for temporal stimuli, with a prior that is independent across time . These methods and our temporal FlowFA extension (Section 4.1) serve as probabilistic baselines.

**Generative models of spike trains.** There have been several approaches to generating spike trains: point-process models (Pillow et al., 2008), generative adversarial networks (Molano-Mazon et al., 2018; Ramesh et al., 2019), and diffusion and flow-matching models (Kapoor et al., 2024; Wei & Pearson, 2026). The closest approach to ours is LDNS (Kapoor et al., 2024), which generates behavior-conditioned population activity from spike counts with latent diffusion. SpikeDiT differs in four ways. First, LDNS compresses the population into a few latent dimensions, so beyond Poisson noise, all variability is shared. Our tokenizer compresses each neuron separately, letting the flow learn both dependencies among neurons and each neuron's own dispersion. Second, LDNS conditions on low-dimensional behavioral variables, whereas ours is an encoding model that predicts responses to natural images and sounds, learning each neuron's stimulus selectivity through neuron-specific queries. Third, LDNS generates firing rates and samples spikes from them, whereas we generate spikes directly at the millisecond resolution, capturing single-neuron timing without post hoc spike-history filters. Finally, we use rectified flow matching (Liu et al., 2022) to reduce sampling from 500 steps to 20 steps, and to 5 during training, rendering it feasible to backpropagate through sampling trajectories for calibration (Section 3.3).

# 3 METHODS

We aim to learn the full distribution of spike trains evoked by a sensory stimulus. For stimulus $x$ and recording session $s$, we represent the response as $\mathbf{Y} \in \{0,1\}^{N_s \times T}$, where $N_s$ is the number of recorded neurons and $T$ the number of 1-ms bins. We model $p(\mathbf{Y} \mid x, s)$ by combining a frozen spike tokenizer with a stimulus-conditioned latent flow transformer (Figure 1). The tokenizer encoder maps individual neuron spike trains to continuous tokens, $Z_i \in \mathbb{R}^{T_z \times D}$, where $T_z$ is the latent time dimension and $D$ the token dimension, while the flow learns their joint distribution across neurons and time windows. The tokenizer decoder maps each token back to spike trains, generating one complete population response.

## 3.1 NEURAL RESPONSE TOKENIZER

We use the single-neuron convolutional VAE tokenizer introduced in Anonymous et al. (2026) (see Appendix L). The tokenizer encoder maps each nonoverlapping 100 ms window of a neuron's spike train to a continuous token of size $D = 16$. Applying the encoder separately across neurons and windows yields the population token grid $\mathbf{Z} \in \mathbb{R}^{N_s \times T_z \times D}$.The tokenizer decoder maps each token to per-millisecond spike probabilities, that are then thresholded to produce binary spikes.

The tokenizer was pretrained on nearly 20 billion spikes from several public and proprietary datasets spanning species and brain regions. Pretraining combined spike reconstruction with KL-divergence regularization of the variational posterior and SIGReg (Balestriero & LeCun, 2025), which encouraged an isotropic Gaussian latent space. Together, this kept latent dimensions stable and discouraged collapse, providing a latent space amenable to flow matching. Appendix E provides architecture and pretraining details.

## 3.2 CONDITIONAL FLOW OVER THE POPULATION TOKEN GRID

Using the tokenizer's continuous representation, we learn $p_\theta(\mathbf{Z} \mid x, s)$, jointly modeling tokens across neurons and time. First, to condition the flow on the stimulus, a modality-specific encoder model extracts features from $x$ and pushes these into keys and values that feed into a cross-attention

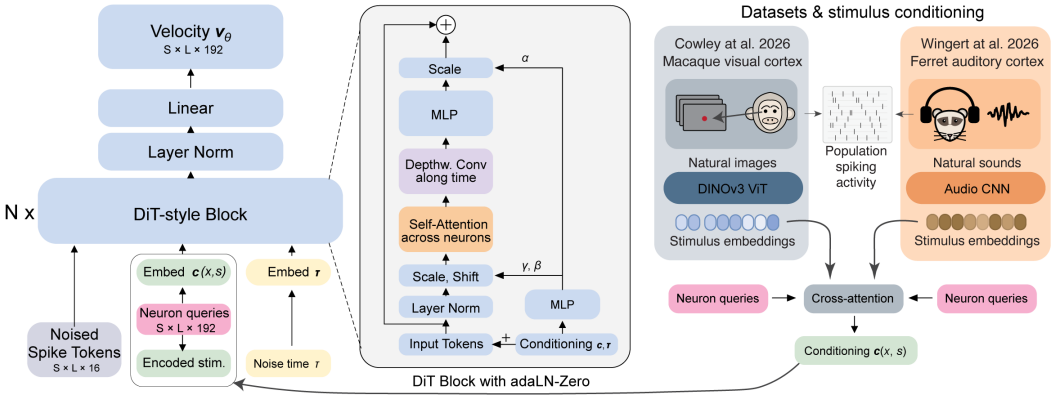


Figure 2: **The SpikeDiT architecture and dataset overview.** Left: Our latent flow transformer predicts the velocity $v_\theta$ of noised spike tokens. Learned per-neuron queries cross-attend to the encoded stimulus to form the conditioning $c$. Middle: Our DiT-style block (Peebles & Xie, 2023) applies self-attention across units and depthwise convolution along time, modulated per token by $c$ and $\tau$ through adaLN-Zero. Right: Data sketches and encoding backbones for the visual cortex (Cowley et al., 2026) and auditory cortex (Wingert et al., 2026) datasets.

module with learnable, neuron-specific queries. The resulting cross-attention output goes through a self-attention block before it is projected into conditioning vectors $c(x, s)$ .

Given a noisy token grid $\mathbf{Z}_\tau$, flow time $\tau \in [0, 1]$, and conditioning $c(x, s)$, a DiT-style transformer (Peebles & Xie, 2023) represents the velocity field $v_\theta$ that guides generation (Figure 2). Each block combines attention across neurons with convolution along temporal token positions. Conditioning enters both at the input and through adaptive layer normalization, with modulation determined separately for each token by $c(x, s)$ and $\tau$. The projected neuron queries from the earlier pre-flow cross-attention are also added to the input to identify the recorded neurons.

During training, we pair the neural tokens $\mathbf{Z}$ with Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$. We interpolate between them and train the flow network to predict the velocity along these straight paths using a rectified flow-matching objective (Lipman et al., 2023; Liu et al., 2022):

$$\begin{aligned} \mathbf{Z}_\tau &= (1-\tau)\epsilon + \tau \mathbf{Z}_1, \\ \mathcal{L}_{\mathrm{FM}} &= \mathbb{E}_{\tau,\epsilon,\mathbf{Z}_1}\left[\left\| v_\theta(\mathbf{Z}_\tau, \tau, c(x,s)) - (\mathbf{Z}_1 - \epsilon)\right\|_2^2\right]. \end{aligned} \tag{1}$$

To generate a population response, we integrate the learned velocity field from a Gaussian draw, then decode the resulting tokens into spike trains. Together, the flow and tokenizer decoder define:

$$p_\theta(\mathbf{Y} \mid x, s) = \int p_{\mathrm{dec}}(\mathbf{Y} \mid \mathbf{Z})\, p_\theta(\mathbf{Z} \mid x, s)\, \mathrm{d}\mathbf{Z}. \tag{2}$$

Although decoder emissions are independent given the tokens, jointly varying tokens produce correlated activity across neurons and time. Each sample constitutes one population trial, and we estimate response statistics from repeated samples.

**Inference-time completion.** Joint modeling also allows observed neural activity to constrain generation of the remaining response. We use the Flower solver (Pourya et al., 2025) to condition generation on recorded activity from selected neurons or time intervals, completing the unobserved response without retraining. Appendix G describes the conditioning procedure.

### 3.3 Calibrating generation with projected CRPS

Flow matching learns the distribution of latent tokens, while the response statistics of interest are measured on decoded spikes. In practice, training with flow matching alone produced overdispersed spike counts during generation, relative to real data (Section 4.3). We therefore calibrate flow matching with a projected continuous ranked probability score (CRPS, Gneiting & Raftery

(2007)) auxiliary loss, computed from generated responses during training. Our projection-based CRPS loss calibrates not just marginals, but also dependencies across units and time.

For a scalar observation $y$ and model samples $Z, Z'$, $\mathrm{CRPS} = \mathbb{E}|Z - y| - \frac{1}{2}\mathbb{E}|Z - Z'|$. Minimizing CRPS mitigates both over- and under-dispersion, and crucially only needs a single observed trial per stimulus, which is necessary to be combined with our flow-matching objective. However, when applied to each unit's spike count, CRPS has no notion of the shared neuronal population covariance. We therefore score fixed randomly-sampled scalar projections of the counts, exploiting that $\mathrm{Var}(N_i + N_j) - \mathrm{Var}(N_i - N_j) = 4\,\mathrm{Cov}(N_i, N_j)$.

At each training step, we generate two independent banks of trials for the same stimulus by integrating the learned field and decoding. These trials, passed through the scalar projections, are used to compute the projected CRPS loss term. We then backpropagate through the generation sequence back to the model parameters. Our full training objective thus becomes

$$\mathcal{L} = \mathcal{L}_{\mathrm{FM}} + \alpha \sum_j w_j \, \mathcal{L}_j^{\mathrm{CRPS}}, \tag{3}$$

with weights $w_j$ fixed by gradient-norm calibration and a scalar dose $\alpha$ (see appendix F for details).

# 4 Experiments

## 4.1 Setup

**Datasets.** We test SpikeDiT on two publicly available electrophysiology datasets, differing in both species and stimulus modality.

The visual dataset contains spike-sorted responses from V4 of three macaques, recorded with chronically implanted Utah arrays (Cowley et al., 2026). Stimuli are mostly natural images from a large natural image corpus. We use 37 sessions containing about 100 units each (3,550 in total), treated as distinct neurons across sessions. Analyses requiring repeats use images with at least six repeats, and are restricted to the 23 sessions with at least 30 such images. Following Cowley et al. (2026), we predict responses during the 200 ms after image presentation onset.

The auditory dataset contains spike-sorted responses from primary and nonprimary auditory cortex in awake ferrets that are presented with natural sounds (Wingert et al., 2026). We use 67 recording sessions across four animals, with 3,259 total units. Stimulus presentations consist of 18 s-long sound sequences. We divide each presentation into 1 s segments and predict responses for each 1 s segment given the stimulus from both that segment and the one immediately preceding it. Because responses in the first second after sound onset are dominated by an onset transient, we score only the latter 16 segments from each presentation.

**Baselines.** We selected our baselines to test SpikeDiT's direct spike generation against count-based encoding with increasingly structured models of variability. Specifically, we compare against an independent Poisson encoder, Temporal FlowFA (our extension of Bashiri et al. (2021)), and the Schmidt time-indexed latent model (Schmidt et al., 2025). All three baselines predict binned spike counts over 20 ms bins. They also use the same stimulus encoder and cross-attention readout as our model, differing only in how they model the conditional response distribution downstream. For the Poisson encoder, we predict mean counts per unit over 20 ms bins, and sample them independently without modeling shared variability. We re-implemented the Schmidt model (Schmidt et al., 2025) for spike counts by adding a latent state shared by all units, drawn independently in each time bin. While the FlowFA (Bashiri et al., 2021) models allow shared variability with a small number Gaussian factors, they only allow a single count per unit per trial. Therefore, we implemented Temporal FlowFA as a fair baseline for FlowFA, in which each factor follows a random trajectory over the response window. Specifically,

$$v_{i,t} = \mu_{i,t}(x, s) + \sum_{k=1}^{K} L_{ik} \, (\Phi a_k)_t + \epsilon_{i,t}, \qquad a_k \sim \mathcal{N}(0, K_k), \tag{4}$$

where $v_{i,t}$ is unit $i$'s transformed count in bin $t$, $L_{ik}$ the weight for factor $k$, and $\Phi a_k$ the factor's trajectory in a temporal basis. We can then recover the counts through FlowFA's neuron-wise

marginal transforms. See Appendix B for implementation details, and Appendix A for a comparison between the generative assumptions of all models, including LDNS. For count-level comparisons, all responses are evaluated as spike counts in 20 ms bins. For 1 ms timing comparisons, each spike sampled by a coarse-bin baseline is independently assigned to a uniformly chosen 1 ms bin within its original 20 ms interval, preserving the sampled count.

**Implementation and evaluation.** Each dataset-specific model uses a flow transformer with four blocks, hidden width 192, four attention heads, an MLP expansion ratio of four, and conditioning width 120. Our DiT-blocks are slightly modified from the canonical DiT formulation (see Figure 2 and Peebles & Xie (2023)), which led to better performance in initial testing and was inspired by LDNS (Kapoor et al., 2024). The visual models use a DINOv3 stimulus encoder (Siméoni et al., 2025), while the auditory models use a convolutional encoder on cochleagrams. Projected-CRPS weights are 0.05 and 0.01, respectively. We evaluate all models on matched neurons, stimuli, and repeats on a held-out test set along four axes: *conditional means*, using count-rate correlation, temporal PSTH correlation, and noise-corrected squared correlation (Pospisil & Bair, 2021); *marginal and shared variability*, using Fano-factor pattern and magnitude, noise-correlation pattern agreement, and covariance amplitude; *spike timing*, using agreement with recorded autocorrelograms and cross-correlograms at 1 ms resolution; and *conditional completion*, using held-out response prediction after observing selected neurons or an initial response interval. For population completion, a different-repeat control tests whether improvements depend on observing activity from the same trial. Full configurations, metric definitions, and completion protocols appear in Appendices F, I, and G, respectively.

**Training.** All models are trained with AdamW (Loshchilov & Hutter, 2019) with $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, a peak learning rate of $5 \times 10^{-4}$, and weight decay 0.01, in bf16 mixed precision. The learning rate warms up linearly, over 1,000 steps on the visual cortex data and 1,200 steps on the auditory cortex datasets, and then decays along a cosine schedule to $10^{-5}$. The stimulus encoder for the visual stimuli (i.e., DINOv3 backbone) weights are frozen during the learning rate warm up. Gradients are clipped at norm 1.0 and accumulated over four micro-batches. Following Esser et al. (2024), continuous timesteps $t \in [0, 1]$ are drawn during training from a logit-normal distribution, $t = \sigma(u)$ where $u \sim \mathcal{N}(\mu, \sigma^2)$. Appendix F lists the full configuration.

### 4.2 Main results

We ask whether modeling the full response distribution preserves the accuracy of mean-prediction models, and what it adds beyond the mean. We evaluate SpikeDiT on stimulus-conditioned mean prediction, trial-to-trial variability through noise correlations, single-neuron dispersion, and spike timing at millisecond resolution. Finally, we show that the learned joint distribution supports completion of partially observed responses without retraining.

**Mean prediction from averaged samples.** We first evaluate SpikeDiT on conditional-mean prediction, the standard benchmark for encoding models. We average its samples and compare the resulting mean response with models trained directly to predict it. On 37 visual cortex sessions, SpikeDiT outperforms the strongest baseline (Schmidt et al., 2025) on all three metrics, namely noise-corrected squared correlation (Figure 3B), count-rate correlation and temporal PSTH correlation (Figure S3). On the auditory dataset, where the species, modality, response duration, population size, and stimulus encoder all differ, SpikeDiT matches performance of the strongest baseline (Figure 3B and S3). Notably, SpikeDiT is never trained directly on mean prediction; only our CRPS term calibrates the whole count distributions and shapes the mean prediction indirectly. Modeling the full response distribution therefore does not come at the cost of stimulus-conditioned mean prediction. We next turn to the part of the response distribution that mean-prediction models do not capture.

**Noise correlations emerge from joint generation.** In Figure 3A, we compare noise correlations predicted by SpikeDiT with those measured from recorded data. Noise correlations measure how pairs of neurons fluctuate together across repeats of the same stimulus. The Poisson baseline samples neurons independently and thus cannot capture them. The latent-variable baselines capture them through low-dimensional shared latents. SpikeDiT generates the tokens of all neurons jointly, and its covariance is learned from the data with no a priori assumptions. On the 23 visual cortex

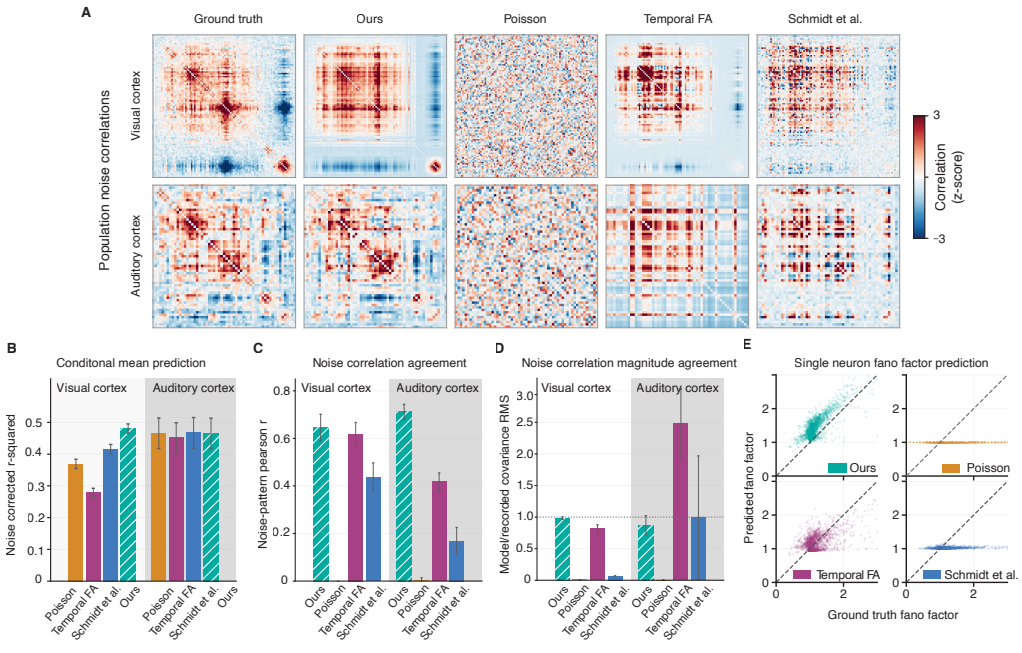


Figure 3: **Shared population variability and marginal firing statistics.** Noise-correlation matrices for one visual cortex (top) and one auditory cortex (bottom) session, z-scored within each matrix. (B) Conditional-mean prediction across visual and auditory recordings measured as noise-corrected squared correlation. (C) Agreement of off-diagonal noise-correlation patterns with the recording. (D) Covariance amplitude relative to the recording. The dotted line marks a ratio of one. (E) Predicted against recorded Fano factor per neuron for all visual cortex sessions. Dashed lines show identity.

sessions with repeated images, SpikeDiT reproduces which neuron pairs are correlated more closely than Temporal FlowFA and the Schmidt-style model (Figure 3C), with its noise-correlation pattern agreement at 0.645, against 0.619 and 0.437, respectively. The same ordering holds on auditory dataset, with a wider margin. SpikeDiT also recovers the magnitude of the covariance: Its amplitude is close to the recorded value on both datasets, while the baselines either miss it substantially or vary widely across sessions (Figure 3D). This shows that shared variability can be captured without a prescribed low-rank structure. We next examine whether SpikeDiT also captures how much each neuron varies on its own.

**SpikeDiT captures per-neuron dispersion from the data.** In Figure 3E, we compare recorded and predicted Fano factors of each neuron. Recorded neurons differ widely in how much their responses vary across repeats. The Poisson baseline by construction fixes every neuron's Fano factor at one, and the Schmidt-style model stays close to it. SpikeDiT imposes no assumption on the Fano factor and learns the per-neuron dispersion from data. On visual cortex, the correlation between predicted and recorded Fano factors is 0.87 for SpikeDiT, against 0.60 for Temporal FlowFA and 0.34 for the Schmidt-style model. On auditory cortex, SpikeDiT again predicts recorded Fano factors most accurately (Figure S4). Fano factors of neuronal responses and noise correlations are statistics of binned spike counts. Because SpikeDiT is able to model spike trains directly, we next ask whether SpikeDiT also captures spike timing accurately at millisecond resolution.

**SpikeDiT reproduces millisecond spike timing within and across neurons.** In Figure 4, we compare the fine temporal structure of generated and recorded spike trains. Within a neuron, we evaluate it with the autocorrelogram (ACG), which reflects effects such as refractoriness and bursting. Across the neuronal population, we evaluate it with the cross-correlogram (CCG), which reflects fine-scale synchrony of neuron-pairs. As SpikeDiT generates spikes directly at 1 ms and all baselines predict spike counts at 20 ms, we place each count within its bin, either at random or evenly spaced (Appendix B). Across 23 visual cortex sessions and 1,994 units, SpikeDiT reproduces ACG

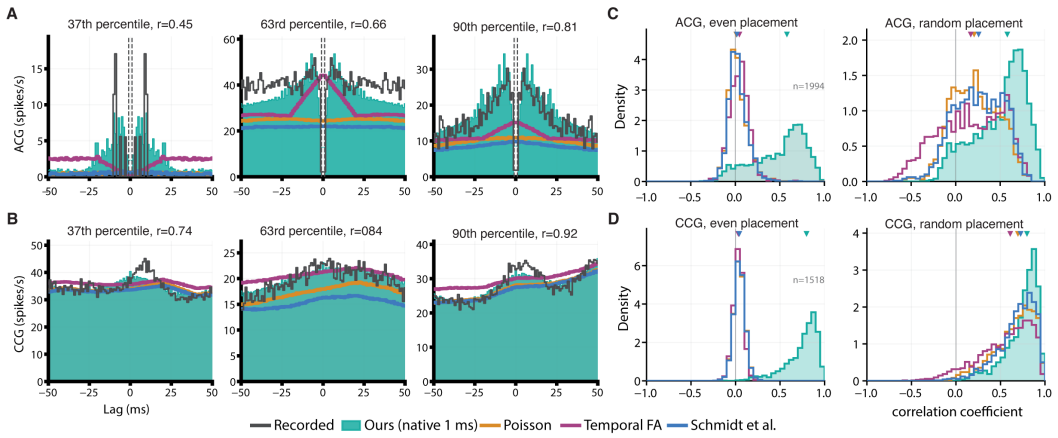


Figure 4: **Accurate spike timing at 1 ms resolution.** (A) Autocorrelograms of example visual cortex units at the 10th, 37th, 63rd, and 90th percentiles of SpikeDiT's ACG-shape agreement $r$. (B) Same for cross-correlograms of example unit pairs. (C-D) Distributions of ACG and CCG shape agreement for all units across all sessions. Baseline 20 ms counts are placed evenly or at random within each bin. SpikeDiT generates spikes at 1 ms directly. Triangles mark medians.

shapes far more closely than the baselines. Its median ACG agreement, measured as the correlation between recorded and generated ACGs, is 0.58, against at most 0.21 for any baseline under either placement rule. Its median CCG agreement is 0.80, against at most 0.70, over 1,518 unit pairs. This shows that generating spikes at native resolution recovers temporal structure that count-based models cannot express. SpikeDiT thus captures the mean, the shared and individual variability, and the fine timing of the population response conditioned on natural stimuli. We next show that SpikeDiT can also be conditioned, without any explicit training, on a subsets of neuron responses.

**Inference-time completion of partially observed responses.** Because SpikeDiT generates the population response jointly and uses a standard flow-matching pipeline, the trained model is able to complete partially observed neural activity at inference time, analogous to image completion, editing, or inpainting in text-to-image models (Rombach et al., 2022; Lugmayr et al., 2022). We use the Flower solver (Pourya et al., 2025) to constrain selected neurons or time points during flow integration while generating the remaining response tokens (Appendix G). Across all sessions, revealing increasing fractions of neurons from the same recorded trial progressively improves single-trial count prediction for the held-out population, reaching a $+0.081$ improvement on average when half of the population is observed. Revealing neurons from a different repeat does not produce this gain, so the gain comes from trial-specific shared variability and not from extra stimulus information. Revealing the first 100 ms improves prediction of counts during the following 100 ms from 0.200 to 0.246. Neither operation requires retraining. Inference methods developed for general flow models therefore apply to SpikeDiT without modification.

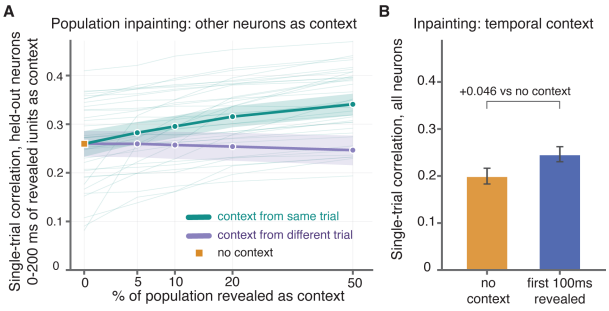


Figure 5: **Generative population inpainting** (A) Single-trial count correlation for the hidden half of the population as more of the population is revealed, using neurons from the same trial or from a different repeat. Thin lines show sessions. (B) Single-trial count correlation for 100–200 ms, with no context or with the first 100 ms revealed.

### 4.3 Ablations

**Removing the calibration term costs mean prediction and calibration.** Training the same transformer with the flow-matching loss alone lowers mean predictivity slightly, the samples become more over-dispersed, and the covariance magnitude is over-estimated (Table 1).

Table 1: Ablating the auxiliary objective and the tokenizer on the visual cortex dataset. Values are means over sessions ± s.e.m.

| | Noise-corrected $R^2$ | Predicted / recorded spike count | Fano factor | Noise-correlation pattern $r$ | Off-diagonal cov. magnitude |
|---|---|---|---|---|---|
| Recording | — | 1.000 | $1.220 \pm 0.049$ | — | 1.000 |
| SpikeDiT | $\mathbf{0.480 \pm 0.015}$ | $\mathbf{0.979 \pm 0.003}$ | $1.426 \pm 0.030$ | $0.645 \pm 0.056$ | $\mathbf{0.983 \pm 0.019}$ |
| – CRPS calibration | $0.464 \pm 0.014$ | $1.192 \pm 0.025$ | $1.470 \pm 0.027$ | $0.644 \pm 0.056$ | $1.255 \pm 0.047$ |
| – Tokenizer regularization | $0.189 \pm 0.011$ | $0.701 \pm 0.017$ | $1.113 \pm 0.027$ | $0.050 \pm 0.021$ | $0.056 \pm 0.008$ |

**Using a less regularized tokenizer makes the model collapse on all metrics.** Replacing the neural response tokenizer by a variant pretrained without SIGReg regularization but with the same reconstruction performance, and with every other component fixed, makes the model collapse on all metric families (Table 1, Appendix J).

## 5 Discussion

**Summary.** Encoding models of neural data are commonly judged by how well they predict the mean response. We instead made the full stimulus-conditioned distribution of population *spike trains* the predictive target, learned jointly across neurons and time at 1 ms resolution with few assumptions about the structure of variability. In macaque V4 and ferret auditory cortex, our model matches or exceeds the strongest baselines on the mean response, outperforms latent-variable models on noise correlations and single-neuron dispersion, and generates millisecond spike timing that count-based models cannot. Encoding models thus no longer have to trade stimulus-driven accuracy for realistic variability, which becomes a prediction target rather than residual noise. We achieve this by transferring the latent generative recipe of image and video synthesis to neural populations: a frozen tokenizer maps each neuron's spike train to a continuous latent space, in which a stimulus-conditioned flow transformer generates population responses. Methods developed for this recipe carry over directly, as training-free completion of partially observed recordings shows. Reconstruction accuracy alone does not make a tokenizer suitable for generation: without latent regularization, a tokenizer that reconstructed spikes almost as well caused the generator to collapse (Section 4.3). With interchangeable stimulus encoders, the same generator applies across modalities and population sizes and can be trained on growing collections of recordings.

**Limitations.** The flexibility of our framework has four main costs. First, the model has no tractable likelihood. Its marginals can be estimated only approximately, so we compare statistics of generated and recorded responses, each testing one aspect of the distribution, whereas a likelihood scores all aspects at once (Bashiri et al., 2021). Second, the model has no interpretable latent state. Shared variability is present in samples but, unlike in latent-variable models (Bashiri et al., 2021; Schmidt et al., 2025), cannot be read out as a few variables per trial and e.g. regressed to behavior. Third, predicting unobserved neurons from observed ones is exact in Gaussian latent models (Yu et al., 2009; Bashiri et al., 2021), whereas our model approximates it by constraining generation to match the observed neurons (Pourya et al., 2025). Fourth, the model is bounded by tokenizer reconstruction, including our (defensible) assumption of one spike per millisecond, and uses stimulus context within the response window, so it describes acausal encoding. Finally, each model was trained with a single seed.

**Variability as integral part of the neural code.** Retaining spike resolution and shared variability across neurons may benefit foundation models for neural encoding, as spike-level tokenization has for decoding (Azabou et al., 2023), provided the approach scales with data and capacity. Because generated responses resemble recorded ones in their single-trial statistics, they can also support in silico experiments that depend on variability, for example to plan how many trials and neurons an experiment needs or to test analysis pipelines before recording. Ultimately, single samples of population activity are what the brain produces. They arise from recurrent dynamics in which feedforward input and feedback carrying context and priors interact on millisecond timescales, and if perception relies on sampling from an internal generative model (Berkes et al., 2011; Orbán et al., 2016), their variability is where these priors should become visible. Generative encoding models thus need to describe the neural code in the form the brain uses, as stochastic spike trains correlated across neurons.

### AI USE STATEMENT

We used generative AI tools to help implement our method. Coding agents assisted in writing parts of the initial code for the model architecture and for data reformatting. Coding agents also helped us write configuration files, launch and manage experiments. The authors reviewed every experiment configuration directly. We did not use generative AI to form hypotheses, design the methodology or experiments, or in interpreting main results. Synthetic data generation, proof assistance, and translation do not apply to this work.

Furthermore, generative AI helped us organize and draft the outline of the first version of this manuscript, which we then revised. We reviewed and tested all AI-generated code before using it in any reported analysis, modifying it where needed. The authors take complete responsibility for the final content of this work, including text, claims and all artifacts produced with the aid of generative AI.

### REPRODUCIBILITY STATEMENT

The code for this submission is available at `https://anonymous.4open.science/r/neu-dit-BCB0`. With it, one can download, process and prepare the two public datasets we use: macaque V4 recordings (Cowley et al., 2026) and ferret auditory cortex recordings (Wingert et al., 2026). The repository implements the flow-matching model, the Poisson, temporal factor analysis and Schmidt et al. baselines, as well as the evaluation metrics and the scripts behind each figure and analysis. The neural data tokenizer module and the model checkpoints belong to a companion paper and will be fully released publicly upon acceptance with the rest of the codebase, after which training, inference and figure regeneration can be reproduced end to end.

## REFERENCES

Anonymous et al. Evaluating neural data tokenizers: A framework for assessing learned representations of spiking activity. Manuscript under review, 2026.

Ján Antolík, Sonja B. Hofer, James A. Bednar, and Thomas D. Mrsic-Flogel. Model constrained by visual hierarchy improves prediction of neural responses to natural scenes. *PLOS Computational Biology*, 12(6):e1004927, 2016. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1004927. URL `http://dx.doi.org/10.1371/journal.pcbi.1004927`.

Amos Arieli, Alexander Sterkin, Amiram Grinvald, and Ad Aertsen. Dynamics of ongoing activity: explanation of the large variability in evoked cortical responses. *Science*, 273(5283):1868–1871, 1996. doi: 10.1126/science.273.5283.1868.

Mehdi Azabou, Vinam Arora, Venkataramana Ganesh, Ximeng Mao, Santosh Nachimuthu, Michael J. Mendelson, Blake Richards, Matthew G. Perich, Guillaume Lajoie, and Eva L. Dyer. A Unified, Scalable Framework for Neural Population Decoding, 2023.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics, 2025. URL `https://arxiv.org/abs/2511.08544`.

Mohammad Bashiri, Edgar Walker, Konstantin-Klemens Lurz, Akshay Jagadish, Taliah Muhammad, Zhiwei Ding, Zhuokun Ding, Andreas Tolias, and Fabian H. Sinz. A flow-based latent state generative model of neural population responses to natural images. In *Advances in Neural Information Processing Systems*, volume 34, pp. 15801–15815, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/84a529a92de322be42dd3365afd54f91-Abstract.html`.

Eleanor Batty, Josh Merel, Nora Brackbill, Alexander Heitman, Alexander Sher, Alan Litke, E.J. Chichilnisky, and Liam Paninski. Multilayer recurrent network models of primate retinal ganglion cell responses. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=HkEI22jeg`.

Maxime Beau, Federico D'Agostino, Ago Lajko, Gabriela Martínez, Michael Häusser, and Dimitar Kostadinov. NeuroPyxels: loading, processing and plotting Neuropixels data in python, September 2021. URL `https://doi.org/10.5281/zenodo.5509776`.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. URL https://arxiv.org/abs/1308.3432.

Pietro Berkes, Gergő Orbán, Máté Lengyel, and József Fiser. Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science*, 331(6013):83–87, 2011. doi: 10.1126/science.1195870.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space, June 2025.

Santiago A. Cadena, George H. Denfield, Edgar Y. Walker, Leon A. Gatys, Andreas S. Tolias, Matthias Bethge, and Alexander S. Ecker. Deep convolutional models improve predictions of macaque V1 responses to natural images. *PLoS Computational Biology*, 15(4):e1006897, 2019.

Marlene R. Cohen and Adam Kohn. Measuring and interpreting neuronal correlations. *Nature Neuroscience*, 14(7):811–819, 2011.

Benjamin R. Cowley, Patricia L. Stan, Jonathan W. Pillow, and Matthew A. Smith. Compact deep neural network models of the visual cortex. *Nature*, 652(8111):947–954, 2026. ISSN 1476-4687. doi: 10.1038/s41586-026-10150-1. URL http://dx.doi.org/10.1038/s41586-026-10150-1.

Alexander S. Ecker, Philipp Berens, R. James Cotton, Manivannan Subramaniyan, George H. Denfield, Cathryn R. Cadwell, Stelios M. Smirnakis, Matthias Bethge, and Andreas S. Tolias. State dependence of noise correlations in macaque primary visual cortex. *Neuron*, 82(1):235–248, 2014.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024.

Zach Evans, C. J. Carr, Josiah Taylor, Scott H. Hawley, and Jordi Pons. Fast timing-conditioned latent audio diffusion. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Robbe L. T. Goris, J. Anthony Movshon, and Eero P. Simoncelli. Partitioning neuronal variability. *Nature Neuroscience*, 17(6):858–865, 2014.

Patrik Hoyer and Aapo Hyvärinen. Interpreting neural response variability as monte carlo sampling of the posterior. In S. Becker, S. Thrun, and K. Obermayer (eds.), *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2002. URL https://proceedings.neurips.cc/paper_files/paper/2002/file/a486cd07e4ac3d270571622f4f316ec5-Paper.pdf.

Jaivardhan Kapoor, Auguste Schulz, Julius Vetter, Felix Pei, Richard Gao, and Jakob H. Macke. Latent diffusion for neural spiking data. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3751. URL https://papers.nips.cc/paper_files/paper/2024/hash/d60b6b7f0ba6bf07d975b3bbdacea702-Abstract-Conference.html.

Stephen L. Keeley, Mikio C. Aoi, Yiyi Yu, Spencer L. Smith, and Jonathan W. Pillow. Identifying signal and noise structure in neural population activity with Gaussian process factor models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 13795–13805, 2020.

David A. Klindt, Alexander S. Ecker, Thomas Euler, and Matthias Bethge. Neural system identification for large populations separating "what" and "where". In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, September 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11451–11461. IEEE, 2022.

Konstantin-Klemens Lurz, Mohammad Bashiri, Konstantin Willeke, Akshay Jagadish, Eric Wang, Edgar Y. Walker, Santiago A Cadena, Taliah Muhammad, Erick Cobos, Andreas S. Tolias, Alexander S Ecker, and Fabian H. Sinz. Generalization in data-driven models of primary visual cortex. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Tp7kI90Htd.

Jakob H. Macke, Lars Buesing, John P. Cunningham, Byron M. Yu, Krishna V. Shenoy, and Maneesh Sahani. Empirical models of spiking in neural populations. In *Advances in Neural Information Processing Systems*, volume 24, pp. 1350–1358, 2011.

Lane McIntosh, Niru Maheswaranathan, Aran Nayebi, Surya Ganguli, and Stephen Baccus. Deep learning models of the retinal response to natural scenes. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/a1d33d0dfec820b41b54430b50e96b5c-Paper.pdf.

Manuel Molano-Mazon, Arno Onken, Eugenio Piasini, and Stefano Panzeri. Synthesizing realistic neural population activity patterns using generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1VVsebAZ.

Gergő Orbán, Pietro Berkes, József Fiser, and Máté Lengyel. Neural variability and sampling-based probabilistic representations in the visual cortex. *Neuron*, 92(2):530–543, 2016. doi: 10.1016/j.neuron.2016.09.038.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, 2023.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer, 2018.

Jonathan W. Pillow, Jonathon Shlens, Liam Paninski, Alexander Sher, Alan M. Litke, E. J. Chichilnisky, and Eero P. Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, 2008. doi: 10.1038/nature07140. URL https://doi.org/10.1038/nature07140.

Dean A. Pospisil and Wyeth Bair. The unbiased estimation of the fraction of variance explained by a model. *PLOS Computational Biology*, 17(8):e1009212, 2021. doi: 10.1371/journal.pcbi.1009212.

Mehrsa Pourya, Bassam El Rawas, and Michael Unser. Flower: A flow-matching solver for inverse problems, 2025.

Poornima Ramesh, Mohamad Atayi, and Jakob H. Macke. Adversarial training of neural encoding models on population spike trains. In *Conference on Cognitive Computational Neuroscience*, 2019. doi: 10.32470/CCN.2019.1263-0. URL https://doi.org/10.32470/CCN.2019.1263-0.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2022.

Finn Schmidt, Polina Turishcheva, Suhas Shrinivasan, and Fabian H. Sinz. Modeling dynamic neural activity by combining naturalistic video stimuli and stimulus-independent latent factors. In *Advances in Neural Information Processing Systems*, volume 38, pp. 20252–20282, 2025. doi: 10.52202/085713-0684. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/1d485d337d5622bdb1462333f4247522-Abstract-Conference.html.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025. URL https://arxiv.org/abs/2508.10104.

Fabian Sinz, Alexander S. Ecker, Paul Fahey, Edgar Walker, Erick Cobos, Emmanouil Froudarakis, Dimitri Yatsenko, Zachary Pitkow, Jacob Reimer, and Andreas Tolias. Stimulus domain transfer in recurrent models for large scale cortical population prediction on video. In *Advances in Neural Information Processing Systems*, volume 31, pp. 7199–7210, 2018.

Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Charu Bai Reddy, Matteo Carandini, and Kenneth D. Harris. Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437):eaav7893, 2019. doi: 10.1126/science.aav7893.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models, 2025.

Eric Y Wang, Paul G Fahey, Zhuokun Ding, Stelios Papadopoulos, Kayla Ponder, Marissa A Weis, Andersen Chang, Taliah Muhammad, Saumil Patel, Zhiwei Ding, Dat Tran, Jiakun Fu, Casey M Schneider-Mizell, MICrONS Consortium, R Clay Reid, Forrest Collman, Nuno Maçarico da Costa, Katrin Franke, Alexander S Ecker, Jacob Reimer, Xaq Pitkow, Fabian H Sinz, and Andreas S Tolias. Foundation model of neural activity predicts response to new stimulus types. *Nature*, 640(8058):470–477, April 2025.

Ganchao Wei and John Pearson. Flow matching for count data, 2026. URL https://arxiv.org/abs/2605.07746.

Konstantin F. Willeke, Polina Turishcheva, Alex Gilbert, Goirik Chakrabarty, Hasan A. Bedel, Paul G. Fahey, Yongrong Qiu, Marissa A. Weis, Michaela Vystrčilová, Taliah Muhammad, Lydia Ntanavara, Rachel E. Froebe, Kayla Ponder, Zheng Huan Tan, Emin Orhan, Erick Cobos, Sophia Sanborn, Katrin Franke, Fabian H. Sinz, Alexander S. Ecker, and Andreas S. Tolias. OmniMouse: Scaling properties of multi-modal, multi-task brain models on 150B neural tokens, 2026.

Jereme C. Wingert, Satyabrata Parida, Sam V. Norman-Haignere, and Stephen V. David. Convolutional neural network models describe the encoding subspace of local circuits in auditory cortex. *Nature Neuroscience*, 29(4):876–887, 2026. ISSN 1546-1726. doi: 10.1038/s41593-026-02216-0. URL http://dx.doi.org/10.1038/s41593-026-02216-0.

Daniel L. K. Yamins, Ha Hong, Charles F. Cadieu, Ethan A. Solomon, Darren Seibert, and James J. DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014. doi: 10.1073/pnas.1403112111.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. Cogvideox: Text-to-video diffusion models with an expert transformer. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LQzN6TRFg9.

Byron M. Yu, John P. Cunningham, Gopal Santhanam, Stephen I. Ryu, Krishna V. Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Journal of Neurophysiology*, 102(1):614–635, 2009.

# A COMPARISON OF POPULATION GENERATIVE MODELS

For stimulus $x$, recording sessions $s$, and $\mathbf{Y} \in \{0,1\}^{N_s \times T}$ the binary population response on a 1 ms grid, the Poisson, Temporal FlowFA and Schmidt models all have a native resolution of 20 ms bins, while FlowFA predicts a single response per window (200 ms). In the corresponding sections, we use $\tilde{t}$ to indicate the bin indices on these coarser grids, $\tilde{y}_{i,\tilde{t}} \in \mathbb{N}_0$ for the spike count of unit $i$ in bin $\tilde{t}$, and $\tilde{\mathbf{Y}} = \{\tilde{y}_{i,\tilde{t}}\} \in \mathbb{N}_0^{N_s \times \tilde{T}}$ for the binned response, where $\tilde{T}$ is the number of bins per response window.

**Independent Poisson.** The Poisson baseline has no shared stochastic component: it treats every unit and bin (20 ms) as conditionally independent given the stimulus,

$$\tilde{y}_{i,\tilde{t}} \mid x, s \;\sim\; \mathrm{Poisson}\big(\lambda_{i,\tilde{t}}(x,s)\big),$$

where $\lambda_{i,\tilde{t}}$ is the predicted rate in bin $\tilde{t}$. Each count's variance equals its mean.

**FlowFA (Bashiri et al., 2021).** FlowFA models a single count, $\tilde{y}_i$, per unit per response window ($\tilde{T} = 1$). Each trial draws a latent response in a Gaussian space,

$$v_i = \mu_i(x,s) + c_i^\top z + \epsilon_i, \qquad z \sim \mathcal{N}(0, I_K), \quad \epsilon_i \sim \mathcal{N}(0, \psi_i),$$

so that $\mathbf{v} \sim \mathcal{N}\big(\boldsymbol{\mu},\, CC^\top + \mathrm{diag}\,\boldsymbol{\psi}\big)$ with loadings $c_i \in \mathbb{R}^K$ (learned per session) forming the rows of $C$. A fixed marginal transform maps this latent response to a non-negative continuous response which is then quantised to a count, $\tilde{y}_i = \big\lfloor g_i^{-1}(v_i) \big\rfloor$ (here $g$ is the square root). A single $K$-dimensional latent $z$ drawn once per trial is shared by all units, giving shared variability across units through the low-rank covariance $CC^\top$. Because the model has no time axis, it represents no variability across time, and its dependence is restricted to Gaussian, low-rank structure on the square-root-transformed window counts $v_i$.

**Temporal FlowFA.** Temporal FlowFA extends FlowFA to $\tilde{T}$ time bins by giving each shared factor a smooth random time course,

$$v_{i,\tilde{t}} = \mu_{i,\tilde{t}}(x,s) + \sum_{k=1}^{K} L_{ik}\, h_k(\tilde{t}) + \epsilon_{i,\tilde{t}}, \qquad \mathbf{h}_k = \Phi\, \mathbf{a}_k, \quad \mathbf{a}_k \sim \mathcal{N}(0, \Sigma_k), \quad \epsilon_{i,\tilde{t}} \sim \mathcal{N}(0, \psi_i),$$

with counts $\tilde{y}_{i,\tilde{t}} = \lfloor g_i^{-1}(v_{i,\tilde{t}}) \rfloor$ after the same dequantization. Here $L_{ik}$ are signed loadings, $\Phi \in \mathbb{R}^{\tilde{T} \times Q}$ is a fixed smooth temporal basis with $Q$ degrees of freedom per factor, $\Sigma_k$ is a learned $Q \times Q$ covariance, and the private variance $\psi_i$ and the learned log transform $g_i$ are shared across bins; the predicted rate sets the median of the transformed response rather than its mean. The $K$ latent time courses drawn on each trial couple units and bins jointly, with covariance $\mathrm{Cov}(v_{i,\tilde{t}}, v_{j,\tilde{r}}) = \sum_k L_{ik} L_{jk}\, [\Phi \Sigma_k \Phi^\top]_{\tilde{t}\tilde{r}} + \delta_{ij}\delta_{\tilde{t}\tilde{r}}\psi_i$ of rank at most $KQ$. This dependence is restricted to Gaussian, low-rank spatiotemporal structure after the marginal transform, with temporal structure limited to the span of $\Phi$.

**Schmidt et al. (2025).** The Schmidt model draws an independent shared state in every bin,

$$\tilde{y}_{i,\tilde{t}} \mid x, s, z_{\tilde{t}} \;\sim\; p_{\mathrm{obs}}\big(\cdot \mid \lambda_{i,\tilde{t}}\big), \qquad \lambda_{i,\tilde{t}} = \ell\big(f_{i,\tilde{t}}(x,s),\, w_i^\top z_{\tilde{t}}\big), \qquad z_{\tilde{t}} \overset{\text{iid}}{\sim} \mathcal{N}(0, I_K),$$

where $f_{i,\tilde{t}}$ is the stimulus drive, $w_i \in \mathbb{R}^K$ are unit loadings, $\ell$ combines drive and latent into a rate, and $p_{\mathrm{obs}}$ is a count observation model applied independently given $z_{\tilde{t}}$. Marginalising the $K$-dimensional state induces shared variability across units within a bin. Because $z_{\tilde{t}}$ and $z_{\tilde{r}}$ are independent for $\tilde{t} \neq \tilde{r}$ under the prior, however, shared fluctuations are not correlated across time, and observations are conditionally independent given the state.

**SpikeDiT (ours).** SpikeDiT generates the 1 ms response directly,

$$u_0 \sim \mathcal{N}(0, I), \qquad \dot{u}_\tau = v_\theta\big(u_\tau, \tau, c(x,s)\big), \qquad \mathbf{Y} \sim p_{\mathrm{dec}}(\cdot \mid u_1),$$

where $u \in \mathbb{R}^{N_s \times L \times D}$ is the token grid of Section 3.1, $v_\theta$ is the learned velocity field, and $p_{\mathrm{dec}}$ is the frozen tokenizer decoder. By not explicitly parameterizing a low dimensional shared state, but rather generating all tokens jointly, the flow model is able to freely learn stimulus-dependent variability across neurons and time.

## B PROBABILISTIC BASELINE IMPLEMENTATIONS

Here we provide the architectural and implementation details for each baseline. All baseline models use the same stimulus-encoder described for SpikeDiT, and the same cross-attention readout, adding unique output heads on top of the readout (and trained separately end-to-end). Baselines generally share the initialization and training settings described in Section 4.1, except for the number of training epochs.

Unlike SpikeDiT, the baselines feed a single query per neuron to the cross-attention (instead of tiling queries along the temporal dimension), yielding one readout vector $o_i \in \mathbb{R}^{d_q}$ per unit and window. On Wingert, the readout is applied separately in non-overlapping 100 ms sub-chunks $w$ of the 1 s target segment. Each baseline maps these vectors to the stimulus-dependent parameters of its response distribution, one value $\eta_{i,\tilde{t}}$ per output bin: on Cowley, a linear projection $W \in \mathbb{R}^{\tilde{T}\times d_q}$ shared across neurons and sessions maps $o_i$ to the trial's bins (ten 20 ms bins, or a single 200 ms bin for FlowFA), while on Wingert a shared FiLM-modulated residual MLP (Perez et al., 2018), conditioned on the unit's query, maps each $o_{i,w}$ to five 20 ms bins from its sub-chunk (which FlowFA then sums over each 200 ms window). Both add a per-unit bias $b_i$. Every baseline then passes $\eta_{i,\tilde{t}}$ through a Softplus activation function, with $\beta$ of $0.1$ or $1$ on Cowley and Wingert, respectively.

**Independent Poisson.** The Poisson baseline (Appendix A) is trained with the Poisson negative log-likelihood, averaged over all units and bins in a batch,

$$\mathcal{L}_{\text{Pois}} = \left\langle \lambda_{i,\tilde{t}} - \tilde{y}_{i,\tilde{t}} \log \lambda_{i,\tilde{t}} \right\rangle_{i,\tilde{t}},$$

with the Softplus outputs serving as the predicted rates,

$$\lambda_{i,\tilde{t}} = \text{softplus}_\beta(\eta_{i,\tilde{t}}).$$

For Cowley we train for 60 epochs. As in SpikeDiT, 2 s of stimulus are presented for Wingert but only the final 1 s of predicted responses are included in the loss. Samples are independent Poisson counts drawn from these rates.

**FlowFA.** FlowFA is trained with the exact Gaussian negative log-likelihood of the dequantised, square-root-transformed counts $v_i = g(\tilde{y}_i + \upsilon_i)$ , $\upsilon_i \sim \mathcal{U}[0,1)$ (see Appendix A), including the log-Jacobian of the transform, which lower-bounds the count likelihood:

$$\mathcal{L}_{\text{FA}} = -\log \mathcal{N}\big(\mathbf{v};\, \boldsymbol{\mu},\, CC^\top + \operatorname{diag}\boldsymbol{\psi}\big) - \sum_i \log g'(\tilde{y}_i + \upsilon_i),$$

with the Softplus outputs, summed over the response window, giving each unit's Gaussian mean in square-root space,

$$\mu_i = \textstyle\sum_{\tilde{t}} \text{softplus}_\beta(\eta_{i,\tilde{t}}).$$

On Cowley the response window is the 200 ms trial and $\tilde{T} = 1$, so $\mu_i = \text{softplus}_\beta(\eta_i)$. On Wingert each 1 s chunk is split into five 200 ms windows, treated as independent, and the sum runs over the ten 20 ms bins of a window (five from each of its two 100 ms sub-chunks). For Cowley we train for 40 epochs. The dependence parameters do not come from the readout: the loadings $C$ ($K = 3$) and private variances $\boldsymbol{\psi}$ are learned per session, with the loadings initialised non-negative, and are held fixed for the first 2,000 steps while the mean trains. Samples of $v_i$ are drawn in Gaussian space, squared and floored, yielding predictions of $\tilde{y}_i$. Note that for Wingert count mean results, instead of sampling we use the closed form, $\mathbb{E}[\tilde{y}_i] = \sum_{c\geq 1} \Pr\big(v_i \geq \sqrt{c}\big)$.

**Temporal FlowFA.** Temporal FlowFA is trained with the same dequantised Gaussian likelihood, now jointly over the $\tilde{T} = 10$ bins of a 200 ms window, with $v_{i,\tilde{t}} = g_i(\tilde{y}_{i,\tilde{t}} + \upsilon_{i,\tilde{t}})$ and the spatiotemporal covariance $\Sigma_{\text{TFA}}$ of Appendix A:

$$\mathcal{L}_{\text{TFA}} = -\log \mathcal{N}\big(\operatorname{vec}\mathbf{v};\, \operatorname{vec}\boldsymbol{\mu},\, \Sigma_{\text{TFA}}\big) - \sum_{i,\tilde{t}} \log g_i'(\tilde{y}_{i,\tilde{t}} + \upsilon_{i,\tilde{t}}),$$

with each Softplus output mapped through the *learned* marginal transform to the Gaussian location of its bin,

$$\mu_{i,\tilde{t}} = g_i\big(\mathrm{softplus}_\beta(\eta_{i,\tilde{t}})\big).$$

Unlike the static counterpart, Temporal FlowFA, the transform is a per-unit log with *learned* affine parameters

$$g_i(y) = a_i^{(2)} \sinh\Big(b_i \,\mathrm{asinh}\big(a_i^{(1)} \log y + t_i^{(1)}\big) - c_i\Big) + t_i^{(2)}, \qquad a_i^{(1)}, a_i^{(2)}, b_i > 0.$$

The parameters are initialised such that $g_i$ starts as $\log y$. Moreover, within the temporal covariance parameters, each factor's time course is spanned by a fixed $Q = 3$ orthogonal-polynomial basis over the ten bins of a window, with a learned covariance $\Sigma_k = G_k G_k^\top$ (initialised to the identity).

***Schmidt et al.*'s latent model.** The Schmidt model (Appendix A) is trained by maximising an evidence lower bound with an amortised posterior $q_\phi(\mathbf{z}_{1:\tilde{T}} \mid \tilde{\mathbf{Y}}, \mathbf{m})$, i.e. by minimising

$$\mathcal{L}_{\mathrm{ELBO}} = \frac{1}{N_s \tilde{T}}\Big(-\sum_{i,\tilde{t}} \log \mathrm{Poisson}\big(\tilde{y}_{i,\tilde{t}};\, \lambda_{i,\tilde{t}}\big) + \beta_{\mathrm{KL}}\, \mathrm{KL}\big(q_\phi(\mathbf{z}_{1:\tilde{T}} \mid \tilde{\mathbf{Y}}, \mathbf{m}) \,\big\|\, \mathcal{N}(0, I)\big)\Big),$$

averaged over the batch, with $\mathbf{z}_{1:\tilde{T}}$ a single reparameterised sample from $q_\phi$ and the Softplus outputs setting the stimulus-driven rate that the shared state scales multiplicatively,

$$\lambda_{i,\tilde{t}} = \mathrm{softplus}_\beta(\eta_{i,\tilde{t}})\, \exp\big(w_i^\top z_{\tilde{t}}\big).$$

Here $\eta_{i,\tilde{t}}$ plays the role of the stimulus drive $f_{i,\tilde{t}}$ in Appendix A, with $K = 12$, one state per 20 ms bin, and signed per-session loadings $w_i$ initialised from $\mathcal{N}(0, 0.1^2)$ and excluded from weight decay. The posterior sits beside the readout rather than on it: a session-specific linear map of the masked responses and their mask $\mathbf{m}$ feeds a shared gated recurrent unit (GRU) that outputs a Gaussian over $\mathbf{z}_{1:\tilde{T}}$. A random fraction of units, drawn uniformly between 25% and 75%, is observed in each training step, the KL term is computed in closed form, and its weight $\beta_{\mathrm{KL}}$ is ramped from 0 to 1 over the first 2,000 steps. For Cowley we train for 60 epochs. At evaluation the posterior is not used: samples draw $z_{\tilde{t}}$ from the prior and then Poisson counts, and the closed-form mean is $\mathbb{E}[\tilde{y}_{i,\tilde{t}}] = \mathrm{softplus}_\beta(\eta_{i,\tilde{t}}) \exp(\|w_i\|^2/2)$. Relative to Schmidt et al. (2025), the calcium observation model is replaced by a Poisson law for spike counts and the state scales the rate multiplicatively rather than entering before the nonlinearity; with $K = 0$ the model reduces exactly to the Poisson baseline.

**Fine-grid emission for count baselines.** As noted in Appendix A, each of the baselines emit response predictions on a courser grid than the 1 ms resolution of SpikeDiT. Accordingly, to compare the baselines on fine-grained metrics, we apply two different upsampling strategies for estimating within-bin timing. Under the first strategy, *random*, given the spike count $c$ for a course bin of width $t_b$ ms we place each of the $c$ spikes into one of the $t_b$ millisecond bins, sampled uniformly with replacement (so multiple spikes can share a slot). For the second, *even*, we distribute the spikes evenly to bins with deterministic offsets, $\lfloor (k+\frac{1}{2})\, t_b/n \rfloor$, $k = 0, \ldots, n-1$. FlowFA is excluded from fine-grid analyses, since with one count per 200 ms window its spike timing would be determined entirely by the placement rule.

## C Training and sampling hyperparameters

Table S1 lists the configuration of the two primary SpikeDiT models. The tokenizer and flow-transformer architecture is identical for both datasets, whereas the stimulus encoder, readout details, projected-CRPS dose and training schedule are dataset-specific.

Table S1: Hyperparameters of the primary models. Both use seed 42 and the terminal checkpoint.

| | Cowley (V4) | Wingert (auditory) |
|---|---|---|
| *Data* | | |
| Stimulus input | RGB image, $224 \times 224$ | 32-channel cochleagram, 100 Hz |
| Response window | 0–200 ms after onset | 1 s segment + 1 s preceding sound |
| Response bin | 1 ms | 1 ms |
| *Tokenizer (frozen, shared)* | | |
| Token length / width | 100 ms / $D = 16$ | 100 ms / $D = 16$ |
| Tokens per unit per trial | $L = 2$ | $L = 10$ |
| Latent scale (train-calibrated) | 0.5037 | 0.4929 |
| *Stimulus encoder* | | |
| Architecture | DINOv3 ViT-S/16 | Dilated conv. (rank 96, kernel 7, dilations 1/2/4/8) + 2 self-attn. layers |
| Initialisation | DINOv3 pretrained | None |
| Frozen steps | 1,000 (then all layers trained) | 0 (all layers trained) |
| Encoder warm-up steps | 500 (LR ramp after unfreezing) | – (uses global warm-up) |
| Encoder LR multiplier | 0.1 | 1.0 |
| *Readout* | | |
| Cross-attention heads / MLP ratio | 4 / 4 | 4 / 4 |
| Query / attention width (heads $\times$ head dim) | 192 ($4 \times 48$) | 384 ($4 \times 96$) |
| Readout-token self-attention | 1 block, 4 heads | none |
| Session / subject embedding | yes / yes | no / yes |
| Conditioning width | 120 | 120 |
| *Flow transformer* | | |
| Blocks / hidden width / heads | 4 / 192 / 4 | 4 / 192 / 4 |
| MLP ratio / temporal conv. kernel | 4 / 3 | 4 / 3 |
| Flow-time distribution | logit-normal$(-1, 0.5)$ | logit-normal$(-1, 0.5)$ |
| Flow-time draws per example | 4 | 1 |
| *Projected CRPS* | | |
| Dose $\alpha$ | 0.05 | 0.01 |
| Weights: count / spatial level / spatial contrast / temporal level / temporal contrast | 5.75e-3 / 3.5e-4 / 2.15e-2 / 3.7e-4 / 5.5e-2 | 1.15e-3 / 7.0e-5 / 4.3e-3 / 7.4e-5 / 1.1e-2 |
| Applied every / draws per bank | 4 batches / 8 | 4 batches / 4 |
| Training-time sampler | Euler, 5 steps | Euler, 5 steps |
| *Optimisation* | | |
| Optimiser | AdamW, $\beta = (0.9, 0.999)$, wd 0.01 | AdamW, $\beta = (0.9, 0.999)$, wd 0.01 |
| Learning rate / schedule | $5 \times 10^{-4}$, cosine to $10^{-5}$ | $5 \times 10^{-4}$, cosine to $10^{-5}$ |
| Warm-up | 1,000 steps | 1,200 steps |
| Batch | 64 images $\times$ 4 accum. | 8 sequences $\times$ 4 crops $\times$ 4 accum. |
| Training length | 50 epochs (76,700 steps) | 60,000 steps |
| Precision / gradient clipping | bf16 mixed / 1.0 | bf16 mixed / 1.0 |
| *Evaluation sampling* | | |
| Sampler | Euler, 20 steps | Euler, 20 steps |
| Draws per presentation | 256 | 256 |

## D MONTE-CARLO ESTIMATION AND THE CHOICE OF $K$.

Our model is generative: for a given stimulus it emits spike trains, not firing rates or the continuous estimate of a Poisson $\lambda$. As such, every quantity we compare with the recording is an expectation under the model's conditional distribution. That distribution is defined only implicitly, by the flow head and the tokenizer decoder, so none of these expectations has a closed form. We estimate each by Monte-Carlo integration, averaging over $K$ independent draws per stimulus presentation. The sampled baselines are evaluated in the same way.

Finite $K$ adds sampling noise to the estimates, and that noise lowers correlation-based scores. A noisy estimate of the mean, for instance, correlates less well with the recording than the true mean does. The reported scores are therefore conservative, and they rise towards their limit as $K$ grows. The Poisson baseline provides a check, because its conditional mean is known in closed form and so gives the exact $K \to \infty$ value.

Figures S1 and S2 score both our model and the Poisson baseline at $K = 16, 32, \ldots, 256$, on both datasets. The noise-corrected $r^2$ saturates. On Cowley its gap to Poisson's exact limit roughly halves with each doubling of $K$, as expected for a bias that decays as $1/K$, and at $K = 256$ it is 0.0035, under a quarter of a session SEM. For Ours, the last doubling (128 to 256) adds 0.004. On Wingert the curves are flat by $K = 128$: Poisson ends within $10^{-4}$ of its limit, and Ours rises by only 0.004 over the whole range and not at all from $K = 128$ to $K = 256$. Wingert's one-second counting windows hold many more spikes, which is consistent with this faster convergence. The noise-pattern correlation barely depends on $K$ on either dataset: Ours moves from 0.639 to 0.645 on Cowley and is flat from $K = 32$ on Wingert. At $K = 256$ further draws would change no reported number beyond its session-level uncertainty. We therefore use $K = 256$ throughout, and draw a second, independent bank of 256 only to check reproducibility.

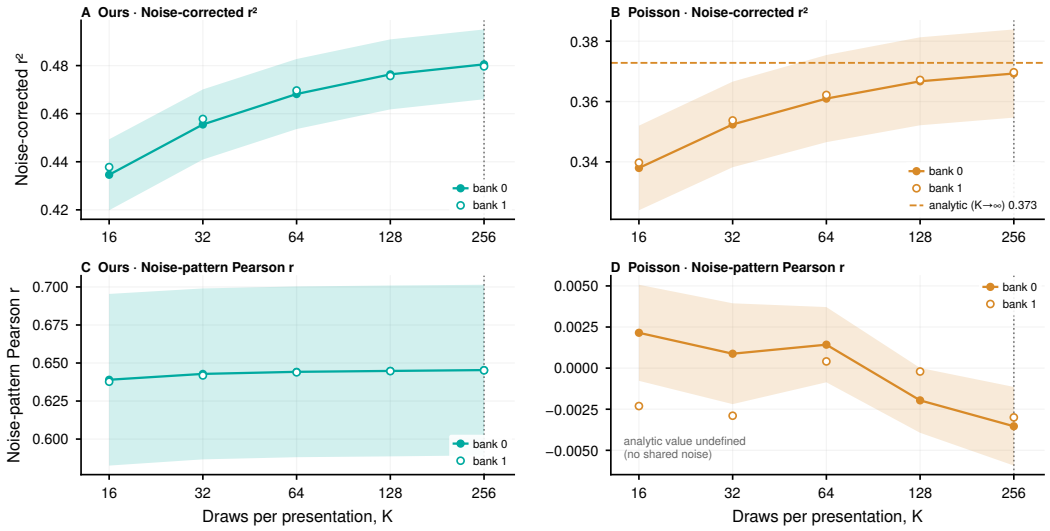


Figure S1: **Convergence of representative mean prediction metrics for our model and the Poisson baseline.** Here shown for the macaque V4 dataset (Cowley et al., 2026).

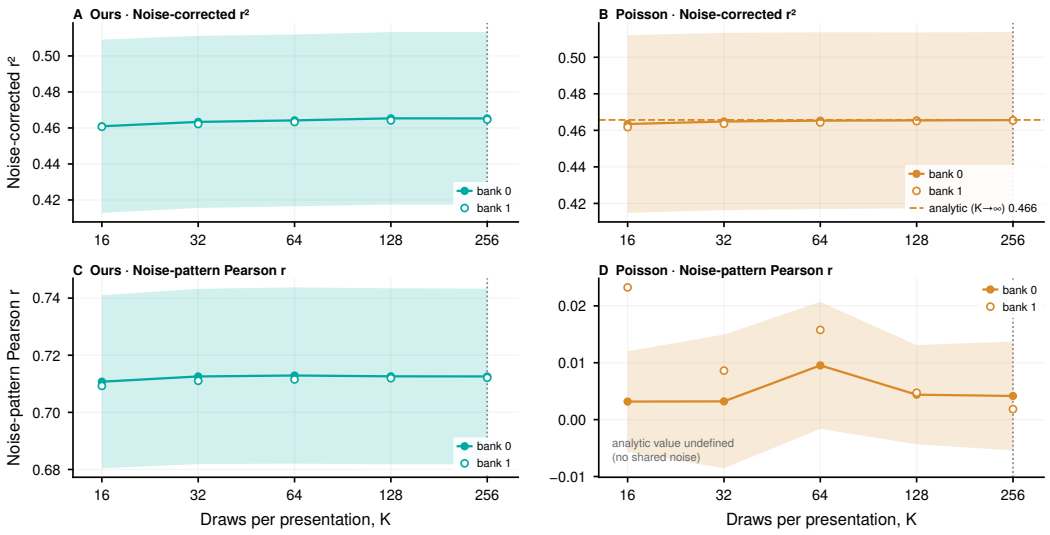


Figure S2: **Convergence of representative mean prediction metrics for our model and the Poisson baseline.** Here shown for the ferret auditory cortex dataset (Wingert et al., 2026).

# E SPIKE TOKENIZER

The Conv-VAE is the single-neuron convolutional tokenizer described in a concurrent study (Anonymous et al., 2026). We use its *frozen* encoder to map each neuron's binary spiking activity in a 200 ms window, $y \in \{0,1\}^{200}$ at 1 ms resolution, to two tokens, one per non-overlapping 100 ms, $u \in \mathbb{R}^{D\times 2}$ with $D = 16$. These continuous tokens serve as the prediction target for SpikeDiT and are passed to the *frozen* decoder to reconstruct spikes. The encoder consists of a patchification stage followed by three stages of residual blocks, the first two of which downsample time by a factor of two; the decoder mirrors this with two upsampling stages and a final de-patchification (Table S2). Note that the encoder downsampling convolution strides equal their kernel size (all other convolutions are $1 \times 1$) so each token depends only on its own 100 ms. The decoder's upsampling kernels cross the token boundary, using temporal information from both windows. No cross-neuron information is used for either.

The output of the decoder's de-patchification is passed through a sigmoid, giving per-bin Bernoulli spike probabilities $\hat{p} \in [0,1]^{200}$, and the tokenizer is trained with a binary cross-entropy objective against the recorded spikes. During this *tokenizer pre-training*, the latent space is parameterized as a per-token diagonal Gaussian,

$$q_\phi(u \mid y) = \mathcal{N}\big(\mu_\phi(y), \mathrm{diag}\big(\sigma^2_\phi(y)\big)\big)\,, \tag{5}$$

regularized towards a standard normal prior by a KL term. For tokenizer inference (i.e., the tokenization step of SpikeDiT) we use only the mean, $\mu_\phi(y)$. We also use the SIGReg regularizer (Balestriero & LeCun, 2025), which encourages an isotropic Gaussian distribution of the latent samples, pooled over units, time steps and windows. The full objective is

$$\begin{aligned}\mathcal{L}_{\mathrm{tok}} = &-\big\langle y_t \log \hat{p}_t + (1-y_t)\log(1-\hat{p}_t)\big\rangle_t \\ &+ \gamma_{\mathrm{KL}}\,\mathrm{KL}\big(q_\phi(u \mid y)\,\big\|\,\mathcal{N}(0,I)\big) + \gamma_{\mathrm{SIG}}\,\mathrm{SIGReg}(u), \qquad u \sim q_\phi(u \mid y),\end{aligned} \tag{6}$$

where $\hat{p} = \mathrm{sigmoid}(\mathrm{Dec}(u))$, $\gamma_{\mathrm{KL}} = 10^{-6}$ (cyclically annealed over training) and $\gamma_{\mathrm{SIG}} = 10^{-3}$.

The tokenizer was trained on the "Full" corpus assembled in Anonymous et al. (2026) (see Appendix L, which contains nearly 20 billion sorted spikes from public as well as unreleased datasets. We also inherit that study's data splits, which hold out the Cowley and Wingert datasets entirely as part of the *xSrc* evaluation tier, so neither encoding dataset used here was seen during tokenizer pre-training. See Table S3 for training settings.

For the SpikeDiT flow targets, we additionally divide the tokenizer latents by a constant $s$, an empirical estimate of the standard deviation of the tokens over each dataset's training split (Table S1), so that the prediction targets have approximately unit variance. Generated latents are multiplied by $s$ before decoding. For SpikeDiT sampling, spikes are emitted by thresholding the decoded probabilities at 0.5 rather than by Bernoulli sampling, so all trial-to-trial variability comes from the generated latents.

Table S2: Conv-VAE architecture. Output shape is channels $\times$ time steps; all convolutions have kernel 1 across units. Res. block: [RMSNorm, SiLU, $1 \times 1$ conv]$\times 2$ with residual connection.

| Stage | Operator | Output |
|---|---|---|
| *Encoder* | | |
| input | binary spikes $y$, 1 ms | $1 \times 200$ |
| patchify | 25 ms patches | $25 \times 8$ |
| stem | $1 \times 1$ conv, 64 | $64 \times 8$ |
| stage 1 | res. block, 64 $\times 2$; conv $k$=2, stride 2 | $64 \times 4$ |
| stage 2 | res. block, 128 $\times 2$; conv $k$=2, stride 2 | $128 \times 2$ |
| stage 3 | res. block, 256 $\times 2$ | $256 \times 2$ |
| middle | res. block, 256 $\times 2$ | $256 \times 2$ |
| head | RMSNorm, SiLU, $1 \times 1$ conv, $2D$; $1 \times 1$ conv, $2D$ | $\mu_\phi, \log \sigma^2_\phi$: $D \times 2$ each |
| *Decoder* | | |
| stem | $1 \times 1$ conv, $D$; $1 \times 1$ conv, 64 | $64 \times 2$ |
| middle | res. block, 64 $\times 2$ | $64 \times 2$ |
| stage 1 | res. block, 64 $\times 2$; 2$\times$ upsample, conv $k$=2 | $64 \times 4$ |
| stage 2 | res. block, 64 $\times 2$; 2$\times$ upsample, conv $k$=2 | $64 \times 8$ |
| stage 3 | res. block, 32 $\times 2$ | $32 \times 8$ |
| head | RMSNorm, SiLU, $1 \times 1$ conv, 25; unpatchify | $1 \times 200$ logits |
| output | sigmoid | $\hat{p} \in [0,1]^{200}$ |
| Parameters: 782,303 | | |

Table S3: Conv-VAE training settings.

| | |
|---|---|
| Training data | training split of Anonymous et al. (2026); C4, Cowley, Wingert held out |
| Training window | 1 s at 1 ms resolution |
| Batch size | 512 windows |
| Training length | 62,500 steps (7.98 epochs); final checkpoint |
| Optimizer | AdamW, betas $(0.95,\ 0.999)$, weight decay 0 |
| Learning rate | $7 \times 10^{-4}$; 100 warm-up steps, cosine decay over 8 epochs to $10^{-5}\times$ peak |
| Reconstruction loss | binary cross-entropy, mean over bins |
| KL weight $\gamma_{\mathrm{KL}}$ | cyclical linear annealing, $0 \to 10^{-6}$, 4 cycles |
| SIGReg weight $\gamma_{\mathrm{SIG}}$ | $10^{-3}$; 256 random projections, 17 knots on $[0, 3]$ |
| Latent width $D$ | 16 per 100 ms token |
| Seed | 42 |

# F Flow training and sampling

This appendix specifies the flow-matching target and loss, the projected CRPS term, and the samplers used for the reported models. See Table S1 for numerical settings and hyperparameters.

**Flow matching.** We train the flow matching model on the tokenizer's posterior means for each recorded trial. Before training, we standardize the tokens by the global standard deviation of the latents using initial training minibatches, and rescale during inference. The starting distribution in flow matching is set to the standard Normal $\epsilon \sim \mathcal{N}(0, I)$, with straight paths $Z_\tau = (1-\tau)\epsilon + \tau Z_1$ as trajectories during training, as introduced in Rectified Flow Matching (Liu et al., 2022; Lipman et al., 2023). We use the mean squared error between the learned velocity field $v_\theta(Z_\tau, \tau, c(x, s))$ and $Z_1 - \epsilon$ as the training objective. During training, we sample the flow times from a Logit Normal distribution $\tau = \mathrm{sigmoid}(-1 + 0.5\,\xi)$ with $\xi \sim \mathcal{N}(0, 1)$. The parameters of this distribution deviate from the typical Logit Normal parameters used for flow matching elsewhere, and place 90% of the mass in the time range $[0.14, 0.46]$. For the Cowley dataset, we draw four pairs $(\tau, \epsilon)$ per example, and a single pair per example for Wingert. We use no conditioning dropout or time-dependent loss weighting.

**Projected CRPS estimator.** To compute the Projected CRPS loss term (Section 3.3), we generate two independent banks of $K$ trials per example, $\widehat{\mathbf{A}}^{(1:K)}$ and $\widehat{\mathbf{B}}^{(1:K)}$. We then score each projection

$a$ against the recorded trial $\mathbf{Y}$:

$$\widehat{\mathrm{CRPS}}_a = \frac{1}{2K}\sum_{k=1}^{K}\Big(\big|a^\top \widehat{\mathbf{A}}^{(k)} - a^\top \mathbf{Y}\big| + \big|a^\top \widehat{\mathbf{B}}^{(k)} - a^\top \mathbf{Y}\big|\Big) - \frac{1}{2K^2}\sum_{k,\ell=1}^{K}\big|a^\top \widehat{\mathbf{A}}^{(k)} - a^\top \widehat{\mathbf{B}}^{(\ell)}\big|. \tag{7}$$

The first term uses all $2K$ draws, and the second compares draws only across the banks, requiring only independent pairs.

We score each unit's window count, together with fixed projections of the window-count vector and of counts binned at 20 and 50 ms (for Cowley) or 20 and 200 ms (for Wingert). We draw these projections once using a fixed random seed, and they depend only on the number of units and bins. For level projections, we use statistics such as population sums and sums of unit pairs. For contrast projections, we enforce the projections to sum to zero across (1) units for window counts, and (2) time within each unit for binned counts. For each projection, we normalize it to unit length and divide it by $\max\{10^{-2}, (\sum_j a_j^2 \mu_j)^{1/2}\}$. This rescaling ensures unit standard deviation under independent Poisson counts at the training means $\mu_j$. For the Wingert dataset, the chunks are not stimulus-locked, and therefore $\mu_j$ is averaged over bins within a unit.

The full CRPS objective is written as a weighted convolution of the various projections over different statistics:

$$\mathcal{L}_{\mathrm{FM}} + \alpha\left(w_{\mathrm{count}}\mathcal{L}_{\mathrm{count}} + w^{\mathrm{lev}}_{\mathrm{space}}\mathcal{L}^{\mathrm{lev}}_{\mathrm{space}} + w^{\mathrm{con}}_{\mathrm{space}}\mathcal{L}^{\mathrm{con}}_{\mathrm{space}} + w^{\mathrm{lev}}_{\mathrm{time}}\mathcal{L}^{\mathrm{lev}}_{\mathrm{time}} + w^{\mathrm{con}}_{\mathrm{time}}\mathcal{L}^{\mathrm{con}}_{\mathrm{time}}\right). \tag{8}$$

The reference weights are set as $(w_{\mathrm{count}}, w^{\mathrm{lev}}_{\mathrm{space}}, w^{\mathrm{con}}_{\mathrm{space}}, w^{\mathrm{lev}}_{\mathrm{time}}, w^{\mathrm{con}}_{\mathrm{time}}) = (0.115, 0.007, 0.43, 0.0074, 1.1)$. These weights were tuned on the Cowley dataset's training split, by comparing gradient norms using the flow network's checkpoint. The reported models scale these weights by $\alpha = 0.05$ on Cowley and $0.01$ on Wingert, see Table S1 for details.

We compute the CRPS loss term once per optimizer step, using only one of the four accumulated micro-batches. On Cowley, this amounts to using 8 images with 2 banks of 8 trials generated from the model, and on Wingert, 2 chunks with 2 banks of 4 generated trials. To generate these trials during training, we use a five-step Euler sampler and backpropagate through it to compute the CRPS loss' gradients. This incorporates the gradients propagated through the tokenizer's decoder itself. Since the decoder thresholds the logits to yield the spike trains, we use a straight-through estimator to approximate the gradients (Bengio et al., 2013). While this surrogate gradient estimator is biased, we found it to empirically work well. On an earlier Cowley checkpoint, where 1.6% of decoder outputs exceeded $1/2$, the variance of the soft window count across generated trials matched that of the emitted count (with Pearson $r = 0.993$–$0.998$ and ratio 0.99–1.02).

**Sampling.** During evaluation, we integrate the learned field from $Z_0 \sim \mathcal{N}(0, I)$ with 20 uniform Euler steps and decode into spikes 200 ms window (two tokens) at a time. A Wingert chunk is decoded as five such windows and concatenated. As mentioned before, the decoder outputs binary spikes $\{0, 1\}$ at each unit and millisecond by a deterministic thresholding of the logit outputs, and consequently, all trial variability arises from the initial sampling noise $Z_0$. We use two independently seeded banks of $K = 256$ draws for each stimulus presentation, and score them separately (see Appendix D).

## G Inference-time completion

In this appendix, we describe how we complete partially observed responses with the trained model, and how we evaluate these completions (Figure 5). We condition the generation on observed activity without retraining the vector field by using the Flower solver (Pourya et al., 2025). All completion experiments are evaluated on the Cowley dataset.

**Observations.** We consider two kinds of partial observations— completion from other neurons, and completion in time. For completion from other neurons, we split each session's units into observed and hidden halves, and score the hidden half by revealing 5, 10, 20, or 50% of the population

as subsets of the observed half. We use the same subsets for every trial. For completion in time, we reveal the first token of every unit, i.e. the first 100 ms, and score the second 100 ms. We represent the observation in both cases as a boolean mask $M$ on the token grid. To obtain the observed tokens $Z^{\text{obs}}$, we (1) set the unobserved spikes to zero, (2) encode the response with the frozen tokenizer, and (3) set the unobserved tokens to zero again. While the tokens are per-unit and constitute 100 ms windows, the tokenizer is not expected to interfere in this evaluation. However, in order to factor out its effect, we still perform this step. This ensures that $Z^{\text{obs}}$ depends only on the observed spikes. Since the tokenizer never mixes units and its encoder tokens do not overlap in time (Appendix E), this masking is equivalent to encoding only the observed units or, for completion in time, only the first 100 ms.

**Solver.** Starting from noise $Z_0 \sim \mathcal{N}(0, I)$, we use the same 20-step Euler-sampled uniform grid as for standard sampling, and at each step from $\tau$ to $\tau'$ compute

$$\begin{aligned} \hat{Z}_1 &= Z_\tau + (1-\tau)\, v_\theta(Z_\tau, \tau, c), \text{ then} \\ Z^\star &= M \odot Z^{\text{obs}} + (1-M) \odot \big(\hat{Z}_1 + \sigma_r(\tau)\, \xi\big), \text{ combined into} \\ Z_{\tau'} &= \tau'\, Z^\star + (1-\tau')\, \epsilon, \end{aligned} \tag{9}$$

with $\sigma_r(\tau) = (1-\tau)/\sqrt{\tau^2 + (1-\tau)^2}$ and fresh $\xi, \epsilon \sim \mathcal{N}(0, I)$ at every step. In a nutshell, each step first predicts the clean endpoint $\hat{Z}_1$ from the current state, then replaces the observed tokens with their recorded values and perturbs the hidden ones, and finally re-noises the result to the next flow time. For the last step, we simply return $Z^\star$ and decode into spikes (Appendix F). With noiseless observations, this is what Flower's measurement step reduces to. Reusing the noise implied by the current state, $(Z_\tau - \tau \hat{Z}_1)/(1-\tau)$, instead of drawing a fresh $\epsilon$ would reduce the update to clamped Euler inpainting. We never copy recorded spikes into the output; instead, we exclude the observed units and the revealed first 100 ms from scoring.

We compare each completion against two controls that use the same solver. In the no-observation control, the mask is empty reducing the solver to unconditional sampling, and yielding the 0% point of each curve in Figure 5A and the baseline in Figure 5B. In the mismatched control, we reveal the same units, but take their activity from a different presentation of the same image. The mismatched control provides the same stimulus information as the matched completion, but without trial-specific fluctuations. Therefore, any gain beyond it comes from shared trial-to-trial variability. All completions share the initial noise and every subsequent noise draw, ensuring that the differences between them arise from the observation itself.

**Evaluation.** We use all test sessions and up to 100 images per session that were presented at least twice. For each image, we score one presentation and, for the mismatched control, another provides the context, giving 3,660 trials in total. We generate $K = 32$ completions per trial. For each hidden unit, we then compute the Pearson correlation across trials (averaged over units) between the mean count over completions and the count recorded on the same trial. We compute 95% paired bootstrap confidence intervals over all sessions using 2,000 resamples. Since averaging a finite number of completions attenuates the correlation, its absolute value increases with $K$, and we therefore compare all arms at the same $K$. When comparing a session's gain to its baseline, we estimate the two on disjoint halves of its trials, since estimating both on the same trials biases their covariance by $-\mathrm{Var}(\text{baseline})$.

## H DATA DETAILS

**Datasets and evaluation supports.** We use visual-cortex recordings from Cowley et al. (2026) and the auditory-cortex recordings of Wingert et al. (2026). Table S4 distinguishes the training cohorts from the supports used for the main-figure evaluations. Cohort sizes count recorded units separately across sessions.

**Recordings, cohort construction, and unit inclusion.** Cowley et al. (2026) recordings were obtained from macaque V4 using chronically implanted Utah arrays. We use the spike sorting of the original study. Of the 50 exported sessions, 44 carry the original authors' training-session designation. Seven of these lack the embedded image data required by our loader, leaving the 37-session

Table S4: Datasets and evaluation supports. The response representation row describes the flow model; baseline count grids are specified separately in the model details.

| | Cowley (visual) | Wingert (auditory) |
|---|---|---|
| Species / recording areas | macaque, V4 | ferret, primarily A1 and PEG; additional labels retained |
| Animal identities | Pepe (PE), Raul (RA), Wakko (WE) | CLT, LMD, PRN, SLJ |
| Training cohort | 37 sessions | 67 sessions after duplicate removal |
| Sessions per animal | PE: 19; RA: 12; WE: 6 | CLT: 21; LMD: 9; PRN: 33; SLJ: 4 |
| Units per session | 95–96 | 3–248 (median 40) |
| Units total | 3,550 | 3,259 |
| Stimuli | static images, including natural and manipulated images | natural-sound sequences |
| Encoder input | RGB image, $224 \times 224$ | 32-channel cochleagram, 100 Hz |
| Stimulus context | current image | current 1 s segment and preceding 1 s of audio |
| Scored response window | 0–200 ms after image onset | current 1 s segment |
| Flow response representation | 1 ms bins, binary | 1 ms bins, binary |
| Downstream split | image identities within each session, approximately 70/15/15% | within-session epoch partitions, with repeated test sequences excluded from training and validation |
| Main-figure evaluation support | test images from 37 sessions | test presentations from all sessions (excluding 11 "donor" sessions) |
| Repeat eligibility / layout | $\geq$30 validation images with $\geq$6 presentations per image | 6 sequences $\times$ 16 scored segments per session; 10–35 presentations per sequence |

cohort in Table S4. Our primary comparisons retain all 3,550 exported units without an additional firing-rate or SNR screen.

Wingert et al. (2026) recordings were obtained from awake, passively listening ferrets using silicon electrode arrays and Neuropixels probes. The original study describes Kilosort2 spike sorting followed by manual curation. The processed export contains 68 files; we remove `SLJ032a-B`, a duplicate recording export of `SLJ032a`, leaving 67 sessions and 3,259 units. We retain the exported units without an additional anatomical filter. Their retained area labels comprise 2,128 A1, 699 PEG, 217 AC, 37 HC, and 178 unknown labels. As such, the full training cohort is broader than an A1/PEG-only population. The donor-panel eligibility rules described below are evaluation-support restrictions, not training-cohort restrictions.

**Downstream splits and repeated presentations.** On data from Cowley et al. (2026), within each retained session, the author-supplied image masks partitions image identities into approximately 70/15/15% training, validation, and test sets. All presentations of an image belong to the same partition. Image IDs are session-local, so split disjointness is checked within each session. The retained cohort contains 393,407 training, 84,222 validation, and 84,301 test presentations.

Wingert's original partitions are intervals within each recording, and the same recording sessions contribute training, validation, and test epochs. We apply two stimulus-level safeguards in addition to duplicate removal. First, the six canonical repeated test sequences are removed from training and validation wherever they occur. Second, ordinary validation is distinguished from `valid_novel`, which retains only sound identities absent from every session's post-exclusion training set. This distinction accounts for the stimulus encoder being shared across sessions: a sound held out within one session may otherwise have appeared in another session's training data.

The six canonical repeated sequences are `STIM_00seq1.wav`, `STIM_00seq2.wav`, `STIM_00seq3.wav`, `STIM_00seq4.wav`, `STIM_00seq5_hand.wav`, and `STIM_00seq6_hand.wav`. Each presentation contains approximately 17.82 s of active sound. We extract 17 complete, nonoverlapping 1 s response segments. Primary evaluation excludes

the onset segment, retaining segments 1–16, or $[1, 17)$ s relative to sound onset; the remaining fractional tail is not scored. Each prediction receives the current audio segment and its preceding 1 s of audio, while only the current response segment enters the loss or metric. Context is restricted to the same presentation. The six sequences therefore define 96 scored stimulus conditions per session. Segments from one presentation are parts of the same experimental trial and are not counted as independent experimental repeats.

**Wingert development "donor" panel and test access.** Because Wingert's repeated sequences are concentrated in the original test partition, we designated a development set for repeat-based evaluation of all models during development. Eleven sessions were designated as donors using an heuristic combining animal/area stratification, canonical-sequence repeat counts. The general eligibility pool required at least ten presentations of each canonical sequence and a session-mean firing rate of at least 5 Hz.

The resulting sessions were `CLT028c`, `CLT045d`, `CLT047c`, `CLT049c`, `LMD019a`, `LMD048a`, `LMD052a`, `PRN007a`, `PRN018a`, `PRN075a`, and `SLJ012a`. They contain 568 units and 10–35 presentations per canonical sequence. These are excluded from the final test set.

**Tokenizer data exposure** Both datasets use the same frozen neural tokenizer. Its training configuration excludes Cowley and Wingert from gradient-based tokenizer training. Both datasets are however present in the tokenizer's test data mixture, used in some final evaluations in Anonymous et al. (2026).

# I METRIC DEFINITIONS

**Notation.** A *stimulus* $s$ is one image (Cowley) or one 1-s chunk of one of the six repeated test sequences (Wingert; Table S5). For unit $u$, $y_{sru}$ is the recorded spike count on repeat $r = 1, \ldots, m_s$ in the counting window. Every model is sampled $K = 256$ times *per presentation*, so stimulus $s$ has $Km_s$ generated counts $\hat{y}_{sku}$: all $K$ draws of every repeat, none reused across repeats.

| | Cowley (V4) | Wingert (auditory) |
|---|---|---|
| stimulus | image | 1-s chunk 1–16 of a sequence |
| count window | 0–200 ms | whole 1-s chunk |
| PSTH grid | $10 \times 20$ ms | $50 \times 20$ ms |
| mean metrics | images, $\geq 2$ repeats | 96 stimuli, $\geq 6$ repeats |
| noise metrics | images, $\geq 6$ repeats | 96 stimuli, $\geq 6$ repeats |

Table S5: Where the two datasets differ. Everything else is identical.

**Conditional-mean metrics.** The predicted mean $\hat{\mu}_{su}$ is the average of all $Km_s$ draws, except for the Wingert baselines, which use their closed-form conditional mean and so carry no Monte-Carlo error in these metrics. Covariance and Fano metrics are computed from draws for every model on both datasets. *Rate* $r$ is the Pearson correlation, across stimuli, of $\hat{\mu}_{su}$ with the repeat mean $\bar{y}_{su}$ of window counts. *PSTH* $r$ correlates the repeat-mean PSTHs, smoothed by a Gaussian with $\sigma = 20$ ms truncated at $\pm 4\sigma$ (zero padding on Cowley, reflecting padding on Wingert), over all (stimulus, bin) entries at once. The *noise-corrected* $r^2$ is the unbiased estimator of Pospisil & Bair (2021). With $a_s = \hat{\mu}_{su} - \langle \hat{\mu}_u \rangle$ and $b_s = \bar{y}_{su} - \langle \bar{y}_u \rangle$ centred over the $S$ stimuli,

$$r^2_{\mathrm{ER}} = \frac{\left(\sum_s a_s b_s\right)^2 - \frac{\sigma^2}{n} \sum_s a_s^2}{\sum_s a_s^2 \left(\sum_s b_s^2 - (S-1)\frac{\sigma^2}{n}\right)}, \tag{10}$$

where $\sigma^2$ is the mean over stimuli of the within-stimulus variance ($\mathrm{ddof} = 1$) and $n$ the median repeat count. Units whose noise-corrected signal variance $\sum_s b_s^2 - (S-1)\sigma^2/n$ is below 5% of $\sum_s b_s^2$ are undefined and excluded, as are units with a constant prediction.

**Noise covariance and correlation.** Each stimulus gets its own covariance, about its own mean, and the matrices are averaged with equal weight per stimulus:

$$\Sigma = \frac{1}{S}\sum_{s=1}^{S}\frac{1}{n_s-1}\sum_{j=1}^{n_s}(\mathbf{z}_{sj}-\bar{\mathbf{z}}_s)(\mathbf{z}_{sj}-\bar{\mathbf{z}}_s)^\top, \qquad \rho_{uv} = \frac{\Sigma_{uv}}{\sqrt{\Sigma_{uu}\Sigma_{vv}}}. \tag{11}$$

The rows $\mathbf{z}_{sj}$ are the $n_s = m_s$ recorded population vectors or the $n_s = Km_s$ generated ones, and $\bar{\mathbf{z}}_s$ is their mean. The row count differs between the two sides, but each summand estimates the same per-stimulus covariance without bias. Correlation is formed *once*, from the pooled $\Sigma$: per-stimulus correlation matrices are never averaged. A unit with $\sqrt{\Sigma_{uu}} \le 10^{-12}$ gets undefined correlations.

**Pattern, magnitude and error.** Let $\mathcal{P}$ be the unordered pairs $u < v$, restricted for correlations to pairs defined on both sides. Then

$$\text{pattern} = \operatorname{corr}_{\mathcal{P}}\big(\rho^{\text{mod}}_{uv}, \rho^{\text{rec}}_{uv}\big), \quad \text{RMS ratio} = \frac{\|\Sigma^{\text{mod}}_{\mathcal{P}}\|}{\|\Sigma^{\text{rec}}_{\mathcal{P}}\|}, \quad \text{error} = \frac{\|\Sigma^{\text{mod}}_{\mathcal{P}} - \Sigma^{\text{rec}}_{\mathcal{P}}\|}{\|\Sigma^{\text{rec}}_{\mathcal{P}}\|}, \tag{12}$$

where $\|\cdot\|$ is the root mean square over pairs. The pattern compares *correlations* and is scale-free. Magnitude and error compare off-diagonal *covariances*, in count units. The plotted pattern is raw. A variant corrected by $\sqrt{R_{\text{mod}}R_{\text{rec}}}$ (Spearman–Brown split-half reliabilities over odd/even draws and interleaved repeats) is computed but not plotted.

**Displayed matrices.** The main figures and Figures S5 and S6 show $\rho$ from equation 11 for one session, computed from one draw of 256 samples, with units ordered by average-linkage clustering of the recorded matrix. The *raw* matrices are the correlations themselves. The *z-scored* matrices standardise the off-diagonal entries by that matrix's own mean and standard deviation, $\tilde{\rho}_{uv} = (\rho_{uv} - \bar{\rho})/s_\rho$, shown on $\pm 3$. The z-score removes each model's overall correlation amplitude, so the displays compare *pattern* alone. In the displayed sessions the off-diagonal RMS of $\rho$ differs 8- to 14-fold between Schmidt et al. and Ours, and 60- to 100-fold between Poisson and the recording. The transform is affine, so the pattern score is identical on raw and z-scored matrices. The panels annotate each model with its session's pattern (z-scored rows) and RMS ratio (raw rows). The RMS ratio is computed on covariances, not the correlations drawn, so a model that overstates single-unit variances can show faint correlations yet a ratio above 1. Poisson has no shared noise, so its z-scored panel shows standardised Monte-Carlo noise.

**Fano factor.** Per unit, $F_u = |V_u|^{-1}\sum_{s\in V_u}\widehat{\text{Var}}_{su}/\widehat{\text{E}}_{su}$, using the same repeat-eligible stimuli and rows as equation 11 ($\text{ddof} = 1$). On Wingert each side's $V_u$ keeps stimuli where *its own* mean exceeds $10^{-6}$, so the recorded Fano is the same for every model. On Cowley, $V_u$ requires a positive mean on *both* sides. A stimulus for which a model predicts an exactly zero mean is therefore also dropped from the recorded Fano, so each model is plotted against its own recorded vector. This happens only for Ours, which emits binary spikes: one of 2,206 units differs from the baselines. Scatter correlations pool units across sessions.

**Aggregation, uncertainty and missing values.** Per-unit metrics are averaged over each session's finite units. Session values are averaged with equal weight per session, never per unit or animal, and error bars are $\pm 1$ SEM, $\text{sd}_{\text{ddof}=1}/\sqrt{n}$, over sessions. Each of the two independent $K = 256$ banks is scored separately, and the draws of the two banks are never pooled. Every bar and scatter is plotted from one $K = 256$ bank of samples. The other bank serves as a Monte-Carlo check: across all sampled metrics, $|\text{bank}_0 - \text{bank}_1|$ is at most 0.39 session SEM. Undefined values (constant predictions, zero-variance units, too few repeats) are NaN and excluded, never set to zero.

**ACG and CCG protocol.** All correlograms are computed on a 1 ms grid over lags $|L| \le 50$ ms, from test presentations of stimuli with at least six repeats. We follow the implementation in `npyx` (Beau et al., 2021): for a trial $i$ — one presentation of stimulus $s(i)$, or one model draw of it — with spike counts $x_{i,u}(t)$, $t = 1, \ldots, T$ ($T = 200$ and 1000 ms for Cowley and Wingert), we form within-trial products only,

$$C_{uv}(L) = \frac{1}{T-|L|}\sum_i w_i \sum_t x_{i,u}(t)\, x_{i,v}(t+L), \qquad w_i = \frac{1}{S\, n_{s(i)}\, K}, \tag{13}$$

where the sum over $t$ runs over the $T-|L|$ pairs of bins that both fall inside the trial. $S$ is the number of eligible stimuli, $n_s$ the number of presentations of stimulus $s$, and $K$ the number of draws per presentation ($K = 1$ for the recording). The weights give every stimulus equal weight, and dividing by $T - |L|$ corrects for the shrinking overlap at long lags. No products are formed across trials, and nothing is smoothed. The correlogram is total, not stimulus-residual: no shift predictor is subtracted. The ACG is $C_{uu}$ with lag 0 (the self-spike term) excluded; the CCG is $C_{uv}$, $u \neq v$, and keeps lag 0.

**Normalisation.** Curves are reported as conditional rates in spikes per second

$$R_{uv}(L) = \frac{C_{uv}(L)}{\bar{r}_u} \cdot \frac{1000}{\Delta}, \qquad \bar{r}_u = \frac{1}{T} \sum_i w_i \sum_t x_{i,u}(t), \tag{14}$$

where $\Delta = 1\,\mathrm{ms}$ is the bin width. $R_{uv}(L)$ is the firing rate of unit $v$ at lag $L$ given a spike of unit $u$ at lag 0, also following the `npyx` (Beau et al., 2021) implementation. At long lags it approaches $v$'s mean rate. Each model curve is normalised by that model's own $\bar{r}_u$. Agreement with the recording is the Pearson correlation across lags, which is invariant to this per-curve scale.

**Models and placement.** Ours emits binary spikes natively at $1\,\mathrm{ms}$. The baselines emit $20\,\mathrm{ms}$ counts, which must be placed on the $1\,\mathrm{ms}$ grid before equation 13 applies. We report two placements, each preserving every bin total exactly. *Even* placement is a deterministic midpoint comb, putting the $j$-th of $n$ spikes at offset $\lfloor (j + \frac{1}{2})\, 20/n \rfloor$. *Random* placement draws offsets iid-uniformly, with replacement. Sampling without replacement is not possible, because some of the baselines can emit up to 33 spikes into a 20-slot bin. Ours is identical under both placements. Static FA emits one window-long bin per trial and is excluded.

## J TOKENIZER ABLATION

The un-regularized tokenizer arm of Table 1 replaces our frozen tokenizer with another checkpoint of the same variational family, trained on the same exact large dataset (which does not include any of the two datasets used in this paper), with the same latent width, token geometry and thresholded binary emission, and changes only latent regularisation. The latent scale is recalibrated for each tokenizer from 50 training batches ($0.5037$ for the reported tokenizer, $0.4143$ for the weak one); it divides the encoder's output to form the flow target and is undone before decoding, both in training and evaluation.

The two tokenizers are nearly equally good autoencoders. Before training, for each run we encode recorded validation trials with its frozen tokenizer and decode them again. This "roundtrip" bypasses the flow model entirely and measures how faithfully the tokenizer alone represents and reconstructs one observed trial. We then correlate the reconstruction with the recorded response: across trials, taking each unit's window count (rate), and within each trial and unit, taking the time course in 33 ms bins (temporal), each averaged over units. On the same 20 validation batches, all from one 96-unit session, the reported tokenizer reaches $0.9945$ (rate) and $0.9957$ (temporal) and the unregularized one $0.9621$ and $0.9359$. Our interpretation is that a near-perfect reconstruction performance is therefore not sufficient for a tokenizer to excel in supporting stimulus-conditioned generation.

## K SUPPLEMENTARY FIGURES

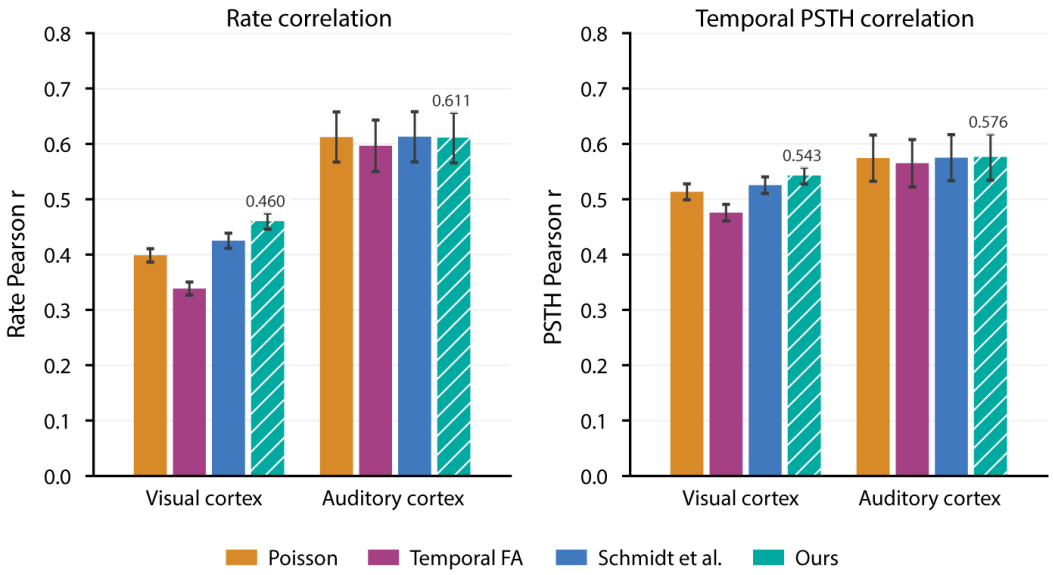


Figure S3: **Conditional mean predictivity** Conditional-mean prediction across visual and auditory recordings measured as firing rate correlation (left) and temporal PSTH correlation (right).

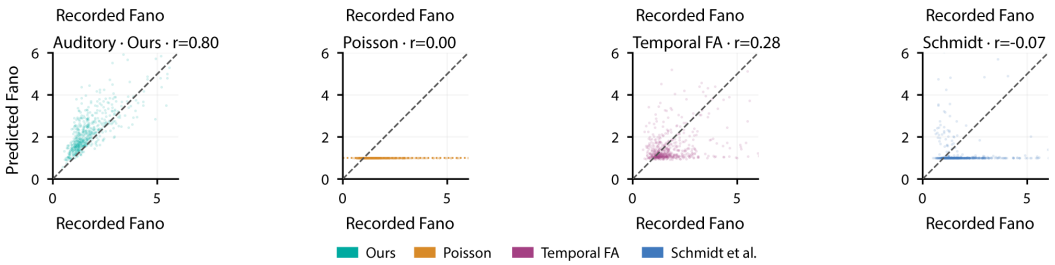


Figure S4: **Fano factor correlations on auditory cortex dataset** Predicted against recorded Fano factor per neuron for all auditory cortex sessions. Dashed lines show identity.

### K.1 NOISE CORRELATION MATRICES

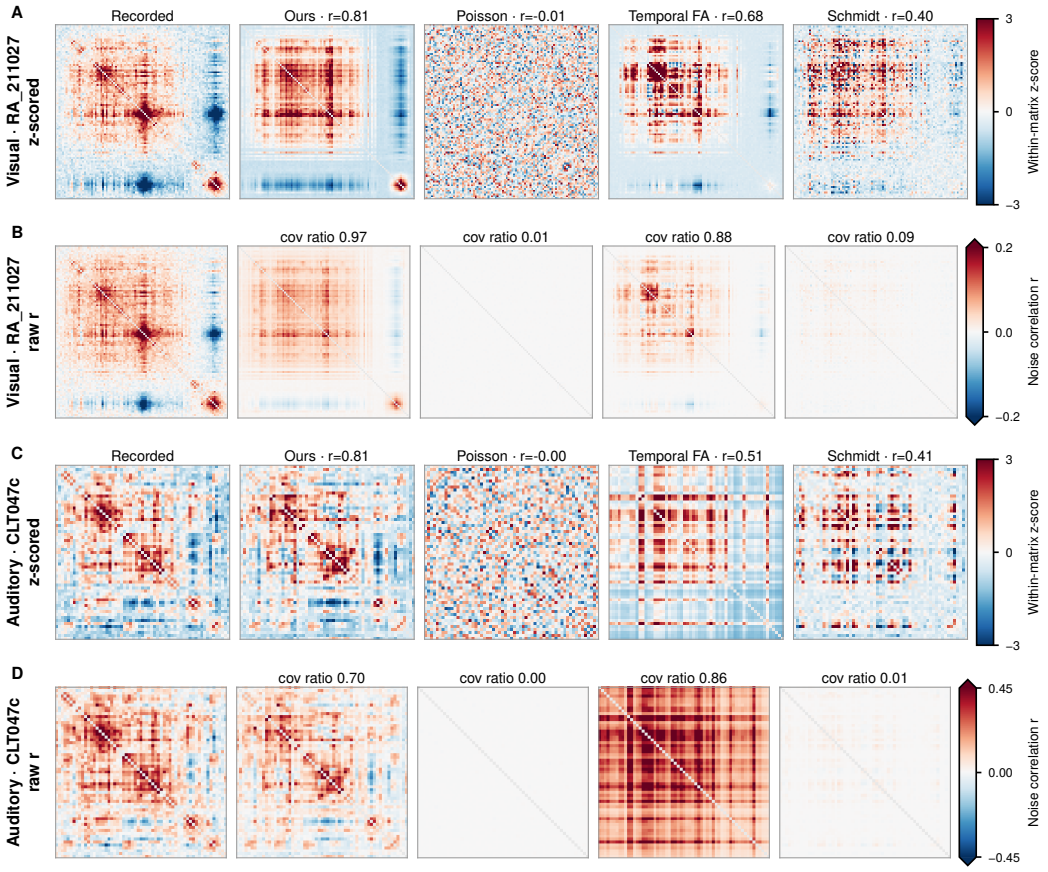


Figure S5: **Noise-correlation matrices, z-scored and raw.** Same of Figure 3 A, with units in the same order. (A, C) Z-scored matrices, same as Figure 3. Each is standardised by its own off-diagonal mean and standard deviation, so the pattern can be compared across models that vary in predicted covariance magnitude. Each title gives that session's noise-pattern r (Figure 3). (B, D) Raw noise correlation matrices. The scalebar is symmetric and shared by every model, per dataset. Each title gives the covariance RMS ratio (Figure 3D metric, 1 = recorded magnitude). The ratio is computed on off-diagonal covariances, not on the correlations drawn, so a model that overstates single-unit variances can show faint correlations yet a ratio near or above 1. Diagonals are blanked.

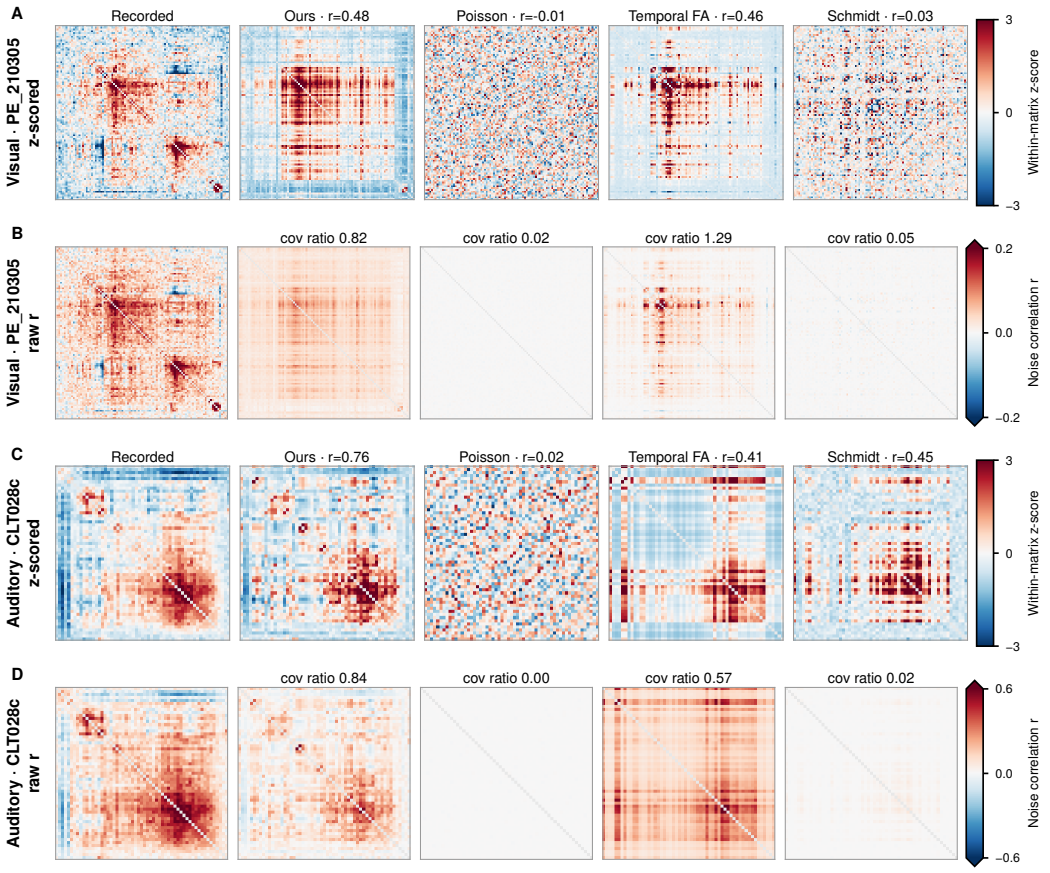


Figure S6: **Noise-correlation matrices for two median sessions.** Same as Appendix Figure S5 for one more session per dataset, chosen by median noise pattern $r$ after ranking over sessions per dataset.

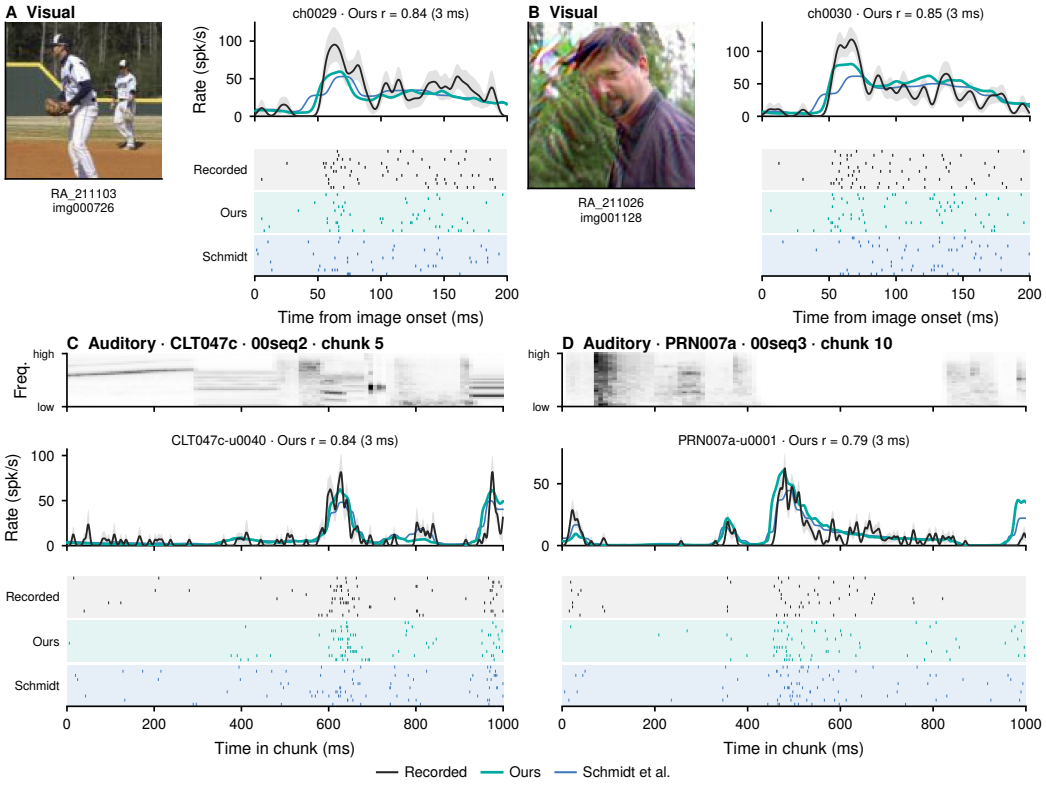


Figure S7: **Example responses at fine timescales.** Four units, each responding to one stimulus: two in visual cortex (A, B; the image shown) and two in auditory cortex (C, D; with the cochleagram of the 1 s chunk). Top: PSTHs of the recording (black, $\pm 1$ SEM over repeats), our model (teal) and Schmidt et al. (blue), all smoothed with the same Gaussian ($\sigma$ = 3 ms) from 1 ms rates. Our model emits spikes natively at 1 ms. Schmidt et al. emits 20 ms counts, which were placed uniformly at random inside their 20 ms bin. Model PSTHs average all presentations of the stimulus × K = 256 sample draws.

## L ANONYMIZED CONCURRENT SUBMISSION

We include the anonymized manuscript cited as Anonymous et al. (2026) for reference. It is not a contribution of this submission.

# Evaluating Neural Data Tokenizers: A Framework for Assessing Learned Representations of Spiking Activity

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Foundation models of brain activity aim to learn representations from large-scale neural recordings that generalize across sessions, subjects, species, and downstream tasks. Transformer-based models have emerged as a promising architecture for this goal and begin all with the same first step – tokenization: transforming sparse binary events into vector representations that downstream networks can consume. In other areas of machine learning, tokenization is its own design problem, distinct from the downstream foundation model. Several mostly implicit tokenization approaches dominate the literature, but their impact on downstream performance has not been systematically evaluated. We propose four explicit desiderata for a neural tokenizer—*reconstruction fidelity*, *downstream utility*, *cross-instance generalization*, and *compression*—and a framework of metrics spanning seven dimensions, including firing statistics, encoding and decoding, and latent structure and robustness. We assemble a benchmark of ten datasets spanning species, brain areas, recording technologies, and behavioral paradigms, and evaluate a representative set of discrete and continuous tokenizers, both single-neuron and population variants. Across the strategies evaluated, we show how tokenizers trained only on reconstruction can support downstream tasks. Yet reconstruction alone is misleading: reconstruction quality and downstream utility can be sharply dissociated. Our multi-axis framework is a response to this failure mode. These findings highlight tokenization as a critical and underexplored design axis in brain foundation models, and suggest that further progress will require continued systematic study of tokenization strategies.

## 1 Introduction

In recent years, advances in neural recording technology have enabled large-scale datasets of brain activity at single-neuron resolution [31, 51, 52, 44]. These datasets have, in turn, driven growing interest in building brain foundation models that ingest neural activity [63, 64, 6, 69, 42, 60]. A central promise of such models is that they are trained in a task-agnostic manner, learning representations that generalize across downstream tasks, experimental conditions, and potentially subjects or species. However, in most current approaches, the tokenization of neural activity is not treated as a reusable component, but is instead implicitly defined by task-specific end-to-end training, undermining the goal of learning general-purpose neural representations. This raises a fundamental question: how should we tokenize spiking activity in a way that preserves the information necessary for a broad range of downstream uses?

In other domains such as language and vision, tokenization is a design choice that critically shapes model behavior. Byte-pair encoding variants systematically impact downstream text performance at a fixed scale [2, 48, 16]. In vision, patch tokenization and learned codebooks (VQ-VAE, ViT,

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

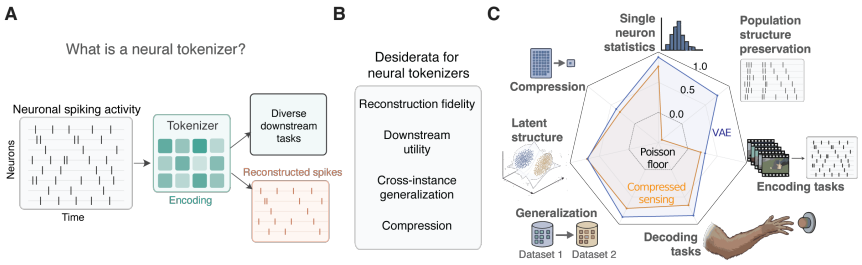


**Figure 1: The neural spike tokenization problem. A**. A tokenizer maps a population spike train to a latent token sequence and back. Two objectives trade off: the reconstructed spike train should resemble the input, and the latents should expose structure useful to downstream tasks. **B**. We propose explicit desiderata for neural tokenizers. **C**. We introduce a multi-axis evaluation framework and provide implementations of diverse tokenizer architectures, bottlenecks, and losses as a community resource to systematically investigate neural tokenizers for foundation models of neural activity.

MAGVIT) play an equivalent role [53, 25, 65]. The impact of tokenization on performance in brain foundation models has rarely been examined [but see e.g. 5].

The tokenizer dictates the model's data interface; any structure it discards cannot be recovered downstream (Figure 1). Work in adjacent neural modalities such as EEG [33, 11] and calcium imaging [62] has begun to develop standalone tokenizers, but spiking activity differs in ways that prevent direct transfer. Spikes are binary at sub-millisecond resolution, sparse, dominated by Poisson-like single-trial variability, and recorded from neuron sets that change from session to session. Tokenizers for spiking activity must thus be designed and evaluated on their own terms.

To date, neural tokenizers have not been systematically evaluated as standalone components. A key question is how to evaluate their quality. The natural default is reconstruction fidelity, which measures how well the original spike train can be recovered from the tokens. However, we show that reconstruction alone is insufficient with the tokenizers we tested: tokenizers with near-perfect reconstruction of held-out spike trains can yield latent representations that range from highly informative to nearly useless for downstream decoding and encoding. Reconstruction alone is therefore an insufficient criterion for evaluating spike tokenizers. Here, we introduce a comprehensive evaluation suite and multi-dataset benchmark for rigorous comparison:

1. We define spike tokenization as a modular component with explicit desiderata, decoupling it from the input layer of end-to-end models.
2. We introduce a multi-axis evaluation framework spanning reconstruction fidelity, single-unit and pairwise temporal statistics, population geometry, and downstream utility, with metrics normalized between a rate-matched Poisson floor and a perfect-reconstruction ceiling.
3. We collate a benchmark of ten datasets spanning four species, multiple brain areas, two recording modalities, and diverse behavioral paradigms, featuring cross-time, cross-session, cross-subject, and cross-source generalization splits.
4. We define a zoo of both continuous and discrete tokenizers for both single-neuron and population-level tokenization—exploring a variety of architectures, bottlenecks, and losses—and apply the evaluation framework to this suite.

Together, this provides a standardized framework for evaluating spike tokenization, enabling the systematic comparison and development of tokenizers for brain foundation models. Applying this evaluation suite across a range of tokenization strategies, we observe that no method is universally optimal across desiderata. Reconstruction quality trades off with downstream utility, and population-level and single-neuron tokenization differ in the aspects of neural structure they preserve. These results highlight spike tokenization as an unresolved design problem in brain foundation models. Just as tokenization has emerged as a key area of study in language and vision, we anticipate that progress in neural foundation models will require more systematic investigation of tokenization strategies, for which this evaluation suite provides a foundation.

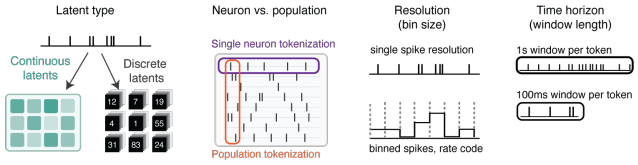


**Figure 2: Tokenizer design space.** The design space of tokenizers in our evaluation framework covers latent types, population, resolution, and window sizes.

## 2 Problem formulation

A neural spike tokenizer maps observed spiking activity into a compressed latent representation that can be reused across modeling objectives. Let $x$ denote spike activity over neurons and time. We focus on tokenizers that define an encoder $q_\phi(z \mid x)$ and a decoder $p_\theta(x \mid z)$, with $z$ as the resulting latent representation. Both components are required because foundation models for spiking activity operate in two directions: from spikes to tokens (encoder), supporting downstream decoding into behavioral or external variables, and from tokens back to spikes (decoder network), supporting generation and reconstruction. The latent $z$ is used directly by these downstream models. The geometry of $z$ determines whether the tokenizer is useful in practice: it sets how easily downstream models extract task-relevant signal [59, 3], whether the same encoder transfers to new sessions and subjects, and whether the representation compresses input efficiently. These considerations motivate four key desiderata:

**1. Reconstruction fidelity.** The encoder and decoder network preserve the structure of the input without distorting its properties, so that closed-loop computation is possible, and to enable use of the model as a digital twin for neuron populations. Properties that should be preserved include individual spike events, single-unit temporal statistics, and population structure.

**2. Downstream utility.** The tokens support downstream tasks either with simple downstream models or by allowing faster training compared to raw data. This has two facets: *cross-task transferability*, where a single frozen tokenizer supports multiple downstream uses (neural decoding, neural encoding, latent analysis) without per-task fine-tuning, and *sample-efficient downstream learning*, where downstream tasks reach competitive performance with few training trials. Two representations can carry identical task-relevant information but differ substantially in how readily a downstream predictor extracts it.

**3. Cross-instance generalization.** The same tokenizer functions without retraining across sessions, subjects, recording sources, and ideally even brain areas or species. This is the property that makes a tokenizer reusable as a foundation model component, rather than a per-session preprocessing step.

**4. Compression.** The tokenizer represents the inputs efficiently, as measured, for example, by the number of *neurons* $\times$ *time bins* per token or by the ratio of input bits to output bits (*compression rate*). Other things being equal, a tokenizer that uses one bit per spike on average would be preferable to a tokenizer requiring ten bits per spike to achieve the same reconstruction fidelity. Compression is motivated by at least two practical goals: more efficient model training (for example, requiring less memory or fewer FLOPs) and better generalization.

These desiderata are not independent in practice. Compression interacts with reconstruction, since higher compression discards more structure. Cross-instance generalization interacts with downstream utility, since a tokenizer that overfits to a single session is unlikely to support efficient learning on a new one.

## 3 Evaluation framework

We provide an evaluation framework for the systematic development of tokenizers on spiking neural data, built on our curated dataset collection (Section 4). Any candidate tokenizer can be evaluated on the same metrics. The set of evaluation metrics in our current implementation is comprehensive, but can be extended with new metrics as the field develops. On the other hand, the four desiderata above (Section 2) are intended as a more stable conceptual framework.

The mapping from desiderata to metrics is not one-to-one: some metrics serve multiple desiderata, and desideratum 3 (cross-instance generalization) applies to every metric. We divide the metrics

into multiple groups below and indicate which desideratum each group serves. Appendix A includes details of metric implementations.

**Single-neuron reconstruction statistics.** These metrics compare the decoded spike train $\hat{x}$ to the input $x$ at the level of individual units, serving desideratum 1 (reconstruction fidelity) and establishing the per-unit floor: reconstruction is a tokenizer's essential function, and information discarded here is lost for all downstream tasks that follow. Metrics include spike-detection F1 score, temporal precision (single-neuron tokenizers only), mutual information, spike-count correlation, smoothed-rate error, ISI divergence, ISI CV and CV2 errors, autocorrelogram correlation, firing-rate error, and Fano factor error.

**Population firing statistics.** These metrics compare $\hat{x}$ to $x$ at the level of joint activity across neurons, also serving desideratum 1: A tokenizer could sufficiently preserve every neuron's marginal statistics while degrading the cross-neuron timing that defines a population code. Metrics include cross-correlogram correlation, functional connectivity F1, participation ratio error, subspace overlap, population coupling preservation, and binned rate difference (population tokenizers).

**Downstream decoding.** These metrics test the tokenizers on a canonical downstream task in spike modeling: reading behavior off neural activity. We train both simple readouts (linear probes and small MLPs) and a sequence-to-sequence non-linear temporal decoder [6] on top of the frozen tokenizer, serving desideratum 2 (downstream utility) across both facets. An asymptotic protocol trains on the full training set and tests cross-task transferability; a learning-curve protocol varies training-set size and tests sample-efficient downstream learning. Targets include cursor position and velocity ($R^2$) and trial-target and movement-phase classification (accuracy).

**Downstream encoding.** Neural encoding is the symmetric counterpart to decoding and the second direction any unified neural foundation model must support, where the token sequence is the prediction target rather than the input. On stimulus-locked data (V4 responses to natural images [22]), we train a shared image backbone with session-specific readouts to predict the frozen tokenizer's latent distribution from each held-out image. The decoded conditional-mean response is then scored against held-out PSTHs (see Appendix F for details), serving desideratum 2 along both facets.

**Latent structure and latent robustness.** These metrics ask whether $z$ itself carries the structure of $x$, independently of whether the tokenizer's decoder network can reconstruct it. Latent geometry is what downstream models actually consume, so probing it directly diagnoses whether the encoder has done useful work independently of the decoder. They are most diagnostic of downstream utility (desideratum 2), with the corruption-robustness sub-metrics also bearing on the stability underlying cross-instance generalization. We measure RSA between input and latent dissimilarity matrices, $k$-NN neighborhood overlap, repeat-trial SNR in the latent space on stimulus-locked datasets, and stability of the latents under input- and latent-side corruption.

**Generalization.** Desideratum 3 (cross-instance generalization) applies to every metric above. All metrics can be reported under four held-out splits: cross-time (held-out time within a session), cross-session (held-out session, same subject), cross-subject (held-out subject, same source), and cross-source (entirely held-out source). For each split and metric, we report a direction-aware retention ratio (held-out vs. within-session performance, clipped to [0, 1]), aggregated as a geometric mean across metrics and tiers (Appendix A.6).

**Compression.** These metrics quantify how efficiently the tokenizer represents the inputs, serving desideratum 4. We report the *compression rate* of tokenizers, as measured by the ratio of input bits to output bits (*i.e.* number of bits required to represent the input *vs.* the output).

Reconstruction metrics (single-neuron and population) and latent-structure metrics are normalized between two anchors: a rate-matched Poisson floor and, when applicable, a perfect-reconstruction ceiling. The Poisson floor corresponds to a tokenizer that preserves per-neuron firing rates and discards all other structure (fine timing, pairwise structure, population geometry). The remaining categories use their native scales: $R^2$ and classification accuracy for decoding and encoding, a retention ratio in $[0, 1]$ for generalization, and natural units for compression.

Each axis is the equal-weighted mean of its constituent metrics in that axis's natural units (Poisson-normalized for reconstruction, $R^2$ for encoding/decoding, soft retention $1/(1+e)$ for input-corruption robustness, geometric-mean retention for generalization); per-metric Poisson-normalized scores are clipped at $-1$ before averaging to bound outliers. We deliberately do not collapse to a single

| | IBL Brain-wide Map [31] | Allen Vis. Behavior [4] | Allen Vis. Coding [51] | C4 Database [12] | Churchland Shenoy 2012 [21] | Flint Slutzky 2012 [26] | O'Doherty Sabes 2017 [40] | Perich Miller 2018 [44] | Cowley Smith 2023 [22] | Wingert David 2026 [61] |
|---|---|---|---|---|---|---|---|---|---|---|
| Recording | Neuropixels | Neuropixels | Neuropixels | Neuropixels | Utah arrays | Utah arrays | Utah arrays | Utah arrays | Utah arrays | Linear arrays |
| Species | mouse | mouse | mouse | macaque & mouse | macaque | macaque | macaque | macaque | macaque | ferret |
| Brain region | brainwide | visual ctx, thal, HPC, MB | visual ctx, thal, HPC, MB | cerebellum | M1, PMd | M1, PMd | M1, S1 | M1, PMd | V4 | A1, PEG |
| Sessions | 331 | 103 | 58 | 321 | 10 | 5 | 47 | 111 | 50 | 68 |
| Units | 101,557 | 82,133 | 40,010 | 4,960 | 1,911 | 957 | 16,566 | 10,410 | 4,774 | 3,382 |
| Spikes | 2.32e9 | 7.01e9 | 3.08e9 | 5.07e8 | 7.39e8 | 7.88e6 | 1.05e8 | 1.11e8 | 2.10e8 | 1.03e8 |
| $xT$ | 0.10 | 0.10 | 0.10 | - | 0.10 | 0.10 | 0.10 | 0.10 | - | - |
| $xS$ | 0.04 | 0.10 | 0.07 | - | 0.45 | 0.43 | 0.12 | 0.07 | - | - |
| $xSub$ | 0.10 | 0.00 | 0.07 | - | 0.00 | 0.00 | 0.41 | 0.10 | - | - |
| $xSrc$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |

**Table 1: Public datasets in the pretraining corpus**. Rows are dataset properties, columns are datasets. Spikes are sourced exclusively from single-unit, spike-sorted data, and counts are aggregated across train/val/test splits. The four lower rows give the fraction of total recording *time* withheld from training under each generalization regime: $xT$, held-out time within a session, $xS$: held-out sessions, same subject, $xSub$: held-out subjects, same dataset, and (4) $xSrc$: held-out datasets, which can also withhold whole species or brain regions.

composite score: cross-axis trade-offs (e.g., compression at the cost of population structure) are precisely what the framework is designed to expose.

## 4 Dataset suite

We compiled a diverse set of public datasets to cover a wide range of settings in which spike tokenizers are typically applied. Each dataset consists of multiple distinct extracellular electrophysiology recordings spanning species and brain regions. Datasets vary along five axes (Figure S1A): (1) recording technology, (2) species, (3) brain region, (4) behavioral paradigm, (5) population size.

Datasets vary by 1-3 orders of magnitude on properties that directly affect tokenizer training: median session duration ranges from ~10 to ~170 min, median population size from 5 to ~1000 units, and median per-neuron firing rate from ~4 to ~38 Hz (Figure S2). Spiking sparsity has similar variability: at 1 ms binning, population sparsity ranges from $< 1\%$ in dense cortical recordings to $> 85\%$ in cerebellar populations, and a strongly dataset-dependent decay as the bin width is coarsened to 10–1000 ms (Fig. S3). Overall, the corpus contains nearly 20 billion sorted spikes: over 14 billion spikes from public datasets (Table 1) and an additional 5.5 billion from proprietary unreleased datasets.

To probe various levels of generalization, we partition the held-out data into four tiers (Figure S1B): (1) $xT$: held-out time within a session, (2) $xS$: held-out sessions, same subject, (3) $xSub$: held-out subjects, same dataset, and (4) $xSrc$: held-out datasets, which can also withhold whole species or brain regions. The four lower rows of Table 1 report, per dataset, the fraction of total recording time withheld from training for each tier. To support the community, we release our data preparation code (here) that builds on existing `brainsets` pipelines [6] to convert each public dataset into a standardized tokenizer-ready format.

## 5 Related work

**Implicit tokenization in neural population models.** In large-scale models of spiking activity, fixed binning is the dominant convention for tokenization: spike times are converted into count vectors at a chosen temporal resolution and then passed to a latent-variable model, transformer, or multimodal architecture. This includes NDT [63, 64, 68], LFADS [42, 36], and NEDS [69]-style pipelines, where the bin size, population vectorization, and embedding layers determine the information available to the downstream model. The tokenizer is trained jointly with the rest of the model without measuring its individual contribution separately.

**Event-level and two-stage neural representations.** Another approach is spike-level tokenization. POYO-like architectures [6, 7, 49] treat individual spikes as events with associated timestamps and neuron identities, preserving fine temporal information before aggregation by a larger model. This makes the tokenization step explicit, but it is still trained jointly with the downstream objective rather than as a reusable module. Two-stage generative approaches are closer to our setting. LDNS

[34], for example, trains an autoencoder before fitting a diffusion model in the inferred latent space, showing that separately learned neural latents can be useful for generation. However, these latents are continuous and are also not evaluated or intended as general-purpose tokens.

**Discrete neural tokenizers in adjacent modalities.** The closest precedents for standalone neural tokenization come from non-invasive signals. EEG tokenizers such as LaBraM [33] and NeuroRVQ [11], among others [45, 14, 45], train discrete codebooks that can be reused by downstream models, and show how tokenizer choice can affect downstream performance. These motivate the modular perspective adopted here, but spiking data differs in ways that make direct transfer from other neural data modalities non-trivial: spikes are sparse, binary events observed at millisecond resolution, with variable neuron identities, session structure, firing-rate regimes, and population statistics.

**Benchmarks for neural data modeling.** Existing neural benchmarks evaluate models built on a fixed input representation. NLB [43] measures how well latent-variable models recover firing rates of held-out neurons (co-smoothing) and decode behavior from population activity. FALCON [35] measures how well decoders transfer across sessions and days with minimal recalibration data. CtDB [56] measures how well dynamical models recover ground-truth dynamics on synthetic data. In each case, the tokenization choice is held fixed and the downstream model is the experimental variable. Our framework evaluates a neural tokenizer, which determines what neural information any subsequent decoder, encoder, or foundation model can access.

## 6 Tokenizer implementations

To evaluate tokenizers systematically, we developed a modular framework in which new tokenizers contribute a minimal `encode`/`decode` interface against one of three supported input datatypes: spike times, binary binned spikes at 1ms resolution, or firing rates at coarser bin sizes. For learned, autoencoder-style models, the framework provides composable bottlenecks, losses, and input transforms. Using this framework, we assembled a zoo spanning several design axes: single-neuron/population inputs, continuous/discrete bottlenecks, and varying levels of compression (as a function of binning resolution, input window length, and downsampling rates).

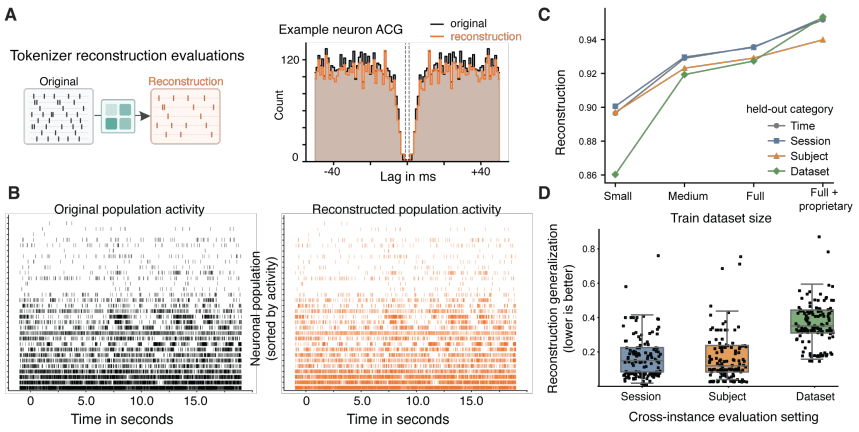


**Figure 3: Reconstruction results.** **A**. Schematic of the spike-tokenization reconstruction task. **B**. Held-out population spike rasters for the best single-neuron tokenizer; ground truth (left) and reconstruction (right) align in spike timing over multi-second windows. **C**. Reconstruction score scales with training corpus size, with the cross-source split starting substantially lower at a small scale and closing most of the gap only at the full corpus. **D**. Generalization across the zoo, reported as the score on each held-out category (held-out sessions, subjects, sources), measured against the held-out-time ($xT$) within-session reference. Reconstruction generalizes well across sessions and subjects, but degrades sharply across held-out datasets.

| Tokenizer | SN Recon ↑ $(-\infty, 1]$ | Pop. Stats ↑ $(-\infty, 1]$ | Decoding ↑ $[0, 1]$ | Encoding ↑ $[0, 1]$ | Latent Rob. ↑ $(0, 1]$ | Generalization ↑ $[0, 1]$ | Compression ↑ |
|---|---|---|---|---|---|---|---|
| *Single-neuron tokenizers (1 ms bins)* | | | | | | | |
| PCA | 0.689 | -1.000 | – | – | 0.652 | 0.737 | – |
| C.S. | 0.513 | -0.732 | 0.650 | 0.220 | 0.590 | 0.669 | 0.3× |
| Conv-VAE (100 : 1) | **0.914** | **0.728** | 0.839 | **0.295** | 0.374 | **0.866** | 0.4× |
| Conv-RFSQ (100 : 1) | 0.710 | -0.725 | 0.840 | 0.274 | 0.669 | 0.813 | 0.6× |
| Transformer-VQ (100 : 1) | 0.812 | -0.705 | **0.850** | 0.125 | 0.267 | 0.713 | 9.1× |
| Conv-VAE (1000 : 1) | 0.240 | -0.946 | 0.820 | – | 0.432 | 0.745 | 2.0× |
| *Population tokenizers (10 ms bins; not directly comparable to single-neuron rows above)* | | | | | | | |
| LDNS | -0.550 | -1.000 | 0.310 | – | 0.631 | 0.686 | **8.6**× |
| Wavelet-FSQ | **0.995** | **0.606** | **0.580** | 0.153 | **0.927** | **0.827** | 1.6× |
| LDNS-Perceiver | 0.125 | -0.689 | 0.121 | **0.230** | 0.815 | – | – |

**Table 2: Evaluating tokenizers across all seven evaluation framework dimensions (Section 3).** We present aggregate scores representing each dimension for a sub-selection from our tokenizer zoo. Single-neuron (*SN Recon*) and population stats are both normalized against a 10 ms Poisson rate baseline (0 matches baseline, see section D.3). *Decoding* and *Encoding* report $R^2$ on our POYO-style [6] behavioral decoding and neural encoding targets (section 7.2), respectively; Latent robustness is also an aggregate across various metrics of latent-space stability under perturbations; *Generalization* aggregates metric retention across our held-out tiers (section 4). *Compression* is the input-to-output bit ratio (section 3). See section B for more aggregation details. Note, $X : 1$ refers to the compression level of input ms to latents (see section 6).

We define a *single-neuron* tokenizer as taking as input a 1-D temporal sequence from a single unit; for our exploration in this regime we restrict to binary spike bins ($\mathbf{x} \in 0, 1^{l_w}$) and two compression levels—100 : 1 and 1000 : 1—producing one token per 100ms and per 1s of activity, respectively. A *population* tokenizer instead encodes the joint activity of all simultaneously recorded neurons, taking as input a $T \times N$ matrix of binned firing rates rather than a single-unit sequence.

Within this space, we explored several design axes. For *architecture*, we implemented convolutional and transformer-based models. We tried continuous VAE-style bottlenecks as well as discrete quantization methods—VQ [53], FSQ [39], BSQ [70], LFQ [66], and RVQ [67]—along with various regularization strategies, for example, a latent straightening penalty [58] or SIGReg [9]. For the *reconstruction objective*, we explored binary cross-entropy and focal loss for binary spike outputs, Poisson NLL for rate-valued decoders, and MSE, L1, Huber, and Wasserstein losses as alternatives, as well as optional target softening transforms. Finally, we tried a variety of hand-designed *input-space transforms* based on fixed discrete wavelet families (e.g. Haar, Db2, Db4). Our framework also supports non-learned algorithms, and along these lines, we included a PCA baseline and a compressed-sensing tokenizer. Full architectural and training details of our models are in Appendix D.

## 7 Results

We deploy the framework on the tokenizer zoo and report what it reveals. Table 2 reports tokenizer scores across the seven evaluation dimensions defined in Section 3, with rows grouped by population vs. single-neuron tokenizers. The scores reveal that while there are substantial differences across tokenizers, no single tokenizer dominates all dimensions—confirming that single-axis evaluation would obscure the design trade-offs our framework is built to expose.

Three claims structure the rest of this section. First, reconstruction quality is high for many tokenizers and scales with training data. Second, reconstruction-trained tokenizers serve as viable backbones for downstream encoding and decoding. Third, reconstruction is necessary but not sufficient: some tokenizers achieve high reconstruction with latents that carry little structure a downstream readout can use, making it the most tractable axis to optimize and the most misleading one to optimize alone.

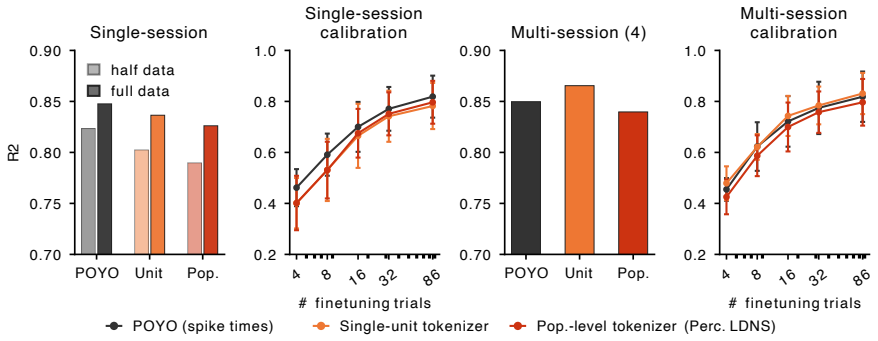


**Figure 4: Tokenized data supports POYO[6]-style temporal decoding.** We compare the native spike-time baseline, a frozen single-unit tokenizer, and a frozen population-level tokenizer on [44] reaching sessions. Left-to-right: **(a)** Single-session scratch decoding with mean held-out test $R^2$ over eight sessions and two seeds. **(b)** Single-session calibration from two base sessions to held-out sessions over different budgets of target-session trials. **(c)** Multi-session training with four sessions over two seeds. **(d)** Calibration to held-out sessions over different budgets of target-session trials.

### 7.1 Reconstruction quality & generalization

Many tokenizers achieve near-perfect reconstruction. Figure 3 illustrates this for the best single-neuron tokenizer evaluated on a held-out session: reconstructed spike trains closely match held-out originals over multi-second windows, and single-unit temporal statistics are recovered with high fidelity. Reconstruction quality scales with training data, and the gains are largest in the cross-source regime, which starts substantially lower at a small scale and closes most of the gap only with the full corpus. Across the full zoo, reconstruction generalizes well across sessions and subjects, with an accuracy loss of approximately 10–15%, but degrades more sharply across sources (loss of ~35%).

### 7.2 Downstream utility of tokenizers

Reconstruction quality establishes that a tokenizer preserves the input, but downstream utility determines its value in foundation model pipelines. We test this directly by evaluating whether frozen tokenizer outputs can replace native spike-time inputs in a POYO[6]-style temporal decoder (Figure 4). This setting is more general than a linear probe, using frozen tokenizer representations as the neural input interface for an attention-based behavioral decoder. Across single-session training, multi-session base training, and cross-session calibrations with different trial budgets, tokenizers performs competitively with the native spike-time POYO baseline. We note that POYO was designed around spike-time inputs, so the ceiling for tokenized inputs could likely be raised with architectural adaptations. For more details, see Appendix E.2.

A parallel evaluation reverses the direction, asking whether stimuli can be used to predict into the tokenizer's latent space and whether then using the frozen decoder can recover PSTH-like responses. Encoding results in Table 2 (full breakdown in Table S11) show substantial variation across tokenizers despite similar reconstruction quality (Fig. 5A), also motivating our dissociation analysis.

On proxy tasks that evaluate latent space alignment with stimulus structure and robustness, most tokenizers underperform Poisson rates S8, showing the difficulty of learning complex neural tokenizers while maintaining structural fidelity to the data.

### 7.3 Reconstruction quality is necessary but not sufficient

A tokenizer that achieves high reconstruction need not produce latents that are useful downstream. Compressed sensing makes the point sharply. As a fixed random projection, it has no learned structure, yet its decoder reconstructs spike trains with fidelity competitive with the best learned tokenizers. On the latent-utility metrics, however, its representational similarity to the input collapses, and downstream probes recover little about the stimulus's relationship to neural activity.

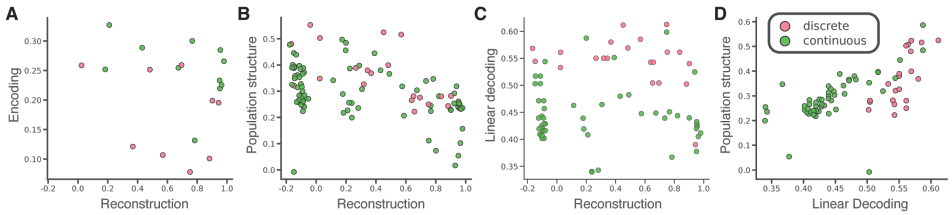


**Figure 5: Tokenizer dissociations.** Each point is a tokenizer in the zoo, colored by bottleneck type (discrete, pink; continuous, green). **A**. Reconstruction quality is uncorrelated with encoding $R^2$: high-fidelity reconstruction does not guarantee latents that support neural encoding. **B**. Population structure preservation falls as single-neuron reconstruction improves, exposing a trade-off between fine spike level fidelity and population geometry. **C**. Reconstruction quality is likewise uncorrelated with linear behavioral decoding $R^2$. **D**. Population-structure preservation, in contrast, is positively correlated with linear decoding, identifying it as the axis that actually predicts downstream readout.

Across the full model zoo, reconstruction quality and downstream utility dissociate along several axes (Figure 5). We observe no clear monotonic relationship between single-neuron reconstruction fidelity and either encoding $R^2$ or linear decoding accuracy. Models that preserve single-unit temporal statistics well are not necessarily best preserving population structure, suggesting that these objectives emphasize different aspects of the data. In contrast, stronger linear decoding performance tends to occur for tokenizers with better preserved population structure, consistent with population geometry being more relevant for simple downstream readouts than precise timing. These trends suggest that the tokenizer's full profile across axes is more diagnostic than any single reconstruction metric.

## 8 Discussion

We introduce a framework for evaluating spike tokenization as a modular component, and use it to compare a range of tokenization strategies systematically. A sharp dissociation emerges between reconstruction fidelity and downstream utility: tokenizers with near-perfect reconstruction can yield latent representations that are either highly informative or nearly useless for tasks the tokenizer was never trained for. This reveals a fundamental open problem. Reconstruction loss is the natural self-supervised objective for a task-agnostic tokenizer, yet it does not appear to capture the structure that downstream models need.

This points to a deeper question that the field has not yet answered: what self-supervised objective should a standalone spike tokenizer optimize? In language, next-token prediction turned out to capture surprisingly universal structure [47, 17, 46]. In vision, masked reconstruction [10, 30] and contrastive objectives [19, 29] transfer broadly across tasks. The analogous question for spikes, i.e., what prediction task over spike trains forces a tokenizer to capture universally useful structure, remains open. Our evaluation framework provides the diagnostic tool to measure progress toward that goal, but identifying the right objective is the central open problem that this work surfaces.

**Limitations.** The framework's predictive validity for full foundation-model performance remains a hypothesis rather than an established result. The metric set is not exhaustive and is designed to grow through community contributions. The current suite covers electrophysiological spikes only, not calcium imaging data or LFP. The evaluation pipeline assumes fixed-rate tokenization, excluding variable-length and event-based designs.

**Future directions.** The framework opens several directions beyond the tokenizers evaluated here. Hybrid tokenization schemes that combine single-neuron and population-level representations can be evaluated within the same framework, turning the trade-off between fine timing and population geometry into a design space rather than a forced choice. The dissociation between reconstruction and downstream utility suggests alternative training objectives—targeting latent structure or downstream alignment—may yield more useful representations than reconstruction alone. Finally, integration with benchmarks such as NLB and FALCON will test whether tokenizer-axis scores predict downstream foundation model performance. By releasing the dataset pipeline, evaluation suite, and tokenizer interface, we aim to enable systematic comparison and extension of spike tokenization methods.

## References


[1] N. Agarwal, A. Ali, M. Bala, Y. Balaji, E. Barker, T. Cai, P. Chattopadhyay, Y. Chen, Y. Cui, Y. Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

[2] M. Ali, M. Fromm, K. Thellmann, R. Rutmann, M. Lübbering, J. Leveling, K. Klug, J. Ebert, N. Doll, J. S. Buschhoff, C. Jain, A. A. Weber, L. Jurkschat, H. Abdelwahab, C. John, P. O. Suarez, M. Ostendorff, S. Weinbach, R. Sifa, S. Kesselheim, and N. Flores-Herr. Tokenizer choice for LLM training: Negligible or crucial?, 2024.

[3] M. Alleman, J. Lindsey, and S. Fusi. Task structure and nonlinearity jointly determine learned representational geometry. In *The Twelfth International Conference on Learning Representations*, Oct. 2023.

[4] Allen Institute for Brain Science. Visual behavior neuropixels technical whitepaper. Technical whitepaper, Allen Institute for Brain Science, 2022. URL `https://portal.brain-map.org/circuits-behavior/visual-behavior-neuropixels`.

[5] V. Arora, D. Lachi, I. J. Knight, M. Azabou, B. A. Richards, C. L. Hurwitz, J. Siegle, and E. L. Dyer. Know Thyself by Knowing Others: Learning Neuron Identity from Population Context. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, Oct. 2025.

[6] M. Azabou, V. Arora, V. Ganesh, X. Mao, S. Nachimuthu, M. Mendelson, B. Richards, M. Perich, G. Lajoie, and E. L. Dyer. A unified, scalable framework for neural population decoding. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[7] M. Azabou, K. X. Pan, V. Arora, I. J. Knight, E. L. Dyer, and B. A. Richards. Multi-session, multi-task neural decoding from distinct cell-types and brain regions. In *The Thirteenth International Conference on Learning Representations*, Oct. 2024.

[8] R. Azouz and C. M. Gray. Cellular mechanisms contributing to response variability of cortical neurons in vivo. *Journal of Neuroscience*, 19(6):2209–2223, 1999.

[9] R. Balestriero and Y. LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics, 2025.

[10] H. Bao, L. Dong, S. Piao, and F. Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[11] K. Barmpas, N. Lee, A. Koliousis, Y. Panagakis, D. A. Adamos, N. Laskaris, and S. Zafeiriou. NeuroRVQ: Multi-scale EEG tokenization for generative large brainwave models, 2025.

[12] M. Beau, D. J. Herzfeld, F. Naveros, M. E. Hemelt, F. D'Agostino, M. Oostland, A. Sanchez-Lopez, Y. Y. Chung, M. Maibach, S. Kyranakis, H. N. Stabb, M. G. Martinez Lopera, A. Lajko, M. Zedler, S. Ohmae, N. J. Hall, B. A. Clark, D. Cohen, S. G. Lisberger, D. Kostadinov, C. Hull, M. Häusser, and J. F. Medina. A deep-learning strategy to identify cell types across species from high-density extracellular recordings. *Cell*, 188(8):2218–2234.e22, 2025. doi: 10.1016/j.cell.2025.01.041. URL `https://www.cell.com/cell/abstract/S0092-8674(25)00110-2`.

[13] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. doi: 10.1137/080716542. URL `https://doi.org/10.1137/080716542`.

[14] R. G. Bettinardi, M. Rahmouni, and U. Gimenez. BioSerenity-E1: A self-supervised EEG model for medical applications, Mar. 2025.

[15] Å. Björck and G. H. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, 27(123):579–594, 1973.

[16] K. Bostrom and G. Durrett. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, 2020.

[17] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[18] E. J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006. doi: 10.1109/TIT.2006.885507.

[19] M. Caron, H. Touvron, I. Misra, H. Jegou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9630–9640, 2021. doi: 10.1109/ICCV48922.2021.00951.

[20] M. M. Churchland, B. M. Yu, J. P. Cunningham, L. P. Sugrue, M. R. Cohen, G. S. Corrado, W. T. Newsome, A. M. Clark, P. Hosseini, B. B. Scott, et al. Stimulus onset quenches neural variability: a widespread cortical phenomenon. *Nature neuroscience*, 13(3):369–378, 2010.

[21] M. M. Churchland, J. P. Cunningham, M. T. Kaufman, J. D. Foster, P. Nuyujukian, S. I. Ryu, and K. V. Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012. doi: 10.1038/nature11129. URL https://www.nature.com/articles/nature11129.

[22] B. R. Cowley, P. L. Stan, J. W. Pillow, and M. A. Smith. Compact deep neural network models of the visual cortex. *Nature*, pages 1–8, Feb. 2026. ISSN 1476-4687. doi: 10.1038/s41586-026-10150-1.

[23] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006. doi: 10.1109/TIT.2006.871582.

[24] D. L. Donoho and J. Tanner. Counting the Faces of Randomly-Projected Hypercubes and Orthants, with Applications, July 2008.

[25] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[26] R. D. Flint, E. W. Lindberg, L. R. Jordan, L. E. Miller, and M. W. Slutzky. Accurate decoding of reaching movements from field potentials in the absence of spikes. *Journal of Neural Engineering*, 9(4):046006, 2012. doi: 10.1088/1741-2560/9/4/046006. URL https://iopscience.iop.org/article/10.1088/1741-2560/9/4/046006.

[27] P. Gao, E. Trautmann, B. Yu, G. Santhanam, S. Ryu, K. Shenoy, and S. Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *bioRxiv*, 2017. doi: 10.1101/214262. URL https://www.biorxiv.org/content/early/2017/11/12/214262.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2020. doi: 10.1109/CVPR42600.2020.00975.

[30] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15979–15988, 2022. doi: 10.1109/CVPR52688.2022.01553.

[31] International Brain Laboratory. A brain-wide map of neural activity during complex behaviour. *Nature*, 645(8079):177–191, 2025. doi: 10.1038/s41586-025-09235-0. URL https://www.nature.com/articles/s41586-025-09235-0. Preprint posted on bioRxiv in 2023 (doi:10.1101/2023.07.04.547681).

[32] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira. Perceiver: General perception with iterative attention. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR, 2021. URL http://proceedings.mlr.press/v139/jaegle21a.html.

[33] W.-B. Jiang, L.-M. Zhao, and B.-L. Lu. Large brain model for learning generic representations with tremendous EEG data in BCI. In *The Twelfth International Conference on Learning Representations*, 2024.

[34] J. Kapoor, A. Schulz, J. Vetter, F. Pei, R. Gao, and J. H. Macke. Latent diffusion for neural spiking data. *Advances in Neural Information Processing Systems*, 37:118119–118154, 2024.

[35] B. M. Karpowicz, J. Ye, C. Fan, P. Tostado-Marcos, F. Rizzoglio, C. B. Washington, T. Scodeler, D. S. de Lucena, S. R. Nason-Tomaszewski, M. Mender, X. Ma, E. M. Arneodo, L. Hochberg, C. Chestek, J. M. Henderson, T. Q. Gentner, V. Gilja, L. E. Miller, A. G. Rouse, R. Gaunt, and C. Pandarinath. Few-shot algorithms for consistent neural decoding (FALCON) benchmark. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2024.

[36] M. R. Keshtkaran, A. R. Sedler, R. H. Chowdhury, R. Tandon, D. Basrai, S. L. Nguyen, H. Sohn, M. Jazayeri, L. E. Miller, and C. Pandarinath. A large-scale neural network training framework for generalized estimation of single-trial population dynamics. *Nature Methods*, 19(12):1572–1577, Dec. 2022. ISSN 1548-7105. doi: 10.1038/s41592-022-01675-0.

[37] N. Kriegeskorte, M. Mur, and P. A. Bandettini. Representational similarity analysis: connecting the branches of systems neuroscience. *Frontiers in Systems Neuroscience*, 2:4, 2008. ISSN 1662-5137. doi: 10.3389/neuro.06.004.2008.

[38] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2003. ISBN 0521642981. URL http://www.inference.org.uk/itila/book.html.

[39] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

[40] J. E. O'Doherty, M. M. B. Cardoso, J. G. Makin, and P. N. Sabes. Nonhuman primate reaching with multichannel sensorimotor cortex electrophysiology. Zenodo dataset, 2017. URL https://zenodo.org/records/3854034.

[41] M. Okun, N. A. Steinmetz, L. Cossell, M. F. Iacaruso, H. Ko, P. Bartó, T. Moore, S. B. Hofer, T. D. Mrsic-Flogel, M. Carandini, and K. D. Harris. Diverse coupling of neurons to populations in sensory cortex. *Nature*, 521(7553):511–515, May 2015.

[42] C. Pandarinath, D. J. O'Shea, J. Collins, R. Jozefowicz, S. D. Stavisky, J. C. Kao, E. M. Trautmann, M. T. Kaufman, S. I. Ryu, L. R. Hochberg, J. M. Henderson, K. V. Shenoy, L. F. Abbott, and D. Sussillo. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10): 805–815, 2018. doi: 10.1038/s41592-018-0109-9.

[43] F. Pei, J. Ye, D. Zoltowski, A. Wu, R. H. Chowdhury, H. Sohn, J. E. O'Doherty, K. V. Shenoy, M. T. Kaufman, M. M. Churchland, M. Jazayeri, L. E. Miller, J. Pillow, I. M. Park, E. L. Dyer, and C. Pandarinath. Neural latents benchmark '21: Evaluating latent variable models of neural population activity. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[44] M. G. Perich, J. A. Gallego, and L. E. Miller. A neural population mechanism for rapid learning. *Neuron*, 100(4):964–976.e7, 2018. doi: 10.1016/j.neuron.2018.09.030.

[45] J. Pradeepkumar, X. Piao, Z. Chen, and J. Sun. Tokenizing Single-Channel EEG with Time-Frequency Motif Learning, Feb. 2026.

[46] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[47] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[48] P. Rust, J. Pfeiffer, I. Vulić, S. Ruder, and I. Gurevych. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, 2021.

[49] A. H.-W. Ryoo, N. H. Krishna, X. Mao, M. Azabou, E. L. Dyer, M. G. Perich, and G. Lajoie. Generalizable, real-time neural decoding with hybrid state-space models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, Oct. 2025.

[50] M. Sahani and J. Linden. How linear are auditory cortical responses? *Advances in neural information processing systems*, 15, 2002.

[51] J. H. Siegle, X. Jia, S. Durand, S. Gale, C. Bennett, N. Graddis, G. Heller, T. K. Ramirez, H. Choi, J. A. Luviano, P. A. Groblewski, R. Ahmed, A. Arkhipov, A. Bernard, Y. N. Billeh, D. Brown, M. A. Buice, N. Cain, S. Caldejon, L. Casal, A. Cho, M. Chvilicek, T. C. Cox, K. Dai, D. J. Denman, S. E. J. de Vries, R. Dietzman, L. Esposito, C. Farrell, D. Feng, J. Galbraith, M. Garrett, E. C. Gelfand, N. Hancock, J. A. Harris, R. Howard, B. Hu, R. Hytnen, R. Iyer, E. Jessett, K. Johnson, I. Kato, J. Kiggins, S. Lambert, J. Lecoq, P. Ledochowitsch, J. H. Lee, A. Leon, Y. Li, E. Liang, F. Long, K. Mace, J. Melchior, D. Millman, T. Mollenkopf, C. Nayan, L. Ng, K. Ngo, T. Nguyen, P. R. Nicovich, K. North, G. K. Ocker, D. Ollerenshaw, M. Oliver, M. Pachitariu, J. Perkins, M. Reding, D. Reid, M. Robertson, K. Ronellenfitch, S. Seid, C. Slaughterbeck, M. Stoecklin, D. Sullivan, B. Sutton, J. Swapp, C. Thompson, K. Turner, W. Wakeman, J. D. Whitesell, D. Williams, A. Williford, R. Young, H. Zeng, S. Naylor, J. W. Phillips, R. C. Reid, S. Mihalas, S. R. Olsen, and C. Koch. Survey of spiking in the mouse visual system reveals functional hierarchy. *Nature*, 592(7852):86–92, 2021. doi: 10.1038/s41586-020-03171-x. URL https://www.nature.com/articles/s41586-020-03171-x.

[52] The MICrONS Consortium. Functional connectomics spanning multiple areas of mouse visual cortex. *Nature*, 640:435–447, 2025. doi: 10.1038/s41586-025-08790-w.

[53] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.

[54] M. C. W. van Rossum. A novel spike distance. *Neural Computation*, 13(4):751–763, 2001. doi: 10.1162/089976601300014321.

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. ukasz Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[56] C. Versteeg, J. D. McCart, M. Ostrow, D. M. Zoltowski, C. B. Washington, L. Driscoll, O. Codol, J. A. Michaels, S. W. Linderman, D. Sussillo, and C. Pandarinath. Computation-through-dynamics benchmark: Simulated datasets and quality metrics for dynamical models of neural activity. *bioRxiv*, 2025. doi: 10.1101/2025.02.07.637062.

[57] T. Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, D. Chen, F. Yu, H. Zhao, J. Yang, J. Zeng, J. Wang, J. Zhang, J. Zhou, J. Wang, J. Chen, K. Zhu, K. Zhao, K. Yan, L. Huang, M. Feng, N. Zhang, P. Li, P. Wu, R. Chu, R. Feng, S. Zhang, S. Sun, T. Fang, T. Wang, T. Gui, T. Weng, T. Shen, W. Lin, W. Wang, W. Wang, W. Zhou, W. Wang, W. Shen, W. Yu, X. Shi, X. Huang, X. Xu, Y. Kou, Y. Lv, Y. Li, Y. Liu, Y. Wang, Y. Zhang, Y. Huang, Y. Li, Y. Wu, Y. Liu, Y. Pan, Y. Zheng, Y. Hong, Y. Shi, Y. Feng, Z. Jiang, Z. Han, Z.-F. Wu, and Z. Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[58] Y. Wang, O. Bounou, G. Zhou, R. Balestriero, T. G. Rudner, Y. LeCun, and M. Ren. Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*, 2026.

[59] W. F. Whitney, M. J. Song, D. Brandfonbrener, J. Altosaar, and K. Cho. Evaluating representations by the complexity of learning low-loss predictors. Oct. 2020.

[60] K. F. Willeke, P. Turishcheva, A. Gilbert, G. Chakrabarty, H. A. Bedel, P. G. Fahey, Y. Qiu, M. A. Weis, M. Vystrčilová, T. Muhammad, et al. Omnimouse: Scaling properties of multi-modal, multi-task brain models on 150b neural tokens. *arXiv preprint arXiv:2604.18827*, 2026.

[61] J. C. Wingert, S. Parida, S. V. Norman-Haignere, and S. V. David. Convolutional neural network models describe the encoding subspace of local circuits in auditory cortex. *Nature Neuroscience*, 2026. doi: 10.1038/s41593-026-02216-0. In press.

[62] X. Xu, Y. Zhang, Q. Qian, and Y. Zhang. Self-Supervised Foundation Model for Calcium-imaging Population Dynamics, Apr. 2026.

[63] J. Ye and C. Pandarinath. Representation learning for neural population activity with neural data transformers. *Neurons, Behavior, Data Analysis, and Theory*, 2021. doi: 10.51628/001c.27358.

[64] J. Ye, J. L. Collinger, L. Wehbe, and R. Gaunt. Neural data transformer 2: Multi-context pretraining for neural spiking activity. In *Advances in Neural Information Processing Systems*, 2023.

[65] L. Yu, Y. Cheng, K. Sohn, J. Lezama, H. Zhang, H. Chang, A. G. Hauptmann, M.-H. Yang, Y. Hao, I. Essa, and L. Jiang. MAGVIT: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[66] L. Yu, J. Lezama, N. B. Gundavarapu, L. Versari, K. Sohn, D. Minnen, Y. Cheng, V. Birodkar, A. Gupta, X. Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.

[67] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[68] Y. Zhang, Y. Wang, D. M. Jiménez-Benetó, Z. Wang, M. Azabou, B. A. Richards, R. Tung, O. Winter, I. B. Laboratory, E. L. Dyer, L. Paninski, and C. L. Hurwitz. Towards a "Universal Translator" for Neural Dynamics at Single-Cell, Single-Spike Resolution. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, Nov. 2024.

[69] Y. Zhang, Y. Wang, M. Azabou, A. Andre, Z. Wang, H. Lyu, International Brain Laboratory, E. L. Dyer, L. Paninski, and C. Hurwitz. Neural encoding and decoding at scale. In *International Conference on Machine Learning*, 2025.

[70] Y. Zhao, Y. Xiong, and P. Krähenbühl. Image and video tokenization with binary spherical quantization. *arXiv preprint arXiv:2406.07548*, 2024.