A PIPELINE FOR INTERPRETABLE NEURAL LATENT DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Mechanistic understanding of the brain requires interpretability of large-scale neuronal computations. Many latent variable model approaches applied to such datasets excel at decoding but yield opaque latent spaces. We address this with NLDisco, a pipeline for interpretable neural latent discovery. Motivated by successful applications of sparse dictionary learning in AI mechanistic interpretability, NLDisco encourages hidden layer neurons in sparse encoder-decoder models to learn interpretable structure in dense neural activity. A flexible, open-source software package supports interpretable latent discovery across recording modalities and experimental paradigms. We validate the pipeline on both synthetic and real extracellular recordings across species and brain regions, showing that it recovers ground-truth features in synthetic data and reveals meaningful representations in real data. These results highlight the pipeline's potential to advance neuroscientific discovery and provide a foundation for future applications and extensions.

1 INTRODUCTION

Advanced neural recording technologies now enable neuroscientists to capture unprecedented volumes of high-dimensional neural[1] data at single-cell resolution (Steinmetz et al., 2021; Raducanu et al., 2017; Cai et al., 2016; Villette et al., 2019; Ouzounov et al., 2017; Ahrens et al., 2013). Understanding the computational principles driving neural activity requires extracting interpretable features from this data. Traditional dimensionality reduction techniques, while useful for visualization and basic analyses, often make invalid assumptions about or altogether fail to disentangle distributed representations, and as a result typically cannot provide a mechanistic understanding of the underlying computations (Cunningham & Yu, 2014; Humphries, 2021). Recent work has produced promising latent variable model (LVM) approaches capable of identifying low-dimensional subspaces that can accurately decode aspects of behavior and environment (Song et al., 2025; Schneider et al., 2023; Le & Shlizerman, 2022; Keshtkaran et al., 2022; Yu et al., 2009; Macke et al., 2011; Gao et al., 2016; Low et al., 2018; Jensen et al., 2020; Hernandez et al., 2020; Kim et al., 2021; Hurwitz et al., 2021; Schimel et al., 2022; Kudryashova et al., 2023; Ye et al., 2023; Gondur et al., 2024; Pellegrino et al., 2024; Sani et al., 2024; Pals et al., 2024; Zhang et al., 2024; George et al., 2025; Perkins et al., 2025; Schmutz et al., 2025). However, these methods typically value decoding accuracy over latent interpretability, and consequently all have at least one of the following limitations: poorly interpretable latents, lossy mapping to and/or poor reconstruction of neural data from latent space, priors on the latent and/or neural data space, requirements of additional behavioral, environmental, and/or trial-structured neural data, supralinear scaling with respect to dataset size, or high implementational complexity.

We address each of these limitations in NLDisco, which provides a simple, user-friendly library for interpretable latent discovery from high-dimensional neural data. We consider a latent's interpretability in two key aspects: 1) its correspondence to a specific external variable – a "natural" behavioral or environmental feature[2]; 2) its explicit composition from contributing neural activity. Unlike other approaches that require a search for meaningful directions or dynamics in a latent space, NLDisco

[1] By default, "neural" and "neuron" refer to neurobiology; e.g. we call brain spike data "neural activity". In contrast, activity and neurons from artificial neural networks are explicitly prefaced; e.g. "model neurons", "hidden layer neuron". For neural data, "unit" denotes a signal source that can but does not necessarily imply correspondence to a single neuron.

[2] A "feature" is a sufficiently interpretable latent.

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.

outputs individual latents that can be directly assessed for interpretability. Additionally, while we showcase NLDisco on spike data, it can be readily used with virtually any other neural recording modality as the only data requirement is any predefined spatiotemporal binning.

NLDisco trains hidden layer neurons in shallow, overcomplete, sparse (Olshausen & Field, 1997) encoder-decoder (SED) models (Vincent et al., 2008) to learn interpretable representations of neural activity. An overcomplete dictionary provides capacity for many candidate features, while sparsity limits how many contribute to any one reconstruction. Together, these encourage explanations of population activity in terms of a set of reusable patterns, making individual latents and their contributions easier to interpret. The overarching field of sparse dictionary learning has had recent successes in mechanistic interpretability, in which models have typically been implemented as sparse autoencoders (SAEs), transcoders (CLTs), or crosscoders (SCCs) (Lindsey et al., 2024; Huben et al., 2024; Bricken et al., 2023; Templeton et al., 2024; Dunefsky et al., 2024; Ameisen et al., 2025; Lindsey et al., 2025). For neural data, SEDs are additionally motivated by the principle that we need not explicitly impose structure on a latent space nor model its dynamics to discover interpretable features.

Specifically, NLDisco bypasses the need to enforce non-Markovian dynamics, which are often a property of an incomplete observation of a system rather than the system itself; any dynamical system can be represented as Markovian via sufficient state augmentation (Takens, 1981). Given this, even methods that appear non-Markovian can be seen as attempts to approximate a complete underlying state. For example, the forward dynamics of a recurrent neural network (RNN) are Markovian in its hidden state (Sussillo & Barak, 2013; Goodfellow et al., 2016), and generalized linear models (GLMs) with history filters (Pillow et al., 2008; Truccolo et al., 2005) use past activity to augment the present state. This perspective aligns with several findings in neuroscience, from activity in motor cortex reflecting a low-dimensional, implicitly Markovian dynamical system (Churchland et al., 2012; Shenoy et al., 2013), to predictive-coding formulations that posit that the brain maintains an internal state sufficient to predict sensory streams, i.e., a Markovian latent process (Rao & Ballard, 1999; Doya et al., 2007; Friston, 2010). Grounded in this perspective, NLDisco forgoes the challenge of modeling intricate dynamics and instead directly extracts meaningful constituent features of the underlying neural state (Figure S1).

2 Related work and positioning

Sparse representation learning. NLDisco builds on sparse coding and dictionary learning methods that represent observations with a small number of elements from an overcomplete dictionary (Olshausen & Field, 1997; Hoyer, 2004). Closely related SAEs have been used in mechanistic interpretability to identify approximately monosemantic features in artificial neural network activations (Huben et al., 2024; Bricken et al., 2023; Templeton et al., 2024; Dunefsky et al., 2024; Lindsey et al., 2024; Ameisen et al., 2025). NLDisco adapts this approach to neural population recordings: each sparse latent can be related both to external experimental variables and, through its encoder weights, to the recorded units and timebins that compose it.

Latent models of neural population activity. Classical approaches such as PCA and ICA expose structure in population activity through linear components whose interpretation is guided by variance or statistical independence (Hotelling, 1933; Comon, 1994). Other LVMs capture richer population geometry and temporal dynamics given a smooth state space assumption, yielding representations useful for generative and decoding tasks, although these objectives do not necessarily make individual latent dimensions interpretable (Yu et al., 2009; Pandarinath et al., 2018; Keshtkaran et al., 2022; Schneider et al., 2023; Song et al., 2025). NLDisco instead makes the individual sparse latent the primary focus. It does not require behavioral supervision or trial structure; it reconstructs the observed population activity such that each latent's neural composition remains explicit, and complements rather than replaces trajectory-level analyses. A detailed comparison of assumptions and capabilities is provided in Table S1.

Temporal representation learning. Sequence models of neural population activity, including recurrent latent dynamics models and transformer-based approaches, use temporal context to improve reconstruction or prediction (Sussillo & Barak, 2013; Le & Shlizerman, 2022; Ye et al., 2023; Zhang et al., 2024). NLDisco uses temporal context for a different objective: discovering representations

that emerge or evolve over time by introduction of the novel yet conceptually simple Window SED (W-SED). A W-SED treats $S > 1$ timebins as a single example and reconstructs the full window, yielding sparse latents that explicitly reconstruct timebin-by-unit activity patterns rather than just serving as directions in a latent space. Unlike standard SAEs used in language-model interpretability, which learn features at an individual sequence position, W-SEDs integrate across sequence positions.

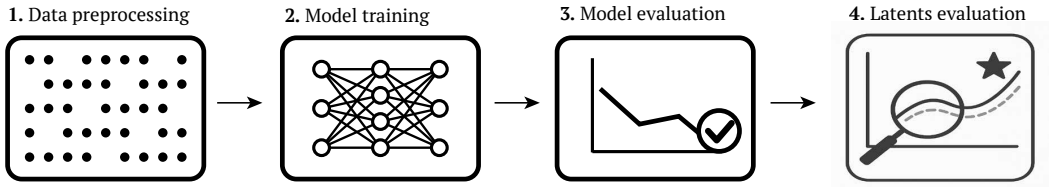


Figure 1: The NLDisco pipeline.
The NLDisco pipeline has four stages: 1) Spatiotemporal binning and processing of neural data; 2) Training a model; 3) Evaluating the model; 4) Evaluating the model's latents for feature interpretability. Steps 1-3 can be either semi- or fully-automated.

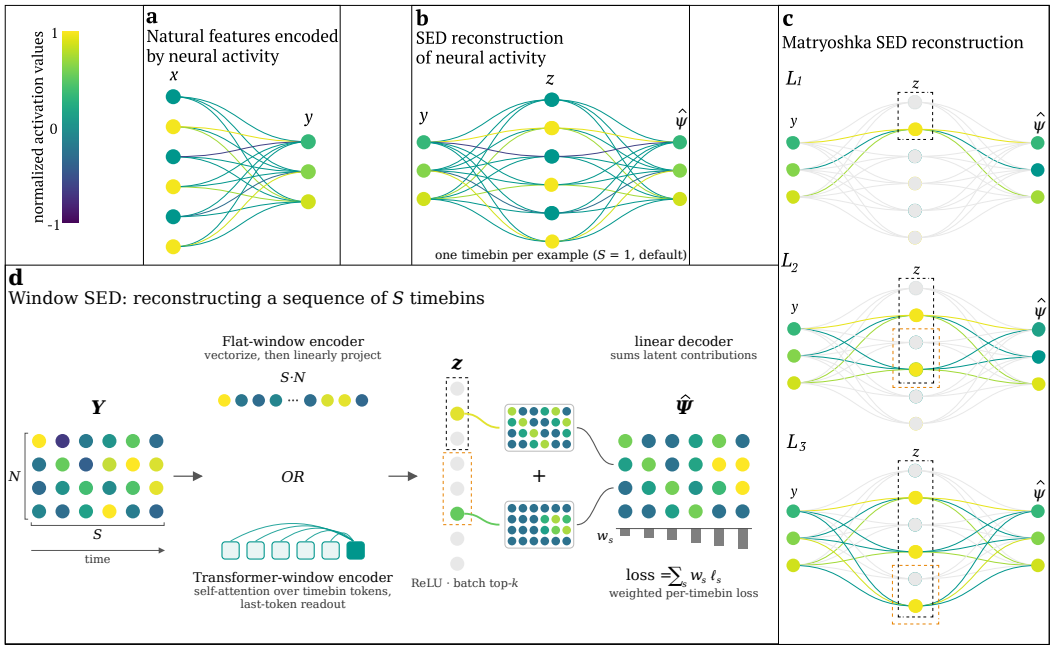


Figure 2: Model architecture
(**a**) Natural, "real-world" features x are encoded by neural activity y. In this example, three active (of six total) features are simultaneously represented by the joint activity of three neurons. (**b**) An SED reconstructs target neural activity ψ as $\hat{\psi}$ from input y through sparsely active latents z. When training is successful, the learned latents z correspond to natural features x. Reconstructing the input population itself ($\psi = y$) gives an autoencoder; reconstructing a separate but related target population gives a transcoder. By default, one training example is a single timebin. (**c**) A Matryoshka SED divides the latents into nested levels $L_1 \subset L_2 \subset L_3$, each of which attempts a full reconstruction of ψ. Black dashed boxes mark the latents included at a given level and orange dashed boxes mark latents added at higher levels. Here, batch top-k selects one additional active latent at each level (yellow), allowing lower (higher) levels to capture coarser (finer) features. (**d**) A W-SED instead treats a sequence of S timebins of N recorded units, $\mathbf{Y} \in \mathbb{R}^{S \times N}$, as one training example and reconstructs all S bins of a target $\mathbf{\Psi} \in \mathbb{R}^{S \times N}$ as $\hat{\mathbf{\Psi}}$ (the batch dimension is omitted throughout). A FW-SED encoder linearly projects the vectorized window to the D latents $\mathbf{z}$; alternatively, a TW-SED encoder treats each timebin's population state as a token, integrates temporal context via self-attention, and projects the temporally contextualized final token to the latents. The latents pass through ReLU and batch top-k (optionally with Matryoshka levels), and each active latent contributes a timebin-by-unit activity pattern to the full reconstruction $\hat{\mathbf{\Psi}}$. The loss is a weighted average of per-timebin losses, $\sum_s w_s \ell_s / \sum_s w_s$; reconstruction quality is also reported per timebin. An optional shift-equivariant mode (each latent is a motif that may occur at any timebin) is described in 6.3.2.

3 METHODS: THE NLDISCO PIPELINE

The NLDisco pipeline transforms high-dimensional neural data into a set of interpretable latents in four stages (Figure 1), the first three of which can be fully automated.

Stage 1: Data preprocessing. The first stage prepares neural data into a timebin $\times$ unit matrix for model training. NLDisco has utilities to work directly with outputs from common spike sorters such as Kilosort (Pachitariu et al., 2016), and can aggregate spikes into timebins for each recorded unit and normalize the resulting activity. Users may also provide their own preprocessed timebin $\times$ unit data, for example two-photon calcium-imaging regions of interest extracted by Suite2p (Pachitariu et al., 2017).

Stage 2: Model training. In the second stage, an SED with configurable dictionary width D is trained to reconstruct target neural activity. Training is unsupervised and uses only neural activity; behavioral and environmental labels enter only post hoc, for latent interpretation and evaluation. The goal is for the model's hidden layer neurons (latents) to capture disentangled, interpretable representations encoded by the population activity (Figure 2a). Sparsity encourages a monosemantic dictionary in which each latent corresponds to a single feature: for example, a latent learned from visual cortical activity may represent a property of a visual stimulus, while one learned from motor cortical activity may represent an aspect of movement. Each active latent contributes to the target reconstruction through explicit decoder weights, and its encoder weights link its activation to constituent source neural activity.

The SED configuration can be selected according to the scientific question and temporal structure of the data (Figure 2b,c,d). A vanilla SED treats each timebin independently. A W-SED treats a sequence $\mathbf{Y} \in \mathbb{R}^{S \times N}$ of $S > 1$ timebins and N recorded units as one example, allowing each latent to represent a timebin-by-unit pattern. W-SEDs have two encoder variants: the Flat-window encoder (FW-SED) linearly projects the vectorized window, while the Transformer-window encoder (TW-SED) integrates timebin tokens using self-attention. Both reconstruct the complete window through a sparse latent code; the TW-SED additionally supports an optional shift-equivariant mode to capture motifs that may occur at different positions within a window across examples. Nested Matryoshka levels (Bussmann et al., 2025) can be used with either the vanilla or W-SED: each nested level attempts to reconstruct the full target using a progressively larger subset of the shared latent dictionary, encouraging multi-scale representations and mitigating feature absorption, in which a specific feature partially subsumes a more general one (Chanin et al., 2025). For example, separate Matryoshka SED latents can recover both an encompassing place field and a narrower field nested within it, without the narrower-field latent hiding the activity associated with the broader field (see 4.1).

Independently of the architectural choice, an SED can be trained as an autoencoder (Huben et al., 2024) that reconstructs its input population, or as a transcoder (Dunefsky et al., 2024) that reconstructs one neural population based on another, potentially across brain regions, recording sessions, or subjects. These configurations support analysis of both the intrinsic representational structure of a population and how representations are transformed across regions, sessions, or subjects. All configurations use batch top-k sparsification, a batch-level variant of TopK (Gao et al., 2025) that preserves the Bk largest latent activations across a batch of size B (Bussmann et al., 2024). Unlike indirect approaches such as an L1 penalty, batch top-k directly controls total sparsity while allowing individual examples to recruit different numbers of latents.

NLDisco also supports automated hyperparameter sweeps of architectural and training optimization configurations across CPUs or GPUs. The complete training procedure and hyperparameter guidance are described in 6.3.2.

Stage 3: Model evaluation. Following training, model quality is evaluated using a suite of quantitative metrics. NLDisco adapts domain-general metrics from SAEBench (Karvonen et al., 2025) and adds metrics tailored to neural data. Although these metrics do not directly establish latent interpretability, they test whether the model has learned a sparse, accurate, and robust representation suitable for subsequent inspection.

Reconstruction fidelity is measured using variance explained ($\mathbf{R^2}$) and reconstruction-to-target cosine similarity across both units and examples (Figure S2, bottom). For W-SEDs, reconstruction quality is

additionally reported for each timebin in the window. Evaluation distinguishes dictionary width D from activation sparsity k. Increasing either can improve reconstruction, but reconstruction alone does not establish interpretability. NLDisco therefore also examines latent **L0** – the number of active latents per example – and latent activation density across the dataset (Figure S2, top), which together can reveal failure modes such as representational collapse into many persistently active latents, or feature death in which latents never activate. Optional diagnostics include latent ablation importance and spectral comparisons between reconstructed and target activity (6.3.3). Raw model parameters, gradients, and activations remain available for bespoke analyses.

Stage 4: Latents evaluation. For exploratory discovery, users can survey all active latents or a random subset, inspecting representative activating and inactive episodes alongside behavior. Recurring patterns suggest feature descriptions; counterexamples refine them. Source activity traceback identifies contributing units and timebins, guiding hypotheses for testing on separate recordings or in targeted experiments. Evaluation methods and metrics are detailed in 6.3.4 and demonstrated in Results.

4 RESULTS

We evaluate NLDisco on three open-source datasets:

1. Simulated single-unit spike data of rat hippocampus during a navigation task (4.1).
2. Unsorted spike data from macaque motor cortex during an active reaching task (4.2).
3. Single-unit spike data from mouse retrosplenial cortex and superior colliculus during an ethological foraging assay (4.3).

Together, these datasets span multiple species, brain regions, and experimental paradigms. We use the first to motivate the various NLDisco model architectures and test recovery of known ground-truth features, the second to assess performance on real neural data and benchmark against established neural LVMs, and the third to assess exploratory feature discovery and hypothesis generation in a naturalistic setting. Evaluation plots and metrics for NLDisco models in 4.1, 4.2, and 4.3 are shown in 6.4. Throughout the results, N denotes the number of input units, S the number of timebins per input example, D the total number of latents, B the batch size, and k the training batch top-k sparsity parameter (Bk latents are selected across a batch, not necessarily k per example). For Matryoshka SEDs, L denotes the number of nested levels, D_ℓ the number of available latents at level ℓ (with $D_L = D$), and k_ℓ its sparsity parameter. A single-level SED has $L = 1$, $D_1 = D$, and $k_1 = k$.

4.1 SIMULATED RAT HIPPOCAMPAL SPIKE DATA IN A NAVIGATION TASK

Using signed Ornstein–Uhlenbeck movement motivated by Theodoni et al. (2018), we simulate a rat's position and velocity over 600 seconds on a linear 1-m track (Figure S3a), together with spike counts from $N = 100$ neurons: four place cells (O'Keefe & Dostrovsky, 1971) with varying peak firing rates and spatial extents, and 96 position-independent "noise" neurons (Figure 3a,b; 6.4.1 for details). The first two place fields are asymmetric and centered at 0.2 and 0.4 m; the other two are symmetric and centered at 0.7 m with differing widths (Figure S4). The simulated neural data are processed into 50-ms binned spike counts per neuron.

We first verify that vanilla SED latents can recover known place cell activity from place cells 1 and 2 despite the presence of the noise neurons. Indeed, across five independently trained models (each with $S = 1$, $D = 128$, $B = 1024$, $k = 4$), we find latents with activations nearly identical to the place cells (Figure 3c).

We then ask whether a Matryoshka SED can additionally recover the activity associated with both the encompassing and nested place cells 3 and 4. Again, across five independently trained models (each with $S = 1$, $(D_1, k_1) = (64, 4)$, $(D_2, k_2) = (128, 4)$, and $B = 1024$), we find latents that recover both fields, with the broader and narrower place fields recovered in latents in the first and second Matryoshka levels respectively, as expected (Figure 3d).

Last, we evaluate whether a W-SED can find features that evolve over time in this setting, e.g., a feature that represents continuous movement. We train five FW-SEDs and five TW-SEDs over 1-s windows (each with $S = 20$, $D = 128$, $k = 4$, $B = 256$) and find latents that are active during 1 s

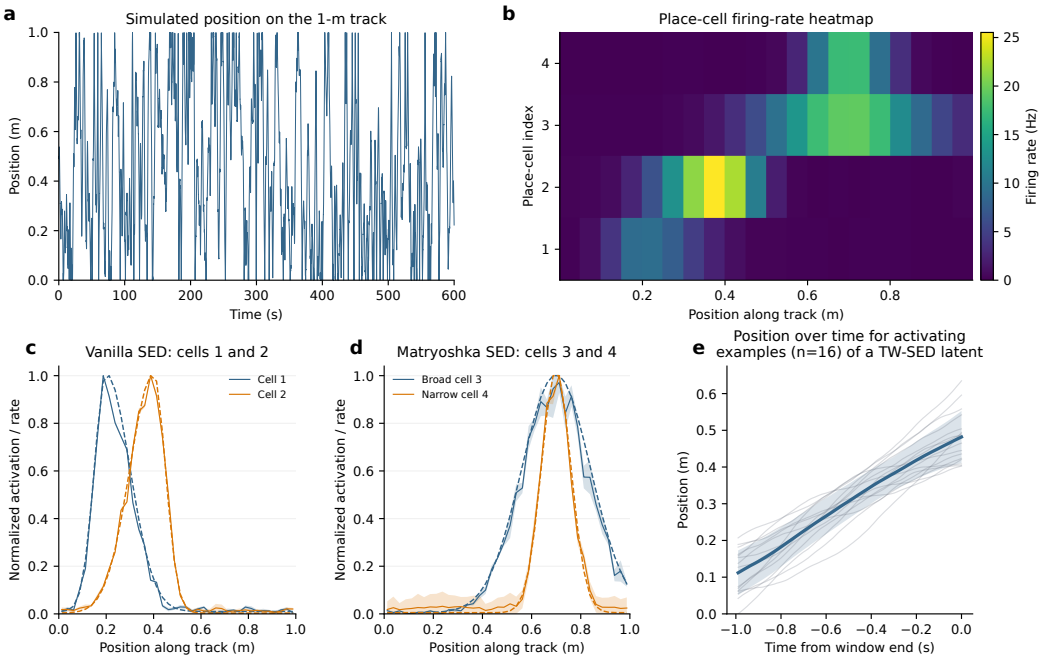


Figure 3: Spatial and temporal features recovered from a controlled simulation.
(a) Simulated position. **(b)** Empirical firing rates of the four simulated place cells in 20 spatial bins. **(c)** For each of five independent training runs with $S = 1$, $D = 128$, $k = 4$, $B = 1024$, SED latents identify the activity of place cells 1 and 2: solid curves and shaded bounds show the mean activation and ± 1 sample S.D. of the best matching latents; dashed curves show the exact place field. **(d)** For each of five independent Matryoshka SED fits with $S = 1$, $(D_1, k_1) = (64, 4)$, $(D_2, k_2) = (128, 4)$, and $B = 1024$, latents identify both broad and narrow place cells 3 and 4. Solid and dashed curves and shaded bounds same as in c. **(e)** Activating examples for an $S = 20$, $D = 128$, $k = 4$, $B = 256$ TW-SED latent that captures forward motion through place fields 1 and 2. Thin gray curves are individual trajectories, with mean (bold curve) and ± 1 sample S.D. (shaded bounds).

of movement in particular directions. In Figure 3e we show for one particular TW-SED latent the position over time for activating examples, which clearly illustrates forward motion through place fields 1 and 2. We find that both FW-SEDs and TW-SEDs can find such features (Figure S5), with directional features recovered in more TW-SED runs. TW-SED latents exhibit preferences for both forward and backward movement (Figure S6 and Figure S7). We also examine the neural activity associated with spatial and temporal latents, finding positive attribution concentrated in the expected place cells and near-zero attribution for noise neurons (Figure S8 and Figure S9), and quantify their selectivity for the expected spatial locations and movement directions (6.4.1).

As a baseline comparison, we apply PCA to an $11{,}981 \times 2{,}000$ matrix of overlapping 1-s neural activity windows, with each row concatenating twenty 50-ms bins from all 100 neurons. We retain 128 principal components to match D from the SED models above. We find that individual principal components exhibit mixed spatial tuning and do not cleanly separate any of the four place fields (Figure S10). Interestingly, a single component can strongly distinguish forward from backward crossings through the first two fields, although its most extreme scores across the full recording are not selective for these traversals (Figure S11). Overall, these results demonstrate that NLDisco can cleanly recover known ground-truth features that a PCA baseline misses, and highlight the utility of each SED model that NLDisco provides.

4.2 Macaque motor cortex spike data in an active reaching task

To test NLDisco on real neural data with unknown ground-truth representations and compare it with established neural LVMs, we analyze neural and behavioral data from a popular macaque motor control task (Churchland et al., 2012), prepared by Brainsets (Azabou et al., 2023). Briefly, the task requires the macaque to wait for a go cue before reaching to a visual target with a hand-controlled cursor while avoiding virtual barriers in different configurations (Figure S3b).

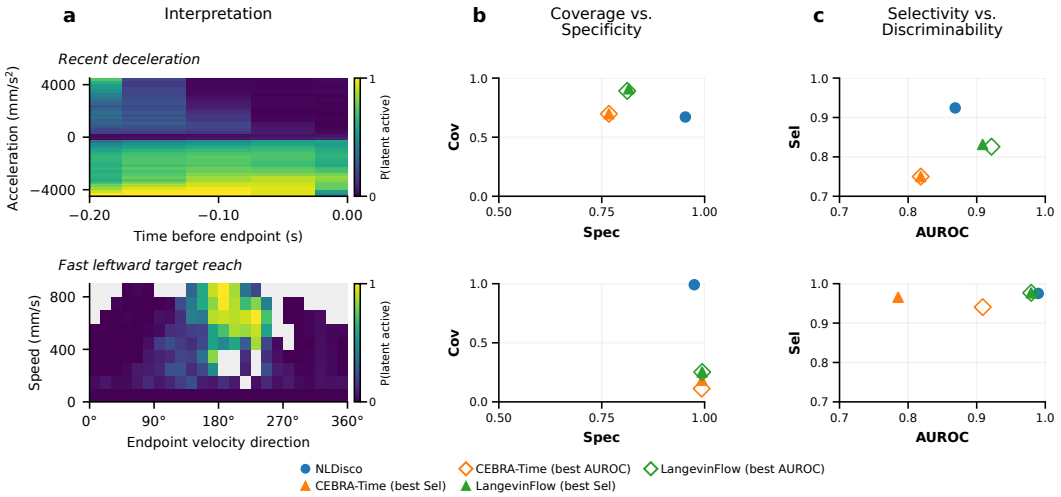


Figure 4: Interpreting and comparing features in 192-dimensional representations. (**a**) NLDisco latent activation probability conditioned on speed slope over the preceding 250 ms (top, latent 83) or endpoint velocity direction and speed (bottom, latent 20), for the second level ($D = 192$, $k = 16$) of the two-level Matryoshka TW-SED trained on examples of $S = 10$ with 50 ms bins. Both heatmaps pool data from all valid experiment trials; they do *not* restrict observations to the scored conditions (gray cells have fewer than 10 distinct trials). (**b**) **Cov** versus **Spec** and (**c**) **Sel** versus **AUROC** for the same NLDisco, CEBRA-Time, and LangevinFlow latents that best match these features. Source activity traceback appears in Figure S14.

We analyze one experiment comprising 1,906 retained trials across 27 configurations with nine total target locations. Unsorted spikes were recorded from dorsal premotor and primary motor cortex using $N = 191$ Utah array channels, and neural and behavioral data are binned at 50 ms. To try to find both instantaneous and temporally-evolving features, we use $S = 10$ to capture 500 ms of activity per example, and train a two-level Matryoshka TW-SED with $(D_1, k_1) = (96, 8)$, $(D_2, k_2) = (192, 16)$, and $B = 1024$. Inspection of latent-conditioned kinematics and task conditions guide feature visualization: behavioral descriptions are developed from trained latents, then fixed for comparison.

Among many features found (see 6.4.2), here we highlight two that together span kinematics and task conditions: 1) *recent deceleration*: a speed slope ≤ -75 mm/s^2 over the preceding 250 ms; 2) *fast leftward target reach*: a movement within $45°$ of leftward at endpoint speed ≥ 500 mm/s on trials with target positioned at $(-144, -52)$ mm of screen center. We visualize them by plotting the probability of latent activation conditioned on behavioral and environmental variables (Figure 4a), and quantify these associations using standard metrics for coverage (**Cov**), specificity (**Spec**), selectivity (**Sel**), and discriminability (**AUROC**; Figure 4b,c). **Cov** measures how often a latent activates within the condition, **Spec** how often it remains inactive outside it, **Sel** summarizes this preference, and **AUROC** measures discrimination across activation thresholds (definitions in 6.3.4). All metrics range from 0 to 1, with a higher value indicating a better score.

We then compare results to other popular neural LVMs. For fair comparisons focused on the main goal of unsupervised interpretable latent discovery, we consider published methods (Table S1) that: 1) prioritize an interpretable latent space; 2) do not require multimodal or trial-structured data. Among these, we show results from CEBRA (which contrastively learns a continuous neural embedding space (Schneider et al., 2023)) and LangevinFlow (which models neural activity through structured, stochastic dynamics (Song et al., 2025)) as they are current top-performing contrastive and sequential variational methods, respectively, on Neural Latents Benchmark (Pei et al., 2021).

We set the latent space for each of these baseline methods to 192 to match the SED D, and perform 12-run sweeps for each method, varying temperature and positive-pair lag for CEBRA-Time, and learning rate and regularization-ramp duration for LangevinFlow (see 6.4.2 for full sweep configurations). For each baseline method, we select the model with the highest mean **AUROC** across features using their best matching latents. And to give the baselines a better opportunity to demonstrate both selectivity and discrimination, we report up to two latents per feature – the highest-**Sel** and

	Recent deceleration						*Fast leftward target reach*					
Method	**Cov**	**Spec**	**Sel**	**AUROC**	Single Decode	All Decode	**Cov**	**Spec**	**Sel**	**AUROC**	Single Decode	All Decode
NLDisco	0.671	**0.953**	**0.924**	0.868	0.768	**0.918**	**0.991**	0.975	**0.975**	**0.989**	**0.981**	**0.995**
CEBRA-Time (best **Sel**)	0.698	0.767	0.749	0.818	0.729	0.899	0.183	0.993	0.965	0.785	0.718	0.989
CEBRA-Time (best **AUROC**)	0.698	0.767	0.749	0.818	0.729	0.899	0.112	0.993	0.940	0.908	0.827	0.989
LangevinFlow (best **Sel**)	**0.909**	0.816	0.831	0.908	**0.862**	0.902	0.252	**0.994**	**0.975**	0.979	0.915	0.958
LangevinFlow (best **AUROC**)	0.892	0.812	0.825	**0.921**	0.850	0.902	0.252	**0.994**	**0.975**	0.979	0.915	0.958

Table 1: Feature recovery and decoding. **Cov**, **Spec**, **Sel**, **AUROC** and feature decoding scores for features and latents shown in Figure 4. "Single decode" uses individual latents, while "All decode" uses the entire latent space. **Decode** reports test balanced accuracy from logistic regression with validation-selected decision thresholds and a 0.8/0.1/0.1 whole-trial split (6.3.4). Column maxima are bolded, including ties.

highest-**AUROC** dimensions – alongside the illustrative NLDisco latent. As in 4.1, this is a targeted search for known features, not a comparison of what each method would discover without specified behavioral descriptions.

As expected, both baselines recover these features. For *recent deceleration*, NLDisco has the highest **Spec** and **Sel**, but lower **Cov** than the baselines, while LangevinFlow has the highest **AUROC**. For *fast leftward target reach*, NLDisco covers nearly all condition-positive examples with much higher **Cov** than the baselines. All shown latents are highly selective for this feature, with NLDisco and LangevinFlow latents also highly discriminative. These comparisons highlight why the metrics should be considered together: a highly selective latent need not detect most condition-positive examples or discriminate well across activation thresholds, as ROC curves further illustrate (Figure S13).

Last, we assess feature decoding from each displayed latent and the full latent space using identical ℓ_2-regularized logistic regression. We report test balanced accuracy with validation-selected regularization and decision thresholds, using a 0.8/0.1/0.1 whole-trial split (Table 1; shared protocol in 6.3.4). We see all methods perform well on feature decoding from the full latent space, while single-latent decoding favors NLDisco and LangevinFlow. Overall, we find that CEBRA and LangevinFlow contain recoverable behavioral information, but their objectives do not explicitly favor individual feature detectors. NLDisco makes these a natural starting point for discovery: inspect activating examples, formulate behavioral interpretations, and quantify selectivity and discrimination. Although our comparison does not directly measure end-to-end discovery effort, it shows that NLDisco supports this direct, latent-centered workflow while retaining strong decoding performance. We compare training times for all three methods in Figure S15.

4.3 Mouse spike data in an ethological foraging assay

Unlike the linear track and trial-structured reaching tasks in 4.1 & 4.2, naturalistic behavior unfolds continuously over extended periods without predefined boundaries. We ask whether NLDisco can support open-ended feature discovery in this setting using Aeon, a platform for continuously recording rodent neural and behavioral activity in naturalistic habitats (Campagner et al., 2025).

We analyze an 18-hour interval from a week-long Aeon experiment in which one mouse in an open environment spontaneously forages by spinning a wheel on a foraging patch (Figure S3c). The neural data comes from a Neuropixels recording, processed into 20-ms binned spike counts from $N = 90$ uncurated Kilosort units across retrosplenial cortex (RSC) and superior colliculus (SC). The behavioral data include the animal's centroid position and segmented image area estimated continuously from an overhead camera, together with foraging-wheel rotation and food-delivery events, with food dispensed every 75 cm of wheel movement (approximately three revolutions).

As before, we train a two-level Matryoshka TW-SED ($S = 20$, $(D_1, k_1) = (96, 6)$, $(D_2, k_2) = (192, 12)$, $B = 1024$) to reconstruct the neural data. Among many features found (see 6.4.3), here we show one descriptive example, latent 52, surprisingly preferentially active for *wheel activity*: wheel speeds > 0.75 cm/s, **Sel** $= 0.839$, **AUROC** $= 0.924$ (Figure 5). The preference persists when controlling for the animal's position, and notably, this association re-emerges across foraging bouts separated by hours: the same latent tracks wheel activity throughout the 18-hour recording (Figure S18). Although RSC and SC are implicated in foraging, that did not predict the surprising wheel-associated feature revealed here. These associations suggest follow-up experiments, for

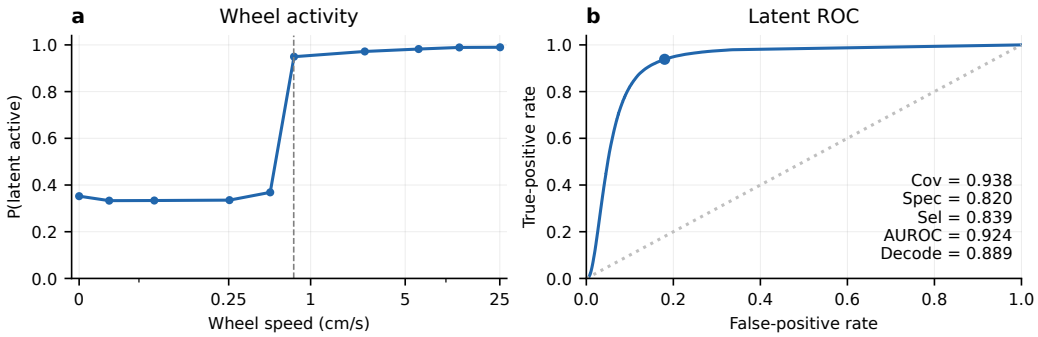


Figure 5: A *wheel activity* latent in a continuous foraging recording. (**a**) Latent activation probability by wheel speed; vertical dashed line at 0.75 cm/s. (**b**) ROC based on latent activations; dot: FPR 0.180, TPR 0.938 at the displayed activation threshold. **Decode**: test balanced accuracy of a single-latent logistic decoder with a validation-selected decision threshold (6.3.4). Source activity traceback appears in Figure S19.

example, varying wheel engagement while measuring posture and controlling location could help distinguish wheel-related activity from accompanying movements.

5 Discussion

NLDisco provides a practical pipeline for discovering interpretable features in neural population activity. Our results show that NLDisco can robustly discover discrete and continuous, environmental and behavioral feature representations across varying timescales and levels of abstraction. Importantly, same feature representations can be found across model training runs and are stable within a subject over time. This work highlights individual sparse latents as useful units of representational analysis, enabling a direct mapping from representation to source neural activity. The illustrated features are candidates for further analysis and experiments, highlighting NLDisco's role in hypothesis generation.

Contributions.

- We provide an open-source software package, NLDisco, that implements an end-to-end pipeline for discovery of interpretable neural representations.
- We introduce novel sparse encoder-decoder architectures that support discovery of temporally-varying and nested feature representations.
- We evaluate on both synthetic and real large-scale neural datasets, demonstrating utility across different species, brain regions, experimental conditions, and spatiotemporal scales.
- We benchmark NLDisco against popular state-of-the-art neural LVMs, CEBRA and LangevinFlow, demonstrating interpretable feature discovery while maintaining competitive neural decoding.

Limitations.

- Latent interpretability remains manual and partly subjective.
- Learned latents reveal correlational structure, not causal mechanisms. Causal claims require additional analyses with explicit assumptions.
- Our targeted comparisons assess recovery of selected features rather than the breadth or effort of discovery across methods.

Future work. Extending empirical validation to other neural recording modalities, such as neuroimaging, would highlight the generality of NLDisco. Evaluating CLTs and SCCs, which are implemented in the codebase but not shown here, would support cross-region prediction and "brain diffing": distinguishing shared and divergent features across regions, sessions, or subjects. Finally, real-time identification of neural representational features could enable closed-loop experiments in which latent activation triggers precisely timed stimulation or task changes: such experiments would connect feature discovery to causal implications. We are excited to share NLDisco with the community and look forward to future applications using this work.

AI USE STATEMENT

Generative AI tools were used to assist with writing and executing code, performing machine learning experiments, data analysis, schematic figure creation, and manuscript formatting and editing. Author review of this work was thorough. The authors take responsibility for reviewing the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

NLDisco is intended to support exploratory analysis and hypothesis generation from neural recordings. This work analyzes simulated neural activity and recordings from macaques and mice; no human participant data are analyzed. Dataset sources and experimental contexts are described in the corresponding results and appendix sections. Animal recording procedures were approved by the relevant institutional ethics committees at Stanford University and Sainsbury Wellcome Centre, and conducted in accordance with applicable animal-welfare regulations set by the Public Health Service (PHS) Policy on Humane Care and Use of Laboratory Animals and the UK Home Office.

The principal risk in interpreting our results is mistaking a learned statistical association for a uniquely identified neural representation or a causal mechanism. We therefore treat latent interpretations as candidate explanations that require independent validation, and report limitations arising from correlated behavioral variables and exploratory feature selection. NLDisco has not been validated for clinical decision-making. Future applications to human neural recordings would require appropriate ethical review, informed consent, and safeguards for the privacy of neural data and any sensitive information inferred from them.

REFERENCES

Misha B Ahrens, Michael B Orger, Drew N Robson, Jennifer M Li, and Philipp J Keller. Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature Methods*, 10(5):413–420, 2013. doi: 10.1038/nmeth.2434. URL https://doi.org/10.1038/nmeth.2434.

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/methods.html.

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. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44937–44956, 2023. doi: 10.52202/075280-1948. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8ca113d122584f12a6727341aaf58887-Abstract-Conference.html.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bart Bussmann, Patrick Leask, and Neel Nanda. BatchTopK sparse autoencoders. *arXiv*, 2024. doi: 10.48550/arXiv.2412.06410. URL https://arxiv.org/abs/2412.06410.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with Matryoshka sparse autoencoders. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 6077–6101. PMLR, 2025. URL https://proceedings.mlr.press/v267/bussmann25a.html.

Denise J Cai, Daniel Aharoni, Tristan Shuman, Justin Shobe, Jeremy Biane, Weilin Song, Brandon Wei, Michael Veshkini, Mimi La-Vu, Jerry Lou, Sergio E Flores, Isaac Kim, Yoshitake Sano, Miou Zhou, Karsten Baumgaertel, Ayal Lavi, Masakazu Kamata, Mark Tuszynski, Mark Mayford, Peyman Golshani, and Alcino J Silva. A shared neural ensemble links distinct contextual memories encoded close in time. *Nature*, 534(7605):115–118, 2016. doi: 10.1038/nature17955. URL https://pubmed.ncbi.nlm.nih.gov/27251287/.

D. Campagner, J. Bhagat, G. Lopes, L. Calcaterra, A. G. Pouget, A. Almeida, T. T. Nguyen, C. H. Lo, T. Ryan, B. Cruz, F. J. Carvalho, Z. Li, A. Erskine, J. Rapela, O. Folsz, M. Marin, J. Ahn, S. Nierwetberg, S. C. Lenzi, J. D. S. Reggiani, and SGEN group – SWC GCNU Experimental Neuroethology Group. Aeon: An open-source platform to study the neural basis of ethological behaviours over naturalistic timescales. *bioRxiv*, 2025. doi: 10.1101/2025.07.31.664513. URL https://www.biorxiv.org/content/early/2025/08/01/2025.07.31.664513.

David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, Satvik Golechha, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. In *Advances in Neural Information Processing Systems*, volume 38, pp. 91489–91526, 2025. doi: 10.52202/085713-2755. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/764ff7477b8e24dbe01531f6791e8bdf-Abstract-Conference.html.

Mark M. Churchland, John P. Cunningham, Matthew T. Kaufman, Justin D. Foster, Paul Nuyujukian, Stephen I. Ryu, and Krishna V. Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012. doi: 10.1038/nature11129. URL https://doi.org/10.1038/nature11129.

Pierre Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3): 287–314, 1994. doi: 10.1016/0165-1684(94)90029-9. URL https://doi.org/10.1016/0165-1684(94)90029-9.

John P Cunningham and Byron M Yu. Dimensionality reduction for large-scale neural recordings. *Nature Neuroscience*, 17(11):1500–1509, 2014. doi: 10.1038/nn.3776. URL https://doi.org/10.1038/nn.3776.

Kenji Doya, Shin Ishii, Alexandre Pouget, and Rajesh P. N. Rao (eds.). *Bayesian Brain: Probabilistic Approaches to Neural Coding*. MIT Press, Cambridge, MA, 2007. URL https://direct.mit.edu/books/edited-volume/2884/Bayesian-BrainProbabilistic-Approaches-to-Neural.

Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. Transcoders find interpretable LLM feature circuits. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24375–24410. Curran Associates, Inc., 2024. doi: 10.52202/079017-0768. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/2b8f4db0464cc5b6e9d5e6bea4b9f308-Abstract-Conference.html.

Karl Friston. The free-energy principle: A unified brain theory? *Nature Reviews Neuroscience*, 11 (2):127–138, 2010. doi: 10.1038/nrn2787. URL https://doi.org/10.1038/nrn2787.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/42ef3308c230942d223c411adf182c88-Abstract-Conference.html.

Yuanjun Gao, Evan Archer, Liam Paninski, and John P Cunningham. Linear dynamical neural population models through nonlinear embeddings. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/76dc611d6ebaafc66cc0879c71b5db5c-Abstract.html.

Tom M. George, Pierre Glaser, Kimberly Stachenfeld, Caswell Barry, and Claudia Clopath. SIMPL: Scalable and hassle-free optimisation of neural representations from behaviour. In *The Thirteenth International Conference on Learning Representations (ICLR)*, pp. 27662–27686,

2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/44d20d542f3f4e3b7097e5e3f78f99f1-Abstract-Conference.html.

Rabia Gondur, Usama Bin Sikandar, Evan Schaffer, Mikio Christian Aoi, and Stephen L. Keeley. Multi-modal Gaussian process variational autoencoders for neural and behavioral data. In *The Twelfth International Conference on Learning Representations (ICLR)*, pp. 14257–14281, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3d226fb8fbd6ee6ec70d0427f1319707-Abstract-Conference.html.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL https://www.deeplearningbook.org/. Chapter 10: Sequence Modeling—Recurrent and Recursive Nets.

Daniel Hernandez, Antonio K. Moretti, Shreya Saxena, Ziqiang Wei, John Cunningham, and Liam Paninski. Nonlinear evolution via spatially-dependent linear dynamics for electrophysiology and calcium data. *Neurons, Behavior, Data Analysis, and Theory*, 3(3), 2020. URL https://nbdt.scholasticahq.com/article/13476.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933. doi: 10.1037/h0071325. URL https://doi.org/10.1037/h0071325.

Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5(Nov):1457–1469, 2004. URL https://jmlr.org/papers/v5/hoyer04a.html.

Robert Huben, Hoagy Cunningham, Logan Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations*, pp. 7827–7845, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/1fa1ab11f4bd5f94b2ec20e794dbfa3b-Abstract-Conference.html.

Mark D Humphries. Strong and weak principles of neural dimension reduction. *Neurons, Behavior, Data Analysis, and Theory*, 5(2):1–28, 2021. ISSN 2690-2664. doi: 10.51628/001c.24619. URL https://doi.org/10.51628/001c.24619.

Cole Hurwitz, Akash Srivastava, Kai Xu, Justin Jude, Matthew G Perich, Lee E Miller, and Matthias H Hennig. Targeted neural dynamical modeling. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29379–29392, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/f5cfbc876972bd0d031c8abc37344c28-Abstract.html.

Kristopher T Jensen, Ta-Chu Kao, Marco Tripodi, and Guillaume Hennequin. Manifold GPLVMs for discovering non-Euclidean latent structure in neural data. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22580–22592, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/fedc604da8b0f9af74b6cfc0fab2163c-Abstract.html.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Isaac Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum Stuart Mcdougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29223–29264. PMLR, 2025. URL https://proceedings.mlr.press/v267/karvonen25a.html.

Mohammad Reza Keshtkaran, Andrew R. Sedler, Raeed H. Chowdhury, Raghav Tandon, Diya Basrai, Sarah L. Nguyen, Hansem Sohn, Mehrdad Jazayeri, Lee E. Miller, and Chethan Pandarinath. A large-scale neural network training framework for generalized estimation of single-trial population dynamics. *Nature Methods*, 19(12):1572–1577, 2022. doi: 10.1038/s41592-022-01675-0. URL https://doi.org/10.1038/s41592-022-01675-0.

Timothy D. Kim, Thomas Z. Luo, Jonathan W. Pillow, and Carlos D. Brody. Inferring latent dynamics underlying neural population activity via neural differential equations. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5551–5561. PMLR, 2021. URL https://proceedings.mlr.press/v139/kim21h.html.

Nina Kudryashova, Matthew G Perich, Lee E Miller, and Matthias H Hennig. Ctrl-TNDM: Decoding feedback-driven movement corrections from motor cortex neurons. In *Computational and Systems Neuroscience (Cosyne) Abstracts*, pp. 144–145, 2023. URL https://www.cosyne.org/s/Cosyne2023_program_book.pdf. Abstract 2-053.

Trung Le and Eli Shlizerman. STNDT: Modeling neural population activity with spatiotemporal transformers. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17926–17939. Curran Associates, Inc., 2022. doi: 10.52202/068431-1303. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/72163d1c3c1726f1c29157d06e9e93c1-Paper-Conference.pdf.

Alexander LeNail. NN-SVG: Publication-ready neural network architecture schematics. *Journal of Open Source Software*, 4(33):747, 2019. doi: 10.21105/joss.00747. URL https://doi.org/10.21105/joss.00747.

Jack Lindsey, Adly Templeton, Jonathan Marcus, Thomas Conerly, Joshua Batson, and Christopher Olah. Sparse crosscoders for cross-layer features and model diffing. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/crosscoders/index.html.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Ryan J. Low, Sam Lewallen, Dmitriy Aronov, Rhino Nevers, and David W. Tank. Probing variability in a cognitive map using manifold inference from neural dynamics. *bioRxiv*, pp. 418939, 2018. doi: 10.1101/418939. URL https://doi.org/10.1101/418939.

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, 2011. URL https://proceedings.neurips.cc/paper/2011/hash/7143d7fbadfa4693b9eec507d9d37443-Abstract.html.

Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. doi: 10.48550/arXiv.1802.03426. URL https://arxiv.org/abs/1802.03426.

Neural Latents Benchmark. MC_Maze dataset, Neural Latents Benchmark, n.d. URL https://neurallatents.github.io/datasets#mcmaze. Accessed 2026-09-25.

J. O'Keefe and J. Dostrovsky. The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain Research*, 34(1):171–175, 1971. doi: 10.1016/0006-8993(71)90358-1. URL https://doi.org/10.1016/0006-8993(71)90358-1.

Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23):3311–3325, 1997. doi: 10.1016/S0042-6989(97)00169-7. URL https://doi.org/10.1016/S0042-6989(97)00169-7.

Dimitre G Ouzounov, Tianyu Wang, Mengran Wang, Danielle D Feng, Nicholas G Horton, Jean C Cruz-Hernández, Yu-Ting Cheng, Jacob Reimer, Andreas S Tolias, Nozomi Nishimura, and Chris Xu. In vivo three-photon imaging of activity of GCaMP6-labeled neurons deep in intact mouse brain. *Nature Methods*, 14(4):388–390, 2017. doi: 10.1038/nmeth.4183. URL https://doi.org/10.1038/nmeth.4183.

Marius Pachitariu, Nicholas A. Steinmetz, Shabnam N. Kadir, Matteo Carandini, and Kenneth D. Harris. Fast and accurate spike sorting of high-channel count probes with KiloSort. In *Advances in Neural Information Processing Systems*, volume 29, pp. 4448–4456, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/1145a30ff80745b56fb0cecf65305017-Abstract.html.

Marius Pachitariu, Carsen Stringer, Mario Dipoppa, Sylvia Schröder, L. Federico Rossi, Henry Dalgleish, Matteo Carandini, and Kenneth D. Harris. Suite2p: Beyond 10,000 neurons with standard two-photon microscopy. *bioRxiv*, 2017. doi: 10.1101/061507. URL https://www.biorxiv.org/content/10.1101/061507v2.

Matthijs Pals, A. Erdem Sağtekin, Felix Pei, Manuel Gloeckler, and Jakob H. Macke. Inferring stochastic low-rank recurrent neural networks from neural data. In *Advances in Neural Information Processing Systems*, volume 37, pp. 18225–18264, 2024. doi: 10.52202/079017-0579. URL https://papers.nips.cc/paper_files/paper/2024/hash/209423f076b6479ab3a4f45886e30306-Abstract-Conference.html.

Chethan Pandarinath, Daniel J. O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D. Stavisky, Jonathan C. Kao, Eric M. Trautmann, Matthew T. Kaufman, Stephen I. Ryu, Leigh R. Hochberg, Jaimie M. Henderson, Krishna V. Shenoy, L. F. Abbott, and David 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. URL https://doi.org/10.1038/s41592-018-0109-9.

Felix Pei, Joel Ye, David Zoltowski, Anqi Wu, Raeed H. Chowdhury, Hansem Sohn, Joseph E. O'Doherty, Krishna V. Shenoy, Matthew T. Kaufman, Mark Churchland, Mehrdad Jazayeri, Lee E. Miller, Jonathan Pillow, Il Memming Park, Eva L. Dyer, and Chethan Pandarinath. Neural Latents Benchmark '21: Evaluating latent variable models of neural population activity. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/979d472a84804b9f647bc185a877a8b5-Abstract-round2.html.

Arthur Pellegrino, Heike Stein, and N. Alex Cayco-Gajic. Dimensionality reduction beyond neural subspaces with slice tensor component analysis. *Nature Neuroscience*, 27(6):1199–1210, 2024. doi: 10.1038/s41593-024-01626-2. URL https://doi.org/10.1038/s41593-024-01626-2.

Sean M. Perkins, Elom A. Amematsro, John Cunningham, Qi Wang, and Mark M. Churchland. An emerging view of neural geometry in motor cortex supports high-performance decoding. *eLife*, 12:RP89421, 2025. doi: 10.7554/eLife.89421.3. URL https://elifesciences.org/articles/89421.

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.

Bogdan C. Raducanu, Refet F. Yazicioglu, Carolina M. Lopez, Marco Ballini, Jan Putzeys, Shiwei Wang, Alexandru Andrei, Veronique Rochus, Marleen Welkenhuysen, Nick van Helleputte, Silke Musa, Robert Puers, Fabian Kloosterman, Chris Van Hoof, Richárd Fiáth, István Ulbert, and Srinjoy Mitra. Time multiplexed active neural probe with 1356 parallel recording sites. *Sensors*, 17(10): 2388, 2017. doi: 10.3390/s17102388. URL https://doi.org/10.3390/s17102388.

Rajesh P. N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999. doi: 10.1038/4580. URL https://doi.org/10.1038/4580.

Omid G Sani, Bijan Pesaran, and Maryam M Shanechi. Dissociative and prioritized modeling of behaviorally relevant neural dynamics using recurrent neural networks. *Nature Neuroscience*, 27(10):2033–2045, 2024. doi: 10.1038/s41593-024-01731-2. URL https://doi.org/10.1038/s41593-024-01731-2.

Marine Schimel, Ta-Chu Kao, Kristopher T. Jensen, and Guillaume Hennequin. iLQR-VAE: Control-based learning of input-driven dynamics with applications to neural data. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=wRODLDHaAiW.

Valentin Schmutz, Ali Haydaroğlu, Shuqi Wang, Yixiao Feng, Matteo Carandini, and Kenneth D. Harris. High-dimensional neuronal activity from low-dimensional latent dynamics: A solvable model. In *Advances in Neural Information Processing Systems*, volume 38, pp. 118390–118424, 2025. doi: 10.52202/085713-3949. URL https://doi.org/10.52202/085713-3949.

Steffen Schneider, Jin Hwa Lee, and Mackenzie Weygandt Mathis. Learnable latent embeddings for joint behavioural and neural analysis. *Nature*, 617(7960):360–368, 2023. doi: 10.1038/s41586-023-06031-6. URL https://doi.org/10.1038/s41586-023-06031-6.

Krishna V. Shenoy, Maneesh Sahani, and Mark M. Churchland. Cortical control of arm movements: A dynamical systems perspective. *Annual Review of Neuroscience*, 36:337–359, 2013. doi: 10.1146/annurev-neuro-062111-150509. URL https://doi.org/10.1146/annurev-neuro-062111-150509.

Joshua H. Siegle, Xiaoxuan Jia, Séverine Durand, Sam Gale, Corbett Bennett, Nile Graddis, Greggory Heller, Tamina K. Ramirez, Hannah Choi, Jennifer A. Luviano, Peter A. Groblewski, Ruweida Ahmed, Anton Arkhipov, Amy Bernard, Yazan N. Billeh, Dillan Brown, Michael A. Buice, Nicolas Cain, Shiella Caldejon, Linzy Casal, Andrew Cho, Maggie Chvilicek, Timothy C. Cox, Kael Dai, Daniel J. Denman, Saskia E. J. de Vries, Roald Dietzman, Luke Esposito, Colin Farrell, David Feng, John Galbraith, Marina Garrett, Emily C. Gelfand, Nicole Hancock, Julie A. Harris, Robert Howard, Brian Hu, Ross Hytnen, Ramakrishnan Iyer, Erika Jessett, Katelyn Johnson, India Kato, Justin Kiggins, Sophie Lambert, Jerome Lecoq, Peter Ledochowitsch, Jung Hoon Lee, Arielle Leon, Yang Li, Elizabeth Liang, Fuhui Long, Kyla Mace, Jose Melchior, Daniel Millman, Tyler Mollenkopf, Chelsea Nayan, Lydia Ng, Kiet Ngo, Thuyahn Nguyen, Philip R. Nicovich, Kat North, Gabriel Koch Ocker, Doug Ollerenshaw, Michael Oliver, Marius Pachitariu, Jed Perkins, Melissa Reding, David Reid, Miranda Robertson, Kara Ronellenfitch, Sam Seid, Cliff Slaughterbeck, Michelle Stoecklin, David Sullivan, Ben Sutton, Jackie Swapp, Carol Thompson, Kristen Turner, Wayne Wakeman, Jennifer D. Whitesell, Derric Williams, Ali Williford, Rob Young, Hongkui Zeng, Sarah Naylor, John W. Phillips, R. Clay Reid, Stefan Mihalas, Shawn R. Olsen, and Christof 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://doi.org/10.1038/s41586-020-03171-x.

Yue Song, T. Anderson Keller, Yisong Yue, Pietro Perona, and Max Welling. Langevin flows for modeling neural latent dynamics. *arXiv preprint arXiv:2507.11531*, 2025. doi: 10.48550/arXiv.2507.11531. URL https://arxiv.org/abs/2507.11531.

Nicholas A. Steinmetz, Cagatay Aydin, Anna Lebedeva, Michael Okun, Marius Pachitariu, Marius Bauza, Maxime Beau, Jai Bhagat, Claudia Böhm, Martijn Broux, Susu Chen, Jennifer Colonell, Richard J. Gardner, Bill Karsh, Fabian Kloosterman, Dimitar Kostadinov, Carolina Mora-Lopez, John O'Callaghan, Junchol Park, Jan Putzeys, Britton Sauerbrei, Rik J. J. van Daal, Abraham Z. Vollan, Shiwei Wang, Marleen Welkenhuysen, Zhiwen Ye, Joshua T. Dudman, Barundeb Dutta, Adam W. Hantman, Kenneth D. Harris, Albert K. Lee, Edvard I. Moser, John O'Keefe, Alfonso Renart, Karel Svoboda, Michael Häusser, Sebastian Haesler, Matteo Carandini, and Timothy D. Harris. Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science*, 372(6539):eabf4588, 2021. doi: 10.1126/science.abf4588. URL https://doi.org/10.1126/science.abf4588.

David Sussillo and Omri Barak. Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Computation*, 25(3):626–649, 2013. doi: 10.1162/NECO_a_00409. URL https://doi.org/10.1162/NECO_a_00409.

Floris Takens. Detecting strange attractors in turbulence. In David A. Rand and L.-S. Young (eds.), *Dynamical Systems and Turbulence, Warwick 1980*, volume 898 of *Lecture Notes in Mathematics*, pp. 366–381. Springer, 1981. doi: 10.1007/BFb0091924. URL https://doi.org/10.1007/BFb0091924.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/.

Panagiota Theodoni, Bernat Rovira, Yingxue Wang, and Alex Roxin. Theta-modulation drives the emergence of connectivity patterns underlying replay in a network model of place cells. *eLife*, 7:e37388, 2018. doi: 10.7554/eLife.37388. URL https://doi.org/10.7554/eLife.37388.

Wilson Truccolo, Uri T. Eden, Matthew R. Fellows, John P. Donoghue, and Emery N. Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology*, 93(2):1074–1089, 2005. doi: 10.1152/jn.00697.2004. URL https://doi.org/10.1152/jn.00697.2004.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL https://jmlr.org/papers/v9/vandermaaten08a.html.

Vincent Villette, Mariya Chavarha, Ivan K. Dimov, Jonathan Bradley, Lagnajeet Pradhan, Benjamin Mathieu, Stephen W. Evans, Simon Chamberland, Dongqing Shi, Renzhi Yang, Benjamin B. Kim, Annick Ayon, Abdelali Jalil, François St-Pierre, Mark J. Schnitzer, Guoqiang Bi, Katalin Toth, Jun Ding, Stéphane Dieudonné, and Michael Z. Lin. Ultrafast two-photon imaging of a high-gain voltage indicator in awake behaving mice. *Cell*, 179(7):1590–1608.e23, 2019. doi: 10.1016/j.cell.2019.11.004. URL https://doi.org/10.1016/j.cell.2019.11.004.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pp. 1096–1103, Helsinki, Finland, 2008. ACM. doi: 10.1145/1390156.1390294. URL https://doi.org/10.1145/1390156.1390294.

Joel Ye, Jennifer L Collinger, Leila Wehbe, and Robert Gaunt. Neural data transformer 2: Multi-context pretraining for neural spiking activity. In *Advances in Neural Information Processing Systems*, volume 36, pp. 80352–80374, 2023. doi: 10.52202/075280-3523. URL https://doi.org/10.52202/075280-3523.

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. doi: 10.1152/jn.90941.2008. URL https://doi.org/10.1152/jn.90941.2008.

Yizi Zhang, Yanchen Wang, Donato M. Jiménez-Benetó, Zixuan Wang, Mehdi Azabou, Blake Richards, Renee Tung, Olivier Winter, The International Brain Laboratory, Eva Dyer, Liam Paninski, and Cole Hurwitz. Towards a "universal translator" for neural dynamics at single-cell, single-spike resolution. In *Advances in Neural Information Processing Systems*, volume 37, pp. 80495–80521, 2024. doi: 10.52202/079017-2559. URL https://doi.org/10.52202/079017-2559.

ACKNOWLEDGEMENTS

To maintain blinding, acknowledgements in this submission are omitted, and will be included in the final version.

6 APPENDIX

6.1 SOFTWARE AVAILABILITY, DATA AVAILABILITY, AND REPRODUCIBILITY

To maintain blinding, software and data availability information can be found in the anonymized repository at `https://anonymous.4open.science/r/F3E2-ocsidln/README.md`. Full reproduction of Results and Additional results is possible through code and artifacts found in the `experiments/` directory.

6.2 METHODS COMPARISON

Many neural LVM approaches focus on decoding from a low-dimensional latent space. However, the structure of this latent space can be inherently complex and difficult to interpret. NLDisco bypasses this challenge by learning individual latents directly, as illustrated in the toy data shown in Figure S1.

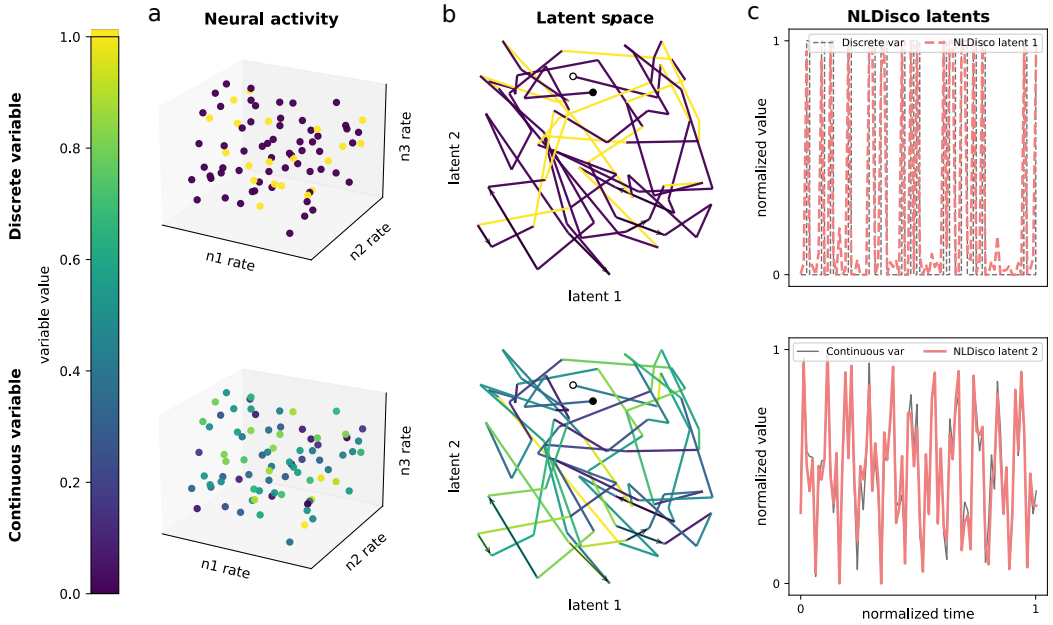


Figure S1: Interpretable latents in a complex latent space.
This toy example highlights the utility of NLDisco. The two rows show two different variables (top: discrete; bottom: continuous), each uniquely encoded by the same underlying neural activity made up of three neurons' firing rates. The colorbar shows the variables' values as a function of this neural activity. (**a**) Each point in the scatterplots represents a moment in time. (**b**) A projection of this activity into a 2D latent space creates tangled trajectories where variable states (e.g. 'on' and 'off' in the discrete case, and 'high' and 'low' in the continuous case) are not easily distinguished. The start and end points of the trajectories are marked by white and black dots, respectively, while arrows indicate trajectory direction. (**c**) In contrast to the tangled latent space, NLDisco finds individual latents corresponding to each variable, demonstrating the potential for improved interpretability of neural representations.

Beyond this conceptual distinction, we compare NLDisco with other neural latent variable methods across 15 methodological features (Table S1):

- *Requires multimodal data*: Whether the method requires multimodal data (e.g. video data, or various forms of behavioral data, in addition to neural data).
- *Requires trial-structured data*: Whether the method requires trial-structured data.
- *Supports multimodal data*: Whether the method can incorporate multimodal data, even if not required.
- *Supports trial-structured data*: Whether the method can use trial-structured data, even if not required.

- *Learns sparse, easily identifiable latents*: Whether the method by default learns sparse, easily identifiable latents.
- *Learns hierarchical latents*: Whether the method by default learns latents that are hierarchically organized (e.g. whether the method can learn one latent that corresponds to a particular behavior, and another, sparser latent that corresponds to a sub-behavior of the first)
- *Learns temporally precise latents*: Whether the method by default learns latents that are time-locked to neural events at the resolution of the desired event, potentially down to single-spike precision.
- *Learns a continuously-valued latent space*: Whether the method by default learns latents that change smoothly as a function of changes in the input neural data.
- *Can use temporal dynamics to update the latent space*: Whether the method can use temporal dynamics (e.g. neural data or latent space history) to update the latent space.
- *Imposes a prior on the latent space*: Whether the method imposes a predfined structure on the latent space (e.g. geometric constraints like orthogonality of latents, or distributional assumptions like a Gaussian latents).
- *Uses nonlinear dynamics*: Whether the method learns latents from nonlinear neural dynamics.
- *Can be used as a generative model*: Whether the method can be used to generate new neural data samples from the learned latent space.
- *Enforces neural data reconstruction*: Whether the method needs to perform neural data reconstruction when learning latents.
- *Has approximate linear time scaling*: Whether the method has linear time scaling with respect to the number of data points in an example and in the dataset.
- *Requires significant hyperparameter tuning*: Whether the method requires significant hyperparameter tuning to learn interpretable latents.

Light-green text indicates that the method has an ideal implementation of the feature, while dark-red text indicates a shortcoming.

Feature	**NLDisco***	LangevinFlow* (Song et al., 2025)	CEBRA* (Schneider et al., 2023)	ST-NDT (Le & Shlizerman, 2022)	AutoLFADS (Keshtkaran et al., 2022)	UMAP (McInnes et al., 2018)	t-SNE (van der Maaten & Hinton, 2008)	sparseNMF (Hoyer, 2004)	ICA (Comon, 1994)	PCA** (Hotelling, 1933)
Requires multimodal data	No	No	No	No	No	No	No	No	No	No
Requires trial-structured data	No	No	No	No	No^	No	No	No	No	No
Supports multimodal data	No†	No	Yes	No	No	No	No	No	No	No
Supports trial-structured data	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Learns sparse, easily identifiable latents	Yes	No	No	No	No	No	No	Yes	No	No
Learns hierarchical latents	Yes	No	No	No	No	No	No	No	No	No
Learns temporally precise latents	Yes	Yes	Yes	Yes	Yes	No	No	Yes	No	No
Learns a continuously-valued latent space	No‡	Yes	Yes	Yes	Yes	Yes	Yes	No‡	Yes	Yes
Can use temporal dynamics to update the latent space	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No
Imposes a prior on the latent space	No	Yes	No	No	Yes	No	No	No	Yes	Yes
Uses nonlinear dynamics	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No
Can be used as a generative model	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No
Enforces neural data reconstruction	Yes	Yes	No	Yes	Yes	No	No	Yes	No	No
Has approximate linear time scaling	Yes#	No	Yes	No	No	Yes	No	No	Yes	Yes
Requires significant hyperparameter tuning	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No

Table S1: Comparison of NLDisco with other neural LVM methods.

*: Latent comparison visualized in Results
**: Latent comparison visualized in Additional results
†: Not a fundamental limitation – could be added as a feature
‡: Enforced sparsity can cause step-like jumps in the data-to-latents mapping
#: In the standard implementation, without a transformer layer in the decoder
^: Can bin data to create "pseudo"-trials

6.3 Additional pipeline details

6.3.1 Data preprocessing

Spike binning and unit selection. The Kilosort loader reads exported spike sample indices and cluster assignments, using a specified sampling rate, timebin duration, and recording interval. Spikes are counted in left-inclusive, right-exclusive timebins, retaining periods without spikes and preserving the correspondence between matrix columns and original cluster identifiers. Users may select units by identifier or cluster-quality label; no quality filtering is applied by default. Binning uses the exported acquisition timebase, while synchronization with behavioral measurements is handled separately.

Normalization. Activity can be z-scored or min–max scaled across timebins independently for each unit, or across units independently within each timebin. These operations serve different purposes: per-unit normalization adjusts differences in activity scale across units, whereas normalization within each timebin adjusts differences in activity scale across time.

Window construction. Consecutive timebins are grouped into windows of length S, with a configurable stride. When trial identifiers, session identifiers, timestamps, and validity masks are supplied, windows crossing corresponding boundaries, temporal gaps, or invalid observations are excluded. Restricting eligible timebins to a data partition also helps prevent windows from crossing training, validation, and test boundaries. For paired input and target windows, these restrictions apply to both windows; their pairing is defined before batching.

6.3.2 Model training

We use S and N for the input window length and unit count, D latent dimensions, B examples per batch, and training sparsity parameter k. The subscript out denotes target dimensions, allowing both window length and unit count to differ from the input in the transcoder case. The dataset defines batch pairs of inputs $\mathbf{Y} \in \mathbb{R}^{B \times S \times N}$ with targets $\mathbf{\Psi} \in \mathbb{R}^{B \times S_{\mathrm{out}} \times N_{\mathrm{out}}}$; the prediction $\hat{\mathbf{\Psi}}$ has the target's shape. Autoencoding is the special case $\mathbf{\Psi} = \mathbf{Y}$, with $S_{\mathrm{out}} = S$ and $N_{\mathrm{out}} = N$. "Overcompleteness" implies $D > N$.

Window SEDs. The FW-SED encoder vectorizes the full $S \times N$ window and applies one learned linear projection to produce D latent pre-activations. This is the most direct temporal extension of a conventional SED: a hidden layer neuron (latent) can assign a different weight to every unit–timebin pair and can therefore represent a motif at a particular position within the window.

The TW-SED encoder treats the N-dimensional population activity at each of the S input timebins as one token, projects it into an embedding space, adds learned absolute-position embeddings, and applies one or more self-attention blocks. The representation at the final token is then projected to the same D latent pre-activations. Causal attention restricts this readout to the present and preceding bins.

In preliminary TW-SED experiments, we also tried concatenating the contextualized outputs from all tokens and projecting this full-window representation into latent pre-activations. This yielded worse recovery of ground-truth latents than projecting only the contextualized final token, motivating the latter readout.

For either Window encoder, ReLU produces nonnegative latent activations, and BatchTopK enforces latent sparsification. The decoder applies a learned linear mapping and bias to the sparse activations, followed by an activation function g, to predict the full target window $\hat{\mathbf{\Psi}} \in \mathbb{R}^{B \times S_{\mathrm{out}} \times N_{\mathrm{out}}}$ jointly. g can be the identity function, ReLU, or softplus: identity output permits signed reconstructions, useful for z-scored activity, whereas ReLU or softplus enforces nonnegativity, for spike counts outputs, for example. The decoder weights specify each latent's contribution to output units at each timebin within the target window before the output activation, making the learned spatiotemporal patterns explicit.

Training objective and algorithm. For L nested Matryoshka levels, D_ℓ, k_ℓ, and λ_ℓ denote the dictionary width, sparsity parameter, and reconstruction-loss weight of level ℓ, with $D_L = D$. A single-level SED has $L = 1$, $D_1 = D$, and $k_1 = k$. At each Matryoshka level ℓ, BatchTopK retains

the Bk_ℓ largest activations across the batch, considering only that level's D_ℓ latent dimensions. Each level attempts to reconstruct the complete target using the time-weighted loss,

$$\mathcal{L}_w(\mathbf{\Psi}, \hat{\mathbf{\Psi}}) = \frac{\sum_{s=1}^{S_{\text{out}}} w_s \, \mathcal{E}(\hat{\mathbf{\Psi}}_{:,s,:}, \mathbf{\Psi}_{:,s,:})}{\sum_{s=1}^{S_{\text{out}}} w_s}, \tag{1}$$

where $\mathcal{E}$ denotes the chosen elementwise reconstruction loss (e.g., $\mathbf{MSE}$), averaged over examples and output units at each target timebin, with one positive weight w_s per target timebin. Reconstruction metrics may be reported separately for each timebin within the target window.

Algorithm 1 specifies one training step. The encoder E_θ is the single-bin, flat-window, or transformer-window encoder; the reconstruction indices $s = 1, \ldots, S_{\text{out}}$ and $n = 1, \ldots, N_{\text{out}}$ refer to target timebins and output units. The shift-equivariant variant uses time-indexed latent occurrences and a convolutional decoder, as described below.

Algorithm 1 Training step with one latent vector per input window

Require: Input $\mathbf{Y} \in \mathbb{R}^{B \times S \times N}$; target $\mathbf{\Psi} \in \mathbb{R}^{B \times S_{\text{out}} \times N_{\text{out}}}$
Require: Encoder E_θ; decoder $\mathbf{W}_{\text{dec}} \in \mathbb{R}^{S_{\text{out}} \times N_{\text{out}} \times D}$
Require: Decoder bias $\mathbf{b}_{\text{dec}} \in \mathbb{R}^{S_{\text{out}} \times N_{\text{out}}}$ and output activation g
Require: Levels $\{D_\ell, k_\ell, \lambda_\ell\}_{\ell=1}^{L}$, time weights $w_s > 0$, auxiliary weight γ

procedure TRAINSTEP($\mathbf{Y}, \mathbf{\Psi}$)
  $\mathbf{Z} \leftarrow \mathrm{ReLU}(E_\theta(\mathbf{Y})) \in \mathbb{R}^{B \times D}$
  $\mathcal{L}_{\text{recon}} \leftarrow 0$
  for $\ell = 1, \ldots, L$ **do**
    $\tilde{\mathbf{Z}}_\ell \leftarrow \mathrm{BatchTopK}(\mathbf{Z}_{:,1:D_\ell}, Bk_\ell)$
    $\hat{\mathbf{\Psi}}_{\ell,b,s,n} \leftarrow g\Big(\sum_{j=1}^{D_\ell} \tilde{Z}_{\ell,b,j} W_{\text{dec},s,n,j} + b_{\text{dec},s,n}\Big)$
    $\mathcal{L}_{\text{recon}} \leftarrow \mathcal{L}_{\text{recon}} + \lambda_\ell \mathcal{L}_w(\mathbf{\Psi}, \hat{\mathbf{\Psi}}_\ell)$
  end for
  $\mathcal{D}_{\text{dead}} \leftarrow \mathrm{InactiveLatents}(\text{activation history})$
  $\mathbf{R} \leftarrow \mathbf{\Psi} - \mathrm{stopgrad}(\hat{\mathbf{\Psi}}_L)$
  $\mathcal{L}_{\text{aux}} \leftarrow \mathrm{DeadLatentResidualLoss}(\mathbf{Z}, \mathbf{R}, \mathcal{D}_{\text{dead}})$
  $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{recon}} + \gamma \mathcal{L}_{\text{aux}}$
  Backpropagate $\mathcal{L}_{\text{total}}$; project decoder gradients tangent to unit-norm dictionary
  Update parameters; normalize decoder atoms; update latent inactivity counters
end procedure

Reconstruction loss functions. For a fixed target timebin s, let $\psi = \Psi_{b,s,n}$ and $\hat{\psi} = \hat{\Psi}_{b,s,n}$ denote target and reconstruction entries, and let $\langle \cdot \rangle_{b,n}$ denote the mean over examples and output units. Three loss functions are mean squared error (MSE), mean squared log error (MSLE), and Poisson KL divergence (PKL):

$$\mathcal{E}_{\text{MSE}} = \left\langle (\hat{\psi} - \psi)^2 \right\rangle_{b,n}, \tag{2}$$

$$\mathcal{E}_{\text{MSLE}} = \left\langle \left[\log(1 + \psi) - \tau \log(1 + \max(\hat{\psi}, 0)) \right]^2 \right\rangle_{b,n}, \tag{3}$$

$$\mathcal{E}_{\text{PKL},\epsilon} = \left\langle (\psi + \epsilon) \log \frac{\psi + \epsilon}{\hat{\psi} + \epsilon} + \hat{\psi} - \psi \right\rangle_{b,n}. \tag{4}$$

The MSLE expression follows the implementation's clipping of negative reconstructions and requires $\psi > -1$. Setting $\tau = 1$ gives the usual squared log error. For nonnegative targets, $\tau > 1$ more heavily penalizes overestimation, whereas $0 < \tau < 1$ more heavily penalizes underestimation. The preliminary PKL implementation used nonnegative targets and reconstructions, adding $\epsilon > 0$ to both to avoid evaluating logarithms at zero (default 10^{-6}).

In preliminary comparisons with all other settings held fixed, MSLE yields better reconstruction fidelity and recovery of synthetic ground-truth features for raw spike counts, whereas we observe no performance difference between MSE and MSLE for standardized spike data. For MSLE, setting

$\tau = 1$ generally gives the best results among the tested values. PKL performs worse than MSE and MSLE in these experiments, and choosing the offset ϵ introduces an additional numerical consideration near zero. The synthetic spatial models in Figure 3c,d use MSLE with $\tau = 1$; all other reported NLDisco models in the synthetic, Churchland, and Aeon results use MSE.

Dead-latent auxiliary loss. Some latents stop activating during training and therefore contribute nothing to reconstruction (marked in $\mathcal{D}_{\text{dead}}$). To encourage their reuse, we train them to fit the residual error $\mathbf{R}$ left by the highest Matryoshka level, similar to the AuxK loss of Gao et al. (2025). This auxiliary loss encourages reuse through gradient updates without explicitly reinitializing inactive latents. This auxiliary branch selects activations only from these dead latents using BatchTopK, with a separate sparsity budget determined by their number and a configured fraction of the dictionary width. This keeps the auxiliary reconstruction sparse, encouraging a small set of dead latents to explain the residual rather than distributing it across many weakly activated features, ostensibly improving interpretability. The residual is treated as a fixed per-batch target during backpropagation for these dead latents only. When no latents are dead, the auxiliary loss is zero.

Optional shift-equivariant mode. The TW-SED's shift-equivariant configuration changes the scientific interpretation from a feature tied to one window alignment to a motif that can occur at different positions within the window. Absolute-position embeddings are removed, and learned relative-position biases encode offsets between timebins, allowing the same detector to operate at every timebin. Rather than reducing the transformer sequence to its final token, the latent projection produces a temporally indexed activation tensor in $\mathbb{R}^{B\times S\times D}$. Batch top-$k$ is then applied across valid time–latent occurrences, so the same latent may activate at different positions in different windows. This can remove obfuscation from the same feature being captured by different latents at different positions within the window.

This mode uses aligned input and target windows on a shared time grid, with $S_{\text{out}} = S$. The corresponding decoder replaces the fixed full-window linear dictionary with a transposed temporal convolution. Each latent has a motif kernel specifying its contribution to each output unit at temporal offsets relative to the motif's activation bin. Every sparse occurrence places a shifted copy of that kernel into the reconstruction. Causal kernels reconstruct the occurrence and preceding support; centered kernels reconstruct symmetrically around it. Because unequal timebin weights would privilege absolute positions and break the intended symmetry, this mode requires all w_s to be equal.

Hyperparameter guidance. NLDisco uses Hydra-based YAML configurations and command-line overrides to specify data, architecture, and optimization settings. Automated sweeps run on CPUs or GPUs using local search or Weights & Biases, with local execution or Slurm scheduling through Submitit.

We recommend jointly varying dictionary width D and sparsity k, testing several sparsity levels at each width rather than automatically holding k/D fixed. For recordings similar to those studied here, $D \in \{128, 192, 256\}$ provides a practical starting range based on our experiments; dictionary width is not determined solely by the number of recorded units or windows. Vary k separately: our synthetic experiments use average budgets as small as $k = 4$, whereas real-data configurations explore larger budgets. The appropriate choices also depend on window duration, preprocessing, and the diversity and frequency of the features to be recovered. Comparisons should hold preprocessing, bin width, window lengths, batch size, and evaluation observations fixed unless explicitly varied. Assess reconstruction fidelity alongside measured mean $\mathbf{L0}$, redundant latents, dead latents, and consistency of recovered features across training seeds (see 6.3.3, 6.3.4 for examples). Larger dictionaries offer more candidates when searching for the best latent; report the search width and any controls for candidate count. Neither the largest dictionary nor the lowest reconstruction error is necessarily preferred.

For training optimization, we use Adam with no weight decay, an initial learning-rate setting of 0.005, $(\beta_1, \beta_2) = (0.9, 0.999)$, and $\epsilon = 10^{-8}$. We do not find additional weight regularization, such as AdamW's decoupled weight decay, necessary in these experiments, where BatchTopK constrains latent activity and the decoder weights are normalized. Exploratory comparisons with NAdam do not yield a clear improvement over Adam. All of our experiments in this paper use the same learning rate schedule, defined as fractions of the total number of optimization steps: linear warmup over the first 10%, a constant learning rate until 50%, and a single cosine decay toward 1% of the initial

rate over the remaining updates. This schedule depends on update count rather than reconstruction loss, so short-term loss fluctuations do not trigger learning-rate reductions. Batch sizes of 256–1,024 windows are used in our experiments. For new datasets, a small learning-rate sweep around the initial setting can be useful.

When monitoring training, it can be useful to distinguish three broad phases: 1) initial representation learning, when latents capture general structure; 2) specialization, when individual latents develop more distinct activation patterns; and 3) refinement, when those patterns undergo smaller adjustments. These phases are a qualitative guide rather than fixed stages that every run must exhibit.

6.3.3 MODEL EVALUATION

Model evaluation considers reconstruction fidelity, dictionary width D, and activation sparsity k separately. NLDisco reviews dictionary quality via latent sparsity metrics (Figure S2 top). The $\mathbf{L0}$ norm measures the number of active latents in an example, set indirectly by the batch top-k parameter, while the latent activation density visualizes the distribution of activation frequencies across latents. Together these metrics reveal whether the model has learned a code that uses many latents efficiently while avoiding common pitfalls such as "representational collapse" into too many constantly active latents and widespread "feature death" in which a large portion of the dictionary never activates. These sparsity metrics are complemented by a couple of reconstruction fidelity metrics: variance explained ($\mathbf{R^2}$) and cosine similarity of reconstruction-to-target neural activity across target units and timebins (Figure S2 bottom). High scores indicate faithful reconstruction under these metrics. Importantly, more available latents D or more active latents k can improve reconstruction, but reconstruction alone does not establish latent interpretability: if the neural code is segmented across too many or too few latents, the resulting features may be difficult to interpret.

Two optional diagnostics examine how individual latents support reconstruction and how well reconstruction preserves frequency content.

Latent ablation importance measures the change in reconstruction error when one latent's activations are set to zero, holding all other activations fixed. We report the increase in mean-squared error and its variance-normalized counterpart, $\Delta R_j^2 = (\mathrm{SSE}_{-j} - \mathrm{SSE}_{\mathrm{full}})/\mathrm{SST}$, where SST centers each target unit at each lag across evaluation windows. Importantly, these scores are not additive shares of explained variance, and unequal contributions need not indicate a problem: meaningful features may differ in prevalence and strength.

Spectral reconstruction fidelity compares each unit's target and reconstructed power spectral densities using Welch's method, with a specified bin duration and spectral segment duration. We assemble one reconstruction per physical timebin at a fixed position within the target window (its endpoint by default), and process contiguous stretches separately. Outputs include the spectra, integrated absolute spectral differences normalized by target power, and optional frequency-band power differences. Spectra are computed in the same normalized units as the targets; zero target power makes relative errors undefined. Matching spectra supports preservation of frequency content, but does not establish preservation of phase or event timing.

6.3.4 LATENTS EVALUATION

Visualization and interpretation. In general, we recommend building a simple dashboard or set of linked plots that places latent activations alongside behavioral and environmental variables: inspect tuning curves and activating examples to develop candidate feature descriptions, then assess these descriptions across the full eligible population, including examples in which the latent is inactive.

In this paper, we evaluate latents using dataset-specific visualizations. In 4.1 we compare spatial tuning and activating trajectories with known place fields and movement directions, in 4.2 we examine latent activity alongside movement kinematics and trial conditions, and in 4.3 we examine latent activity during spontaneous foraging, including wheel activity and movement around the patch. These observations inform candidate feature descriptions, which we quantify over explicitly defined comparison populations using the shared metrics described below.

Coverage, specificity, selectivity, and discriminability. To assess a candidate feature, we need to measure both how often a latent detects its proposed condition and how often it activates outside that

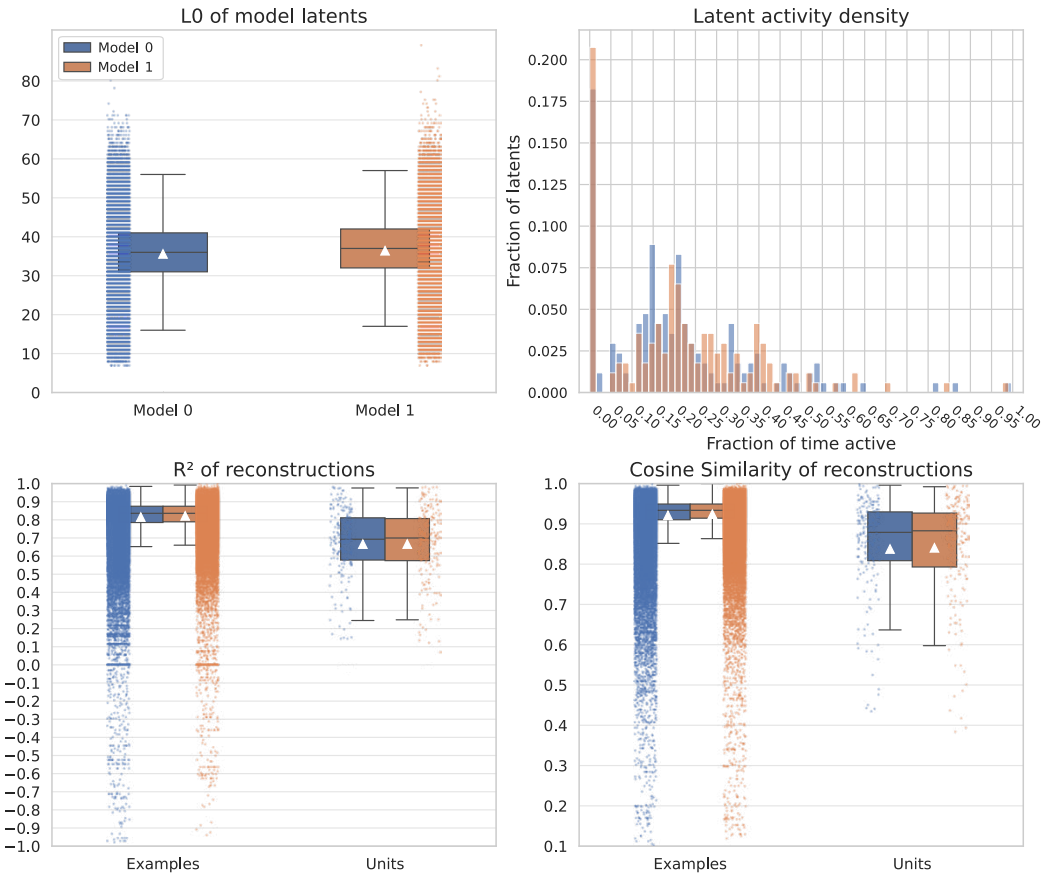


Figure S2: Model evaluation metrics.
This example uses Allen Institute Visual Coding Neuropixels data (Siegle et al., 2021). The SED model has $L = 3$ nested Matryoshka levels with $(D_1, k_1) = (128, 12)$, $(D_2, k_2) = (224, 24)$, and $(D_3, k_3) = (320, 36)$, and $B = 1024$ examples per batch. Each level selects Bk_ℓ activation entries across the batch from its first D_ℓ latents. The recording contains $N = 200$ units, and the model is trained on 182,868 single-bin examples ($S = 1$) of 50-ms spike counts. The model exhibits an approximately normal distribution of the latent **L0** norm, few dead or tonically active latents, and high neural reconstruction fidelity as measured by $\mathbf{R^2}$ and cosine similarity across both units and examples, all indicators of healthy training.

condition. Coverage and specificity describe these two aspects, selectivity summarizes the relative preference, and discriminability measures how well activation values separate these two aspects across thresholds.

Let A_l denote the event that latent l is active under the stated activation rule, and let C denote a behavioral or environmental condition, such as speed in a specified interval. The true-positive rate (**TPR**) measures how often the latent is active when the condition holds; the false-positive rate (**FPR**) measures how often it is active otherwise. Coverage (**Cov**) = **TPR**, specificity (**Spec**) = $1 - \mathbf{FPR}$, and selectivity (**Sel**) compares the two activation probabilities (estimated as fractions of eligible examples in each condition class):

$$\mathbf{TPR}_l(C) = P(A_l \mid C),$$
$$\mathbf{FPR}_l(C) = P(A_l \mid \neg C),$$
$$\mathbf{Cov}_l(C) = \mathbf{TPR}_l(C), \tag{5}$$
$$\mathbf{Spec}_l(C) = 1 - \mathbf{FPR}_l(C), \tag{6}$$
$$\mathbf{Sel}_l(C) = \frac{\mathbf{TPR}_l(C)}{\mathbf{TPR}_l(C) + \mathbf{FPR}_l(C)}. \tag{7}$$

Selectivity should be read alongside coverage. For example, $\mathbf{TPR} = 0.10$, $\mathbf{FPR} = 0.01$ and $\mathbf{TPR} = 0.90$, $\mathbf{FPR} = 0.09$ both give $\mathbf{Sel} \simeq 0.909$, despite very different coverage. A highly selective latent may therefore detect only a small subset of a condition's occurrences.

To quantify discriminability across activation thresholds, we use the area under the receiver operating characteristic curve ($\mathbf{AUROC}$), which measures how reliably condition-positive examples have higher latent activity than condition-negative examples:

$$\mathbf{AUROC} = P(z_C > z_{\neg C}) + \tfrac{1}{2}P(z_C = z_{\neg C}), \tag{8}$$

where z_C and $z_{\neg C}$ denote the same latent's activation value for an example drawn from inside and outside condition C, respectively. Each comparison counts as correct when the positive example has the larger activation, incorrect when it has the smaller activation, and half-correct when the activations are equal.

Decoding. We also assess how well a feature can be classified as present ($y = 1$) or absent ($y = 0$) based on latent activity by fitting an ℓ_2-regularized logistic-regression decoder,

$$\widehat{P}(y = 1 \mid \mathbf{z}) = \sigma\big(b + \mathbf{w}^\top \widetilde{\mathbf{z}}\big), \qquad \sigma(a) = \frac{1}{1 + \exp(-a)}, \tag{9}$$

where $\widetilde{\mathbf{z}}$ is standardized $\mathbf{z}$ latent activations, $\mathbf{w}$ the learned regression weight, and b an intercept term. A single-latent decoder uses a scalar w to weight the specified latent; a full-space decoder uses a vector w to weight all latents in the LVM.

We fit the decoder by minimizing binary cross-entropy with an ℓ_2 penalty on its weights:

$$\begin{aligned} \mathcal{L}_{\mathrm{dec}}(\mathbf{w}, b) &= -\sum_{i=1}^{M} [y_i \log \widehat{p}_i + (1 - y_i) \log(1 - \widehat{p}_i)] + \frac{\lambda_{\mathrm{dec}}}{2} \|\mathbf{w}\|_2^2, \\ \widehat{p}_i &= \sigma\big(b + \mathbf{w}^\top \widetilde{\mathbf{z}}_i\big)\,. \end{aligned} \tag{10}$$

Here, M is the number of training examples, y_i indicates whether the behavioral condition holds, and λ_{dec} controls regularization strength, which we select from $\lambda_{\mathrm{dec}} \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100, 1000\}$ to maximize validation set $\mathbf{AUROC}$. We optimize the decoder weights and intercept using L-BFGS over the full training set, with a convergence tolerance of 10^{-5} and a maximum of 1,000 iterations per fit.

The reported $\mathbf{Decode}$ score is test balanced accuracy: the average fraction of condition-positive examples predicted positive and condition-negative examples predicted negative,

$$\mathbf{Decode} = \tfrac{1}{2}\big(\mathbf{TPR}_{\mathrm{decoder}} + 1 - \mathbf{FPR}_{\mathrm{decoder}}\big). \tag{11}$$

This gives both classes equal weight, with 0.5 for a constant prediction and 1 for perfect classification. For each fitted decoder, we select a single probability threshold t that maximizes balanced accuracy on the validation set. An example is classified as condition-positive when its predicted probability $\widehat{P}(y = 1 \mid \mathbf{z}) \geq t$, and condition-negative otherwise. The decoder weights and this threshold are then held fixed for all test examples.

Source activity traceback. To identify which recorded units and timebins contribute to a latent's activation, we trace its response back through the encoder. For an input window $\mathbf{Y}$, let $h_l(\mathbf{Y})$ denote latent l's pre-activation, before ReLU and sparsification. We assign each input entry the attribution score:

$$a_{l,s,n}(\mathbf{Y}) = Y_{s,n} \frac{\partial h_l(\mathbf{Y})}{\partial Y_{s,n}}. \tag{12}$$

Here, $Y_{s,n}$ is the activity of unit n at timebin s (post input data preprocessing). The derivative measures how sensitive the latent's pre-activation is to a small change in that input entry. For a linear encoder, the derivative is simply the corresponding encoder weight, so the score is weight times activity. For a transformer encoder, the derivative also depends on the surrounding activity in the input window. For models with time-indexed latent activations, we attribute the endpoint pre-activation.

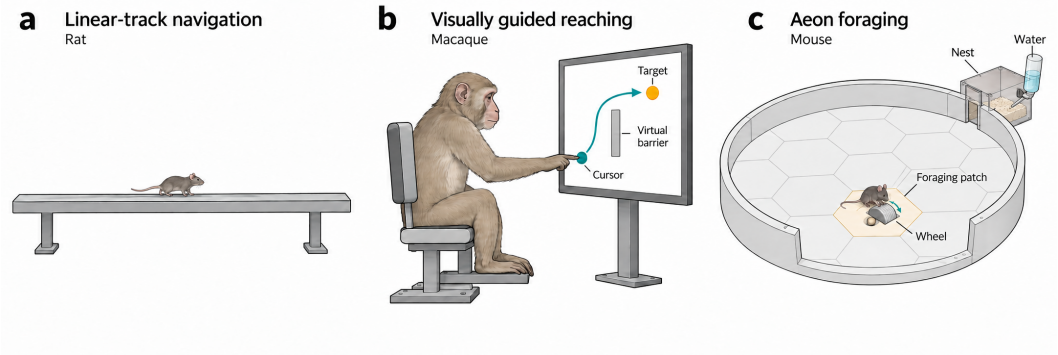


Figure S3: Schematic experiment assays for the three datasets. (**a**) Simulated rat navigation along a linear track (Section 4.1). (**b**) Macaque visually guided reaching: after a go cue, hand movements guide a cursor to a visual target while avoiding virtual barriers (Churchland et al., 2012) (Section 4.2). (**c**) Mouse foraging in an Aeon arena with a single food patch, where turning a small wheel provides access to food; (Campagner et al., 2025) (Section 4.3)

6.4 Additional results

All results are reproduced in publicly available notebooks – see Software availability, data availability, and reproducibility.

6.4.1 Simulated rat hippocampal spike data in a navigation task

Dataset design. We simulate four place cells and 96 position-independent noise units for 600 s at 100 Hz, giving a $100 \times 60{,}000$ spike matrix. The signed Ornstein–Uhlenbeck movement model is motivated by Theodoni et al. (2018). Position starts at 0.5 m on a 1-m track, with velocity

$$dv_t = \theta(\mu - v_t)\,dt + \sigma\,dW_t, \qquad \theta = 1\,\mathrm{s}^{-1}, \quad \mu = 0, \quad \sigma = 0.4\,\mathrm{m\,s}^{-3/2}, \quad v_0 = 0. \tag{13}$$

Euler–Maruyama updates velocity before advancing position. At track endpoints, overshoot is reflected and velocity changes sign without damping. Motion uses NumPy's PCG64 generator with seed 42.

Cells 1–4 have centers $c = (0.2, 0.4, 0.7, 0.7)$ m and amplitude parameters $a = (10, 28, 20, 20)$ Hz. Their position-dependent firing rates are split Gaussians:

$$r_i(x) = 0.1 + a_i \exp\left[-\frac{1}{2}\left(\frac{x - c_i}{s_i(x)}\right)^2\right], \qquad s_i(x) = \begin{cases} s_{i,L}, & x < c_i, \\ s_{i,R}, & x \geq c_i. \end{cases} \tag{14}$$

The left widths are $s_L = (0.05, 0.10, 0.14, 0.06)$ m and right widths are $s_R = (0.10, 0.05, 0.14, 0.06)$ m. All rates depend only on position, with no direction or velocity modulation (Figure S4).

Each noise unit has a constant rate independently sampled from Uniform$(1, 5)$ Hz. Spikes are independent Bernoulli draws with probability $r_i(x_t)\,dt$. Empirical rate maps divide spike counts by occupancy in 20 equal bins over the observed position range, including the maximum-position sample. Display smoothing uses a Gaussian standard deviation of 0.5 bins with nearest-edge extension. Exact generative maps are saved on a 501-point grid. Throughout this subsection, cells 1–4 correspond to zero-based unit IDs 0–3 in the saved artifacts; model seeds and latent indices remain zero-based.

FW-SED versus TW-SED. Full-recording reconstruction **MSE** is 0.152 ± 0.000 for FW-SED and 0.153 ± 0.000 for TW-SED (mean ± 1 sample standard deviation across five initializations of each; standard deviations below 0.001 display as 0.000). Both architectures recover directional features under the same criteria, using the direction index (DI) and selection procedure defined below, though the TW-SED latents have higher selected composite direction scores. Figure S5 compares the strongest passing forward feature from each architecture. These observations indicate TW-SED generates more interpretable latents for directionality in this dataset, though do not establish superiority of either architecture generally.

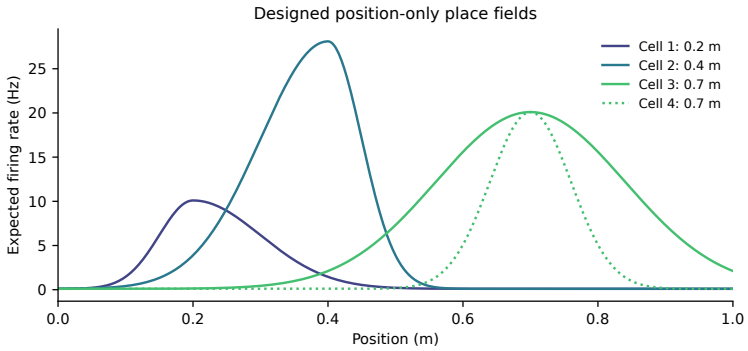


Figure S4: Synthetic place-field design. Exact expected firing rates of the four place cells, including the 0.1-Hz baseline. Cells 1 and 2 have asymmetric tails; cells 3 and 4 share a center but differ in width. Colors indicate field center, and the dotted curve distinguishes cell 4 from cell 3.

Oppositely directed temporal features. TW-SED finds features associated with ordered traversal of the first pair of place fields. We show the highest-scoring passing forward feature (seed 0, latent 20) and backward feature (seed 2, latent 66), using the selection criteria below. Figure S6 shows each latent's place-cell decoder pattern alongside its 16 highest-activating nonoverlapping windows. Examples are selected greedily from all 11,981 windows with positive activation, without filtering position, displacement, reversals, or direction.

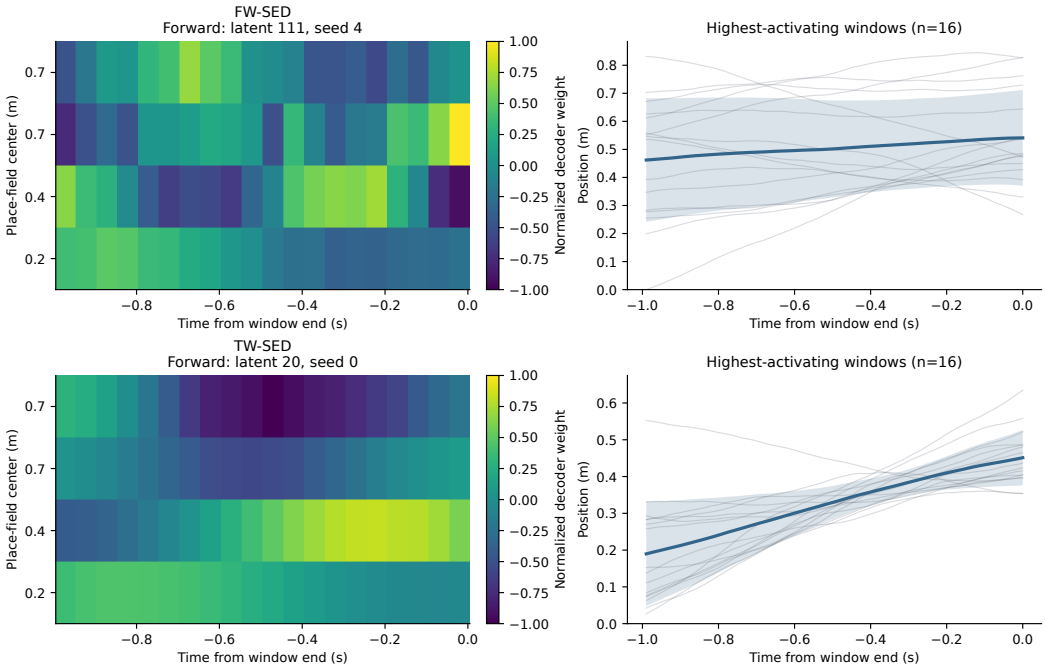


Figure S5: Comparison of FW-SED and TW-SED temporal features. Top: strongest passing forward FW-SED latent; bottom: strongest passing forward TW-SED latent. Left panels show decoder weights for the four known place cells over the window, normalized by their maximum absolute weight. Right panels show the 16 highest-activating windows for each latent, selected without behavioral filtering. Thin curves are trajectories, the mean is emphasized, and shading is ± 1 sample standard deviation across trajectories. Both architectures use the same recording, window duration, latent count, sparsity, and training budget.

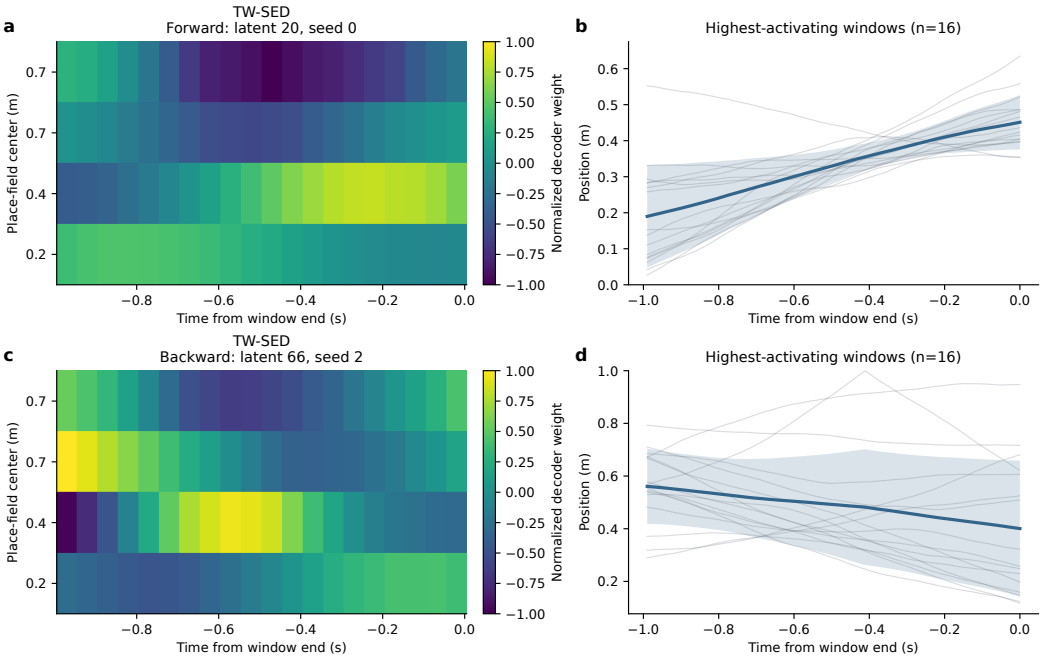


Figure S6: Forward- and backward-preferring TW-SED latents. Left panels show decoder weights for the four known place cells over the window, normalized by their maximum absolute weight. Right panels show the 16 highest-activating windows for each latent, selected without behavioral filtering. Thin curves are trajectories, the mean is emphasized, and shading is ± 1 sample standard deviation across trajectories. The preferred direction occurs in 15/16 and 13/16 examples, respectively.

Conditioning on clean crossings. To further examine traversals over the first pair of place fields, we restrict the eligible pool to strict nonreversing first-pair crossings with mean position in $[0.2, 0.4]$ m, i.e. the animal's average position within the window must lie between 0.2 and 0.4 m, but its trajectory may extend beyond this interval. There are 218 forward and 232 backward eligible windows. Both directions compete when selecting the 16 highest-activating windows for each of the same two latents; no preferred-direction filter is applied. All examples per latent now follow its preferred direction (Figure S7).

Source activity traceback for spatial latents. For each selected spatial latent in Figure 3c, we apply the source activity traceback to standardized unit activity in its same model input form: unique 50-ms binned counts. For the linear vanilla encoder, this multiplies each unit's activity by its encoder weight. We average equally over active bins within each of 20 equal position bins over the 1-m track, regardless of latent activation value. At each position, we then average equally across the selected runs with active bins there, retaining one curve per recorded unit (Figure S8). Each field has five selected runs; a run contributes no value at positions where its latent never activates.

Source activity traceback over time. For the same two TW-SED latents in Figure S6, Figure S9 applies the source activity traceback to the same model input form: unstandardized 50-ms counts, retaining the unit and timebin axes across the full window. At each relative time, we then average these attributions equally across all windows with positive latent activation: 9,991 windows for the forward latent and 7,281 for the backward latent. This analysis does not restrict to clean crossings or top examples, and does not weight windows by activation value.

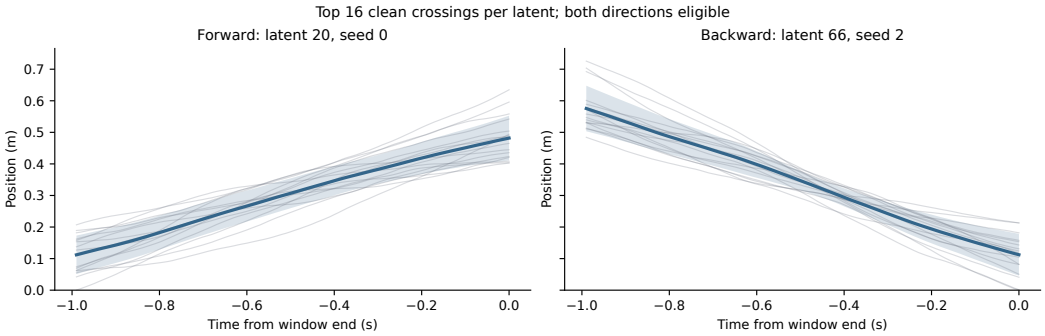


Figure S7: The same temporal latents conditioned on clean first-pair crossings. Left: seed 0, latent 20; right: seed 2, latent 66. Each panel contains its 16 highest-activating disjoint eligible crossings, with both directions competing. All 16 follow the respective preferred direction. Shading is ± 1 sample standard deviation across trajectories.

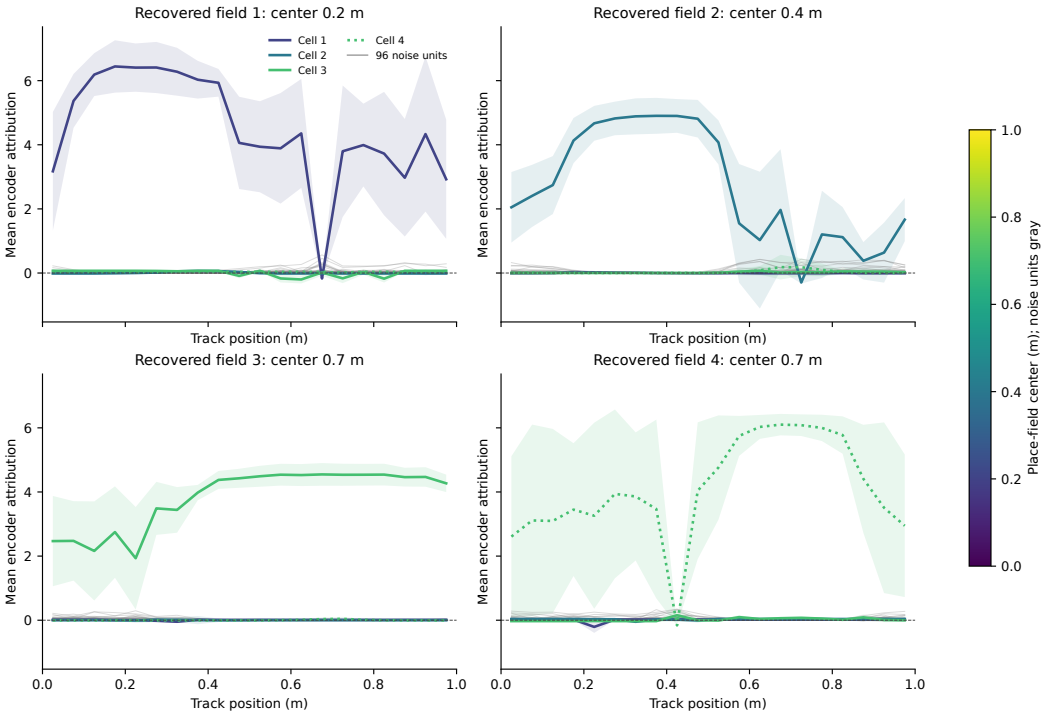


Figure S8: Source activity traceback for the spatial latents. One panel per recovered place field shows each unit's mean encoder attribution (standardized activity times encoder weight) versus track position. Curves average active-bin attributions within 20 position bins, then average equally across the selected runs with observations in each bin. Shaded bands for place cells indicate ± 1 sample standard deviation across contributing runs. All 100 units are retained: place cells use the field-center color scale, cell 4 is dotted, and the 96 noise units are gray.

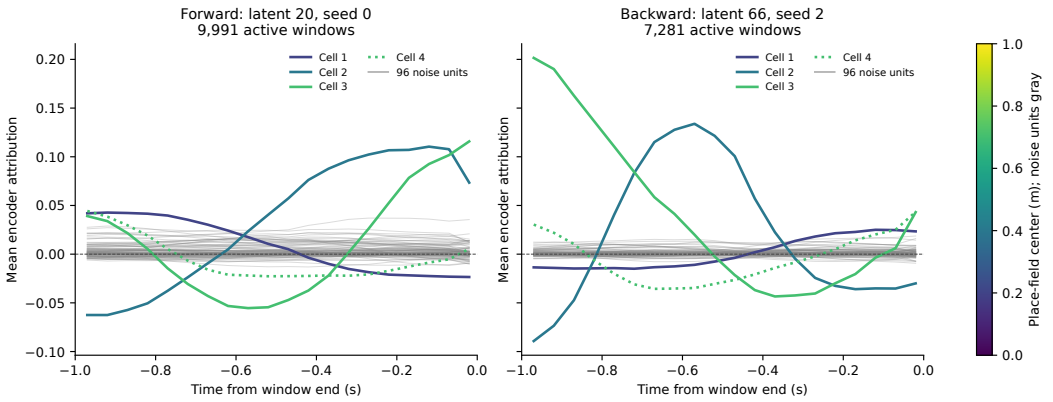


Figure S9: Source activity traceback over time for the two TW-SED latents. Each curve is one unit's mean encoder attribution (50-ms count times the gradient of the latent's pre-activation) at each time within active windows. Left: seed 0, latent 20, using all 9,991 positive-activation windows; right: seed 2, latent 66, using all 7,281. Windows contribute equally, without behavioral filtering or activation weighting. Place cells use the field-center color scale, cell 4 is dotted, and the 96 noise units are gray.

Direction index. To measure a latent's preference for forward or backward movement, we compare its mean activation in the two directions:

$$\mathrm{DI} = \frac{\bar{z}_{\mathrm{forward}} - \bar{z}_{\mathrm{backward}}}{\bar{z}_{\mathrm{forward}} + \bar{z}_{\mathrm{backward}} + 10^{-12}}. \tag{15}$$

DI ranges from -1 to 1: positive values indicate greater forward activation, negative values greater backward activation, and zero equal activation. Here, $\bar{z}_{\mathrm{forward}}$ and $\bar{z}_{\mathrm{backward}}$ are mean activations after matching the distribution of mean window position across directions, including windows in which the latent is inactive.

Matching position reduces differences caused by unequal sampling of locations. We group clean first-pair crossings by mean position into 0.025-m bins, retaining bins with at least ten windows in each direction. Within each bin, we calculate a mean activation for each direction. We then weight both means by the number of windows in the less frequent direction, giving greater influence to positions with more examples in both directions. The resulting mean for direction d is

$$\bar{z}_d = \frac{\sum_{j \in \mathcal{J}} w_j \bar{z}_{j,d}}{\sum_{j \in \mathcal{J}} w_j}, \qquad w_j = \min(n_{j,\mathrm{forward}}, n_{j,\mathrm{backward}}), \tag{16}$$

where $\mathcal{J}$ contains the qualifying position bins, $n_{j,d}$ is the number of windows in bin j for direction d, and $\bar{z}_{j,d}$ is their mean latent activation.

For clean crossings, eight position bins qualify, with weights summing to 209. To assess preference across the full track, we also analyze windows with no reversals and at least 0.1 m of net movement. The same procedure with wider, 0.1-m position bins retains eight bins, with weights summing to 939. Each total is the denominator for both directional means above. Both analyses use all windows in the retained bins, including overlapping windows, rather than only the illustrated examples.

Temporal feature selection. We apply the same selection criteria to FW-SED and TW-SED latents, assessing each movement direction separately. A passing latent must have a clean-crossing DI of at least 0.25 in the tested direction, decoder-center displacement of at least 0.01 m in that direction, and at least 20 active clean-crossing windows across both directions, with at least two matched position bins.

To measure decoder-center displacement, we weight the known centers of cells 1 and 2 by their positive squared decoder weights, then subtract the weighted center over the first six timebins from that over the last six. The pair's mean positive squared weight per entry must be at least twice that of the 96 noise units. Each cell must contribute at least 10% of the pair's total positive squared weight, and the pair at least 50% of that across all four place cells. We rank passing latents by the product of direction-signed DI, direction-signed displacement, and $\log(1 + \mathrm{enrichment})$, where enrichment

is the pair-to-noise ratio above. Decoder displacement describes a learned pattern, not a calibrated trajectory estimate.

Directional preferences. The two selected TW-SED latents show opposite preferences during clean crossings, with weaker preferences across the full track:

TW-SED latent	Seed	Crossing DI	Full-track DI	Preferred-direction examples Figure S6	Figure S7
20 (forward)	0	$+0.558$	$+0.118$	$15/16$	$16/16$
66 (backward)	2	-0.479	-0.199	$13/16$	$16/16$

Neither latent activates exclusively in its preferred direction. These results support preferences for particular traversals, rather than uniform direction tuning throughout the recording.

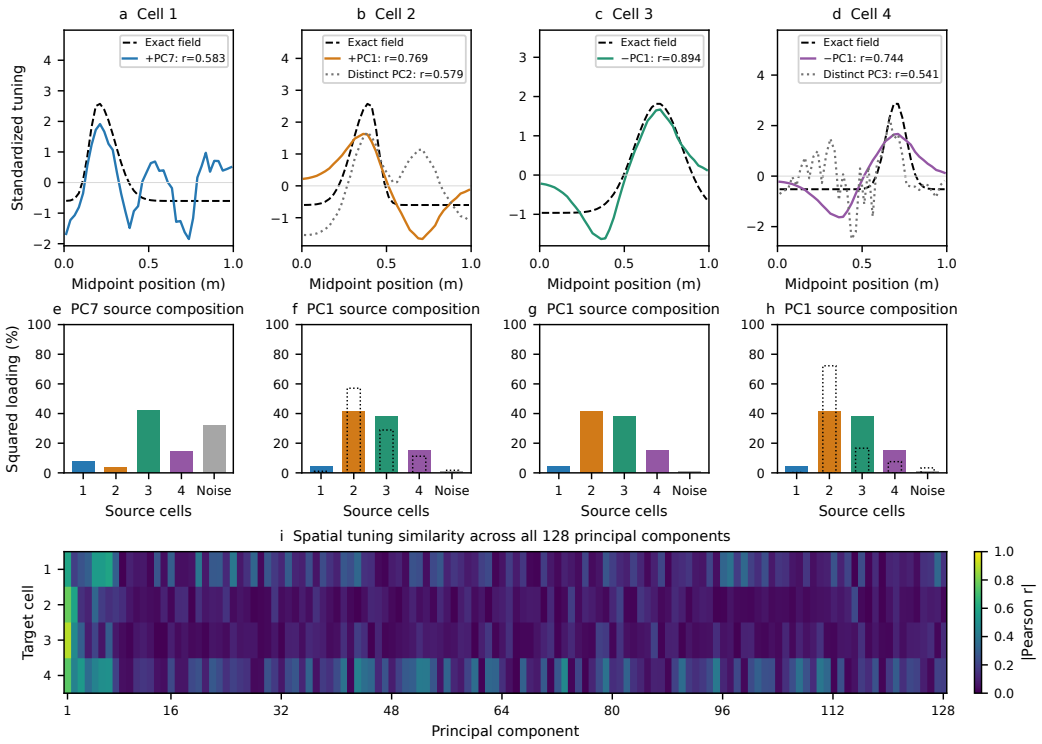


Figure S10: Individual PCs cannot isolate place fields. (a–d) Best independent PC matches to the four exact place fields. Each signed tuning curve and ground-truth curve is centered and scaled to unit standard deviation over spatial bins for display only. Gray dotted curves show the alternative when four distinct PCs are required. **(e–h)** Source composition of each best-matched PC, measured by squared loading. Dotted outlines show distinct-PC-assignment alternatives where applicable. **(i)** Absolute tuning correlations for all four place fields against each PC's spatial tuning curve.

Windowed PCA baseline. We fit PCA to the same complete 600-s recording and 1-s windows used by the W-SEDs, retaining 128 principal components to match their latent count, $D = 128$. Each window contains 20 timebins across 100 neurons, which we flatten into a vector of 2,000 values, in "time-then-neuron" order, preserving temporal order. Sliding the window forward in 50-ms steps gives 11,981 windows and an input matrix $X \in \mathbb{R}^{11,981\times 2,000}$, with one row per window. We mean-center each column without scaling its variance and fit PCA using a full singular value decomposition. The retained components explain 15.645% of the centered input variance. We use the resulting component scores for both spatial and directional analyses.

To describe each PC's spatial tuning, we assign each window a position by averaging the animal's position at its two central 10-ms samples. We divide the 1-m track into 40 equal spatial bins and average the signed PC scores of windows assigned to each bin:

$$T_{p,j} = \frac{1}{|\mathcal{W}_j|} \sum_{w \in \mathcal{W}_j} z_{w,p}. \tag{17}$$

Here, $z_{w,p}$ is the signed score of PC p in window w, $\mathcal{W}_j$ is the set of windows assigned to spatial bin j, and $|\mathcal{W}_j|$ is their number. The 40 values $T_{p,j}$ form the PC's spatial tuning curve.

We compare this PC spatial tuning curve with each known place field using absolute Pearson correlation. Taking the absolute value treats positively and negatively correlated patterns as equally strong matches, since a PC's sign is arbitrary. Computing $C_{i,p}$ for all four fields and 128 PCs gives the 4×128 correlation matrix in Figure S10i.

Independent best matches for place cells 1–4 are +PC7, +PC1, −PC1, and −PC1, with correlations 0.583, 0.769, 0.894, and 0.744. Here three targets reuse the same component; requiring four distinct components gives PCs 7, 2, 1, and 3, with correlations 0.583, 0.579, 0.894, and 0.541. The best

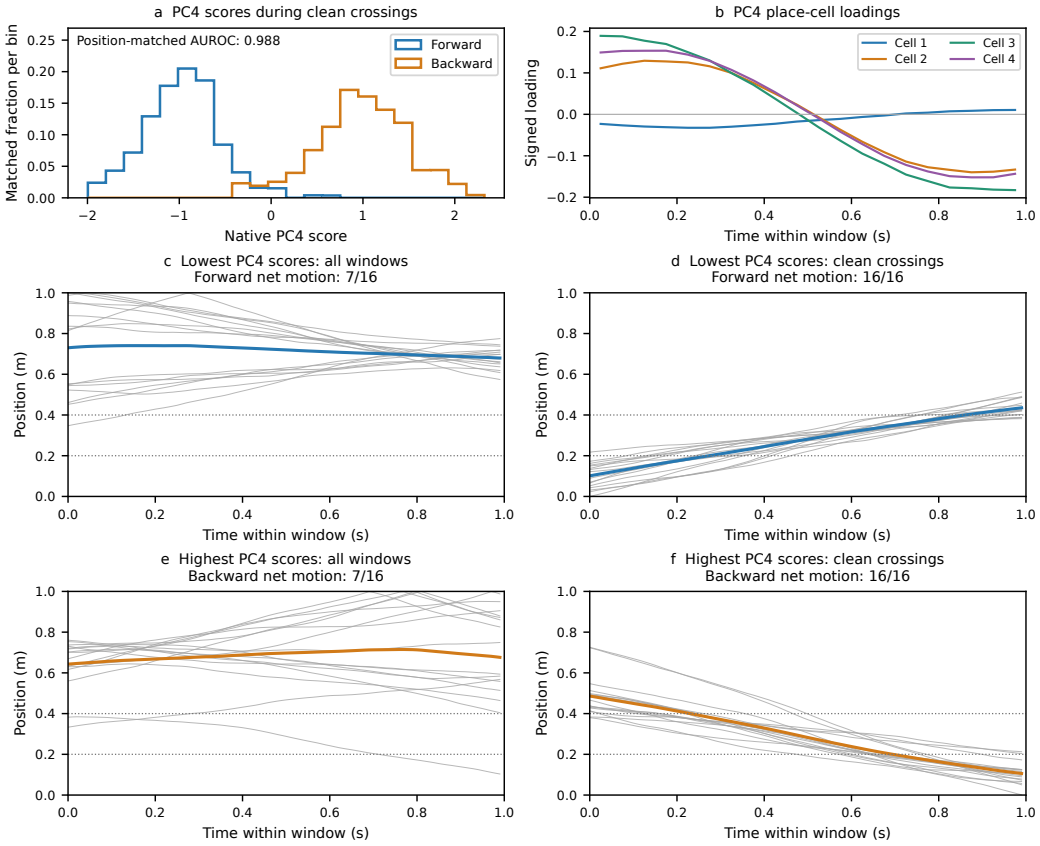


Figure S11: PC4 distinguishes traversal direction within clean crossings. (**a**) Native PC score distributions for forward and backward clean crossings, reweighted to match mean-position occupancy. Lower scores favor forward motion within crossings, and higher scores favor backward motion. (**b**) PC4's signed weights for the four place cells at each time within the window. Cells 2–4 have positive weights early and negative weights late, revealing an early-versus-late activity contrast shared across several place cells. **c–f:** The 16 lowest (c,d) or highest (e,f) score windows, selected either from the full recording (c,e) or only clean first-pair crossings (d,f). Gray curves show individual nonoverlapping trajectories; colored curves show their means.

spatial matches do not cleanly isolate any individual place field: they retain additional peaks, contrasts between locations, or mixed source contributions (Figure S10).

Interestingly, PC4 strongly distinguishes forward from backward movement among the same clean first-pair crossings used above (Figure S11). To compare directions at similar locations, we group crossings by mean position into 0.025-m bins, retaining bins with at least ten windows in each direction. Within each bin, we calculate how well the signed PC scores distinguish forward from backward crossings using $\mathbf{AUROC}$. We then average these values, weighting each bin by its less frequent direction count. This gives greater influence to locations with more examples in both directions and reduces confounding from differences in where forward and backward crossings occur. The weighted average is

$$A = \frac{\sum_{j \in \mathcal{J}} w_j \, \mathbf{AUROC}_j}{\sum_{j \in \mathcal{J}} w_j}, \qquad w_j = \min(n_{j,\text{forward}}, n_{j,\text{backward}}), \tag{18}$$

where $\mathcal{J}$ contains the qualifying position bins, $\mathbf{AUROC}_j$ is the direction AUROC within bin j, and $n_{j,\text{forward}}$ and $n_{j,\text{backward}}$ count its forward and backward windows. We report $\max(A, 1 - A)$ to account for PCA's arbitrary sign, using one sign orientation across all bins. PC4 achieves a position-matched $\mathbf{AUROC}$ of 0.988 after this averaging, and has lower native PC scores favoring forward crossings and higher PC scores favoring backward crossings.

6.4.2 MACAQUE MOTOR CORTEX SPIKE DATA IN AN ACTIVE REACHING TASK

Dataset. We analyze the 12 August 2009 recording of Nitschke, a male macaque (*Macaca mulatta*), in the Churchland–Shenoy reaching dataset (Churchland et al., 2012), using the Brainsets-prepared recording (Azabou et al., 2023). The recording has 191 retained Utah-array channels containing threshold-crossing spiking events from dorsal premotor and primary motor cortex; these are not 191 spike-sorted single cells. We retain 1,906 valid, correct reaches across 27 maze conditions. Of 3,038 source trials, 34 fail the validity flag and a further 1,098 are incorrect reaches. Channel identities and source trial indices are preserved in the data manifest; channels are not selected using model performance. This analysis is related to, but is not the official MC_Maze benchmark evaluation (Neural Latents Benchmark, n.d.).

Each trial is binned independently into 50-ms bins anchored at trial start. Zero-count bins are retained. The resulting matrix has 147,661 timebins and 191 channels, contains 53,625,900 spike threshold crossings, and covers 7,383.050 s of retained trial time. Only the final incomplete bin of each trial is omitted.

Neural inputs are not temporally smoothed. Behavioral samples within a bin are averaged and the result timestamped at its center. Hand kinematics are unavailable in 32,544 bins, principally because their recorded intervals end before the corresponding trial stops; these bins remain in neural training but are omitted from speed-specific measurements. Target labels remain available throughout each retained trial. Positions and target locations are expressed in mm, speed and velocity in mm/s, and speed slope in mm/s^2.

Models are fit without behavioral supervision using all retained neural inputs. We form 130,507 stride-one windows of $S = 10$ consecutive 50-ms bins, entirely within individual trials. Each scored representation corresponds to the final bin of its window and has access to exactly those ten bins.

Feature discovery. We define candidate features by inspecting latent activity in a reference NLDisco model with $D = 256$, alongside hand kinematics, target locations, and movement timing. Each feature description summarizes the behavioral condition associated with a latent's activation; we refine it when the latent responds to only part of a broader condition. We then use these same descriptions to assess recovery of all four features in CEBRA, LangevinFlow, and a freshly trained NLDisco model, each with a 192-dimensional latent space. The four descriptions are:

1. *Recent deceleration*: the ordinary-least-squares slope of speed over the last five bins (250 ms) is ≤ -75 mm/s^2.
2. *Fast leftward target reach*: endpoint speed is at least 500 mm/s, $v_x < 0$ & $|v_x| \geq |v_y|$, and the target is $(-144, -52)$ mm.
3. *Early leftward movement*: the movement endpoint is $[75, 150)$ ms after movement onset and its bin-averaged horizontal velocity $v_x < 0$.
4. *Late, target-specific reach*: the target is $(125, -18)$ mm and the movement endpoint is at least 250 ms after movement onset, in the direction towards the target.

For NLDisco, we use a two-level Matryoshka TW-SED with $(D_1, k_1) = (96, 8)$ and $(D_2, k_2) = (192, 16)$. Its causal, shift-equivariant Transformer has one layer, internal width 128, four attention heads, feedforward width 512, and no dropout. A learned projection maps the 128-dimensional hidden state to the 192 latent pre-activations. Channel counts are standardized using full-recording statistics. A linear temporal decoder reconstructs the ten-bin input with an identity output activation. Training uses equally weighted MSE loss across timebins and levels, Adam optimizer with initial learning rate 0.005, and the warmup/cosine schedule described above over ten shuffled epochs (1,280 updates, $B = 1,024$).

For CEBRA (version 0.4.0), we use the label-free temporal contrastive variant, CEBRA-Time, with 128 hidden units and a 192-dimensional output embedding. We use the official `offset10` convolutional encoder, cosine InfoNCE loss, and Adam optimizer with learning rate 3×10^{-4}. Channel counts are standardized using full-recording channel statistics. The 12-run sweep crosses output widths $\{128, 192, 256\}$, temperatures $\{0.3, 1\}$, and forward-positive lags $\{1, 5\}$ bins. For InfoNCE, positive endpoints must have complete ten-bin context within the same trial, negatives are sampled globally. With $B = 1,024$, 1,275 updates give 1,305,600 reference draws, approximately

ten dataset epochs in expectation. The sweep selects temperature 1 and lag 1; we retain these settings in the subsequent 192-dimensional fit used here.

For LangevinFlow (revision `68dcbfaa78b2371851842333ae933eea1f961832`), the selected model uses three 64-dimensional states—position, momentum, and GRU state—whose concatenation gives the 192-dimensional full latent space. We use the model's standard quadratic potential, underdamped transitions, Transformer/Poisson decoder, Adam optimizer with learning rate 0.003, solver step 0.01, damping 0.55, dropout 0.05, and coordinated dropout 0.5. Inputs are raw channel counts. Our adapter initializes from bin 0, consumes bins 1–9 once, and resets all states for each window. At extraction, dropout is disabled and Gaussian innovations are replaced by their means. In this implementation, position and momentum evolve autonomously after initialization, whereas the GRU state incorporates later observations. The 12-run sweep crosses state widths $\{32, 64, 128\}$, (corresponding to concatenated full latent space widths $\{96, 192, 384\}$), learning rates $\{0.001, 0.003\}$, and KL/weight-decay ramp durations $\{5, 500\}$ epochs. Each fit goes through ten shuffled epochs (1,280 updates, $B = 1,024$). At zero-based epoch e, the KL coefficient is $0.1 \min(e/r, 1)$ and weight decay is $2 \times 10^{-5} \min(e/r, 1)$ for ramp duration r.

For CEBRA and LangevinFlow, sweep runs are ranked by the mean best eligible-latent $\mathbf{AUROC}$ across the four descriptions. For CEBRA, each candidate latent is one coordinate of the output embedding; for LangevinFlow, it is one coordinate of the concatenated position, momentum, and recurrent states. We score these coordinates individually against each behavioral description using development trials. These dense coordinates may be sign-reversed so that larger values indicate a stronger association with the condition, and activation thresholds are calibrated as described below.

In the results below, we show only NLDisco's illustrative main-text latent for each of the features mentioned in the main-text, but show NLDisco's highest-$\mathbf{Sel}$ and highest-$\mathbf{AUROC}$ eligible latents for the two supplementary features, matching the eligible latents shown for CEBRA and LangevinFlow.

METRICS. We report $\mathbf{Cov}$, $\mathbf{Spec}$, $\mathbf{Sel}$, $\mathbf{AUROC}$, and single-latent and full-space $\mathbf{Decode}$ using the shared definitions in 6.3.4. An NLDisco latent counts as active when its activation is positive, $z > 0$. For CEBRA and LangevinFlow, we first choose each coordinate's sign separately for each feature: if its development $\mathbf{AUROC}$ is below 0.5, we multiply its values by -1, so that larger values indicate a stronger association with the condition. We then set an activation threshold so that approximately the highest r fraction of its values across all eligible development windows count as active.

For each feature, we calculate r as the fraction of eligible development windows in which a latent from the reference NLDisco model had positive activation, including windows both inside and outside the behavioral condition. This gives $r = 31.921\%$ for recent deceleration, 0.793% for fast leftward target reach, 1.473% for early leftward movement, and 2.401% for late, target-specific reach. The aim is to compare baseline coordinates with the reference latent at approximately the same overall activation frequency. The rates differ because the reference latents activate with different frequencies; BatchTopK does not require equal activity across latents. These frequencies can reflect how often and for how long a behavior occurs, activation outside the described condition, and differences in which windows are eligible for each feature. Thus, r measures the reference latent's activity, not the prevalence of the behavioral condition. We retain these original rates, so the baseline activation frequencies are not matched to the current NLDisco-192 latents.

To be eligible for selection, a latent must activate in at least 5% of windows satisfying the behavioral condition, and do so in at least 20 distinct trials. We select the eligible latents with the highest $\mathbf{Sel}$ and highest $\mathbf{AUROC}$, except where we retain NLDisco's main-text illustrative latent as described above.

The two supplementary features illustrate the difference between a selective detector and a broadly discriminative latent (Figure S12). For each, NLDisco's best-$\mathbf{Sel}$ latent activates more narrowly, while its best-$\mathbf{AUROC}$ latent covers a larger fraction of condition-positive windows. Movement-aligned curves use 50-ms bins around movement end or 25-ms bins around movement onset, with at least 20 observations per plotted cell. Feature scores use all eligible development examples.

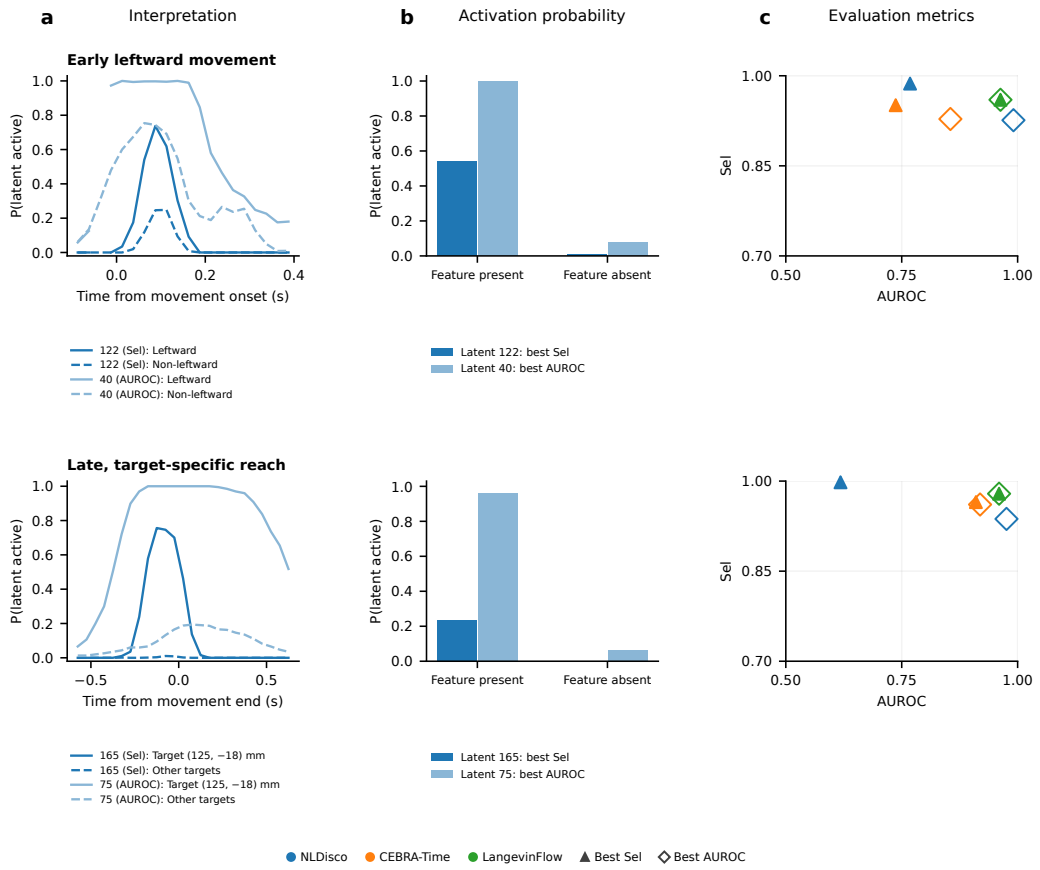


Figure S12: *Early-leftward* and *late-reaching* features. (**a**) Activation probabilities for NLDisco's best-**Sel** and best-**AUROC** latents, aligned to movement onset or end and separated by direction or target. (**b**) Their activation probabilities inside and outside each condition. (**c**) **Sel** versus **AUROC** for the two selected latents from each of the three 192-dimensional representations; coincident selections are overlaid. Scores use all valid development windows, including those outside the displayed timing ranges.

Method / selector	Latent	**Cov**	**Spec**	**Sel**	**AUROC**	Single Decode	All Decode
Recent deceleration							
NLDisco	83	0.6708	**0.9532**	**0.924**	0.868	0.7682	**0.9176**
CEBRA-Time (best **Sel**)	126	0.6976	0.7674	0.749	0.818	0.7291	0.8995
CEBRA-Time (best **AUROC**)	126	0.6976	0.7674	0.749	0.818	0.7291	0.8995
LangevinFlow (best **Sel**)	136	**0.9089**	0.8158	0.831	0.908	**0.8620**	0.9021
LangevinFlow (best **AUROC**)	178	0.8918	0.8119	0.825	**0.921**	0.8497	0.9021
Fast leftward target reach							
NLDisco	20	**0.9914**	0.9748	**0.975**	**0.989**	**0.9814**	**0.9946**
CEBRA-Time (best **Sel**)	93	0.1830	0.9935	0.965	0.785	0.7184	0.9890
CEBRA-Time (best **AUROC**)	56	0.1124	0.9929	0.940	0.908	0.8268	0.9890
LangevinFlow (best **Sel**)	179	0.2522	**0.9940**	**0.975**	0.979	0.9152	0.9578
LangevinFlow (best **AUROC**)	179	0.2522	**0.9940**	**0.975**	0.979	0.9152	0.9578
Early leftward movement							
NLDisco (best **Sel**)	122	0.5422	**0.9931**	**0.987**	0.768	0.7556	**0.9924**
NLDisco (best **AUROC**)	40	**0.9977**	0.9206	0.926	**0.991**	**0.9726**	**0.9924**
CEBRA-Time (best **Sel**)	93	0.2289	0.9884	0.951	0.737	0.6652	0.9457
CEBRA-Time (best **AUROC**)	54	0.1641	0.9875	0.928	0.855	0.7549	0.9457
LangevinFlow (best **Sel**)	179	0.2672	0.9890	0.960	0.963	0.9260	0.9815
LangevinFlow (best **AUROC**)	179	0.2672	0.9890	0.960	0.963	0.9260	0.9815
Late, target-specific reach							
NLDisco (best **Sel**)	165	0.2369	**0.9996**	**0.998**	0.618	0.6202	**0.9894**
NLDisco (best **AUROC**)	75	**0.9624**	0.9354	0.937	**0.976**	**0.9510**	**0.9894**
CEBRA-Time (best **Sel**)	142	0.3937	0.9859	0.965	0.910	0.7703	0.9778
CEBRA-Time (best **AUROC**)	44	0.3675	0.9852	0.961	0.919	0.8029	0.9778
LangevinFlow (best **Sel**)	136	0.5149	0.9891	0.979	0.960	0.8932	0.9741
LangevinFlow (best **AUROC**)	136	0.5149	0.9891	0.979	0.960	0.8932	0.9741

Table S2: Feature recovery and decoding for all four behavioral descriptions. **Cov**, **Spec**, **Sel**, and **AUROC** use development windows. Single-latent and full-space **Decode** report test balanced accuracy with validation-selected decision thresholds. NLDisco uses its illustrative latents for the two main-text features and both selectors for the supplementary features; both selectors are shown for each baseline. Repeated coordinates reflect coincident selections. Column maxima within each feature are bolded, including ties at the displayed precision.

The combined scores (Table S2) show why these metrics are useful together. The main-text deceleration latent has higher **Spec** and **Sel** than the baselines, but lower **Cov**; for fast leftward target reaches, it covers nearly all positive examples. For the supplementary features, NLDisco's best-**Sel** latents are more selective than either baseline, while its best-**AUROC** latents provide stronger discrimination. NLDisco has the highest full-space decoding score for all four descriptions. Its shown single latents lead fast-reach, early-leftward, and late-reaching decoding, while LangevinFlow leads single-latent deceleration decoding.

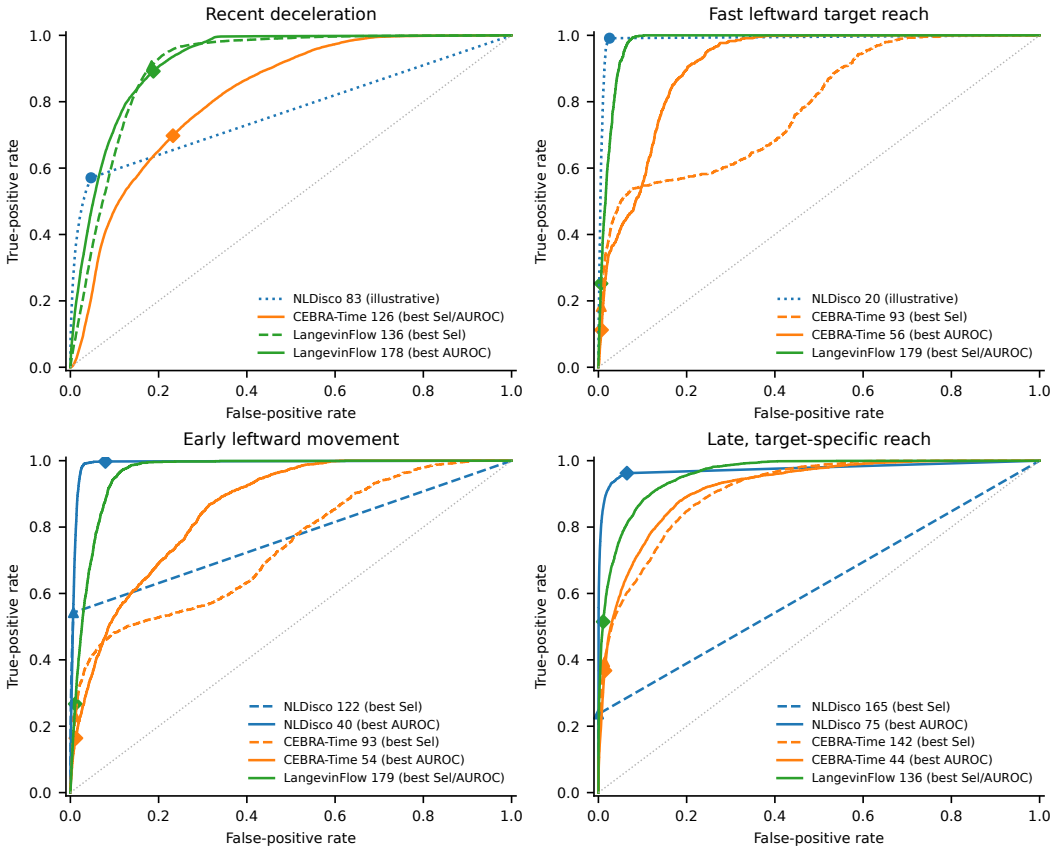


Figure S13: Discriminability of the selected feature latents. ROC curves for the four behavioral descriptions in the three 192-dimensional representations. NLDisco uses its main-text illustrative latent for *recent deceleration* and *fast leftward target reach*, and its best-**Sel** and best-**AUROC** latents for the supplementary features. Baselines use both selectors. Curves use the same development windows as the association scores in Table S2.

SOURCE ACTIVITY TRACEBACK. We apply the source activity traceback to NLDisco latents for all four features, using the same selections as in Table S2: the illustrative latents for the main-text features and both the best-**Sel** and best-**AUROC** latents for the supplementary features. Inputs are channel counts standardized using the same full-recording statistics as training. We retain all ten timebins and 191 channels, averaging attribution maps equally over development windows in which each latent is active. The corresponding signed decoder weights show how these latents contribute to reconstruction (Figure S14).

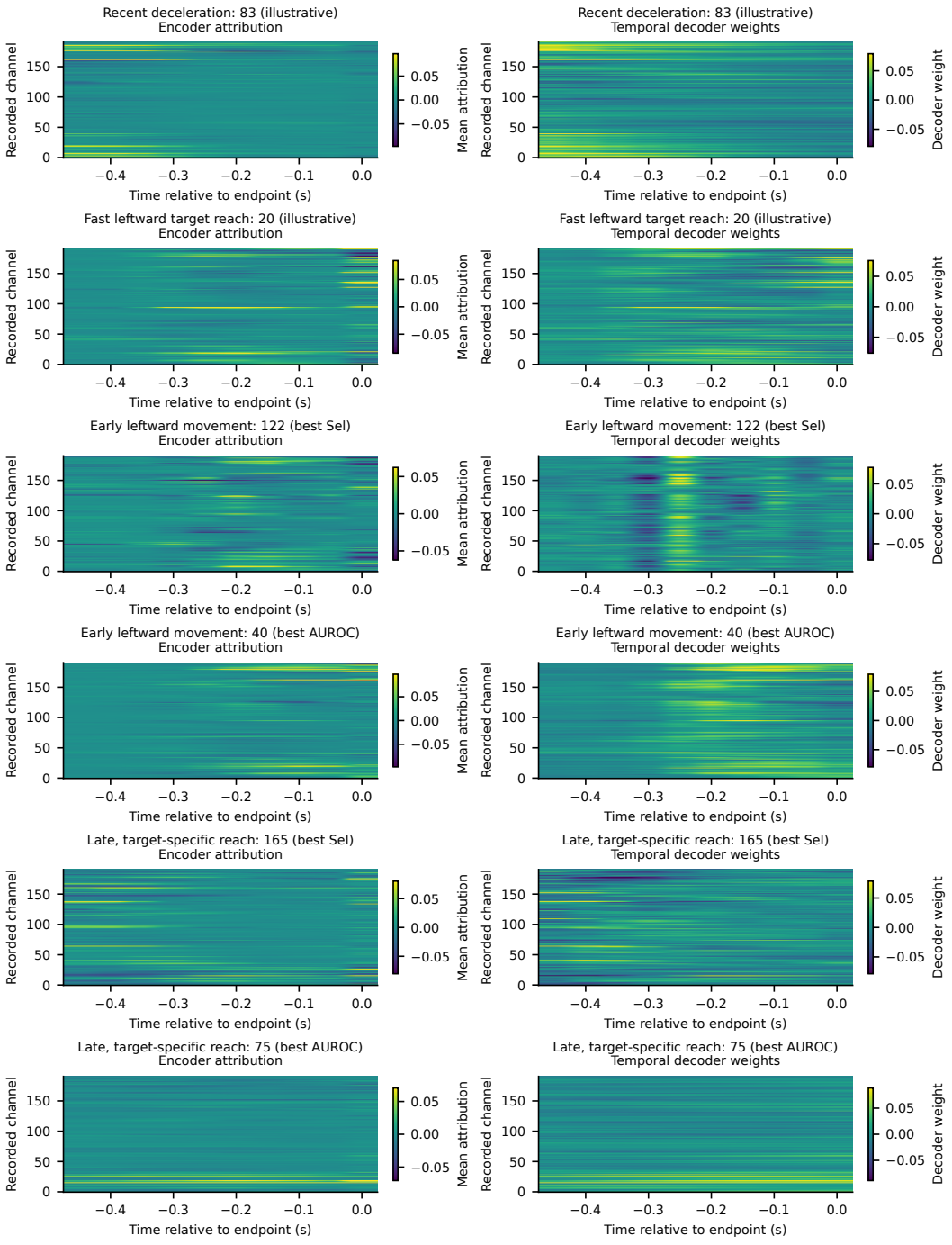


Figure S14: Source activity traceback for all four behavioral features. Rows follow the feature order in Table S2: *recent deceleration* (illustrative latent 83), *fast leftward target reach* (illustrative latent 20), *early leftward movement* (best-**Sel** latent 122 and best-**AUROC** latent 40), and *late, target-specific reach* (best-**Sel** latent 165 and best-**AUROC** latent 75). All latents are from seed 1. Left: mean encoder attribution over development windows with positive latent activation. Right: temporal decoder weights. All 191 recorded channels and ten input timebins are retained. Colors are clipped symmetrically at each panel's 99th absolute percentile for readability.

Methods' training time comparison. We compare the fixed 192-dimensional NLDisco, CEBRA-Time, and LangevinFlow configurations used in Section 4.2 on the same NVIDIA H100 80-GB GPU. Each method is trained from scratch five times (seeds 0–4), for 15 sequential fits in separate processes. All fits use the same 130,507 complete within-trial windows, each containing ten 50-ms bins from 191 channels, with batch size 1,024. NLDisco and LangevinFlow make ten exhaustive shuffled passes (1,280 updates). CEBRA-Time samples temporal-positive reference pairs and negatives with replacement: its ten epoch-equivalents comprise 1,275 updates and 1,305,600 reference draws, rather than exhaustive passes through the windows. For this dataset, GPU memory (VRAM) usage remained below 5 GB throughout all 15 fits across the three methods.

Wall-clock timing is synchronized with CUDA immediately before and after optimization. It includes minibatch preparation, device transfers during training, loss checks and the trainer's local progress bookkeeping; it excludes process startup, initial data loading/preprocessing, model and optimizer construction, checkpoint writing, and latent export. Each fit starts in a fresh process, without discarded training warm-up updates. External experiment tracking is disabled. All fits use four CPU threads and PyTorch 2.5.1 with CUDA 12.1. GPU process occupancy was sampled every two seconds; no competing compute process is observed on the benchmark GPU. Other GPUs on the host were not reserved.

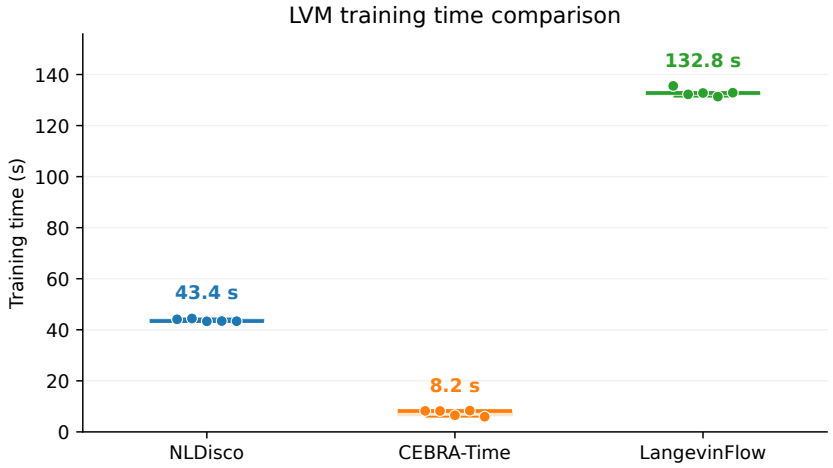


Figure S15: LVM training time comparison. All three models expose a **192-dimensional latent representation**: NLDisco's largest dictionary (with a nested 96-dimensional level), CEBRA-Time's embedding, and LangevinFlow's concatenated position, velocity and recurrent states (64 dimensions each). Total/trainable parameter counts, excluding buffers, are NLDisco, 614,859/614,859; CEBRA-Time, 270,784/270,784; LangevinFlow, 449,199/449,199. Each dot is one fresh fit; five seeds (0–4) per method give 15 fits run sequentially on the same H100 80-GB GPU, with batch size 1,024 and four CPU threads. All five observations per method are shown.

6.4.3 Mouse spike data in an ethological foraging assay

Dataset. We analyze an 18-hour interval from an Aeon experiment, which records neural activity alongside spontaneous behavior in a naturalistic foraging environment (Campagner et al., 2025). The recording is from subject `BAA-1104292` in `social-ephys0.1-aeon3`, from June 6, 2024, 23:00 to June 7, 17:00. It contains 56,191,057 spikes from 90 Kilosort 4 clusters across retrosplenial cortex and superior colliculus. We retain all 90 clusters.

Spikes are binned at 20 ms after aligning the neural acquisition and behavioral clocks. We exclude bins with unsupported synchronization or clock anomalies and require complete neural support throughout each input window. This leaves 3,238,600 overlapping windows of $S = 20$ bins, each spanning 400 ms. Unsmoothed counts are standardized per cluster using all valid neural bins.

An overhead camera provides centroid position and segmented image area; the foraging patch provides wheel rotation and food-delivery events. Centroid velocity uses a causal 100-ms position average followed by a 100-ms backward difference. Wheel speed is the absolute net rotation per

bin, averaged over the preceding 100 ms and converted to cm/s speed using the 4-cm wheel radius. Camera analyses exclude missing tracking data.

Feature discovery. We inspect latent activity alongside wheel movement, position, segmented area, and movement direction to identify candidate features. The nine-model sweep crosses $D \in \{128, 192, 256\}$ and $k \in \{8, 12, 16\}$, using two Matryoshka levels, $(D/2, k/2)$ and (D, k). Each model uses a causal, shift-equivariant TW-SED with one Transformer layer, internal width 128, four attention heads, feedforward width 512, and no dropout. A learned projection maps the final token to the latent dictionary, and a linear temporal decoder reconstructs the preceding 400-ms window with an identity output activation. Training uses 5,000 Adam updates with $B = 1,024$, initial learning rate 0.005, the warmup/cosine schedule described above, and equally weighted reconstruction losses across timebins and levels. All fits use seed 0; no behavioral labels enter training.

The displayed latents are selected by inspecting behavioral tuning, high-activation episodes, and source activity across the sweep. Feature descriptions are refined during this exploration and then fixed for scoring:

1. *Wheel activity*: endpoint wheel-rim speed exceeds 0.75 cm/s. All input bins must have valid wheel measurements; proximity to the patch is not required. This is the main-text feature, illustrated by latent 52 of the $D = 192$, $k = 12$ model.
2. *Smaller segmented area near the patch*: endpoint area lies in the lowest quintile, $101.5 \leq a < 512.5$ px^2, among camera-valid windows wholly within 100 px of the patch. The comparison is therefore between smaller and larger segmented areas near the patch. Area is an image measurement, not a verified posture or feeding label.
3. *Highest centroid-speed quintile*: endpoint centroid speed lies in the highest quintile of speed-valid windows, beginning at 9.837 px/s. This is a relative speed preference within the recording, rather than a threshold for fast running.
4. *Radially outward movement*: among camera-valid windows with endpoint centroid speed at least 100 px/s, movement heading is within 45° of the outward radial direction from the arena center. Other fast-moving windows form the comparison population, allowing us to assess direction preference within fast movement.

The supplementary features use latents 107, 109, and 10, respectively, from the $D = 256$, $k = 12$ model. All latent indices are zero-based.

METRICS. We report **Cov**, **Spec**, **Sel**, and **AUROC** using the shared definitions in 6.3.4. A latent is active when $z > 0$ (for *wheel activity*, $z \geq 3.31$, the operating point marked in Figure 5b), and **AUROC** uses its continuous activation values. Scores include every eligible window inside and outside each condition, rather than only the examples used for visualization (Table S3). Because windows overlap, their counts do not represent independent observations.

The area and speed tuning plots show activation probability within five empirical quintiles (Figure S16). The directional plot instead groups fast-moving windows by heading relative to the outward radial direction (Figure S17). Together, these plots show how the selected latents' activity varies around the scored conditions.

We evaluate each feature using the shared logistic-regression protocol, fitting separate decoders to the selected latent and all D latents from its model. Both use the feature's eligible comparison population. The recording is divided chronologically into 80/10/10 training, validation, and test intervals, excluding windows that cross partition boundaries. Standardization and decoder weights use training examples only; validation selects regularization and one probability threshold per decoder, both fixed for testing. The single-latent wheel decoder uses $\lambda_{\text{dec}} = 1000$ and $t = 0.034469$, achieving test **Decode** 0.889. This decoder threshold differs from the latent-activation threshold used for the wheel association scores ($z \geq 3.31$), which gives **TPR** 0.938 at **FPR** 0.180 (Figure 5b).

BEHAVIORAL CONTROLS. To test whether the *wheel activity* feature reflects simply being at the patch, we compare wheel-active and wheel-inactive windows while restricting both groups to camera-valid windows wholly within 100 px of the patch. The association remains (**Sel** 0.698, **AUROC** 0.842). We then compare the two states within 25-px-square position cells containing at least 20

Method	D/k	Latent	**Cov**	**Spec**	**Sel**	**AUROC**	Single Decode	All Decode
Wheel activity								
NLDisco	192/12	52	0.9383	0.8200	0.839	0.924	0.8890	0.9328
Smaller area near the patch								
NLDisco	256/12	107	0.8367	0.7607	0.778	0.849	0.7989	0.7937
Highest centroid-speed quintile								
NLDisco	256/12	109	0.7180	0.7814	0.767	0.778	0.7118	0.7949
Radially outward movement								
NLDisco	256/12	10	0.8400	0.8051	0.812	0.872	0.8550	0.9005

Table S3: Feature recovery and decoding for the four behavioral descriptions. Each row group reports the selected NLDisco latent for one feature. **Cov**, **Spec**, and **Sel** use native positive activation, except for *wheel activity*, which uses the $z \geq 3.31$ operating point in Figure 5b; **AUROC** uses continuous activation values. Association scores use all eligible windows in the feature-specific population. Single and All **Decode** report test balanced accuracy using the selected latent or all D latents from its model, respectively, with a separately validation-selected decision threshold for each decoder.

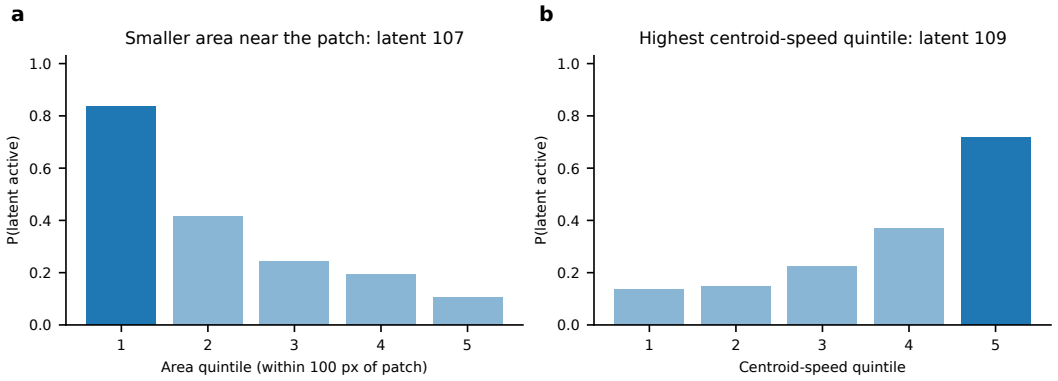


Figure S16: *Smaller area* and *higher speed* features. Activation probabilities for NLDisco latents 107 and 109, respectively, in the $D = 256$, $k = 12$ model. Bars group eligible windows into empirical quintiles; dark bars identify the scored conditions. Area quintiles compare camera-valid windows wholly within 100 px of the patch. Speed quintiles use all windows with valid centroid-speed measurements.

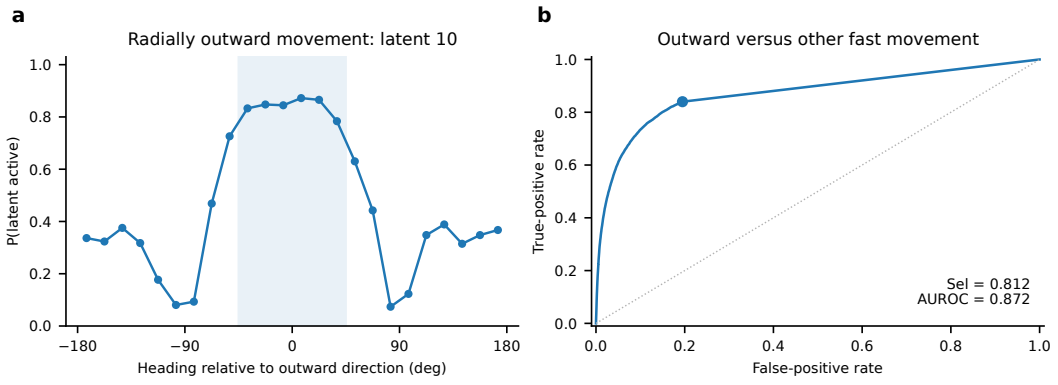


Figure S17: *Radially outward movement* feature. (**a**) Activation probability for NLDisco latent 10 ($D = 256$, $k = 12$), grouped by heading relative to the outward radial direction during movement at ≥ 100 px/s. Shading marks the scored $\pm 45^\circ$ sector. (**b**) ROC curve comparing outward with other fast movement; the marker indicates native positive activation. Camera measurements must be valid throughout each input window.

windows of each state. Five cells cover 99.98% of near-patch positive windows. Averaging the within-cell activation rates and AUROCs with weights proportional to each cell's positive-window count gives position-adjusted **Sel** 0.617 and within-cell **AUROC** 0.755. These checks support an association beyond coarse patch location, while leaving body orientation and movements required to spin the wheel unresolved. Two six-minute excerpts show the distinction directly: the latent activates during wheel bouts but remains largely silent during extended patch occupancy without wheel movement (Figure S18).

The *radially outward* association also persists after coarse control for speed and position. Within fast-moving windows, we group endpoints into speed bins $[100, 150)$, $[150, 200)$, $[200, 300)$, and $[300, \infty)$ px/s and 100-px-square position cells. Groups with at least 20 outward and 20 other-direction windows cover 91.6% of outward examples. Weighting each group's activation rates and AUROC by its positive-window count gives adjusted **Sel** 0.710 and within-group **AUROC** 0.833. These results support direction preference beyond coarse speed and location. They do not establish movement toward a particular gate: the recording lacks a calibrated gate location, and an exploratory entrance-location proxy yields **AUROC** 0.422 for this latent. We therefore retain the directly measured radial description.

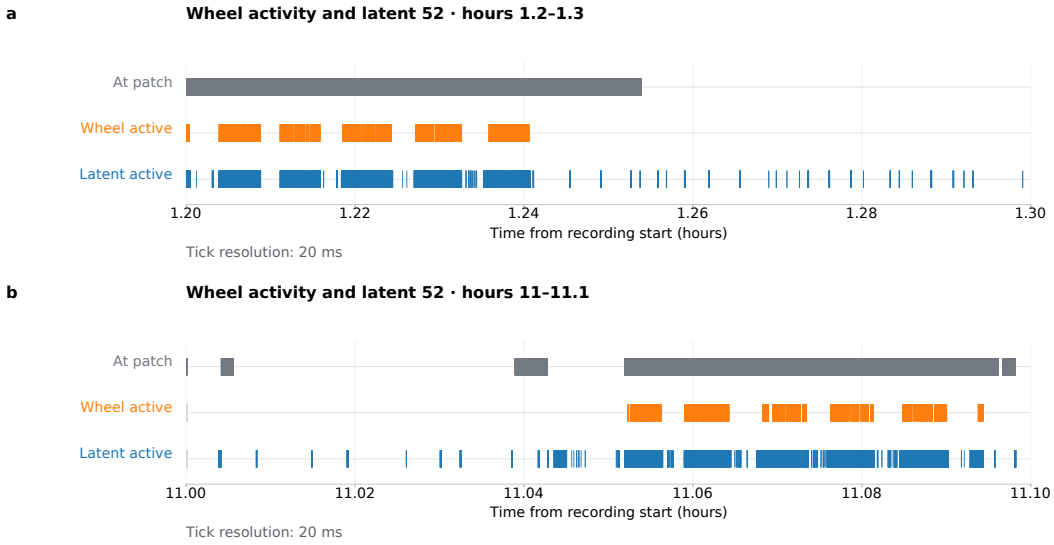


Figure S18: Wheel activity and patch occupancy. Two excerpts from hours 1.2–1.3 (**a**) and 11–11.1 (**b**) of the same recording. Rows show 20-ms bins with centroid position within 100 px of the patch, wheel-rim speed above 0.75 cm/s, and positive activation of latent 52 ($D = 192$, $k = 12$). Invalid observations are excluded. Patch occupancy without wheel movement often leaves the latent silent, whereas wheel bouts coincide with activation in both excerpts.

SOURCE ACTIVITY TRACEBACK. We apply the source activity traceback to the four selected latents using the same standardized counts as in training. Attribution maps retain all 90 recorded clusters and all 20 input timebins. We average equally over neural-valid windows with positive latent activation, regardless of behavioral condition. The wheel feature's time-averaged profile provides a compact view of its source contributions (Figure S19); the complete maps show when each cluster contributes within the input window, alongside the decoder weights describing reconstruction (Figure S20). These are model attributions, not causal effects or evidence of biological excitation or inhibition.

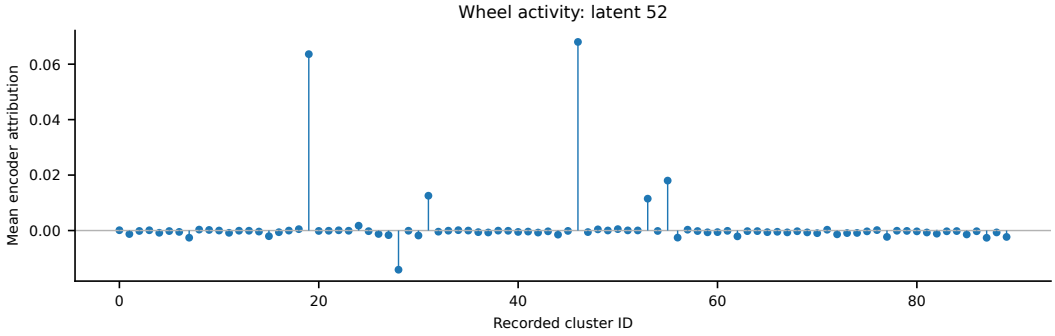


Figure S19: Source activity traceback for the *wheel activity* feature. Mean encoder attribution for latent 52 ($D = 192$, $k = 12$), averaged over the 20 input timebins and all neural-valid windows with positive latent activation. Each point identifies one recorded cluster; all 90 are shown. The full temporal map appears in Figure S20.

Across the Churchland and Aeon analyses, neural preprocessing and representation fitting use all retained neural inputs. Churchland feature descriptions use decoder-training and validation trials, whereas Aeon feature discovery and association scoring use the full recording. Decoder weights and decision thresholds use their designated training and validation partitions, but the resulting scores assess decoding within these recordings rather than independent validation of the full discovery procedure. The reported associations are exploratory and do not establish exclusive encoding or causal function.

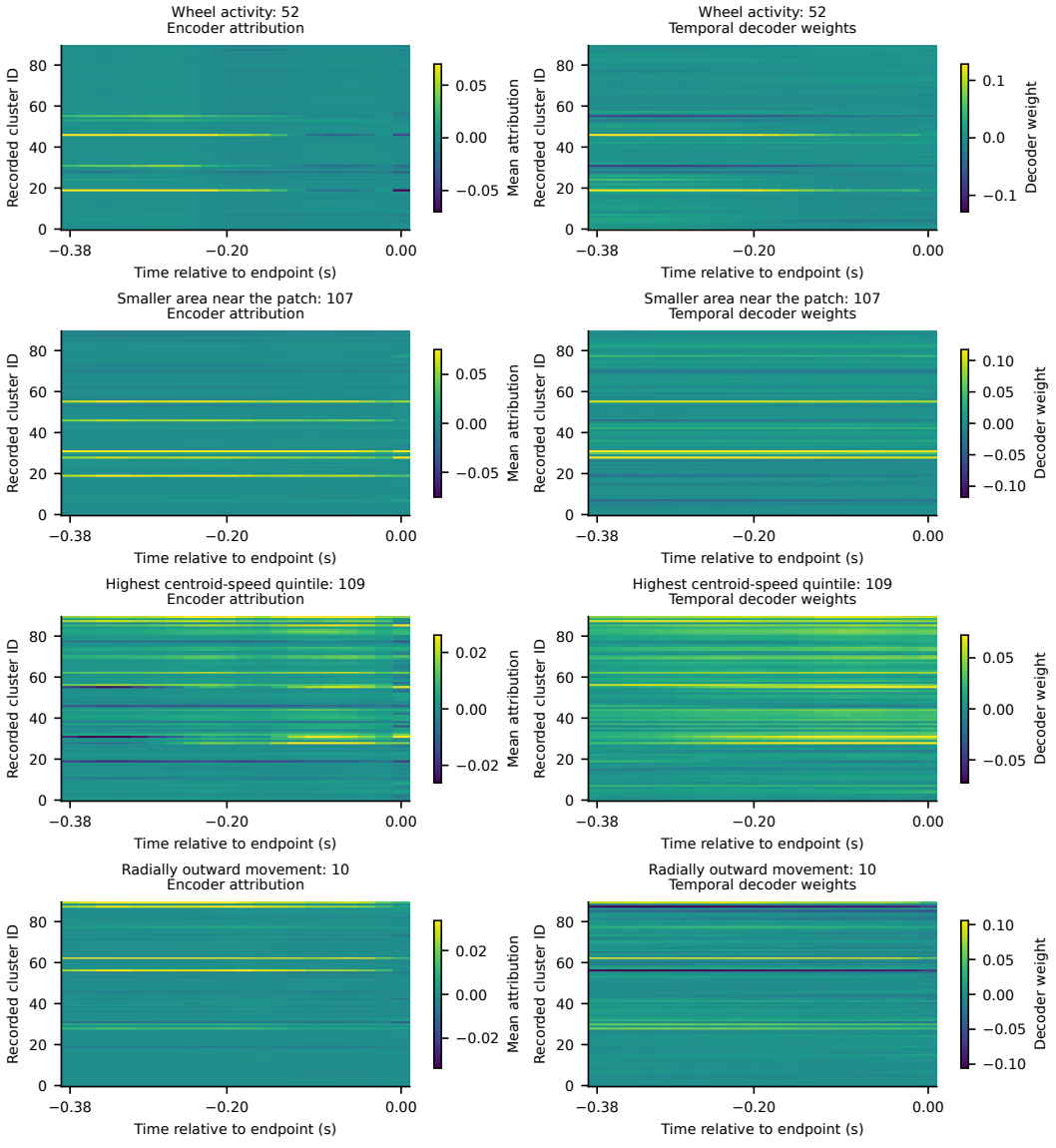


Figure S20: Source activity traceback for all four behavioral features. Rows follow the feature order in Table S3: wheel activity (latent 52), smaller area near the patch (107), highest centroid-speed quintile (109), and radially outward movement (10). Left: mean encoder attribution over neural-valid windows with positive latent activation. Right: temporal decoder weights. All 90 recorded clusters and 20 input timebins are retained. Colors are clipped symmetrically at each panel's 99th absolute percentile for readability.