R-NDT: ROBUST NEURAL DATA TRANSFORMER FOR GENERALIZED CAUSAL MOTOR DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Intracortical brain motor decoding aims to interpret and translate neural spike activity into behaviors. Causal decoding models that can generalize across variations, such as sessions, behaviors, subjects, and recording sites, are critical for real-world brain-computer interface (BCI) applications. However, every recording session presents the decoder with a new set of neural units and a continuous stream of nonstationary population spike activity. In this work, we present *R-NDT*, a robust neural data transformer that is designed to adapt to new recording sessions without need to re-learn unit/session embeddings, and it decodes long-term recordings in a streaming causal manner. Specifically, 1) unit identity is inferred from a few seconds of unlabeled calibration activity by a permutation-equivariant encoder to weight units in a population read-in; 2) a sliding-window causal transformer processes spike count bins with a bounded key-value cache; 3) a moving-average state conditions every block through adaptive layer norm; and 4) behavior covariates read the latent state through cross-attention. We pretrained R-NDT using several public datasets. After pretraining, we evaluated R-NDT generalization by fine-tuning it on cross-session, cross-type, cross-subject, and cross-site downstream behavior decoding tasks. Our results show that, on average across 20 sessions, R-NDT outperforms the decoding performance of existing decoding models, which has implications for future BCI decoders.

1 INTRODUCTION

An intracortical brain-computer interface (BCI) aims to decode the intended movement from the spiking patterns of a neural population. Performance of decoding models is typically dependent on careful pairing between well-structured neural-behavior activity recorded from stable microelectrode arrays and stereotyped behavior activities. However, in practice, every recording session, for example, a daily session, usually records from a different set of neural units. This creates a situation where a decoder trained on one day has limited generalization on the next. Meanwhile, the recorded neural activity is typically nonstationary (Chestek et al., 2011), which requires the decoder to operate with windows over longer periods of time, e.g., minutes, resulting in a mode that is session-based. Ideally, motor decoding is aimed to be in continuous time, i.e. causal and streaming, such that each prediction of the decoder uses only the past instances, and the recording arrives as one uninterrupted stream rather than as a set of trials. Recent neural transformers such as NDT2, SPINT, POYO, POSSM, and MOJO (Ye et al., 2023; Le et al., 2025; Azabou et al., 2023; Ryoo et al., 2025; Mao et al., 2026) have made significant progress toward these requirements. However, three gaps still remain for future practical BCI.

The first gap is the generalization over sessions (*cross-session generalization*). Works have proposed to amend it by either a dedicated model for per-session alignment (Zhang et al., 2025b) or a per-unit embedding table (Azabou et al., 2023). Both solutions have limitations, since the first requires a separate supervised trained model and the second grows linearly with the number of sessions. Recently, SPINT (Le et al., 2025) proposed an alternative approach which inferred unit identity from a short unlabeled calibration segment within a shared network. Such an approach circumvents the need for separate alignment; however, the proposed model does not allow pretraining since it is a supervised decoder that requires end-to-end training. The second gap is in the discrepancy between the time scale of the evaluation and the time scale of the decoder. Evaluation is performed on a trial-by-trial basis, which is much shorter time scale than session-by-session basis. Trials last a few

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.

	Infer unit identity from spikes	Flexible unit size	Causal streaming	State across windows	Label-free pretraining
RNN, LSTM, GRU, Mamba	✗ (channel order)	✗	✓	✓	✗
LFADS Pandarinath et al. (2018)	✗ (channel order)	✗	✗	✓	✓
NDT-2 (Ye et al., 2023)	✗ (channel order)	✗ (patches)	✓	✗	✓
POYO (Azabou et al., 2023)	✗ (learned table)	✓	✗	✗	✗
POSSM (Ryoo et al., 2025)	✗ (learned table)	✓	✓	✓	✗
BIT (Zhang et al., 2025b)	✗ (channel order)	✗ (patches)	✗	✗	✓
SPINT (Le et al., 2025)	✓ (calibration)	✓	✓	✗	✗
R-NDT (ours)	✓ (calibration)	✓	✓	✓	✓

Table 1: Design components of spike decoders. *Infer unit identity from spikes*: the representation of a unit is computed from its own spike activity. *Flexible unit size*: input tensor whose shape is not tied to a particular session population. *Causal streaming*: every prediction uses only past spikes. *State across windows*: information is carried beyond the attention window at constant memory. *Label-free pretraining*: the self-supervised pretraining objective uses spikes only.

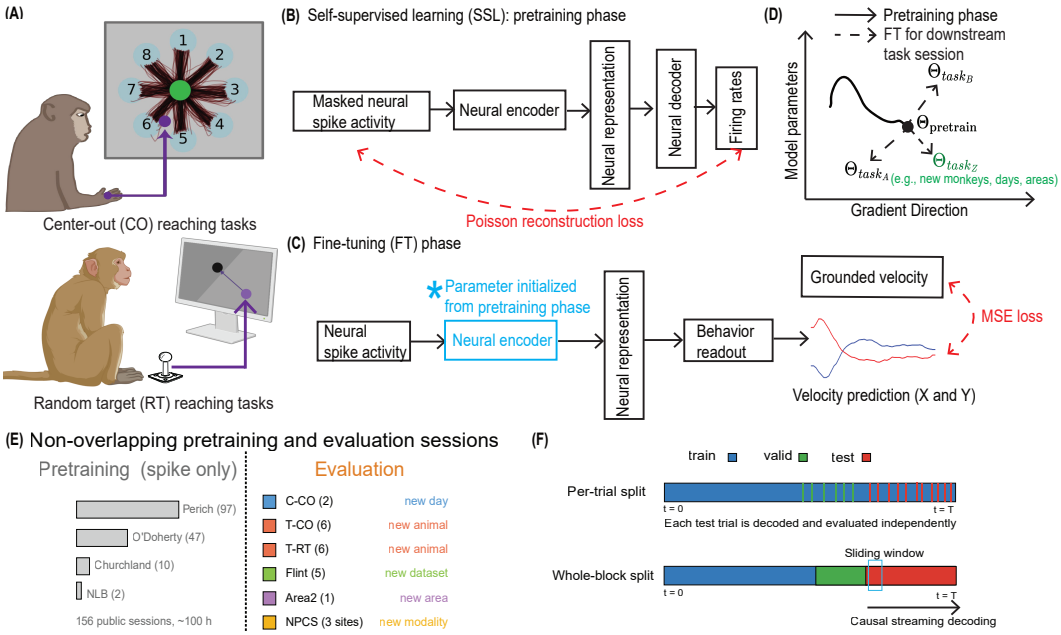


Figure 1: Overall illustration of the pretraining and fine-tuning workflow for motor decoding. (A) Typical examples of non-human primates performing motor tasks. (B) and (C) General schemes for SSL and FT, respectively. (D) Illustration of model adaptation during pretraining and fine-tuning. (E) Data used for pretraining and evaluation, respectively. (F) Two causal evaluation protocols, trial-based causal decoding and whole block-based streaming decoding.

seconds and any recurrent state is reset between them (Ryoo et al., 2025), whereas a deployed decoder runs on a continuous, minutes long stream in which the state is accumulating. The third gap is that unsupervised label-free pretraining of a decoder backbone has yet to be shown to perform better than supervised pretraining under an identical fine-tuning protocol on same sessions (Ye et al., 2023; Ryoo et al., 2025). Table 1 outlines the current state-of-the-art decoder model design considerations.

In this work, we introduce a Robust Neural Data Transformer (*R-NDT*) to address the above gaps. R-NDT follows the conventional tokenization perspective on binned spike counts (Ye & Pandarinath, 2021). Specifically, it has five components: 1) **Neural context** which infers an in-context vector for every unit from a few seconds of unlabeled calibration activity with a permutation-equivariant encoder. This eliminates the requirement of per-session/unit parameters; 2) **Population read-in** which turns the counts of one bin into one neural token by weighting each unit with its identity. Thus, units never attend to one another and the same network is flexible for different number of units; 3) **Sliding-window causal backbone** which processes these neural tokens with a key-value

cache bounded to one window, such that the model streams at constant attention memory with one bin of emission latency; 4) **Drift memory** which employs a multi-timescale moving average of the backbone output computed only from past block windows and every block is conditioned through adaptive layer normalization; 5) **Behavior-query readout** which maps behavior timestamps with cross-attention to the most recent backbone states for kinematics decoding.

R-NDT is pretrained by Poisson reconstruction of stochastic masked counts for the unit dropout and contiguous multi-bin blackouts and does not require behavior labels. We evaluate our model architecture and streaming decoding by training a small model from scratch (R-NDT-SS) and fully fine-tuning the pretrained R-NDT (FT). Specifically, we evaluate the models under two protocols. The first is POSSM's evaluation protocol, which includes evaluation on 20 public sessions (across five different categories) spanning new days, a new subject, and new datasets. The second evaluation protocol is a whole-block causal streaming that decodes each test block, consisting of several minutes, as a single causal sample with no intermediate trial reset. The evaluation is on random-target (RT) sessions under a temporal split and on a new evaluation dataset of Neuropixels insertion sites spanning dorsal premotor and primary motor cortex in one macaque (Canfield et al., 2026). Under both protocols, R-NDT has been trained from scratch. The results show that R-NDT exceeds baselines, in particular the strongest single-session baseline. Further, after label-free pretraining and fine-tuning it exceeds supervised-pretrained baselines. In summary, our contributions are:

- **Architecture**: Unit identity is computed from an unlabeled calibration segment rather than learned per session, along with a read-in that accepts a flexible number of units. Architecture is designed with causal sliding-window attention that streams at a constant KV-cache.
- **Label-free pretraining**: Poisson reconstruction of structurally masked counts on public sessions that, under the fine-tuning protocol, exceeds supervised pretraining.
- **Comprehensive evaluation**: We implement two evaluation protocols. Trial-based evaluation and a whole-block causal protocol with comprehensive comparison of spiking decoders given Utah-array and Neuropixels recordings.

2 Related Work

Neural decoders and pretrained population models. Continuous motor decoding in BCIs was long carried out with Wiener and Kalman filters (Van Drongelen, 2018; Wu et al., 2002; Orsborn et al., 2014). Nonlinear Bayesian decoders add priors on movement statistics (Wu et al., 2006; Shanechi et al., 2016). Deep decoders followed, from MLPs (Glaser et al., 2020) to recurrent networks (Sussillo et al., 2016; Sani et al., 2024), and latent-variable models such as GPFA, MRNN and LFADS (Yu et al., 2008; Sussillo et al., 2016; Pandarinath et al., 2018) extracted low-dimensional dynamics. Trained on the particular labeled data of one session, each of these is usually tied to that particular session's neural units. With the development of Transformers, methods leveraging them introduced pretraining. The Neural Data Transformer (Ye & Pandarinath, 2021) proposed to model binned spike counts with masked reconstruction, STNDT (Le & Shlizerman, 2022) added a spatial stream over units, NDT2 (Ye et al., 2023) pretrained across sessions, subjects and tasks with patch tokens and per-context read-ins, NDT3 (Ye et al., 2025) scaled the approach to hundreds of datasets, and BIT (Zhang et al., 2025b) and NEDS (Zhang et al., 2025a) extended binned masked modeling to decoding and joint encoding. In addition, another line of work proposed to tokenize individual spikes. Indeed, POYO (Azabou et al., 2023; 2024) paired a learnable unit embedding with a PerceiverIO encoder and pretrained them with behavior supervision across subjects, POSSM (Ryoo et al., 2025) fed chunks to a recurrent backbone so that decoding is causal, and MOJO (Mao et al., 2026) added a self-supervised pathway alongside the supervised one. PopT and BrainBERT adapted encoders to intracranial local field potentials (Wang et al., 2023; Chau et al., 2025).

Cross-session adaptation and unit identity. Changes in recorded neural unit identity and other properties have been a major cause of compromised BCI decoder robustness and require frequent recalibration (Chestek et al., 2011; Collinger et al., 2013). Several approaches have aimed to address this challenge. In particular, multiple works kept a fixed decoder and aligned each new day's activity to it, through latent-space stabilization (Degenhart et al., 2020; Gallego et al., 2020) and few-shot alignment (Karpowicz et al., 2024). In contrast, pretrained decoders assign each unit a learned embedding. Examples of existing work include adapting POYO or POSSM to a new session by initializing new unit and session embeddings and fitting them via gradient descent on that session's

labels for unit identification (Azabou et al., 2023; Ryoo et al., 2025). More recent approaches, such as MOJO, lowered the label requirement to a few trials (Mao et al., 2026). Furthermore, NDT2 attached per-context tokens yet requires read-in weights (Ye et al., 2023). In all the scenarios, the identity of a unit is a parameter fitted to the new session. An alternative could be to compute identity from the unit's own activity. Indeed, approaches such as NuCLR learn a neuron embedding from population context (Arora et al., 2026), and the recent SPINT model infered a per-unit identity from a short unlabeled calibration segment with one shared MLP network (Le et al., 2025).

Causal streaming and long context modeling. Real-time decoding requires causal models with bounded latency and memory. Causal models, such as POSSM (Ryoo et al., 2025), have been introduced. It proposed to evaluate every model causally on chunks. Most computational neuroscience comparisons remain to be based on evaluation scores per trial, resetting the state between trials of a few seconds. Therefore, determination of how a decoder behaves over longer periods such as minutes or longer, remains unclear. Such uncertainty is tightly connected to practical motor BCI which requires deploying deep decoders in a closed loop (Orsborn et al., 2014; 2012) which corresponds to considering long-context sequence modeling. Addressing this aspect produced attention variants that keep a bounded cache of past keys and values; examples include Transformer-XL (Dai et al., 2019), streaming attention with attention sinks (Xiao et al., 2024), block-recurrent and hybrid attention-recurrence models (Hutchins et al., 2022), and structured state-space models (Gu et al., 2021; Gu & Dao, 2023) that carry a state at linear cost. Conditioning a network on a slowly varying summary through adaptive normalization is standard in generative models (Peebles & Xie, 2023).

3 METHODS

We study causal decoding of behavior from a stream of population spiking activity. Within a recording session s, spikes of N_s sorted units are binned at width Δ into counts $\mathbf{X} \in \mathbb{Z}_{\geq 0}^{T \times N_s}$, where $x_{t,i}$ is the count of unit i in bin t, and the behavior covariate $\mathbf{y}_\tau \in \mathbb{R}^{d_y}$ is sampled at timestamps τ based on the sampling rates. The most challenging part for joint multi-session recording training is that the number and the identity of the units differ from session to session, and explicit correspondence between the units of two sessions is unknown. Here, we consider a short *unlabeled* calibration segment $\mathbf{C} \in \mathbb{Z}_{\geq 0}^{T_{\mathrm{cal}} \times N_s}$ of the same population is accessible. We then seek a generalizable brain decoder

$$\hat{\mathbf{y}}_\tau = f_\theta\big(\mathbf{X}_{\leq t(\tau)},\, \mathbf{C}\big), \tag{1}$$

where $t(\tau)$ is the bin containing τ and $\mathbf{X}_{\leq t}$ denotes all counts up to and including bin t, such that 1) θ is shared by all sessions, without parameters specified by a session or unit; 2) the prediction is causal with at most one bin of latency; and 3) the computation per bin is constant in the length of the stream. We propose R-NDT (see Figure 2), which factorizes f_θ into a neural context and population read-in (Section 3.1), a sliding-window causal backbone conditioned on a drift memory (Section 3.2), and a behavior readout (Section 3.3); Section 3.4 gives the pretraining and adaptation objective. We defer implementation details to Appendix E.

3.1 NEURAL CONTEXT AND POPULATION READ-IN

Neural context. A unit has no label that carries across sessions, but it has a signature: how much it fires, how regularly, and how it co-fluctuates with the rest of the population. Thus, we can compute this signature from the calibration segment. For every unit i,

$$\boldsymbol{\phi}_i = \psi\big(\mathbf{C}_{:,i},\ \tfrac{1}{N_s - 1} \textstyle\sum_{j \neq i} \mathbf{C}_{:,j}\big) \in \mathbb{R}^F, \tag{2}$$

where ψ collects F fixed statistics of three families: the rate and dispersion of the unit's counts, their autocorrelation at short lags, and their lagged cross-correlation with the leave-one-out population mean (Appendix E.1). Importantly, no statistic needs to scale with the length of the segment, so $\boldsymbol{\phi}_i$ is defined for any T_{cal}. The sample estimated statistics $\tilde{\boldsymbol{\phi}}_i$ are then mapped to the *neural context* by a permutation-equivariant set encoder (Zaheer et al., 2017) of L_E layers,

$$\mathbf{h}_i^{(\ell+1)} = \sigma\Big(\mathbf{W}_\ell\, \mathbf{h}_i^{(\ell)} + \mathbf{V}_\ell\, \tfrac{1}{N_s} \textstyle\sum_{j=1}^{N_s} \mathbf{h}_j^{(\ell)}\Big), \qquad \mathbf{h}_i^{(0)} = \tilde{\boldsymbol{\phi}}_i, \qquad \mathbf{E}_i = \mathbf{U}\, \mathbf{h}_i^{(L_E)} \in \mathbb{R}^{d_E}, \tag{3}$$

where σ is a pointwise nonlinearity and $\mathbf{W}_\ell$, $\mathbf{V}_\ell$, $\mathbf{U}$ are shared by all units and all sessions. The pooled term lets the encoder place a unit relative to the population it was recorded with, so that

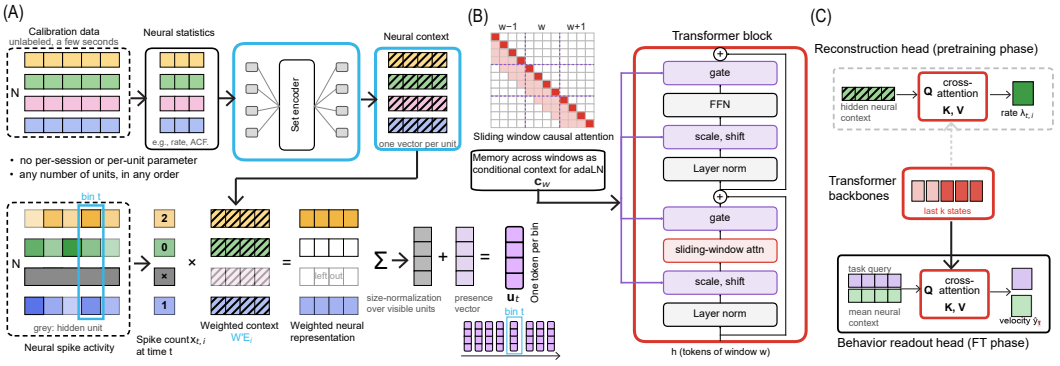


Figure 2: **Architecture of R-NDT**. (A) Neural context and read-in: parameter-free statistics of an unlabeled calibration segment data are mapped by a set encoder to the neural context of every unit; each bin is read in as one token $\mathbf{u}_t$, a count-weighted sum of $\mathbf{W}'\mathbf{E}_i$ over the visible units plus a presence term. (B) Transformer blocks with sliding-window causal attention restricted to a sliding window of T_w bins. A drift memory conditions every block of window w through adaLN-Zero. (C) A per-unit query predicts the Poisson rate of hidden counts (pretraining); a behavior query at timestamp τ cross-attends to the k most recent states and reads out behavior covariates $\hat{\mathbf{y}}_\tau$ (decoding).

the same statistics can map to different contexts in a quiet and in an active population. We write $\mathbf{E} = f_E(\mathbf{C}) \in \mathbb{R}^{N_s \times d_E}$. Although it seems to play the role of the learned unit embedding, it only depends on the population's own activity rather than a look-up table (Azabou et al., 2023).

Population read-in. Let $\mathcal{V}_t \subseteq \{1, \ldots, N_s\}$ be the units recording in bin t and $N_t = |\mathcal{V}_t|$. The counts of the bin are summarized into a single token

$$\mathbf{u}_t = \frac{1}{\sqrt{N_t}} \sum_{i \in \mathcal{V}_t} x_{t,i}\, \mathbf{W}'\mathbf{E}_i \;+\; \log(1 + N_t)\, \mathbf{v} \;\in \mathbb{R}^D, \tag{4}$$

where $\mathbf{W}' \in \mathbb{R}^{D \times d_E}$ and $\mathbf{v} \in \mathbb{R}^D$ are shared across sessions. The first term is the read-in of a linear decoder, $\sum_i x_{t,i}\mathbf{w}_i$, whose per-unit weight $\mathbf{w}_i = \mathbf{W}'\mathbf{E}_i$ is *computed* from the above neural context. Thus, no per-session read-in parameters are required in our model input. The $\sqrt{N_t}$ normalization keeps the scale of $\mathbf{u}_t$ comparable across populations of different sizes. The presence term evaluates how many active units were recorded to distinguish the modeling case: a bin in which the population was silent ($x_{t,i} = 0$ for all $i \in \mathcal{V}_t$) and a bin in which no unit was observed at all ($\mathcal{V}_t = \emptyset$, as under the later masking training strategy) would yield $\mathbf{u}_t = \mathbf{0}$.

Proposition 1 (Permutation symmetry and size agnosticism)**.** *Let π be any permutation of the units of a session. Then $f_E(\pi\mathbf{C}) = \pi f_E(\mathbf{C})$ and $\mathbf{u}_t(\pi\mathbf{X}, \pi\mathbf{C}) = \mathbf{u}_t(\mathbf{X}, \mathbf{C})$ for every t. Moreover, no parameter in Equation 2–Equation 4 has a dimension that depends on N_s.*

The proof is in Appendix E.6. Consequently, f_θ is invariant to the order of the units and applies unchanged to a population of any size, which allows robust pretraining across *heterogeneous* sessions.

3.2 Sliding-window causal backbone with memory conditioning

Sliding-window causal attention. The tokens $\mathbf{u}_{1:T}$ are processed by pre-norm transformer blocks with rotary position encoding (Su et al., 2024) and query-key normalization (Henry et al., 2020); we write $\mathbf{h}_t^{(\ell)} \in \mathbb{R}^D$ for the state of bin t after block ℓ, with $\mathbf{h}_t^{(0)} = \mathbf{u}_t$. Every attention layer is restricted to a causal window of T_w bins,

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\Big(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}} + \mathbf{M}\Big)\mathbf{V}, M_{tk} = \begin{cases} 0 & \text{if } k \le t < T_w + k \text{ and } r(t) = r(k) \\ -\infty & \text{otherwise} \end{cases} \tag{5}$$

where d_h is the head dimension normalization and $r(t)$ indexes the recording that bin t belongs to, so attention never crosses a recording boundary. Equation 5 bounds the key-value cache of every layer to $T_w - 1$ positions, so a stream of any length is decoded at constant memory, while the receptive field still grows with depth to $\ell\,(T_w - 1) + 1$ bins after block ℓ (Dai et al., 2019; Xiao et al., 2024).

Drift memory with adaptive layer-norm conditioning. Attention starts to forget beyond its receptive field, whereas the statistics of neural activity drift over minutes. Thus, we maintain a bank of K exponential moving averages of its own output to access this slow drift,

$$\mathbf{z}_t = \mathbf{P}\,\mathbf{h}_t \in \mathbb{R}^{K\times d_m}, \qquad \mathbf{m}_t^{(\kappa)} = (1-a_\kappa)\,\mathbf{m}_{t-1}^{(\kappa)} + a_\kappa\,\mathbf{z}_t^{(\kappa)}, \qquad a_\kappa = 1-2^{-\Delta/\eta_\kappa}, \tag{6}$$

for $\kappa = 1,\dots,K$, where $\mathbf{P}$ is a learned projection, $\mathbf{z}_t^{(\kappa)} \in \mathbb{R}^{d_m}$ is the κ-th row of $\mathbf{z}_t$, and $\eta_1 < \cdots < \eta_K$ defines half-lives spanning seconds to minutes to mimic the fast and slow neural activity. The memory is a recurrent state of fixed size Kd_m, and it is reset to zero at the start of a recording. The memory component does not enter the token sequence; rather, it modulates every block through adaLN-Zero (Peebles & Xie, 2023). Let window w cover bins $(w-1)T_w+1,\dots,wT_w$. The memory *entering* the window is mapped to a conditioning vector $\mathbf{c}_w = g\big(\mathbf{m}_{(w-1)T_w}\big) \in \mathbb{R}^D$ by a small network g, each block forms $(\boldsymbol{\gamma}_1,\boldsymbol{\beta}_1,\boldsymbol{\alpha}_1,\boldsymbol{\gamma}_2,\boldsymbol{\beta}_2,\boldsymbol{\alpha}_2) = \mathrm{Linear}(\mathrm{SiLU}(\mathbf{c}_w))$, and the states of the window are updated as

$$\begin{aligned}\mathbf{h} &\leftarrow \mathbf{h} + \boldsymbol{\alpha}_1 \odot \mathrm{Attn}\big((1+\boldsymbol{\gamma}_1)\odot \mathrm{LN}(\mathbf{h}) + \boldsymbol{\beta}_1\big),\\ \mathbf{h} &\leftarrow \mathbf{h} + \boldsymbol{\alpha}_2 \odot \mathrm{FFN}\big((1+\boldsymbol{\gamma}_2)\odot \mathrm{LN}(\mathbf{h}) + \boldsymbol{\beta}_2\big).\end{aligned} \tag{7}$$

While attention carries the fine temporal structure of the last T_w bins, the memory carries a slowly varying summary. Thus, it keeps the modulation constant within the window and explicitly causal.

3.3 Behavior readout

Behavior is read by query-based cross-attention following the PerceiverIO architecture, as widely used in most prior works (Azabou et al., 2023; Le et al., 2025).

$$\mathbf{q}_\tau = \mathbf{e}_{\mathrm{task}} + \mathbf{W}_s\bar{\mathbf{E}}, \quad \mathbf{e}_\tau = \mathbf{q}_\tau + \mathrm{CrossAttn}\big(\mathbf{q}_\tau,\, \mathbf{H}_{t(\tau)-k+1\,:\,t(\tau)}\big), \quad \hat{\mathbf{y}}_\tau = \mathbf{W}_y\big(\mathbf{e}_\tau + \mathrm{FFN}(\mathbf{e}_\tau)\big), \tag{8}$$

where $\mathbf{e}_{\mathrm{task}} \in \mathbb{R}^D$ is a learned task embedding; $\bar{\mathbf{E}}$ is the mean neural context of the session; τ and k define the recent referring states. We note that the term $\mathbf{W}_s\bar{\mathbf{E}}$ functions as a per-session context in the behavior readout. Further, timestamps are decoupled from bins; thus the readout supports any behavioral sampling rate. We train it under the standard mean-squared error (MSE)

$$\mathcal{L}_{\mathrm{dec}} = \frac{\sum_\tau \omega_\tau \,\frac{1}{d_y}\big\|\hat{\mathbf{y}}_\tau - \mathbf{y}_\tau\big\|_2^2}{\sum_\tau \omega_\tau}, \tag{9}$$

where $\omega_\tau \geq 0$ is a per-timestamp weight set by the behavioral epoch that τ falls in.

3.4 Pretraining and adaptation

Pretraining. We first pretrain the model without behavior labels by reconstructing hidden spike counts from visible ones. Three masks are applied jointly to every training window: 1) *unit dropout* hides a random subset of the units throughout, 2) *temporal blackout* hides the whole population for a short contiguous run of bins, and 3) a sparse i.i.d. mask hides individual entries (Appendix E.5). Hidden entries are removed from the visible set $\mathcal{V}_t$. Each hidden unit is queried by its own neural context, which attends to the k' most recent backbone states and predicts a Poisson rate,

$$\lambda_{t,i} = \mathrm{softplus}\Big(\mathbf{w}_\lambda^\top\,\mathrm{CrossAttn}\big(\mathbf{W}_q\mathbf{E}_i,\ \mathbf{H}_{t-k'+1\,:\,t}\big) + b_\lambda\Big), \tag{10}$$

and the loss is the Poisson negative log-likelihood averaged over the set $\mathcal{M}$ of hidden entries,

$$\mathcal{L}_{\mathrm{ssl}} = \frac{1}{|\mathcal{M}|}\sum_{(t,i)\in\mathcal{M}}\big(\lambda_{t,i} - x_{t,i}\log\lambda_{t,i}\big). \tag{11}$$

Adaptation to a new session. Adapting to a session starts by computing its neural context $\mathbf{E} = f_E(\mathbf{C})$ once, from an unlabeled calibration segment. The decoder is then obtained by minimizing $\mathcal{L}_{\mathrm{dec}}$ (Equation 9) on the labeled training data of the session, with unit dropout retained as an augmentation, in one of two ways. **R-NDT-SS** trains all of θ from a random initialization, and measures the architecture alone. **R-NDT (FT)** initializes θ from the pretrained model, replaces the reconstruction head by the behavior readout, and fine-tunes all parameters.

4 EXPERIMENTS AND RESULTS

4.1 EXPERIMENTAL SETUP

Data. R-NDT is pretrained without behavior labels on 156 public sessions of Utah-array recordings from motor and somatosensory cortex of nonhuman primates (Perich 97, O'Doherty 47, Churchland 10, Neural Latents Benchmark 2; roughly 100 h, see Figure 1E left). Public evaluation uses the 20 sessions (see Figure 1E right), none of which is in the pretraining data: from the Perich et al. dataset (Perich et al., 2018; Perich et al.), two center-out sessions of monkey C (C-CO 2016; 520 units), six center-out sessions of monkey T (T-CO; 336 units) and six random-target sessions of monkey T (T-RT; 349 units); five center-out sessions of monkey C from Flint et al. (Flint et al., 2012) (Flint; 957 units); and the Area2_Bump session of the Neural Latents Benchmark (Pei et al., 2021) (Area2; 65 units, somatosensory cortex). Monkey T, Flint, and Area2 are new to the pretrained model: a new animal, a new dataset, and a new cortical area, respectively. Details about the pretraining and evaluation datasets are listed in Appendix Table 2. Data splits are listed in Appendix B.

NPCS. We further evaluate on a new dataset, "NeuroPixels across Cortical Sites" (NPCS) (Canfield et al., 2026): 3 insertion sites of a Neuropixels 1.0 probe (Jun et al., 2017) in one rhesus macaque, spanning around 4 mm of dorsal premotor and primary motor cortex over about one year, with no neuron shared between sites. More details are in Appendix Table 2 and Table D.

Two test protocols. First, under the *per-trial* protocol in the Figure 1F upper panel, we follow POSSM's evaluation exactly: the upstream per-dataset train/validation/test splits, 1 s training windows, whole test trials at their native length (3 to 5 s) as test samples with the recurrent state of every model restarting per trial, R^2 pooled over all test trials of a session and averaged over the two velocity axes. Second, under the *whole-block streaming* protocol (see Figure 1F lower panel) the per-trial reset is removed: on the six T-RT sessions of monkey T we replace the trial split by a contiguous 60/10/30 temporal split, so the test set is one 4 to 6 min block, and on NPCS we use the dataset's native split into 10 contiguous blocks (8/1/1), so the test set is one 61 to 368 s block. The whole block is decoded causally as a single sample, without trial boundary and state reset.

Models and baselines. R-NDT-SS is trained from scratch on one session; R-NDT (FT) is the pretrained model fine-tuned on that session (Section 3.4). From-scratch baselines are a Wiener filter (Van Drongelen, 2018), a binned MLP (Glaser et al., 2020), a GRU (Cho et al., 2014), S4D (Gu et al., 2021), Mamba (Gu & Dao, 2023), a unidirectional LFADS (Pandarinath et al., 2018), SPINT (Le et al., 2025), POYO-SS (Azabou et al., 2023) and POSSM-SS with S4D, Mamba and GRU backbones (Ryoo et al., 2025); pretrained baselines are NDT-2, LFADS, POYO-1 and o-POSSM under unit identification (UI) and full fine-tuning (FT). Every row was trained by us on identical splits with the GitHub-released code and the same test protocol; architectures and training recipes are in Appendix C. We report the mean and standard deviation, and run a two-sided paired Wilcoxon signed-rank test across evaluation sessions.

4.2 PRETRAINING AND DECODING ON PUBLIC BENCHMARKS

We started by showing the decoding-trace comparison for typical session trials in Figure 3A. Zooming in on those decoding kinematic traces, we observed that R-NDT models aligned quite well with the ground-truth traces (traces of baseline models are in Appendix Figure 1). Figure 3B gave the overall statistical comparisons of R-NDT against the recent SOTA POSSM-GRU and POYO. Specifically, against POSSM-GRU-SS, R-NDT-SS gained $+0.0147$ on average and wins 16 of 20 sessions ($p = 2.6 \times 10^{-4}$); against POYO-SS, it gained $+0.0352$ (18/20, $p = 3.6 \times 10^{-5}$). Those results suggested the power of inferring neural unit identity for spike modeling. Meanwhile, pretrained models followed by full fine-tuning improved on R-NDT-SS by $+0.0128$ on average and won 17 of the 20 test sessions ($p = 5.9 \times 10^{-4}$). Further, R-NDT (FT) also significantly outperformed its supervised pretrained counterparts (POSSM and POYO), giving us strong confidence that both the model architecture and pretraining strategy help. Having established the above, we next investigated the data scaling behaviors of R-NDT models. In Figure 3C, we showed that with more pretraining data (from 25% to 100%), the pretraining loss decreased, aligned with the current scaling trend in LLM training (upper panel), and the corresponding test session performance increased (lower panel). In Figure 3D, we also observed the test-time data scaling trend. Interestingly, we found that with pretraining, the R-NDT (FT) model significantly outperformed the R-NDT-SS in the lower data regions,

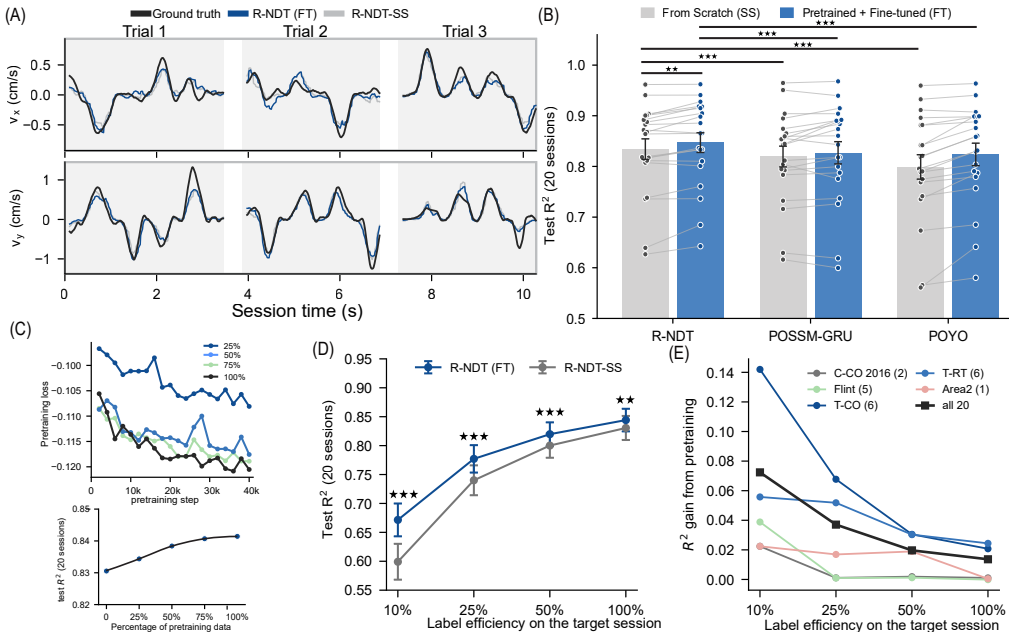


Figure 3: (A) Behavior decoding traces for R-NDT (FT) and R-NDT-SS on test trials. (B) Pretraining and fine-tuning against training from scratch on the 20 public evaluation sessions over POSSM and POYO baselines. Points are individual sessions, and a line joins the from-scratch (SS) and pretrained-then-fine-tuned (FT) arms of the same session. (C) Pretrain data scaling trend. (D) Test-time data scaling trend. (E) Per-dataset decoding performance gain from pretraining models across different percentages of target sessions. Bars are the mean test R^2, and error bars are computed by the standard error of the mean (s.e.m) over sessions; $**$ and $***$ stand for $p < 0.005$ and $p < 0.0005$, using a Wilcoxon signed-rank test over the 20 sessions. Per-session results are in Appendix Table 3.

	Method	C-CO 2016 (2)	Flint (5)	T-CO (6)	T-RT (6)	Area2 (1)	**Average (20)**
From scratch	Wiener filter	0.8801 ± 0.0127	0.6924 ± 0.0805	0.6777 ± 0.0289	0.6150 ± 0.0768	0.6644	0.6821 ± 0.0947
	MLP	0.9416 ± 0.0108	0.7620 ± 0.1002	0.8317 ± 0.0245	0.7012 ± 0.0858	0.7537	0.7822 ± 0.1015
	S4D	0.9164 ± 0.0183	0.6418 ± 0.1328	0.7126 ± 0.0355	0.5904 ± 0.0916	0.7367	0.6798 ± 0.1272
	Mamba	0.9326 ± 0.0164	0.7395 ± 0.1009	0.7097 ± 0.0257	0.6576 ± 0.0900	0.7679	0.7267 ± 0.1053
	GRU	0.9385 ± 0.0146	0.7480 ± 0.0910	0.7964 ± 0.0260	0.7061 ± 0.0805	0.7948	0.7713 ± 0.0931
	LFADS (unidirectional)	0.9530 ± 0.0108	0.7716 ± 0.0980	0.8344 ± 0.0346	0.7066 ± 0.0792	0.7949	0.7903 ± 0.1004
	SPINT	0.9346 ± 0.0077	0.7331 ± 0.1040	0.8533 ± 0.0196	0.7479 ± 0.0627	0.8062	0.7974 ± 0.0927
	POYO-SS	0.9460 ± 0.0135	0.7270 ± 0.1005	0.8679 ± 0.0387	0.7375 ± 0.0806	0.8208	0.7990 ± 0.1048
	POSSM-S4D-SS	0.9510 ± 0.0099	0.7213 ± 0.0920	0.8497 ± 0.0280	0.6729 ± 0.0731	0.7713	0.7708 ± 0.1121
	POSSM-Mamba-SS	0.9511 ± 0.0097	0.7341 ± 0.1198	0.8540 ± 0.0193	0.7399 ± 0.0838	0.8473	0.7992 ± 0.1027
	POSSM-GRU-SS	**0.9577** ± 0.0070	0.7740 ± 0.0896	0.8690 ± 0.0240	0.7551 ± 0.0668	0.8594	0.8195 ± 0.0895
	R-NDT-SS	0.9511 ± 0.0105	**0.7848** ± 0.0950	**0.8921** ± 0.0122	**0.7733** ± 0.0663	**0.8640**	**0.8342** ± 0.0881
Pretrained	NDT-2 (FT)	0.8507 ± 0.0132	0.6815 ± 0.1135	0.6487 ± 0.0225	0.5924 ± 0.1251	0.8325	0.6694 ± 0.1064
	LFADS (FT)	0.9176 ± 0.0133	0.5228 ± 0.1815	0.7341 ± 0.0593	0.6351 ± 0.0995	0.7287	0.6697 ± 0.1600
	POYO-1 (UI)	0.9518 ± 0.0097	0.7377 ± 0.1111	0.8213 ± 0.0324	0.7147 ± 0.0782	0.7456	0.7777 ± 0.1024
	POYO-1 (FT)	0.9520 ± 0.0116	0.7576 ± 0.1115	0.8867 ± 0.0263	0.7672 ± 0.0607	0.8584	0.8237 ± 0.0969
	o-POSSM-S4D (UI)	0.9481 ± 0.0076	0.5214 ± 0.1292	0.8388 ± 0.0086	0.6790 ± 0.0841	0.7343	0.7172 ± 0.1616
	o-POSSM-Mamba (UI)	0.9537 ± 0.0068	0.6121 ± 0.1256	0.8150 ± 0.0273	0.6990 ± 0.0803	0.7627	0.7407 ± 0.1303
	o-POSSM-GRU (UI)	0.9562 ± 0.0089	0.7258 ± 0.1202	0.8604 ± 0.0200	0.7340 ± 0.0677	0.7836	0.7946 ± 0.1064
	o-POSSM-S4D (FT)	0.9533 ± 0.0077	0.7446 ± 0.1119	0.8753 ± 0.0208	0.7126 ± 0.0728	0.8079	0.7982 ± 0.1094
	o-POSSM-Mamba (FT)	**0.9577** ± 0.0085	0.7640 ± 0.0950	0.8796 ± 0.0186	0.7584 ± 0.0721	0.8505	0.8207 ± 0.0941
	o-POSSM-GRU (FT)	0.9537 ± 0.0142	0.7694 ± 0.0928	0.8938 ± 0.0129	0.7599 ± 0.0767	0.8605	0.8269 ± 0.0958
	R-NDT (FT)	0.9514 ± 0.0106	**0.7862** ± 0.0889	**0.9137** ± 0.0097	**0.7930** ± 0.0551	**0.8653**	**0.8470** ± 0.0847

Table 2: Velocity decoding on the 20 public evaluation sessions under POSSM's per-trial causal protocol. Values are mean R^2 ± SD across the sessions (Area2 is one session); the last column is the mean over all 20 sessions. The best value per column is in boldface and the second best is underlined. SS: single session from scratch; UI: unit identification; FT: full fine-tuning.

which suggested fewer target (test-time) session data required for future BCI decoder deployment. We presented per-dataset gain according to the percentage of label efficiency in Figure 3E.

Having established Figure 3, we next present the comprehensive quantitative evaluation results across all baselines and testing datasets in Table 2. **Training from scratch (Table 2, top)** R-NDT-SS is the best from-scratch model in four of the five columns. The margins are largest on the two

columns of the new animal, T-CO (0.8921 v.s. second best 0.8690) and T-RT (0.7733 v.s. second best 0.7551). Beyond what is shown in Figure 3B, R-NDT-SS also outperformed the unidirectional LFADS and SPINT models, RNN family models, the traditional binned MLP, and wiener filter. The significant gaps to R-NDT-SS reflect the component innovations in the neural context, sliding causal backbone, and behavior-query readout design. **Pretrained models (Table 2, bottom)** As illustrated in Figure 3, the bottom table confirms the benefits of pretraining consistently in all five datasets when we compare R-NDT (FT) to R-NDT-SS (avg. 0.8470 v.s. 0.8342). Further, R-NDT (FT) also significantly outperformed other baselines after averaging over 20 sessions (0.8470 v.s. the second best 0.8269). We note that the pretraining regimes differ in both models, as o-POSSM requires supervised behavior labels and adapts by fitting unit embeddings from labels; R-NDT sees spikes only and adapts from an unlabeled calibration segment for neural context embedding. Last, we also report the POSSM's published values next to our reproduction in Appendix A.

4.3 Whole-block causal streaming

Streaming decoding. Under the whole-block protocol, R-NDT (FT) and R-NDT-SS rows are the best and the second best on both datasets (Table 3). Importantly, R-NDT-SS (0.7314) outperformed POSSM-GRU-SS (0.7044) and SPINT (0.7038) on the six T-RT blocks, and R-NDT-SS (0.8960) outperformed POSSM-GRU-SS (0.8793) and SPINT (0.8830) on the 3 NPCS blocks. The SPINT model is the closest single-session baseline, which illustrates the benefits of calibration data for neural context inference in whole-block streaming decoding. The gap between the SPINT and our R-NDT-SS is attributed to the potential benefits of state across the decoding windows, where the SPINT decoder is reset every on every decoding window. We also observed that supervised pretraining does not help the POSSM-GRU models in general (T-RT: o-POSSM-GRU-FT 0.6902 v.s. POSSM-GRU-SS 0.7044; NPCS: FT 0.8523 v.s. SS 0.8793), which was also observed in the original papers Ryoo et al. (2025), showing the potential limitations of the supervised pretraining strategy. On the other hand, our FT model was better than the SS models, further illustrating the power of the label-free pretraining strategy. Per-session streaming decoding results are in the Appendix Table 4 and Table 5.

Method	T-RT temporal (6)	NPCS (3)
Wiener filter	0.5576 ± 0.0677	0.7310 ± 0.0238
MLP	0.6627 ± 0.0882	0.8633 ± 0.0081
GRU	0.6425 ± 0.0830	0.8580 ± 0.0210
S4D	0.5288 ± 0.0961	0.6903 ± 0.0405
Mamba	0.5672 ± 0.1044	0.8423 ± 0.0061
LFADS(unidirectional)	0.6611 ± 0.0878	0.8603 ± 0.0064
SPINT-SS	0.7038 ± 0.0790	0.8830 ± 0.0121
POSSM-S4D-SS	0.5593 ± 0.1488	0.6723 ± 0.0301
POSSM-Mamba-SS	0.5824 ± 0.1985	0.8430 ± 0.0424
POSSM-GRU-SS	0.7044 ± 0.0702	0.8793 ± 0.0072
o-POSSM-S4D (UI)	0.4147 ± 0.0552	0.5790 ± 0.0477
o-POSSM-S4D (FT)	0.4699 ± 0.0792	0.7503 ± 0.0533
o-POSSM-Mamba (UI)	0.4119 ± 0.0236	0.7558 ± 0.1334
o-POSSM-Mamba (FT)	0.6666 ± 0.0668	0.8829 ± 0.0182
o-POSSM-GRU (UI)	0.6657 ± 0.0694	0.8090 ± 0.0114
o-POSSM-GRU (FT)	0.6902 ± 0.0672	0.8523 ± 0.0295
R-NDT-SS	$\underline{0.7314} \pm 0.0731$	$\underline{0.8960} \pm 0.0190$
R-NDT (FT)	$\mathbf{0.7325} \pm 0.0589$	$\mathbf{0.8983} \pm 0.0205$

Table 3: Whole-block causal streaming. The test set of each session is one contiguous block, decoded causally as a single sample with no trial reset. Values are mean $R^2 \pm$ SD across sessions; best in boldface, second best underlined.

Last, we discuss both causal evaluation protocols and their implications. The trial-based protocol needs to restart the decoder from an empty window every second; a practical deployed decoder instead runs on one unbroken stream for minutes, motivating the whole-block protocol. Thus, models with a recurrent (state-space evolution) or attention state (sliding-window attention as a bounded KV cache) are necessary for efficiently using the preceding minutes as causal context; R-NDT obtains such a state without the usual cost of a quadratic expansion of the transformer attention blocks, which gives nonlinear and efficient long-range modeling for BCI decoders.

5 Conclusions

Building a causal brain-motor decoding model is challenging. We introduce R-NDT, a robust neural transformer that integrates novel components for robust motor decoding performance. Our results show that R-NDT outperforms across different generalization sessions. **Limitations and implications.** Our modeling and evaluation were restricted to cortical motor areas, excluding other regions (Trautmann et al., 2025). Second, validating R-NDT on broader behavioral datasets will be the next step (Willett et al., 2023; Banga et al., 2025). Beyond that, while demonstrated on intracortical motor decoding, our work can extend to future closed-loop BCIs (Orsborn et al., 2012).

DISCLOSE USE OF LLMS

We used LLMs, for initial idea brainstorming and later to check grammar and spelling and improve the clarity of our manuscript's language. All scientific content, including figures, tables, experimental design, analysis, and interpretations, is entirely the original work.

ETHICS STATEMENT

Our work focuses on algorithm design and evaluation for neural data. To evaluate the R-NDT model, we used multiple public datasets that have been well-cited and introduced in both Section 4.1 and Appendix Section 2. The NPCS dataset was from (Canfield et al., 2026). Data privacy and ethics protocols for experimental procedures related to neural data collection were addressed in the original publication.

REPRODUCIBILITY STATEMENT

All hyperparameters, architectural details, and training configurations necessary for reproducing our results are provided in Appendix E and Appendix F. Evaluation protocols and baselines are described in Appendices B, C and D. The code will be made publicly available on GitHub upon acceptance of the paper.

REFERENCES

Vinam Arora, Divyansha Lachi, Ian Knight, Mehdi Azabou, Blake Richards, Cole Hurwitz, Joshua H Siegle, and Eva L Dyer. Know thyself by knowing others: Learning neuron identity from population context. *Advances in Neural Information Processing Systems*, 38:99472–99501, 2026.

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

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

Kush Banga, Julius Benson, Jai Bhagat, Dan Biderman, Daniel Birman, Niccolò Bonacchi, Sebastian A Bruijns, Kelly Buchanan, Robert AA Campbell, Matteo Carandini, et al. Reproducibility of in vivo electrophysiological measurements in mice. *Elife*, 13:RP100840, 2025.

Ryan A Canfield, Tomohiro Ouchi, Hao Fang, Beatrice Macagno, Lydia I Smith, Leo R Scholl, and Amy L Orsborn. The spatiotemporal structure of neural activity in motor cortex during reaching. *The Journal of Neuroscience*, 46(20):e1965252026, 2026.

Geeling Chau, Christopher Wang, Sabera Talukder, Vighnesh Subramaniam, Saraswati Soedarmadji, Yisong Yue, Boris Katz, and Andrei Barbu. Population transformer: Learning population-level representations of neural activity. In *International Conference on Learning Representations*, volume 2025, pp. 69100–69121, 2025.

Cynthia A Chestek, Vikash Gilja, Paul Nuyujukian, Justin D Foster, Joline M Fan, Matthew T Kaufman, Mark M Churchland, Zuley Rivera-Alvidrez, John P Cunningham, Stephen I Ryu, et al. Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex. *Journal of neural engineering*, 8(4):045005, 2011.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

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.

Jennifer L Collinger et al. High-performance neuroprosthetic control by an individual with tetraplegia. *The Lancet*, 381(9866):557–564, 2013.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 2978–2988, 2019.

Alan D Degenhart et al. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. *Nature biomedical engineering*, 4(7):672–685, 2020.

Robert D Flint, Eric W Lindberg, Luke R Jordan, Lee E Miller, and Marc W Slutzky. Accurate decoding of reaching movements from field potentials in the absence of spikes. *Journal of neural engineering*, 9(4):046006, 2012.

Juan A Gallego et al. Long-term stability of cortical population dynamics underlying consistent behavior. *Nature neuroscience*, 23(2):260–270, 2020.

Joshua I Glaser, Ari S Benjamin, Raeed H Chowdhury, Matthew G Perich, Lee E Miller, and Konrad P Kording. Machine learning for neural decoding. *eneuro*, 7(4), 2020.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4246–4253, 2020.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. Block-recurrent transformers. *Advances in neural information processing systems*, 35:33248–33261, 2022.

James J Jun, Nicholas A Steinmetz, Joshua H Siegle, Daniel J Denman, Marius Bauza, Brian Barbarits, Albert K Lee, Costas A Anastassiou, Alexandru Andrei, Çağatay Aydın, et al. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, 2017.

Brianna Karpowicz, Joel Ye, Chaofei Fan, Pablo Tostado-Marcos, Fabio Rizzoglio, Clayton Washington, Thiago Scodeler, Diogo de Lucena, Samuel Nason-Tomaszewski, Matthew Mender, et al. Few-shot algorithms for consistent neural decoding (falcon) benchmark. *Advances in Neural Information Processing Systems*, 37:76578–76615, 2024.

Trung Le and Eli Shlizerman. Stndt: Modeling neural population activity with spatiotemporal transformers. *Advances in Neural Information Processing Systems*, 35:17926–17939, 2022.

Trung Le, Hao Fang, Jingyuan Li, Tung Nguyen, Lu Mi, Amy Orsborn, Uygar Sümbül, and Eli Shlizerman. Spint: Spatial permutation-invariant neural transformer for consistent intracortical motor decoding. *arXiv preprint arXiv:2507.08402*, 2025.

Ximeng Mao, Nanda H Krishna, Avery Hee-Woon Ryoo, Matthew G Perich, and Guillaume Lajoie. Leveraging unlabelled data for generalizable neural population decoding. *arXiv preprint arXiv:2607.14086*, 2026.

Amy L Orsborn, Siddharth Dangi, Helene G Moorman, and Jose M Carmena. Closed-loop decoder adaptation on intermediate time-scales facilitates rapid bmi performance improvements independent of decoder initialization conditions. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 20(4):468–477, 2012.

Amy L Orsborn, Helene G Moorman, Simon A Overduin, Maryam M Shanechi, Dragan F Dimitrov, and Jose M Carmena. Closed-loop decoder adaptation shapes neural plasticity for skillful neuroprosthetic control. *Neuron*, 82(6):1380–1393, 2014.

Joseph E O'Doherty, Mariana MB Cardoso, Joseph G Makin, and Philip N Sabes. Nonhuman primate reaching with multichannel sensorimotor cortex electrophysiology. *Zenodo http://doi.org/10.5281/zenodo*, 583331, 2017.

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, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15(10):805–815, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

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, et al. Neural latents benchmark'21: Evaluating latent variable models of neural population activity. *arXiv preprint arXiv:2109.04463*, 2021.

Matthew G. Perich, Lee E. Miller, Mehdi Azabou, and Eva L. Dyer. Long-term recordings of motor and premotor cortical spiking activity during reaching in monkeys. URL `https://dandiarchive.org/dandiset/000688/0.250122.1735`. DANDI:DANDI:000688/0.250122.1735.

Matthew G Perich, Juan A Gallego, and Lee E Miller. A neural population mechanism for rapid learning. *Neuron*, 100(4):964–976, 2018.

Avery Hee-Woon Ryoo, Nanda H Krishna, Ximeng Mao, Mehdi Azabou, Eva L Dyer, Matthew G Perich, and Guillaume Lajoie. Generalizable, real-time neural decoding with hybrid state-space models. *arXiv preprint arXiv:2506.05320*, 2025.

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.

Maryam M Shanechi, Amy L Orsborn, and Jose M Carmena. Robust brain-machine interface design using optimal feedback control modeling and adaptive point process filtering. *PLoS computational biology*, 12(4):e1004730, 2016.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

David Sussillo, Sergey D Stavisky, Jonathan C Kao, Stephen I Ryu, and Krishna V Shenoy. Making brain–machine interfaces robust to future neural variability. *Nature communications*, 7(1):13749, 2016.

Eric M Trautmann, Janis K Hesse, Gabriel M Stine, Ruobing Xia, Shude Zhu, Daniel J O'Shea, Bill Karsh, Jennifer Colonell, Frank F Lanfranchi, Saurabh Vyas, et al. Large-scale high-density brain-wide neural recording in nonhuman primates. *Nature Neuroscience*, pp. 1–14, 2025.

Wim Van Drongelen. *Signal processing for neuroscientists*. Academic press, 2018.

Christopher Wang, Vighnesh Subramaniam, Adam Uri Yaari, Gabriel Kreiman, Boris Katz, Ignacio Cases, and Andrei Barbu. Brainbert: Self-supervised representation learning for intracranial recordings. *arXiv preprint arXiv:2302.14367*, 2023.

Francis R Willett, Erin M Kunz, Chaofei Fan, Donald T Avansino, Guy H Wilson, Eun Young Choi, Foram Kamdar, Matthew F Glasser, Leigh R Hochberg, Shaul Druckmann, et al. A high-performance speech neuroprosthesis. *Nature*, 620(7976):1031–1036, 2023.

Wei Wu, M Black, Yun Gao, M Serruya, A Shaikhouni, J Donoghue, and Elie Bienenstock. Neural decoding of cursor motion using a kalman filter. *Advances in neural information processing systems*, 15, 2002.

Wei Wu, Yun Gao, Elie Bienenstock, John P Donoghue, and Michael J Black. Bayesian population decoding of motor cortical activity using a kalman filter. *Neural computation*, 18(1):80–118, 2006.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks, 2024. *URL https://arxiv. org/abs/2309.17453*, 1(5.1), 2024.

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

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

Joel Ye, Fabio Rizzoglio, Adam Smoulder, Hongwei Mao, Xuan Ma, Patrick Marino, Raeed Chowdhury, Dalton Moore, Gary Blumenthal, William Hockeimer, et al. A generalist intracortical motor decoder. *bioRxiv*, 2025.

Byron M Yu, John P Cunningham, Gopal Santhanam, Stephen Ryu, Krishna V Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Advances in neural information processing systems*, 21, 2008.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Yizi Zhang, Yanchen Wang, Mehdi Azabou, Alexandre Andre, Zixuan Wang, Hanrui Lyu, The International Brain Laboratory, Eva Dyer, Liam Paninski, and Cole Hurwitz. Neural encoding and decoding at scale. *ArXiv*, pp. arXiv–2504, 2025a.

Yizi Zhang et al. Decoding inner speech with an end-to-end brain-to-text neural interface. *arXiv preprint arXiv:2511.21740*, 2025b.

A PUBLISHED VALUES AGAINST OUR REPRODUCTION

Below table 1 places the values published by POSSM (Ryoo et al., 2025) next to ours.

	C-CO 2016 (2)		Flint (5)		T-CO (6)		T-RT (6)		Area2 (1)	
Method	POSSM	ours	POSSM	ours	POSSM	ours	POSSM	ours	POSSM	ours
MLP	0.9210	0.9416	0.6946	0.7620	0.7976	0.8317	0.7007	0.7012	0.4179	0.7537
S4D	0.9381	0.9164	0.6619	0.6418	0.8526	0.7126	0.7145	0.5904	0.3942	0.7367
Mamba	0.9287	0.9326	0.6708	0.7395	0.7692	0.7097	0.6694	0.6576	0.7231	0.7679
GRU	0.9376	0.9385	0.7308	0.7480	0.8453	0.7964	0.7279	0.7061	0.8128	0.7948
POYO-SS	0.9427	0.9460	0.7381	0.7270	0.8705	0.8679	0.7156	0.7375	0.4974	0.8208
POSSM-S4D-SS	0.9515	0.9510	0.7768	0.7213	0.8838	0.8497	0.7505	0.6729	0.8163	0.7713
POSSM-Mamba-SS	0.9550	0.9511	0.7959	0.7341	0.8747	0.8540	0.7418	0.7399	0.8350	0.8473
POSSM-GRU-SS	0.9549	0.9577	0.7930	0.7740	0.8863	0.8690	0.7687	0.7551	0.8161	0.8594
NDT-2 (FT)	0.8507	0.8507	0.6876	0.6815	0.6549	0.6487	0.5903	0.5924	0.8237	0.8325
POYO-1 (UI)	0.9534	0.9518	0.7779	0.7377	0.8214	0.8213	0.7099	0.7147	0.7666	0.7456
POYO-1 (FT)	0.9611	0.9520	0.8209	0.7576	0.8859	0.8867	0.7591	0.7672	0.8676	0.8584
o-POSSM-S4D (UI)	0.9572	0.9481	0.7651	0.5214	0.8739	0.8388	0.7228	0.6790	0.7941	0.7343
o-POSSM-Mamba (UI)	0.9609	0.9537	0.5381	0.6121	0.8630	0.8150	0.7340	0.6990	0.8014	0.7627
o-POSSM-GRU (UI)	0.9581	0.9562	0.7815	0.7258	0.8738	0.8604	0.7518	0.7340	0.7941	0.7836
o-POSSM-S4D (FT)	0.9618	0.9533	0.8147	0.7446	0.9069	0.8753	0.7587	0.7126	0.8701	0.8079
o-POSSM-Mamba (FT)	0.9574	0.9577	0.8092	0.7640	0.9011	0.8796	0.7621	0.7584	0.8674	0.8505
o-POSSM-GRU (FT)	0.9587	0.9537	0.8154	0.7694	0.9021	0.8938	0.7717	0.7599	0.8646	0.8605

Table 1: Mean test R^2 per column as published by POSSM (Ryoo et al., 2025) ("POSSM") and as reproduced by us on identical splits

For the three POSSM-GRU rows, which anchor both comparisons in the main text, 12 of the 15 cells agree within 0.02; the three that do not are POSSM-GRU-SS on Area2 and the two o-POSSM-GRU rows on Flint. Flint is the systematic miss across the table: our values run below the published ones for every pretrained row except o-POSSM-Mamba (UI), and the o-POSSM S4D and Mamba variants are further below the GRU variant than in the paper. Two facts about our o-POSSM pretraining corpus are relevant. It omits the two C-CO 2016 sessions, which POSSM pretrained on and we hold out because they are evaluation sessions, and it omits most of the Churchland maze sessions, so it has 142 sessions and 24,286 units against the paper's 140 sessions and 24,504 units.

B PUBLIC DATASETS AND SPLIT PROTOCOLS

Datasets. We list the detailed public dataset information in Table 2.

Splits. Every split is taken unchanged from the upstream data pipelines that POSSM uses (Ryoo et al., 2025), and they differ by dataset. Perich sessions split the *successful* trials 70/10/20 into train, validation, and test and keep every inter-trial interval in training; Flint sessions split trials 70/10/20 with no inter-trial data; the NLB Area2 session uses the benchmark's own train and test masks, with the train mask split 80/20 into train and validation. For the T-RT temporal split, the trial split is replaced by a contiguous 60/10/30 division of the recording in time, so that the test set is one 4 to 6 min block, and POSSM is re-run on bit-identical intervals.

	Dataset	Regions	Task	# Subj.	# Sess.	# Units	Hours
Pretrain	Perich et al. (Perich et al., 2018)	M1, PMd	CO, RT	3	97	9,205	39.1
	O'Doherty et al. (O'Doherty et al., 2017)	M1, S1	RT	2	47	16,566	13.7
	Churchland et al. (Churchland et al., 2012)	M1, PMd	CO	2	10	1,911	47.8†
	NLB MC_Maze (Pei et al., 2021)	M1, PMd	Maze	1	2	324	2.0
Evaluation	Perich et al. monkey C (C-CO 2016)	M1, PMd	CO	1	2	520	0.6
	Perich et al. monkey T (T-CO)	M1, PMd	CO	1	6	336	1.7
	Perich et al. monkey T (T-RT)	M1, PMd	RT	1	6	349	1.6
	Flint et al. (Flint et al., 2012) (Flint)	M1	CO	1	5	957	0.9
	NLB Area2_Bump (Pei et al., 2021) (Area2)	S1	CO	1	1	65	0.6
	NPCS (Canfield et al., 2026)	PMd, M1	CO	1	3	903	1.76

Table 2: Datasets. Pretraining uses 156 public Utah-array sessions (28,006 units, 102.6 h of recording) without behavior labels; no evaluation session, and no Flint, Area2 or NPCS session, is in the pretraining data. CO: center-out reaching; RT: random-target reaching. Hours are the span of each recording. † The Churchland recordings contain inter-trial gaps and cover 28.9 h of data.

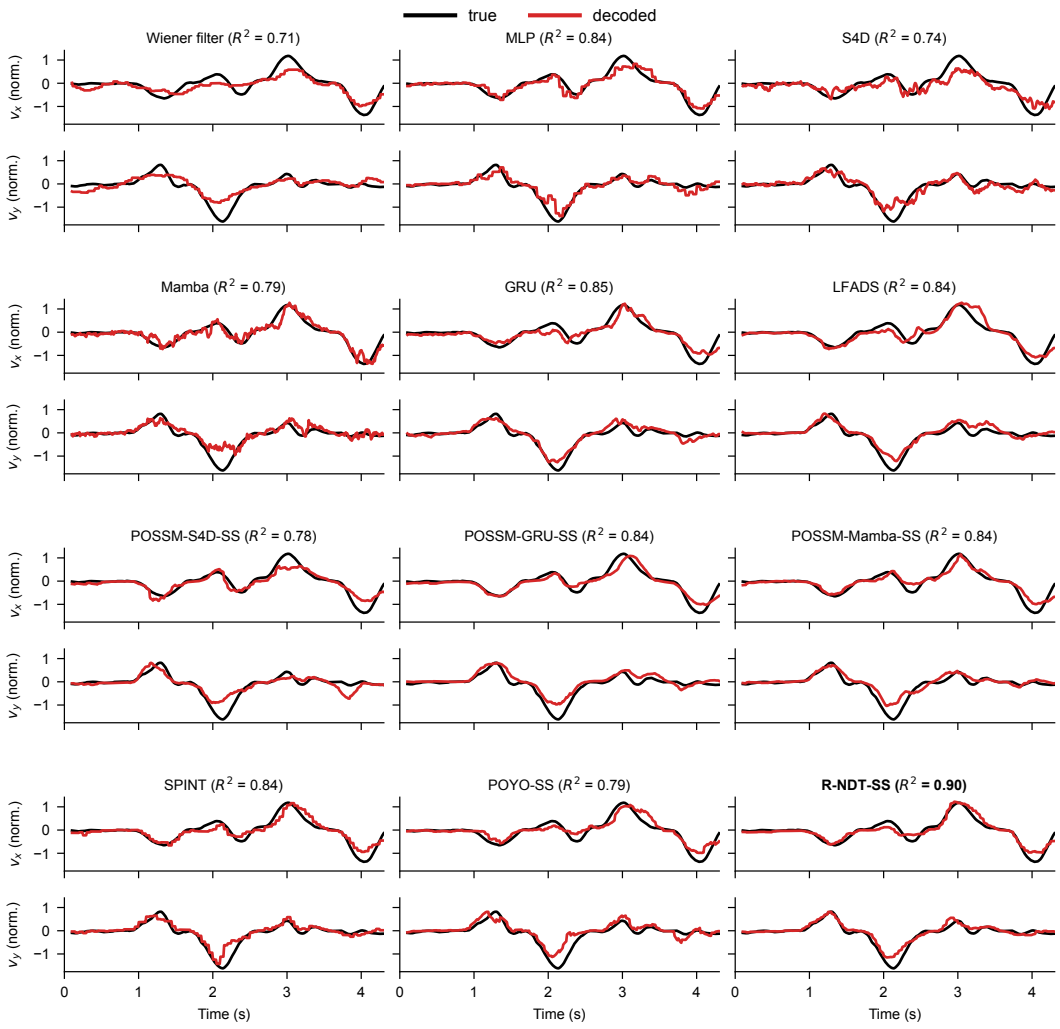


Figure 1: Decoded cursor velocity of every from-scratch model on a typical test trial. Each block shows v_x (top) and v_y(bottom) for three models; black is the true velocity, red the decoded velocity under per-trial causal protocol. The value in each title is the R^2 pooled over the shown trial.

C BASELINES

This section shows all the baseline architectures. Throughout, $\mathbf{x}_t \in \mathbb{Z}_{\geq 0}^N$ is the vector of counts of the N units in bin t, $\mathbf{y}_\tau \in \mathbb{R}^2$ the velocity at timestamp τ, and $t(\tau)$ the bin containing τ.

Shared recipe of the from-scratch binned rows. MLP, GRU, S4D, Mamba, LFADS and SPINT are trained by one trainer: 1,000 epochs at batch size 128, peak learning rate 4×10^{-3} obtained from a per-sample base rate of 3.125×10^{-5} by linear scaling with the batch size, the LAMB optimizer with weight decay 10^{-4}, a one-cycle schedule, validation every epoch and the best-validation checkpoint. The behavior targets of each window are subsampled to 100 queries as in POSSM. No unit dropout is applied to these rows. All use 1 s windows of 50 ms bins and share one read-in and one readout,

$$\mathbf{z}_t = \mathrm{LN}\big(\mathbf{A}\,\mathbf{x}_t\big) \in \mathbb{R}^{64}, \qquad \hat{\mathbf{y}}_\tau = \mathbf{W}_y\,\mathbf{h}_{t(\tau)} + \mathbf{b}_y, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{64 \times N}$ is a learned per-session read-in (one 64-dimensional embedding per unit, so each row is a per-unit parameter), $\mathbf{h}_t \in \mathbb{R}^{256}$ is the output of the model core at bin t, and $\mathbf{W}_y$ is linear; dropout 0.2 is applied to $\mathbf{z}_t$ and $\mathbf{h}_t$. The bin geometry and the one-bin emission latency are shared with POSSM. Each core has 0.2 to 0.3 M parameters.

Column	Session	R-NDT SS	R-NDT FT	POSSM-GRU SS	POSSM-GRU FT	POYO SS	POYO FT
C-CO 2016	C 2016-10-13	0.9406	0.9408	0.9507	0.9395	0.9325	0.9404
C-CO 2016	C 2016-10-21	0.9616	0.9621	0.9647	0.9679	0.9595	0.9636
Flint	Flint e1	0.8826	0.8855	0.8651	0.8836	0.8417	0.8758
Flint	Flint e2	0.7357	0.7364	0.7160	0.7262	0.6734	0.6849
Flint	Flint e3	0.8718	0.8655	0.8555	0.8429	0.8186	0.8603
Flint	Flint e4	0.6266	0.6425	0.6293	0.6187	0.5657	0.5802
Flint	Flint e5	0.8075	0.8010	0.8043	0.7756	0.7354	0.7870
T-CO	T 2013-08-19	0.8877	0.9048	0.8427	0.8737	0.8812	0.8881
T-CO	T 2013-08-21	0.8679	0.8982	0.8445	0.8885	0.7954	0.8293
T-CO	T 2013-08-23	0.9038	0.9184	0.8947	0.9148	0.9113	0.9073
T-CO	T 2013-09-03	0.8919	0.9180	0.8537	0.8901	0.8419	0.8986
T-CO	T 2013-09-05	0.9010	0.9277	0.9041	0.9040	0.8960	0.8982
T-CO	T 2013-09-09	0.9005	0.9150	0.8743	0.8919	0.8818	0.8990
T-RT	T 2013-08-20	0.7382	0.7606	0.7323	0.7369	0.7413	0.7569
T-RT	T 2013-08-22	0.8173	0.8337	0.7923	0.7995	0.7688	0.7794
T-RT	T 2013-08-30	0.6393	0.6845	0.6161	0.5996	0.5610	0.6412
T-RT	T 2013-09-04	0.8114	0.8102	0.7946	0.8181	0.7749	0.7905
T-RT	T 2013-09-06	0.8155	0.8360	0.7834	0.7875	0.7888	0.8037
T-RT	T 2013-09-10	0.8183	0.8330	0.8116	0.8179	0.7902	0.8314
Area2	Area2_Bump	0.8640	0.8653	0.8594	0.8605	0.8208	0.8584
Average over 20 sessions		**0.8342**	**0.8470**	0.8195	0.8269	0.7990	0.8237

Table 3: Per-session test R^2 behind Figure 3: the from-scratch (SS) and pretrained-then-fine-tuned (FT) arms of R-NDT, POSSM-GRU (POSSM-GRU-SS and o-POSSM-GRU (FT)) and POYO (POYO-SS and POYO-1 (FT)) on the 20 public evaluation sessions.

T-RT Session (2013)	08-20	08-22	08-30	09-04	09-06	09-10
Wiener filter	0.517	0.590	0.449	0.576	0.567	0.647
MLP-SS	0.617	0.712	0.503	0.716	0.696	0.733
GRU-SS	0.609	0.714	0.492	0.667	0.664	0.709
S4D-SS	0.481	0.552	0.354	0.607	0.524	0.655
Mamba-SS	0.511	0.598	0.387	0.634	0.594	0.679
LFADS-SS	0.607	0.722	0.506	0.718	0.692	0.721
SPINT-SS	0.680	0.751	0.553	0.740	0.739	0.760
POSSM-S4D-SS	0.613	0.721	0.492	0.264	0.608	0.658
POSSM-Mamba-SS	0.623	0.739	0.508	0.219	0.658	0.748
POSSM-GRU-SS	0.681	0.765	0.579	0.715	0.714	0.771
o-POSSM-S4D (UI)	0.415	0.427	0.313	0.440	0.416	0.478
o-POSSM-S4D (FT)	0.446	0.537	0.348	0.430	0.491	0.567
o-POSSM-GRU (UI)	0.616	0.712	0.550	0.703	0.682	0.732
o-POSSM-GRU (FT)	0.652	0.740	0.573	0.724	0.702	0.750
o-POSSM-Mamba (UI)	0.416	0.445	0.419	0.390	0.379	0.422
o-POSSM-Mamba (FT)	0.664	0.715	0.536	0.680	0.710	0.694
R-NDT-SS	0.714	0.787	0.593	0.761	0.745	0.788
R-NDT (FT)	0.689	0.777	0.634	0.747	0.763	0.785

Table 4: Every row of main paper Table 3 per T-RT session on the whole test block.

Wiener filter. A causal linear filter over the last K bins,

$$\hat{\mathbf{y}}_t = \mathbf{b} + \sum_{k=0}^{K-1} \mathbf{W}_k \, \tilde{\mathbf{x}}_{t-k}, \qquad (\mathbf{W}, \mathbf{b}) = \arg\min \sum_t \omega_t \big\| \hat{\mathbf{y}}_t - \mathbf{y}_t \big\|_2^2 + \lambda \|\mathbf{W}\|_F^2, \tag{2}$$

where $\tilde{\mathbf{x}}$ are the counts standardized per unit and tap, ω_t are the same per-epoch weights as for every other row, and $K \in \{5, 10, 20\}$ and λ are chosen on the validation split. The filter is fitted in closed form and is stateless, so its value is the same at every evaluation length.

MLP. The core concatenates the last $H = 20$ read-in tokens (1 s of history) and applies two GELU layers,

$$\mathbf{h}_t = \mathrm{GELU}\big(\mathbf{W}_2 \, \mathrm{GELU}(\mathbf{W}_1 [\mathbf{z}_{t-H+1}; \ldots; \mathbf{z}_t] + \mathbf{b}_1) + \mathbf{b}_2\big), \tag{3}$$

with hidden width 256, so the model is memoryless with a fixed 1 s receptive field.

GRU. One layer of width 256 with the state reset at the start of every sample,

$$\begin{aligned} \mathbf{r}_t &= \sigma(\mathbf{W}_r \mathbf{z}_t + \mathbf{U}_r \mathbf{h}_{t-1}), \quad \mathbf{u}_t = \sigma(\mathbf{W}_u \mathbf{z}_t + \mathbf{U}_u \mathbf{h}_{t-1}), \\ \mathbf{h}_t &= (1 - \mathbf{u}_t) \odot \mathbf{h}_{t-1} + \mathbf{u}_t \odot \tanh\big(\mathbf{W}_h \mathbf{z}_t + \mathbf{U}_h (\mathbf{r}_t \odot \mathbf{h}_{t-1})\big), \end{aligned} \tag{4}$$

biases omitted (Cho et al., 2014).

NPCS Site	045	019	020
Wiener filter	0.740	0.749	0.704
MLP-SS	0.869	0.854	0.867
GRU-SS	0.843	0.849	0.882
S4D-SS	0.645	0.723	0.703
Mamba-SS	0.837	0.841	0.849
LFADS-SS	0.865	0.863	0.853
SPINT-SS	0.885	0.870	0.894
POSSM-S4D-SS	0.663	0.706	0.648
POSSM-Mamba-SS	0.819	0.818	0.892
POSSM-GRU-SS	0.883	0.871	0.884
o-POSSM-S4D (UI)	0.605	0.608	0.524
o-POSSM-S4D (FT)	0.785	0.689	0.777
o-POSSM-GRU (UI)	0.814	0.796	0.817
o-POSSM-GRU (FT)	0.822	0.854	0.881
o-POSSM-Mamba (UI)	0.829	0.837	0.602
o-POSSM-Mamba (FT)	0.875	0.870	0.904
R-NDT-SS	0.891	0.880	0.917
R-NDT (FT)	0.898	0.878	0.919

Table 5: Every row of main paper Table 3 on the three NPCS sites, whole test block.

S4D. One residual block, $\mathbf{h}_t = \mathbf{z}'_t + \mathbf{W}_o\,\mathrm{GELU}(\mathrm{S4D}(\mathrm{LN}(\mathbf{z}')))_t$ with $\mathbf{z}'_t = \mathbf{W}_i\mathbf{z}_t \in \mathbb{R}^{256}$, where each channel of the S4D layer is a diagonal linear state-space model (Gu et al., 2021),

$$\mathbf{s}_t = \bar{\mathbf{A}}\,\mathbf{s}_{t-1} + \bar{\mathbf{B}}\,z'_t, \qquad o_t = \mathrm{Re}\big(\mathbf{C}^\top \mathbf{s}_t\big) + D\,z'_t, \qquad \bar{\mathbf{A}} = \exp(\Delta_s \mathbf{A}), \;\; \mathbf{A} = \mathrm{diag}(a_1, \ldots, a_{64}), \tag{5}$$

with the S4D-Lin initialization $a_n = -\frac{1}{2} + i\pi n$, a learned step size Δ_s per channel, and the recurrence evaluated in training as a convolution by FFT over the window. Its state is 64 complex numbers per channel.

Mamba. One selective state-space block (Gu & Dao, 2023) with state size 16, in which the step size and the input and output projections are functions of the input,

$$\Delta_t = \mathrm{softplus}(\mathbf{w}_\Delta^\top \mathbf{z}'_t + b_\Delta), \quad \mathbf{B}_t = \mathbf{W}_B\mathbf{z}'_t, \quad \mathbf{C}_t = \mathbf{W}_C\mathbf{z}'_t, \quad \mathbf{s}_t = \exp(\Delta_t\mathbf{A})\,\mathbf{s}_{t-1} + \Delta_t\mathbf{B}_t\,z'_t,$$
$$o_t = \mathbf{C}_t^\top \mathbf{s}_t + D\,z'_t, \tag{6}$$

preceded by a causal convolution of width 4 and gated by a parallel SiLU branch, with expansion factor 2.

LFADS (unidirectional). Traditional LFADS backbone uses bidirectional GRUs, thus is not causal. To make it causal, we replace the bidirectional GRU with the unidirectional GRU and pretrain it with a Poisson observation model,

$$\begin{aligned} \mathbf{e}_t &= \mathrm{GRU}_{\mathrm{enc}}(\mathbf{z}_t, \mathbf{e}_{t-1}), \qquad \mathbf{u}_t \sim \mathcal{N}\big(\boldsymbol{\mu}(\mathbf{e}_t), \mathrm{diag}\,\boldsymbol{\sigma}^2(\mathbf{e}_t)\big) \in \mathbb{R}^{32}, \\ \mathbf{g}_t &= \mathrm{GRU}_{\mathrm{gen}}(\mathbf{u}_t, \mathbf{g}_{t-1}), \qquad \mathbf{f}_t = \mathbf{W}_f\,\mathbf{g}_t \in \mathbb{R}^{40}, \qquad \log r_{t,i} = \mathbf{w}_i^\top \mathbf{f}_t + b_i, \\ \hat{\mathbf{y}}_\tau &= \mathbf{W}_y \mathbf{f}_{t(\tau)} + \mathbf{b}_y, \end{aligned} \tag{7}$$

where $\mathrm{GRU}_{\mathrm{enc}}$ (width 128) runs forward only, $\mathbf{u}_t$ is the inferred input sampled by the reparameterization trick in training and set to its mean at test, $\mathrm{GRU}_{\mathrm{gen}}$ (width 200) has a learned initial state, $\mathbf{W}_f$ is bias-free, and $(\mathbf{w}_i, b_i)$ is a per-unit rate readout. The loss is

$$\mathcal{L} = \mathcal{L}_{\mathrm{dec}} + 0.1\,\frac{1}{|\mathcal{T}|}\sum_{(t,i)\in\mathcal{T}} \big(r_{t,i} - x_{t,i}\log r_{t,i}\big) + \beta_{\mathrm{step}}\,\mathrm{KL}\big(q(\mathbf{u}_t \mid \mathbf{e}_t)\,\|\,\mathcal{N}(\mathbf{0}, \mathbf{I})\big), \tag{8}$$

with $\mathcal{L}_{\mathrm{dec}}$ the weighted velocity error of equation 9, $\mathcal{T}$ the valid (bin, unit) entries, and β ramped linearly from 0 to 10^{-4} over the first 2,000 steps. The whole model has 0.23 M dense parameters against the GRU row's 0.26 M.

SPINT. Follows the authors' implementation (Le et al., 2025) line by line. Each unit is a token whose feature is its window of the last $W = 20$ counts, offset by a context-dependent identity computed from unlabeled calibration segments $\{\mathbf{C}^{(m)}\}_{m=1}^{M}$ of the training split,

$$\mathbf{E}_i = \psi\Big(\tfrac{1}{M}\textstyle\sum_{m=1}^{M}\phi\big(\mathbf{C}^{(m)}_{:,i}\big)\Big) \in \mathbb{R}^W, \qquad \mathbf{Z}_{t,i} = d_i\,\big(\mathbf{x}_{t-W+1:t,\,i} + \mathbf{E}_i\big), \tag{9}$$

where ϕ and ψ are multi-layer perceptrons, $d_i \in \{0, 1/(1-p)\}$ is a channel-dropout mask with $p \sim \mathcal{U}(0, 1.0)$ per sample, and the tokens of a bin are read by a single learnable query,

$$\mathbf{h}_t = \mathrm{CrossAttn}\big(\mathbf{W}_{\mathrm{in}}\mathbf{q}_0;\ \{\mathbf{W}_{\mathrm{in}}\mathbf{Z}_{t,i}\}_{i=1}^{N}\big), \qquad \hat{\mathbf{y}}_\tau = \mathbf{W}_y\,\mathbf{h}_{t(\tau)} + \mathbf{b}_y, \tag{10}$$

one pre-norm cross-attention layer (width 256, 8 heads, feed-forward width 1,024, dropout 0.1) with the read-in $\mathbf{W}_{\mathrm{in}}$ shared by the query and the unit tokens. The model has 1.14 M parameters and no per-unit parameters; it is memoryless with a fixed 1 s history. Calibration is 64 fixed 2 s segments taken evenly from the training split, of which a random 32 are used per training step and an evenly spread 32 at evaluation. Departures from the released configuration are 50 ms instead of 20 ms bins, fixed-length segments instead of interpolated calibration trials (the random-target and NPCS sessions are not cleanly trialized) and the shared trainer above in place of the Adam optimizer.

POYO-SS and POYO-1. POYO (Azabou et al., 2023) tokenizes every spike: the j-th spike of the window, emitted by unit i_j at time t_j, is the token $\mathbf{x}_j = \mathrm{UnitEmb}(i_j)$ carrying timestamp t_j, where $\mathrm{UnitEmb}$ is a learned table with one row per unit. A set of N_z learned latents compresses the M spike tokens and behavior is read by timestamp queries,

$$\begin{aligned} \mathbf{Z}_1 &= \mathrm{CrossAttn}\big(\mathbf{Z}_0;\ \{\mathbf{x}_j\}_{j=1}^{M}\big), \quad \mathbf{Z}_{l+1} = \mathrm{SelfAttn}(\mathbf{Z}_l), \\ \hat{\mathbf{y}}_\tau &= \mathbf{W}_y\,\mathrm{CrossAttn}\big(\mathbf{q}_\tau;\ \mathbf{Z}_L\big), \quad \mathbf{q}_\tau = \mathbf{e}_{\mathrm{task}} + \mathrm{SessionEmb}(s), \end{aligned} \tag{11}$$

with rotary encoding of the timestamps in every attention. Because the latents attend to the whole 1 s window, the readout is not causal; test windows are therefore anchored at their end and only their final 50 ms is scored, without which the model averages in up to 0.5 s of future activity. POYO-SS is the 1.3 M-parameter configuration (width 64, depth 6, 16 latents per 125 ms) trained per session for 1,000 epochs at batch size 128 and peak rate 4×10^{-3} with POSSM's unit-dropout and output-subsampling transforms. POYO-1 is the 11.8 M-parameter configuration (width 128, depth 24) pretrained by us on the same 142-session corpus as o-POSSM for 500 epochs, then adapted per session for 500 epochs at peak rate 4×10^{-3}: under unit identification (UI) only the new rows of $\mathrm{UnitEmb}$ and $\mathrm{SessionEmb}$ are trained; under full fine-tuning (FT) the model is unfrozen after 100 epochs of UI.

POSSM-SS and o-POSSM. POSSM (Ryoo et al., 2025) keeps POYO's spike tokens but compresses each 50 ms chunk c separately into one latent and integrates the chunks with a recurrent backbone,

$$\begin{aligned} \mathbf{z}^{(c)} &= \mathrm{CrossAttn}\big(\mathbf{q};\ \{\mathbf{x}_j : t_j \in \text{chunk } c\}\big), \qquad \mathbf{h}^{(c)} = f_{\mathrm{RNN}}\big(\mathbf{z}^{(c)}, \mathbf{h}^{(c-1)}\big), \\ \hat{\mathbf{y}}_\tau &= \mathbf{W}_y\,\mathrm{CrossAttn}\big(\mathbf{q}_\tau;\ \mathbf{h}^{(c(\tau)-2\,:\,c(\tau))}\big), \end{aligned} \tag{12}$$

where $\mathbf{q}$ is a learned query, f_{RNN} is a four-layer GRU, S4D or Mamba stack of width 512 in the same slot (so the three variants are capacity-matched), and the behavior query attends to the three most recent chunk states, which is the causal boundary R-NDT adopts. POSSM-SS uses the authors' single-session configuration (width 256, one latent per chunk), 1,000 epochs at batch size 128 and peak rate 4×10^{-3}, with unit dropout and 100 output queries per window. o-POSSM is pretrained by us with behavior labels on 142 sessions (Perich 97, O'Doherty 44, NLB Maze 1; 24,286 units, 37.6 h of training data). Adaptation follows the authors' procedure: fresh unit and session embeddings are initialized for the new session and trained with the rest of the model frozen (UI), or the whole model is unfrozen after 100 epochs of UI and trained for 400 more (FT), 500 epochs in total at a constant rate. Appendix A compares the published values with this reproduction. **NDT-2 (FT)** uses the model checkpoint from the same POSSM paper but later fine-tuned by us for evaluations.

D THE NPCS DATASET

The dataset was collected at the National Primate Research Center; all procedures were conducted in compliance with the NIH Guide for the Care and Use of Laboratory Animals and were approved by the Institutional Animal Care and Use Committee, and details of the recording setup are in work (Canfield et al., 2026).

E MODEL AND TRAINING DETAILS

This section gives the values of every symbol of Section 3 and the implementation details omitted there. Spikes are binned at $\Delta = 50$ ms, a window is $T_w = 20$ bins (1 s), and the behavior is

Site	Units	Minutes	x (mm)	y (mm)
020	261	10.2	0.00	4.30
019	445	61.3	-1.07	4.30
045	197	33.8	2.15	2.15

Table 6: The 3 NPCS insertion sites. Units are Kilosort-good units; Minutes stands for the session length; x, y are positions on the insertion grid (1 grid unit = 1.0745 mm). It holds 903 sorted units (197 to 445 per site), 8.0 M spikes, and 1.76 h of recording of a center-out cursor task with behavior sampled at 1 kHz. Different from the Utah arrays, the Neuropixels recordings have more units per site and lower firing rates.

two-dimensional cursor or hand velocity ($d_y = 2$). The pretrained model has width $D = 256$ and $L = 6$ blocks (7.9 M parameters plus a 0.55 M readout); R-NDT-SS has $D = 64$ and $L = 2$ (0.32 M parameters, readout included). The neural context has $d_E = 64$ dimensions and is computed once per session from the earliest 8 s of its training split. Table 7 lists the dimensions of the two model sizes; the training recipes are in Appendix F.

Symbol	Meaning	Pretrained / R-NDT (FT)	R-NDT-SS
D	model width	256	64
L	backbone blocks	6	2
–	attention heads $\times$ head width d_h	8×32	2×32
T_w	attention and training window (bins)	20	20
F	statistics per unit	21	21
L_E, –	set-encoder layers, hidden width	2, 128	2, 128
d_E	neural-context width	64	64
K, d_m	memory bands, band width	4, 64	4, 64
η_κ	memory half-lives (s)	5, 20, 60, 180	5, 20, 60, 180
k	states attended by the behavior query	3	3
k'	states attended by the reconstruction query	4	–
–	parameters	7.9 M + 0.55 M readout	0.32 M

Table 7: Values of the symbols.

E.1 Neural context

Statistics. For a calibration segment $\mathbf{C} \in \mathbb{Z}_{\geq 0}^{T_{\text{cal}} \times N}$, with $\bar{x}_i$ the mean count of unit i, the $F = 21$ entries of $\boldsymbol{\phi}_i$ are: (1) $\log(1 + \bar{x}_i)$; (2–4) the Fano factor of the counts aggregated over 1, 2 and 4 consecutive bins; (5) the coefficient of variation of the counts; (6–7) the fraction of bins with at least one and with at least two spikes; (8–12) the autocorrelation at lags 1 to 5 bins; (13–19) the correlation with the leave-one-out population mean $\bar{\mathbf{c}}_{-i}$ at lags -3 to $+3$ bins; (20) the 90th-percentile count; and (21) the rate trend $\log(1+\bar{x}_i^{\text{2nd half}}) - \log(1+\bar{x}_i^{\text{1st half}})$. A statistic with fewer than three usable samples is set to zero. The only cross-unit quantity is the population mean, which is symmetric in the units. **Standardization.** $\tilde{\boldsymbol{\phi}}$ is a per-feature z-score with running corpus statistics (momentum 0.01) that are updated during pretraining only and frozen afterwards; they are buffers, not parameters. **Set encoder.** Two layers of equation 3 with GELU and widths $21 \to 128 \to 128$, followed by the linear map $\mathbf{U} : 128 \to 64$; 47 k parameters. When sessions of different sizes are batched, the mean in equation 3 runs over the real units only. **Calibration segment.** During pretraining, whenever a training stream enters a recording, a length is drawn uniformly from $\{1, 2, 4, 8\}$ s and floored to the longest of these values that the chosen interval of the training split can supply, and the start is drawn uniformly within that interval. At adaptation and at test the segment is the earliest 8 s of the session's training split, and $\boldsymbol{\phi}$ is computed once per session. During pretraining $\mathbf{E}$ is computed from the whole population, including the units that unit dropout hides, because the reconstruction head queries them by $\mathbf{E}_i$ (Equation 10); during adaptation, units removed by the dropout augmentation are also removed from the mean of equation 3 and from $\bar{\mathbf{E}}$.

E.2 Backbone

Each block is pre-norm with an affine-free layer normalization (the affine transformation is supplied by $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ of Equation 7), bias-free query, key, value and output projections, and a feed-forward

network $\mathrm{Linear}(D, 4D) \to \mathrm{GELU} \to \mathrm{Linear}(4D, D)$. Queries and keys are RMS-normalized per head and then rotated by rotary position encoding of the absolute bin index with base 30 at head width 32; the base is chosen so that over the 19-bin span of the attention window every rotary band turns between 0.78 and 19 rad, so none is constant and none aliases. Keys are cached after normalization and rotation, which makes windowed and single-pass processing identical. The cache of each layer is trimmed to the last $T_w - 1 = 19$ positions, so bin t attends to bins $t - 19, \ldots, t$ at every layer and the receptive field after $L = 6$ blocks is 115 bins. A final affine-free layer normalization produces $\mathbf{h}_t$. Dropout acts on the feed-forward and attention paths of the backbone only; the read-in, the neural-context encoder and both heads have none.

E.3 Drift memory

With $\Delta = 50$ ms the half-lives $\eta = (5, 20, 60, 180)$ s give per-bin rates $a = (6.91, 1.73, 0.577, 0.193) \times 10^{-3}$. A whole window is advanced in one closed-form step equivalent to the per-bin recursion of equation 6, $\mathbf{m}_{\mathrm{out}}^{(\kappa)} = (1 - a_\kappa)^n \mathbf{m}_{\mathrm{in}}^{(\kappa)} + \sum_t a_\kappa (1 - a_\kappa)^{n-1-t} \mathbf{z}_t^{(\kappa)}$ over the n valid bins of the window; padded bins neither contribute nor advance the decay. The memory is kept in 32-bit floating point even under reduced-precision training: in bfloat16 three of the four retention factors $1 - a_\kappa$ round to exactly one, and the corresponding bands would neither decay nor update. The conditioning network is $g(\mathbf{m}) = \mathbf{W}_2 \,\mathrm{SiLU}\big(\mathbf{W}_1 \,\mathrm{LN}(\mathrm{vec}(\mathbf{m})) + \mathbf{b}_1\big) + \mathbf{b}_2$ with dropout 0.1 on the hidden layer, and the modulation projection of every block is zero-initialized in both weight and bias, so the backbone starts as an unmodulated residual network.

E.4 Heads

Behavior readout. The query attends to the $k = 3$ backbone states of the bin containing τ and the two before it, which is the causal boundary of POSSM (Ryoo et al., 2025): the lookahead is at most one bin. Queries and keys are RMS-normalized per head; the relative position of the query, $\tau/\Delta - t(\tau) \in [-1, 0)$, and those of the keys, $(-2, -1, 0)$, are encoded rotarily. The feed-forward layer has hidden width $2D$ and $\mathbf{W}_y$ is a linear map to $\mathbb{R}^{d_y}$. ω_τ follows the per-epoch weights of the dataset configurations of Ryoo et al. (2025) (for Perich center-out sessions: reach 5, return and random 1, hold 0.1; the NPCS configuration uses the same values and 0.1 for invalid trials; Flint has uniform weights).

Reconstruction head. The query $\mathbf{W}_q \mathbf{E}_i$ attends to the $k' = 4$ states $\mathbf{h}_{t-3}, \ldots, \mathbf{h}_t$, each with a learned slot embedding, with query-key normalization and no feed-forward layer; the output projection and $\mathbf{w}_\lambda$ are folded into one vector per head so that no tensor of size $T \times N \times D$ is formed. $\mathcal{M}$ is a uniform random subsample of at most 4,096 hidden entries per training sequence, the constant $\log x!$ is dropped, and the likelihood is computed in 32-bit floating point. Validation reports $\mathcal{L}_{\mathrm{SSL}}$ against three floors: the per-unit mean rate, persistence of the previous count, and the mean rate estimated from the calibration segment alone.

E.5 Masking and the training loop

Masks. Pretraining uses $\rho_{\max} = 1.0$, $p_b = 0.5$ per window with ℓ_b drawn uniformly from $\{1, 2, 4, 8\}$ bins (50 to 400 ms) and the start s drawn uniformly among the positions that keep the run inside the window and leave its first bin visible, and $p_{\mathrm{iid}} = 0.1$. The unit-dropout pattern is drawn when a stream enters a recording and held until it leaves; it is never allowed to hide every unit. During adaptation only unit dropout is used, redrawn for every window of every batch with $\rho \sim \mathcal{U}(0, 1.0)$. **Pretraining loop.** A step processes 32 persistent streams, each advancing through 4 consecutive windows of one recording in time order. Gradients flow through the memory and the attention cache across the 4 windows of a step and are truncated at the step boundary: the memory is detached and carried to the next step, while the attention cache is rebuilt. The memory is reset whenever a stream enters a new recording or a new contiguous segment of its training split. **Adaptation loop.** In the trial-based causal evaluations, adaptation samples are independent 1 s windows; In whole-block causal evaluations, adaptation streaming sample length is the same as in pretraining.

E.6 PROOFS

Proposition 1. Every entry of $\boldsymbol{\phi}_i$ is a function of the unit's own column $\mathbf{C}_{:,i}$ and of $\bar{\mathbf{c}}_{-i} = (\sum_j \mathbf{C}_{:,j} - \mathbf{C}_{:,i})/(N_s - 1)$, in which the sum is symmetric in the units; hence $\boldsymbol{\phi}(\pi\mathbf{C}) = \pi\boldsymbol{\phi}(\mathbf{C})$, and the per-feature standardization preserves this. Each layer of equation 3 applies the same map to every unit and a symmetric mean, so it is permutation equivariant, and so is the pointwise map $\mathbf{U}$; by composition $f_E(\pi\mathbf{C}) = \pi f_E(\mathbf{C})$. In equation 4 the pair $(x_{t,i}, \mathbf{E}_i)$ moves with the unit under π, $\mathcal{V}_t$ is mapped to $\pi(\mathcal{V}_t)$ with the same cardinality, and a sum over a set does not depend on the order of its elements, so $\mathbf{u}_t$ is unchanged. Finally, the parameters are $\{\mathbf{W}_\ell, \mathbf{V}_\ell\}$, $\mathbf{U}$, $\mathbf{W}'$, whose shapes involve only F, d_E, D and the set-encoder width. □

F TRAINING RECIPES AND COMPUTE

Tables 8, 9 and 10 list the three recipes. We use 3 NVIDIA A40 GPUs for model training; pretraining takes roughly 2 to 3 h per 20k steps.

Pretraining	
Model	$D = 256$, $L = 6$, 8 heads $\times$ 32; 7.9 M parameters
Initialization	random
Objective	Poisson NLL on masked counts (Equation 11)
Corpus	156 public sessions, no evaluation session
Steps	100k
Batch	32 persistent streams $\times$ 4 consecutive windows
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$), weight decay 0.01, gradient clip 1.0
Learning rate	peak 2×10^{-4}, warm-up 1,000 steps, cosine to 5 %
Dropout	0.1 (backbone)
Unit dropout	$\rho \sim \mathcal{U}(0, 1.0)$, drawn per stream on entering a recording
Blackout	$\ell_b \in \{1, 2, 4, 8\}$ bins with probability 0.5 per window
i.i.d. mask	0.1
Calibration segment	length in $\{1, 2, 4, 8\}$ s, random position, resampled per recording

Table 8: Pretraining recipe.

R-NDT-SS (from scratch)	
Model	$D = 64$, $L = 2$, 2 heads $\times$ 32; 0.32 M parameters including the readout
Initialization	random
Objective	weighted MSE on velocity (Equation 9)
Steps	10k, validation every 500, best-validation checkpoint
Batch	128 windows
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$), weight decay 0.01, gradient clip 1.0
Learning rate	peak 10^{-3}, warm-up 100 steps, cosine to 5 %
Dropout	0.2 (backbone)
Unit dropout	$\rho \sim \mathcal{U}(0, 1)$, redrawn per window
Calibration segment	earliest 8 s of the training split, computed once

Table 9: From-scratch recipe.

R-NDT (FT) (full fine-tuning)	
Model	$D = 256$, $L = 6$, 8 heads $\times$ 32; 7.9 M parameters + 0.55 M readout
Initialization	pretraining checkpoint at 100k steps; all parameters trained
Objective	weighted MSE on velocity (Equation 9)
Steps	10k, validation every 500, best-validation checkpoint
Batch	128 windows
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$), weight decay 0.01, gradient clip 1.0
Learning rate	peak 10^{-3}, warm-up 100 steps, cosine to 5 %
Dropout	0.7 (backbone), chosen on validation
Unit dropout	$\rho \sim \mathcal{U}(0, 1)$, redrawn per window
Calibration segment	earliest 8 s of the training split, computed once

Table 10: Fine-tuning recipe.