Know Thyself by Knowing Others: Learning Neuron Identity from Population Context

Vinam Arora[1†], **Divyansha Lachi**[1], **Ian J. Knight**[1], **Mehdi Azabou**[2], **Blake Richards**[3,4], **Cole Hurwitz**[2], **Joshua H. Siegle**[5], **Eva L. Dyer**[1†]
[1]University of Pennsylvania, [2]Columbia University,
[3]McGill University, [4]Mila, [5]Allen Institute for Neural Dynamics

Abstract

Neurons process information in ways that depend on their cell type, connectivity, and the brain region in which they are embedded. However, inferring these factors from neural activity remains a significant challenge. To build general-purpose representations that allow for resolving information about a neuron's identity, we introduce `NuCLR`, a self-supervised framework that aims to learn representations of neural activity that allow for differentiating one neuron from the rest. `NuCLR` brings together views of the same neuron observed at different times and across different stimuli and uses a contrastive objective to pull these representations together. To capture population context without assuming any fixed neuron ordering, we build a spatiotemporal transformer that integrates activity in a permutation-equivariant manner. Across multiple electrophysiology and calcium imaging datasets, a linear decoding evaluation on top of `NuCLR` representations achieves a new state-of-the-art for both cell type and brain region decoding tasks, and demonstrates strong zero-shot generalization to unseen animals. We present the first systematic scaling analysis for neuron-level representation learning, showing that increasing the number of animals used during pretraining consistently improves downstream performance. The learned representations are also label-efficient, requiring only a small fraction of labeled samples to achieve competitive performance. These results highlight how large, diverse neural datasets enable models to recover information about neuron identity that generalize across animals. Code is available at `https://github.com/nerdslab/nuclr`.

1 Introduction

Neurons operate in ways that reflect their cell type, connectivity, and the brain region in which they are embedded [55, 46, 37]. Understanding how these structural and physiological factors shape neural activity is an important open question [37] and is central to modern theories of neural function [16, 10]. Cell type- and circuit-specific effects are also increasingly implicated in neurological disorders, where it has been shown that particular neuronal subtypes can be selectively affected in neurodegenerative disease [33, 29] and that neuron subtypes can be targeted to improve efficacy in real time in closed-loop brain stimulation [57]. Together, these directions highlight a broader need to understand how features such as cell type and brain region contribute to a neuron's role in the brain, motivating the development of methods that can extract aspects of neuron identity.

Much of what we know about neuronal cell types and their connectivity comes from molecular, anatomical, and morphological labeling techniques that tag neurons based on gene expression profiles

[†]Contact: vinam@upenn.edu, dyer1@upenn.edu
Reviewed on OpenReview: `https://openreview.net/forum?id=zt3RKc6VBp`.

or structural features [55, 45, 12, 40]. These approaches have revealed a rich taxonomy of neuronal classes, but they are resource intensive and typically provide only a sparse labeling of one or a few classes at a time. Currently, there is no molecular access to different cell types or projection classes in nonhuman primates and humans [17, 32, 22], which has created major roadblocks in studying neuron diversity in humans. For these reasons, there is a growing interest in methods that can infer aspects of neuronal identity directly from large-scale neural population activity.

Recent work has started to explore deep learning approaches for tackling this question [38, 30, 46, 6, 54, 52]. Early methods relied on intrinsic single-neuron features such as waveform shape or interspike interval profiles [38, 46, 6, 54], but these signals capture only part of what distinguishes one neuron from another. More recent approaches have attempted to incorporate some information about the ongoing population activity [30, 52], but reduce the surrounding activity to a fixed low-dimensional summaries, which obscures the fine-grained coordination patterns that are often key to neuronal identity. Furthermore, these models usually require retraining or finetuning on each new set of neurons, limiting their ability to generalize in a zero-shot manner across animals. These limitations point to the need for methods that leverage rich population context while producing neuron-level representations that transfer robustly across recordings.

To fill this gap, we introduce `NuCLR`, a self-supervised approach for learning neuron-level representations from large-scale, neural activity datasets. At a high-level, the goal of `NuCLR` is to build a representation that is stable across time and makes each neuron distinguishable from other neurons in the population, thereby capturing abstract features that reflect its identity. Within these representations, we expect to uncover information related to cell type, brain region, connectivity, and potentially other latent attributes. To learn these representations in a self-supervised manner, `NuCLR` constructs two population-level views by sampling and encoding different temporal segments from the same recording, and applies a contrastive objective that treats representations of the same neuron across these views as positives and all other neurons in the population as negatives. To allow for integration of rich population context in each neuron's representation, we introduce a spatiotemporal transformer that aggregates information from surrounding neurons without assuming a fixed ordering or requiring session-specific alignment. Together, these components allow `NuCLR` to learn population-aware representations that generalize reliably across animals, enabling zero-shot decoding of neuron-level attributes from previously unseen recordings.

To evaluate the identity-related information learned by `NuCLR`, we first pretrain the model in an unsupervised manner and then train linear classifiers on the resulting representations to decode cell type and brain region. Across multiple electrophysiology and calcium imaging datasets, `NuCLR` achieves a new state-of-the-art on both tasks. We show that `NuCLR`'s population-based self-supervision yields representations that transfer robustly across sessions and animals in a *zero-shot* manner, supporting decoding neuron identities on entirely new subjects without retraining or additional metadata. In addition to strong zero-shot generalization, `NuCLR` is highly label efficient: using only 12.5% of labels to train the classifier still outperforms all baselines even when they use all labeled data. We further provide the first systematic scaling analysis for neuron-level tasks, demonstrating that increasing the number of animals used in pretraining consistently improves zero-shot cell type and region decoding accuracy. On the Allen Visual Coding Neuropixels dataset [41], doubling the unlabeled pretraining data improves cell-type decoding more than doubling the supervised labels, highlighting the advantages of scaling pretraining across many animals.

Our core contributions are:

- We introduce `NuCLR`, a self-supervised framework for learning neuron-level representations from neural population activity. After unsupervised pretraining, these representations support linear decoding of cell type and brain region and set a new state-of-the-art across multiple electrophysiology and calcium imaging datasets.
- We show that `NuCLR` generalizes in a *zero-shot* manner: the same pretrained model and embeddings transfer robustly to entirely new sessions and animals without retraining or requiring additional metadata, enabling out-of-the-box decoding of cell type and brain region.
- We provide the first systematic scaling analysis for neuron-level representation learning, demonstrating that increasing the number of animals used during pretraining yields consistent gains in zero-shot cell-type and region decoding. This underscores the value of large, diverse unlabeled neural corpora for inferring neuronal identity from activity alone.

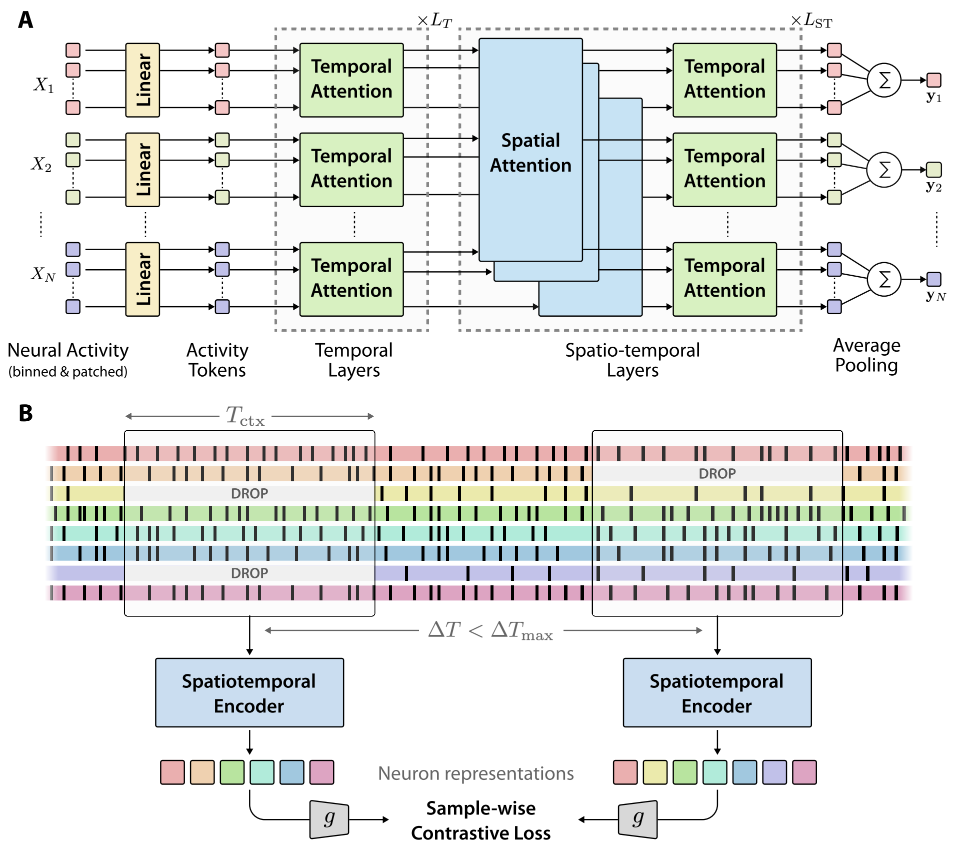

Figure 1: **Overview of the NuCLR framework.** (A) The model takes as input the activity of a neural population over a fixed context window. Each neuron's activity is treated as a patched-token sequence and encoded across time using temporal transformer blocks. These temporally encoded tokens are then passed through spatio-temporal transformer layers that attends across neurons to incorporate population context. Finally, the model outputs one vector for each neuron in the population. (B) The resulting representations are trained with a sample-wise contrastive objective. We sample two temporally-spaced views of the same population and encode them into neuron-level representations. Neurons are also randomly dropped before encoding to build robustness to partial observations. Representations of the same neuron in both views are pulled closer, and all other neurons in the population are considered negatives and pushed apart. As a result, each neuron's representation is encouraged to be stable across time and distinguishable from other neurons in the population.

2 Method

The goal of this work is to design and train deep learning models that, given the activity of a population of neurons, produce a latent representation for each neuron that captures its identity within the population. Our approach is grounded in two core principles:

1. Capturing a neuron's identity requires not only its own activity but also the surrounding population activity, which provides essential contextual information.
2. A neuron's identity remains stable over time, so representations computed from different temporal segments of the population activity should be consistent with one another.

Designing models that effectively incorporate population context is challenging because the number of recorded neurons varies widely across experiments, individuals, and sessions, making it non-trivial to define a consistent input structure or population-level operation. Recordings also span multiple experimental conditions—including different stimuli and behaviors—which makes it hard for a model to generalize across recording sessions. Together, these sources of variability make it difficult to learn neuron-level representations that are truly general-purpose and transferable to new recordings. To satisfy our principles while accommodating these sources of variability, we introduce a model architecture (Section 2.1) and training objective (Section 2.2).

2.1 Model architecture

We aim to learn a function that maps the collective activity of a neural population to general-purpose neuron-level representations. To support learning across recordings from different sessions and animals, each with a varying number of neurons, this function must be applicable to populations of arbitrary size. Moreover, we do not assume any relational bias between neurons, and treat them as a permutation-equivariant set where all neuron-neuron interactions are modeled. Finally, to enable representations that reflect a neuron's role within the circuit, the function should allow for information exchange among neurons, allowing each one to contextualize its activity relative to the population. We formalize this as a set-to-set transformation:

$$\{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\} = \mathcal{F}\Big(\{X_1, X_2, \dots, X_N\}\Big), \tag{1}$$

where X_n denotes the spike-train sequence of the n-th neuron, and $\mathbf{y}_n$ is its corresponding representation. We implement $\mathcal{F}$ using attention-based transformer blocks (Figure 1A), which naturally support set-to-set mappings through their permutation-equivariant structure [24].

Given spike trains of a population of N neurons over a temporal context window T_{ctx}, we begin by binning the spike train[1], and partitioning the bins into non-overlapping temporal patches of length T_{patch}. The resulting binned-and-patched activity of n-th neuron is a sequence of vectors $X_n = (\mathbf{x}_{n,1}, \mathbf{x}_{n,2}, \dots, \mathbf{x}_{n,P})$, where $P = T_{\text{ctx}}/T_{\text{patch}}$ is the number of patches. Each patch is linearly projected into a D-dimensional latent space, yielding a token sequence $Z_n^{(0)} = (\mathbf{z}_{n,1}^{(0)}, \dots, \mathbf{z}_{n,P}^{(0)})$ for each neuron.

These latent tokens are first processed independently per neuron using L_T layers of self-attention, which we refer to as **temporal transformer layers**. Each temporal layer $\mathcal{T}_{\text{temp}}$ operates on the patch sequence of a single neuron:

$$(\mathbf{z}_{n,1}^{(l+1)}, \dots, \mathbf{z}_{n,P}^{(l+1)}) = \mathcal{T}_{\text{temp}}\left((\mathbf{z}_{n,1}^{(l)}, \dots, \mathbf{z}_{n,P}^{(l)})\right), \quad \forall\, n \in [N]. \tag{2}$$

Within these layers, the temporal structure is maintained by the use of rotary position embeddings [44, 3], which encode the relative timing of each patch without requiring absolute positional indices.

Following the stack of temporal layers, we apply L_{ST} layers of **spatio-temporal transformer layers**, which alternate between spatial and temporal attention (Figure 1A). In the spatial transformer layer, tokens at the same time index interact across the population via a shared transformer $\mathcal{T}_{\text{spatial}}$:

$$\{\mathbf{z}_{1,p}^{(l+1)}, \dots, \mathbf{z}_{N,p}^{(l+1)}\} = \mathcal{T}_{\text{spatial}}\left(\{\mathbf{z}_{1,p}^{(l)}, \dots, \mathbf{z}_{N,p}^{(l)}\}\right) \quad \forall\, p \in [P]. \tag{3}$$

No positional embeddings were used in the spatial blocks. The temporal blocks in these layers reuse the structure defined in Equation (2). So, $\mathcal{T}_{\text{spatiotemporal}} = \mathcal{T}_{\text{spatial}} \circ \mathcal{T}_{\text{temp}}$.

Finally, to obtain a fixed-dimensional representation for each neuron, we apply mean pooling over the temporal axis:

$$\mathbf{y}_n = \frac{1}{P}\sum_{p=1}^{P} \mathbf{z}_{n,p}^{\text{final}} \quad \forall\, n \in [N]. \tag{4}$$

The core idea behind this architecture is to first build a temporally informed representation of each neuron based solely on its own activity. This is accomplished through the initial stack of temporal transformer layers, which process each neuron's token sequence independently. The resulting latent sequence captures the internal dynamics of each neuron over time[2]. Next, the spatial transformer layers allow neurons to exchange information at each timepoint, injecting population-level context into each neuron's representation. By alternating spatial and temporal layers, information received from other neurons at a given timepoint can be distributed to all other timepoints, creating more informed tokens to query the population again through the next spatial layer.

This architecture can also be readily adapted for calcium imaging data, with minimal modifications (see Appendix B.3). Additional details are provided in Appendix B.

[1]We use a fixed bin-size of 20ms, and also present a sweep over this value in Appendix A.1. We also attempted a spike-tokenization based approach, and discuss it in Appendix A.2.

[2]The initial temporal-only layers, while not essential for performance, contribute to improved computational efficiency. Please see Appendix A.6 for a longer discussion.

What is considered a *population*? For electrophysiology datasets, we treat each probe insertion as a distinct population, both for the spatial transformer layers and for the contrastive loss, where negative pairs are limited to neurons recorded on the same insertion (i.e. a single Neuropixels [42] probe). In early experiments, we observed that allowing interactions across insertions led the model to cluster neurons based on probe identity rather than biologically meaningful properties. This is undesirable, as our goal is to produce representations that reflect intrinsic neuronal attributes such as brain region and cell type.

2.2 Self-supervised training objective

Our goal is to learn the transformation $\mathcal{F}$ in Equation (1) such that it produces neuron-level representations that capture biologically meaningful properties such as cell type and brain region. Since such labels are costly and difficult to obtain at scale, we adopt a self-supervised approach. `NuCLR` leverages the natural spatial and temporal structure present in neural population activity and applies a contrastive learning objective to train $\mathcal{F}$ without requiring any explicit labels.

We begin by sampling two T_{ctx}-long windows, or *views*, of neural population activity within $\Delta T_{\max}$ of each other (Figure 1B). Because identity-related properties such as cell type or brain region remain static, we encode this temporal-invariance using a contrastive loss that encourages corresponding neurons across the two views to produce similar representations. At the same time, the representation should capture each neuron's distinct identity, so representations of different neurons within a population are pushed apart.

To promote robustness to partial observations and prevent overfitting to specific populations, we apply *neuron dropout* independently to each view, randomly removing *up to* 50% neurons per view (Appendix B.1). As a result, the number of neurons present in each view usually differs, and only a subset of neurons in one view have a corresponding neuron in the other view. We define the indices of neurons presented in both views as

$$\mathcal{M} = \{(n, m) \mid \text{neuron } n \text{ in view 1 corresponds to neuron } m \text{ in view 2}\}, \tag{5}$$

which identifies all valid positive pairs for contrastive training.

Let $\tilde{\mathcal{X}}^1 = \{X_1^1, \ldots, X_{N_1}^1\}$ and $\tilde{\mathcal{X}}^2 = \{X_1^2, \ldots, X_{N_2}^2\}$ denote the two augmented views after neuron dropout. The encoder maps these views into corresponding sets of representations:

$$\mathcal{F}(\tilde{\mathcal{X}}^1) = \{\mathbf{y}_1^1, \ldots, \mathbf{y}_{N_1}^1\} \text{ and } \mathcal{F}(\tilde{\mathcal{X}}^2) = \{\mathbf{y}_1^2, \ldots, \mathbf{y}_{N_2}^2\}. \tag{6}$$

Similar to SimCLR [9], these representations are passed through a projection head $g(\cdot)$—a one-hidden-layer MLP—to obtain projected vectors $\mathbf{p}_n^v = g(\mathbf{y}_n^v)$, where $v \in \{1, 2\}$ indexes the view. We then apply an InfoNCE-based contrastive loss over the set of valid positive pairs $\mathcal{M}$:

$$\mathcal{L}(\mathcal{F}, g \mid \tilde{\mathcal{X}}^1, \tilde{\mathcal{X}}^2) = \frac{1}{|\mathcal{M}|} \sum_{(n,m)\in\mathcal{M}} -\log\Bigg(\frac{\exp\left(\langle \mathbf{p}_n^1, \mathbf{p}_m^2\rangle/\tau\right)}{\sum\limits_{n'\neq n} \exp\left(\langle \mathbf{p}_n^1, \mathbf{p}_{n'}^1\rangle/\tau\right) + \sum\limits_{(n,k)\notin\mathcal{M}} \exp\left(\langle \mathbf{p}_n^1, \mathbf{p}_k^2\rangle/\tau\right)}\Bigg)$$
$$+ \text{ symmetric term for view 2 to 1} \tag{7}$$

where $\langle\cdot,\cdot\rangle$ denote cosine-similarity, and τ is a temperature hyperparameter [3]. This loss encourages representations of the same neuron to be close across views while pushing apart those of different neurons within the same population. As a result, the model learns neuron-level representations that are temporally stable and discriminative with respect to their functional roles.

A key distinction from standard SimCLR [9] lies in how we handle the minibatch setting. In `NuCLR`, the contrastive loss is computed *independently* within each view pair. That is, when training on minibatches containing samples from different animals or sessions, we do not treat neurons across samples as negatives. This design choice avoids an overabundance of *easy negatives* in the denominator of Equation (7), which can degrade contrastive learning performance [18]. As a result, the number of negatives per sample is limited to the neurons within a single recording (typically in the hundreds). This motivates our omission of positive-pair similarity terms from the

[3]We found slightly better results by using different temperatures for within-view and across-view cosine similarities. Please refer to Appendix A.5 for more details.

denominator—following the decoupled contrastive loss (DCL) [53]—which has been shown to improve performance in regimes with few negatives.

In practice, for a minibatch containing B independently sampled view pairs (potentially from different recordings), the overall loss is computed as a weighted average across samples:

$$\mathcal{L}_{\text{batch}}(\mathcal{F}, g) = \frac{1}{\sum_{b=1}^{B} N_b} \sum_{b=1}^{B} N_b \cdot \mathcal{L}(\mathcal{F}, g | \tilde{\mathcal{X}}_b^1, \tilde{\mathcal{X}}_b^2), \tag{8}$$

where $\mathcal{X}_b^{1,2}$ is the b-th view pair, and N_b is the number of valid positive pairs (i.e., neurons present in both views after dropout). The inclusion of the N_b terms helps to deal with the imbalance otherwise created between views with very different numbers of valid positive pairs. We provide additional details about the model, hyperparameter choices, and other training details in Appendix B.

3 Results

3.1 Experimental setup

To test the quality of the learned representations, we follow the standard linear-evaluation protocol used in self-supervised learning [9, 13]. We first pre-train the model with our contrastive objective (Section 2.2), then freeze the encoder and compute a single representation per neuron by sequentially sampling windows from each session and averaging the resulting outputs. Using these frozen neuron representations, we train linear classifiers to evaluate performance on two downstream tasks: *cell type* decoding and *brain region* decoding.

Evaluation strategies. For each downstream task, we perform evaluation across three generalization settings: (1) *Transductive*, where the testing populations are seen during self-supervised pretraining, and partial labels from these populations are used to train the classification head; (2) *Transductive zero-shot*, where the test populations are present during pretraining, but no labels from them are used when training the classifier; and (3) *Inductive zero-shot*, where the test populations are entirely unseen during pretraining, and no fine-tuning is performed on the encoder or classifier before evaluation. For all three settings, we use a linear head on the output representations for the classification probe.

Settings (1) and (2) emulate *offline* (or post-hoc) analysis of neural recordings, where pretraining is allowed on the test populations. Setting (1) reflects a scenario where a subset of neurons in a recording is labeled, and the goal is to infer labels for the rest. Setting (2) captures the case where the test population is collected but is not labeled, so it used for self-supervised pretraining but cannot be used to train the classification probe. Setting (3) corresponds to the *online* use-case, where a pretrained model must operate on completely new neural populations without any retuning. This final setting is the most stringent and directly tests the model's ability to generalize in a truly out-of-the-box fashion.

Baseline methods. We compare `NuCLR` to several baselines: **NeuPRINT** [31], a population-context model that uses summary statistics of neighboring neurons and the recorded behavior; **NEMO** [54], a CLIP-style contrastive approach that encodes individual neurons using their activity autocorrelograms and waveform templates; and **LOLCAT** [38], a supervised model that uses the interspike interval distribution for individual neurons and leverages trial structure in the recording. We refer the reader to Appendix E for further details on our implementation of these methods. Our main metric is the **macro F1-score**, which is robust to class imbalance and enables fair comparison across datasets with varying label distributions [4].

We note that not all methods can be tested across all settings: NeuPRINT learns a neuron embedding table through back-propagation so it cannot be tested inductively without re-training. NEMO is designed only for electrophysiology data and cannot be applied to Bugeon et al. which consists of calcium imaging recordings. Finally LOLCAT is a supervised method so the transductive zero-shot setting is not possible. The performance of baseline methods in such cases will be reported as "N/A".

3.2 Cell type decoding

We begin by evaluating the quality of `NuCLR`'s representations on their ability to predict neuronal cell types. The first dataset used to test our approach is the Allen Brain Observatory Visual Coding

[4]We attempted to compare with a recent method NeurPIR [52] but were not able to receive code from the authors to reproduce their method.

Table 1: **Macro F1-score for cell type classification across different generalization settings.** Reported as mean ± std. dev. across 5 training seeds. N/A indicates the method cannot operate in that evaluation setting.

Dataset	# Classes	Setting	NuCLR (Ours)	NeuPRINT	NEMO	LOLCAT
Allen VC[5]	3	Transd. zero-shot	0.7218 ± 0.0113	0.4020 ± 0.0238	0.4256 ± 0.0114	N/A
		Ind. zero-shot	0.7200 ± 0.0267	N/A	0.4194 ± 0.0099	0.4121 ± 0.0800
Bugeon (E vs. I)	2	Transductive	0.8110 ± 0.0035	0.6658 ± 0.0090	N/A	0.7205 ± 0.0127
		Transd. zero-shot	0.6826 ± 0.0293	0.6362 ± 0.0073	N/A	N/A
		Ind. zero-shot	0.6738 ± 0.0563	N/A	N/A	0.7463 ± 0.0095
Bugeon (Subclass)	5	Transductive	0.6101 ± 0.0249	0.4952 ± 0.0222	N/A	0.2900 ± 0.0388
		Transd. zero-shot	0.4014 ± 0.0268	0.3529 ± 0.0194	N/A	N/A
		Ind. zero-shot	0.3938 ± 0.0556	N/A	N/A	0.2418 ± 0.0078

(VC) Neuropixels dataset [41], which contains 58 sessions, each from a different mouse. Each animal is presented visual stimuli ranging from drifting gratings to natural images and video. Within the 58 sessions, 16 include optotagged inhibitory neurons labeled as one of 3 subclasses: Pvalb, Sst, or Vip. Since all labeled neurons within a session belong to the same subclass, we evaluate performance only in the two zero-shot settings [5]. In the *transductive zero-shot* setting, all sessions are used during pretraining (including labeled ones), but the decoder is trained and tested on neurons from non-overlapping subsets of mice. In the more stringent *inductive zero-shot* setting, pretraining is restricted to unlabeled sessions only. Further details on our data splits and validation methodology are provided in Appendix D.

The second dataset that we evaluate on is a spatial transcriptomics dataset from Bugeon et. al. [7], which consists of calcium imaging recordings from 17 sessions across 4 mice, with cell types labeled as excitatory (E) or inhibitory (I), and inhibitory neurons further divided into 5 subclasses (Lamp5, Pvalb, Vip, Sncg, Sst). We test both binary classification (E vs. I) and five-way subclass classification. Additionally, in Appendix A.8, we also test an 11-way subclass classification.

As shown in Table 1, `NuCLR` achieves strong performance across both datasets and consistently outperforms all baselines in zero-shot settings. On the Allen VC dataset, `NuCLR` achieves a macro F1-score of 0.7218 in the transductive zero-shot setting and 0.7200 in the inductive setting—more than 0.29 F1 higher than the next best method (NEMO) in the inductive case. On the Bugeon dataset, `NuCLR` achieves a macro F1 of 0.6738 on inductive zero-shot E vs. I classification and 0.3938 on the more challenging five-way subclass task on a held-out subject. We also compared with POYO+ [5] on the Allen VC in (Appendix A.4), a multi-session multi-task decoding method that learns a neuron-level embedding table, and found that `NuCLR` outperforms these embeddings by a strong margin on cell type classification.

These results demonstrate that `NuCLR` produces stable, transferable neuron representations that generalize to previously unseen populations without retraining, enabling accurate zero-shot decoding of cell type identity across diverse datasets and experimental conditions.

3.3 Brain region decoding

For brain region identification, we evaluate on two electrophysiological datasets with well-curated anatomical annotations: the International Brain Laboratory (IBL) Brain-wide Map [2] and the Steinmetz et al. 2019 dataset [43]. The IBL dataset comprises recordings from 139 mice across approximately 700 probe insertions. Following the evaluation setup used by NEMO, we perform classification across 10 brain regions (Figure 2A). The Steinmetz et al. dataset includes 39 recordings from 10 mice, with classification over four regions: HPF, MB, TH, and VIS. Zero-shot test populations correspond to entirely unseen experimental sessions (and thus unseen probe insertions) in the IBL dataset, and to unseen subjects in the Steinmetz et al. dataset. Further details on our evaluation methodology and data folds are provided in Appendix D.

As shown in Table 2, `NuCLR` achieves the highest macro F1-scores across all evaluation settings. In the inductive zero-shot regime on IBL, `NuCLR` reaches an F1-score of 0.53 compared to the next best performing method NEMO with a 0.38. We also observe in Figure 2A that the representations pro-

[5]The transductive setting cannot allow for proper testing in this case, as a “perfect” classifier only has to infer which of the previously seen sessions does the neuron belong to and the cell-type corresponding to that session.

Table 2: **Macro F1-scores for brain region classification across different generalization settings.** Reported as mean ± std. dev. across 5 training seeds. N/A indicates the method cannot operate in that evaluation setting.

Dataset	Classes	Setting	NuCLR (Ours)	NeuPRINT	NEMO	LOLCAT
IBL	10	Transductive	0.6686 ± 0.0034	0.2734 ± 0.0153	0.4188 ± 0.0041	0.2851 ± 0.0008
		Transd. zero-shot	0.5343 ± 0.0115	0.2531 ± 0.0137	0.3804 ± 0.0011	N/A
		Ind. zero-shot	0.5295 ± 0.0040	N/A	0.3793 ± 0.0011	0.2532 ± 0.0016
Steinmetz et. al.	4	Transductive	0.9594 ± 0.0027	0.4476 ± 0.0166	0.6989 ± 0.0044	0.3205 ± 0.0055
		Transd. zero-shot	0.7338 ± 0.0226	0.4122 ± 0.0326	0.6681 ± 0.0016	N/A
		Ind. zero-shot	0.5810 ± 0.0110	N/A	0.5595 ± 0.0048	0.3191 ± 0.0311

duced by `NuCLR` are organized according to brain region in the IBL dataset. Embedding visualizations for the remaining datasets are provided in Appendix C.

3.4 Data scaling and label efficiency

We study how zero-shot performance scales with the amount of unlabeled pretraining data and the availability of labeled neurons for training the classification probe (Figure 2B). We increase the amount of data used for pretraining while holding the test populations fixed and distinct from those used in pretraining. To vary the level of supervision, we randomly subsample the labeled neurons used to train the linear classifier (reported as the "label ratio"), where a ratio of 1.0 corresponds to using the same full set of labels across all pretraining scales.

We find that for the Allen VC dataset, increasing the number of unlabeled pretraining data leads to dramatic improvements in cell type classification performance. In fact, in some cases, doubling the amount of pretraining data is significantly more beneficial than doubling the number of labeled neurons, as annotated in Figure 2B. For the IBL dataset, we observe similar trends: brain region decoding performance continues to improve with additional pretraining data.

As expected, classification accuracy increases when more labeled data is used to train the classifier heads at a fixed pretraining data scale. However, `NuCLR` is highly label-efficient: when only 12.5% of the labeled data is used, it still achieves scores 0.54 and 0.50 on Allen VC and IBL respectively, outperforming all baselines even when they use 100% of the labels. For instance, NEMO reaches only 0.42 and 0.38 under full supervision on these datasets. Taken together, these results show that our approach is both label-efficient and highly scalable, capable of leveraging large unlabeled datasets to improve downstream performance with minimal supervision.

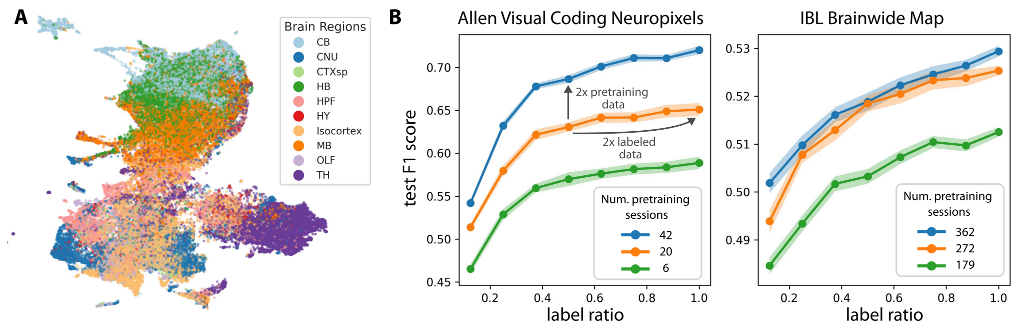

Figure 2: **Representation visualization and scaling trends.** (A) A 2-D UMAP visualization of `NuCLR`'s neuron representations for the IBL Brainwide Map dataset colored by brain region. (B) Data scaling trends of inductive zero-shot decoding performance for cell type (Allen VC) and brain region (IBL) tasks. The number of animals used in pretraining are varied along with the amount of labels used to train the linear classifier heads (label ratio). Performance improves with larger pretraining data and amount of supervision. In many cases, increasing the amount of unlabeled pretraining data is more effective than increasing the amount of labeled data.

3.5 Ablations

We conducted a series of ablation experiments to assess the contribution of key architectural and training components in our model (Table 3). First, we removed the spatial attention layers which enable population-level interactions across neurons.[6] Ablating the spatial layers (population context) led to large performance drops on both the Allen VC (from 0.72 to 0.55) and IBL datasets (from 0.53 to 0.36). These results highlight the importance of spatial context: integrating information from surrounding neurons provides complementary signals to single-neuron activity and improves the model's ability to capture features of neuronal identity. It's interesting to note that even with this huge drop in accuracy, our model still outperforms most baselines in this restricted setting.

We also examined the impact of neuron dropout, a regularization technique that randomly masks a subset of neurons during training. This component improved performance on the Allen VC dataset but had minimal effect on IBL. We attribute this difference to dataset scale: Allen VC pretraining dataset includes only 42 populations and therefore benefits more from regularization, whereas IBL comprises over 600 populations and may be less prone to overfitting.

Table 3: **Ablation study on the Allen Visual Coding and IBL datasets.** We test the impact of spatial attention layers and neuron dropout on the Macro-F1 score. Values are reported as mean ± std. across 5 seeds.

Model Variant	**Allen VC (Cell type)**	**IBL (Brain region)**
Full NuCLR	0.7200 ± 0.0267	0.5295 ± 0.0040
w/o neuron dropout	0.6181 ± 0.0252	0.5248 ± 0.0164
w/o spatial attn layers	0.5550 ± 0.0796	0.3573 ± 0.0019

4 Related Work

Functional organization of brain circuits. From early days of neuroscience and experiments of Hubel and Wiesel [15], mapping the responses or "tuning" of different neurons has been an important part of building our understanding of how the brain works. At scale, and with modern large-scale datasets, this has allowed for extensive work in characterizing the functional organization in primate V4 [51], in chromatic feature detectors in retina [14], visual cortex in mouse [47], and combinatorial codes in mouse V1 [49]. In vision, maximally exciting or most discriminative inputs [8, 50, 47] can be used to find stimuli that help to differentiate neurons with different functional properties.

Deep learning approaches for cell type and brain region classification. Attempts to build deep learning solutions for the problem of extracting cell types and brain regions from neural activity are still nascent [37]. In [38, 46], interspike interval (ISI) distributions are used for cell type and brain region prediction, respectively. LOLCAT [38] extracts the ISI distribution over individual trials and builds an multi-head attention layer to attend to subsets of trials and classify neurons in a supervised manner. NEMO [54], PhysMAP [23] and VAE-based models [6] combine waveform and autocorrelogram views from a single neuron; NEMO uses a CLIP-style contrastive loss between both modalities, and Beau et al. concatenating both modalities after separate VAE-based encoders are applied to each.

Recent methods have begun to explore integration of population context into neuron embeddings. NeuPRINT [30] uses a reconstruction task and stimulus information to learn an embedding table for neurons, and adds population context through a set of fixed summary statistics. NeurPIR [52] learns time-invariant neuron representations from population dynamics using a contrastive VICReg loss and a pretrained population-level encoder to provide population context for each neuron. While these approaches offer improvements over single-neuron models, their reliance on compressed population features can discard fine-grained structure in the dynamics. Additionally, because these models must be retrained or finetuned for each new recording, they provide limited ability to generalize in a zero-shot manner across sessions or animals.

Channel-level transformer architectures and functional embeddings. A growing line of work applies transformer architectures at the channel or neuron level to generate embeddings from population activity [3, 5, 56, 27, 20]. POYO and POYO+[7] learn a unit embedding table and EIT generates a channel-level tokenization; all focus on supervised behavior decoding tasks. NEDS uses both encoding (neural activity prediction) and decoding (behavior prediction) objectives, and learns a set of neuron level weights at the decoder which can be used to read out neuron-level attributes. STNDT

[6]When ablating the spatial attention layers, the spatial layers were replaced with additional temporal layers to match the overall model depth and parameter count.

[7]While POYO+ reports Cre-line and brain-region classification, these results are obtained using session-level averaged latent representations, and are not based on embeddings of individual neurons. See Appendix A.4 for more details and an extended analysis of the unit embeddings learned by POYO+.

employs a masked modeling objective with both neuron-level and population-level tokens to achieve strong performance on prediction of neural activity on held-out neurons. However, none of these approaches are designed to learn identity-relevant neuron embeddings or to generalize in zero-shot settings. `NuCLR` is specifically built for this purpose, with an objective and architecture tailored to recover neuron identity from population activity and transfer robustly across sessions and animals.

Contrastive methods and their use in neural population data analysis. Contrastive learning maps similar examples to nearby points in an embedding space while separating dissimilar ones [21, 9], and has achieved broad success across language [11], vision [9], audio [36], and multimodal representation learning [35]. Motivated by these advances, contrastive objectives have increasingly been used to learn representations from neural population activity [4, 26, 34, 48, 39]. Swap-VAE and MYOW introduce temporal and dropout-based augmentations for population responses [26, 4]; Swap-VAE combines contrastive and generative losses across two latent spaces to learn disentangled population embeddings, while MYOW employs a BYOL-based [13] contrastive objective with nearest-neighbor mining to identify harder positives. Urzay et al. apply contrastive metric learning [1] to detect change points in neural time series [48]. Peterson et al. demonstrate cross-stream self-supervised learning between neural activity and behavior [34]. CEBRA uses contrastive learning to align recordings across sessions, modalities, and animals [39]. While these approaches successfully learn representations of neural activity that can be connected to behavior, they operate at the population level and do not construct embeddings for individual neurons. `NuCLR` applies contrastive learning at the neuron level: different temporal views of the same neuron serve as positives, all other neurons serve as negatives. This design enables `NuCLR` to learn identity-relevant embeddings for individual neurons in the context of other neurons, rather than global representations which compress the entire population into a single embedding.

5 Discussion

We introduced `NuCLR`, a self-supervised framework for learning population-aware *neuron identity embeddings* from both electrophysiology and calcium imaging datasets. Using a spatiotemporal transformer trained with a contrastive objective, `NuCLR` captures within-neuron dynamics and population context, and enables *zero-shot* prediction of neuronal properties in new subjects without supervised labels or session-specific tuning.

Even though we don't use stimulus information in our pretraining, our current evaluations use datasets with similar underlying task structure. Thus, testing and building robustness to changes in the underlying stimulus distribution will be an interesting line of future research. As we extend the approach to train on datasets spanning different tasks, sensory modalities, and behavioral states, it may be useful to incorporate task-conditioned encoders, or metadata-aware prompts to support training across heterogeneous datasets.

While our current evaluations show the applicability of `NuCLR` on both electrophysiology and calcium imaging, we currently need two different models for each modality. In the future, building a unified pretraining approach for both modalities [19] would allow for a joint model pretraining that could take advantage of both modalities and train at an even larger scale than with a single modality alone.

Our zero-shot scaling results show strong benefits of pretraining with more data on more animals. Thus, we anticipate that `NuCLR` could be trained on even more data to achieve even more performance gains, and ultimately building a path toward general-purpose neuron identity estimation and a foundation for models that integrate across modalities, species, and experimental settings. As larger and more diverse neural datasets emerge, such approaches may help reveal how neuronal diversity shapes computation across brain-wide circuits.

Acknowledgments and Disclosure of Funding

We would like to thank Anna Lakunina, Shivashriganesh P. Mahato, Alexandre André, and Liam Paninski for insightful discussions and feedback. This project was supported by the NIH (1R01EB029852), the NSF (CAREER IIS-2146072), as well as generous gifts from the CIFAR Learning in Machines & Brains Program and The Hypothesis Fund.

References

[1] N. Ahad, E. L. Dyer, K. B. Hengen, Y. Xie, and M. A. Davenport. Learning sinkhorn divergences for change point detection. *IEEE Transactions on Signal Processing*, 73:4347–4363, 2025.

[2] D. Angelaki, B. Benson, J. Benson, D. Birman, N. Bonacchi, K. Bougrova, S. A. Bruijns, M. Carandini, J. A. Catarino, et al. A brain-wide map of neural activity during complex behaviour. *Nature*, 645(8079):177–191, 2025.

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

[4] M. Azabou, M. G. Azar, R. Liu, C.-H. Lin, E. C. Johnson, K. Bhaskaran-Nair, M. Dabagia, B. Avila-Pires, L. Kitchell, K. B. Hengen, et al. Mine your own view: Self-supervised learning through across-sample prediction. *arXiv preprint arXiv:2102.10106*, 2021.

[5] M. Azabou, K. X. Pan, V. Arora, I. J. Knight, E. L. Dyer, and B. A. Richards. Multi-session, multi-task neural decoding from distinct cell-types and brain regions. In *The Thirteenth International Conference on Learning Representations*, 2025.

[6] M. Beau, D. J. Herzfeld, F. Naveros, M. E. Hemelt, F. D'Agostino, M. Oostland, A. Sánchez-López, Y. Y. Chung, M. Maibach, H. N. Stabb, et al. A deep-learning strategy to identify cell types across species from high-density extracellular recordings. *Cell*, 2025.

[7] S. Bugeon, J. Duffield, M. Dipoppa, A. Ritoux, I. Prankerd, D. Nicoloutsopoulos, D. Orme, M. Shinn, H. Peng, H. Forrest, A. Viduolyte, C. B. Reddy, Y. Isogai, M. Carandini, and K. D. Harris. A transcriptomic axis predicts state modulation of cortical interneurons. *Nature*, 607(7918):330–338, July 2022.

[8] M. F. Burg, T. Zenkel, M. Vystrčilová, J. Oesterle, L. Höfling, K. F. Willeke, J. Lause, S. Müller, P. G. Fahey, Z. Ding, K. Restivo, S. Sridhar, T. Gollisch, P. Berens, A. S. Tolias, T. Euler, M. Bethge, and A. S. Ecker. Most discriminative stimuli for functional cell type clustering. In *The Twelfth International Conference on Learning Representations*, 2024.

[9] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[10] A. J. Christensen, T. Ott, and A. Kepecs. Cognition and the single neuron: How cell types construct the dynamic computations of frontal cortex. *Current Opinion in Neurobiology*, 77:102630, 2022.

[11] T. Gao, X. Yao, and D. Chen. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, 2021.

[12] N. W. Gouwens, S. A. Sorensen, J. Berg, C. Lee, T. Jarsky, J. Ting, S. M. Sunkin, D. Feng, C. A. Anastassiou, E. Barkan, et al. Classification of electrophysiological and morphological neuron types in the mouse visual cortex. *Nature neuroscience*, 22(7):1182–1195, 2019.

[13] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[14] L. Höfling, K. P. Szatko, C. Behrens, Y. Deng, Y. Qiu, D. A. Klindt, Z. Jessen, G. W. Schwartz, M. Bethge, P. Berens, K. Franke, A. S. Ecker, and T. Euler. A chromatic feature detector in the retina signals visual context changes. *eLife*, 13, Oct. 2024.

[15] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574, 1959.

[16] A. Jha, D. Gupta, C. Brody, and J. W. Pillow. Disentangling the roles of distinct cell classes with cell-type dynamical systems. *Advances in Neural Information Processing Systems*, 37:33668–33690, 2024.

[17] A. L. Juavinett, G. Bekheet, and A. K. Churchland. Chronically implanted neuropixels probes enable high-yield recordings in freely moving mice. *Elife*, 8:e47188, 2019.

[18] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus. Hard negative mixing for contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21798–21809. Curran Associates, Inc., 2020.

[19] I. J. Knight, V. Arora, M. Azabou, and E. L. Dyer. Unified pretraining on mixed optophysiology and electrophysiology data across brain regions. In *Foundation Models for the Brain and Body Workshop — NeurIPS 2025*, 2025.

[20] T. Le and E. Shlizerman. STNDT: Modeling neural population activity with spatiotemporal transformers. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[21] P. H. Le-Khac, G. Healy, and A. F. Smeaton. Contrastive representation learning: A framework and review. *Ieee Access*, 8:193907–193934, 2020.

[22] A. T. Lee, E. F. Chang, M. F. Paredes, and T. J. Nowakowski. Large-scale neurophysiology and single-cell profiling in human neuroscience. *Nature*, 630(8017):587–595, 2024.

[23] E. K. Lee, A. E. Gül, G. Heller, A. Lakunina, S. Jaramillo, P. F. Przytycki, and C. Chandrasekaran. Physmap-interpretable in vivo neuronal cell type identification using multi-modal analysis of electrophysiological data. *BioRxiv*, pages 2024–02, 2024.

[24] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 3744–3753, 2019.

[25] B. Lefaudeux, F. Massa, D. Liskovich, W. Xiong, V. Caggiano, S. Naren, M. Xu, J. Hu, M. Tintore, S. Zhang, P. Labatut, D. Haziza, L. Wehrstedt, J. Reizenstein, and G. Sizov. xformers: A modular and hackable transformer modelling library. `https://github.com/facebookresearch/xformers`, 2022.

[26] R. Liu, M. Azabou, M. Dabagia, C.-H. Lin, M. Gheshlaghi Azar, K. Hengen, M. Valko, and E. Dyer. Drop, swap, and generate: A self-supervised approach for generating neural activity. *Advances in neural information processing systems*, 34:10587–10599, 2021.

[27] R. Liu, M. Azabou, M. Dabagia, J. Xiao, and E. L. Dyer. Seeing the forest and the tree: Building representations of both individual and collective dynamics with transformers. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[28] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[29] J. N. McGregor, C. A. Farris, S. Ensley, A. Schneider, L. J. Fosque, C. Wang, E. I. Tilden, Y. Liu, J. Tu, H. Elmore, et al. Failure in a population: Tauopathy disrupts homeostatic set-points in emergent dynamics despite stability in the constituent neurons. *Neuron*, 112(21):3567–3584, 2024.

[30] L. Mi, T. Le, T. He, E. Shlizerman, and U. Sümbül. Learning time-invariant representations for individual neurons from population dynamics. *Advances in Neural Information Processing Systems*, 36, 2023.

[31] L. Mi, T. Le, T. He, E. Shlizerman, and U. Sümbül. Learning time-invariant representations for individual neurons from population dynamics. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 46007–46026. Curran Associates, Inc., 2023.

[32] C. P. Mosher, Y. Wei, J. Kamiński, A. Nandi, A. N. Mamelak, C. A. Anastassiou, and U. Rutishauser. Cellular classes in the human brain revealed in vivo by heartbeat-related modulation of the extracellular action potential waveform. *Cell reports*, 30(10):3536–3551, 2020.

[33] V. Pak, Q. Adewale, D. Bzdok, M. Dadar, Y. Zeighami, and Y. Iturria-Medina. Distinctive whole-brain cell types predict tissue damage patterns in thirteen neurodegenerative conditions. *Elife*, 12:RP89368, 2024.

[34] S. M. Peterson, R. P. Rao, and B. W. Brunton. Learning neural decoders without labels using multiple data streams. *Journal of Neural Engineering*, 19(4):046032, 2022.

[35] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[36] A. Saeed, D. Grangier, and N. Zeghidour. Contrastive learning of general-purpose audio representations. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3875–3879. IEEE, 2021.

[37] A. Schneider. *Robust Embeddings of Genetics, Anatomy, and State Decoded from a Neuron's Activity*. PhD thesis, Washington University in St. Louis, 2025.

[38] A. Schneider, M. Azabou, L. McDougall-Vigier, D. F. Parks, S. Ensley, K. Bhaskaran-Nair, T. Nowakowski, E. L. Dyer, and K. B. Hengen. Transcriptomic cell type structures in vivo neuronal activity across multiple timescales. *Cell reports*, 42(4), 2023.

[39] S. Schneider, J. H. Lee, and M. W. Mathis. Learnable latent embeddings for joint behavioural and neural analysis. *Nature*, 617(7960):360–368, 2023.

[40] C. M. Schneider-Mizell, A. L. Bodor, D. Brittain, J. Buchanan, D. J. Bumbarger, L. Elabbady, C. Gamlin, D. Kapner, S. Kinn, G. Mahalingam, et al. Inhibitory specificity from a connectomic census of mouse visual cortex. *Nature*, 640(8058):448–458, 2025.

[41] J. H. Siegle, X. Jia, S. Durand, S. Gale, C. Bennett, N. Graddis, G. Heller, T. K. Ramirez, H. Choi, J. A. Luviano, et al. Survey of spiking in the mouse visual system reveals functional hierarchy. *Nature*, 592(7852):86–92, 2021.

[42] N. A. Steinmetz, C. Aydin, A. Lebedeva, M. Okun, M. Pachitariu, M. Bauza, M. Beau, J. Bhagat, C. Böhm, M. Broux, S. Chen, J. Colonell, et al. Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science*, 372(6539), Apr. 2021.

[43] N. A. Steinmetz, P. Zatka-Haas, M. Carandini, and K. D. Harris. Distributed coding of choice, action and engagement across the mouse brain. *Nature*, 576(7786):266–273, Nov. 2019.

[44] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[45] B. Tasic, Z. Yao, L. T. Graybuck, K. A. Smith, T. N. Nguyen, D. Bertagnolli, J. Goldy, E. Garren, M. N. Economo, S. Viswanathan, et al. Shared and distinct transcriptomic cell types across neocortical areas. *Nature*, 563(7729):72–78, 2018.

[46] G. B. Tolossa, A. M. Schneider, E. Dyer, and K. B. Hengen. Neurons throughout the brain embed robust signatures of their anatomical location into spike trains. *eLife*, 13:RP101506, 2025.

[47] R. Tong, R. da Silva, D. Lin, A. Ghosh, J. Wilsenach, E. Cianfarano, P. Bashivan, B. Richards, and S. Trenholm. The feature landscape of visual cortex. bioRxiv 2023.11.03.565500, Nov. 2023.

[48] C. Urzay, N. Ahad, M. Azabou, A. Schneider, G. Atamkuri, K. B. Hengen, and E. L. Dyer. Detecting change points in neural population activity with contrastive metric learning. In *2023 11th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 1–4. IEEE, 2023.

[49] I. Ustyuzhaninov, M. F. Burg, S. A. Cadena, J. Fu, T. Muhammad, K. Ponder, E. Froudarakis, Z. Ding, M. Bethge, A. S. Tolias, and A. S. Ecker. Digital twin reveals combinatorial code of non-linear computations in the mouse primary visual cortex. bioRxiv 2022.02.10.479884, Feb. 2022.

[50] E. Y. Walker, F. H. Sinz, E. Cobos, T. Muhammad, E. Froudarakis, P. G. Fahey, A. S. Ecker, J. Reimer, X. Pitkow, and A. S. Tolias. Inception loops discover what excites neurons most using deep predictive models. *Nature Neuroscience*, 22(12):2060–2065, Nov. 2019.

[51] K. F. Willeke, K. Restivo, K. Franke, A. F. Nix, S. A. Cadena, T. Shinn, C. Nealley, G. Rodriguez, S. Patel, A. S. Ecker, F. H. Sinz, and A. S. Tolias. Deep learning-driven characterization of single cell tuning in primate visual area v4 unveils topological organization. bioRxiv 2023.05.12.540591, May 2023.

[52] W. Wu, C. Liao, Z. Deng, Z. Guo, and J. Wang. Neuron platonic intrinsic representation from dynamics using contrastive learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[53] C.-H. Yeh, C.-Y. Hong, Y.-C. Hsu, T.-L. Liu, Y. Chen, and Y. LeCun. Decoupled contrastive learning. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision – ECCV 2022*, pages 668–684, Cham, 2022. Springer Nature Switzerland.

[54] H. Yu, H. Lyu, Y. Xu, C. Windolf, E. K. Lee, F. Yang, A. M. Shelton, O. Winter, I. B. Laboratory, E. L. Dyer, C. Chandrasekaran, N. A. Steinmetz, L. Paninski, and C. L. Hurwitz. In vivo cell-type and brain region classification via multimodal contrastive learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[55] H. Zeng and J. R. Sanes. Neuronal cell-type classification: challenges, opportunities and the path forward. *Nature Reviews Neuroscience*, 18(9):530–546, 2017.

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

[57] Y. Zhao, H. M. Stealey, H.-Y. Lu, E. Contreras-Hernandez, Y.-J. Chang, P. Tobler, and S. R. Santacruz. Distinct roles of neuronal phenotypes during neurofeedback adaptation. *bioRxiv*, pages 2025–05, 2025.

Appendix

A Additional Results

A.1 Effect of bin-size

The encoder used in this work assumes the neural activity is binned with a finite bin-size, which sets the minimum time resolution at which the encoder can views the data itself. A fair hypothesis would be that certain cell-types or neurons from certain brain-regions would be more (or less) identifiable at a certain bin-size setting.

We test this by performing a sweep over the bin-size, and measuring the class-wise and macro F1-scores. As we see in Figure 3, some kinds of neurons do indeed prefer certain bin-sizes, however, the overall effect of this hyperparameter is surprisingly small. In other words, the overall classifier performance is relatively robust to the choice of bin-size.

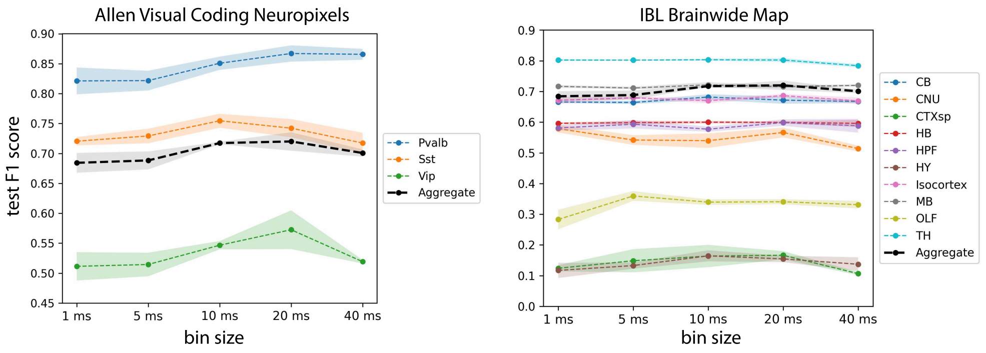

Figure 3: **Results of bin-size sweep.** Classification performance variation for different classes upon sweeping the bin-size hyperparameter of the encoder. “Aggregate” refers to the overall macro-F1 score. Error polygons represents standard error of mean (SEM) measured across 3 pretraining seeds.

A.2 Experiments with a spike-based encoder

We also explored a spike-tokenization strategy inspired by POYO [3]. Unlike POYO, our model aims to *produce* neuron-level embeddings at its outputs, and therefore cannot assume or learn neuron identities through separate “unit embeddings.” Instead, we assign the same learnable embedding to all neurons, so that each spike token consists only of its timestamp and this shared embedding. Each neuron’s spike train is processed using perceiver-style cross-attention layers, which are queried by learnable vectors placed at linearly spaced timestamps—using a spacing equal to the patch length in our main model. The shared embedding serves only to enable cross-attention over the spike sequence and does not encode any neuron-specific information. The resulting set of tokens is then treated as activity tokens (as shown in Figure 1A) and passed through the spatio-temporal transformer identically to our main architecture. A schematic of this spike-tokenization design is shown in Figure 4A.

We find that this spike-based encoder performs significantly worse on the Allen VC cell-type classification task, while achieving nearly comparable performance on the IBL brain-region classification task (Table 4). We also note that the training stability and convergence speed is much better in the binned-and-patched version of the model, as shown in Figure 4B. Alternative implementations of spike tokenization may yield improved results, however, we leave that for future work.

Table 4: **Spike-tokenization ablation.** Values are reported as mean ± std. dev. across 5 seeds for the 20ms binning model (main NuCLR), and 3 seeds for the spike tokenization based model.

Model Variant	**Allen VC**	**IBL**
20ms binning (main NuCLR)	0.7200 ± 0.0267	0.5295 ± 0.0040
Spike-tokenization based	0.6665 ± 0.0167	0.5268 ± 0.0136

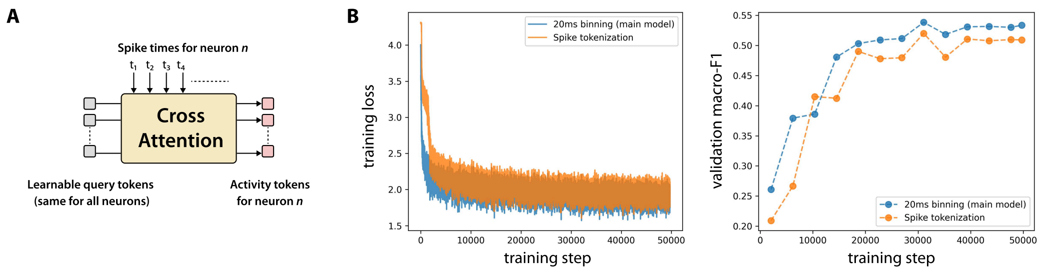

Figure 4: **Spike-tokenization based encoder.** (A) A spike-tokenization layer for the NuCLR encoder. (B) Pretraining-time metrics comparison for spike-tokenization version and binning version of NuCLR. These plots are for pretraining on the IBL dataset.

A.3 Probing intermediate layer representations

In all our experiments, we used a 6-layer implementation of NuCLR, with the first two layers being temporal-only layers followed by two spatio-temporal layers (each having 2 transformer blocks). In Figure 5 we measure the cell-type and brain-region decoding accuracies of the embeddings obtained at all 6 layers. The results show a general improvement of classification performance as we go deeper in the network, with major performance jumps being observed whenever the representation passes through spatial layers. This is another confirmation of the importance of spatial layers, as shown in Section 3.5.

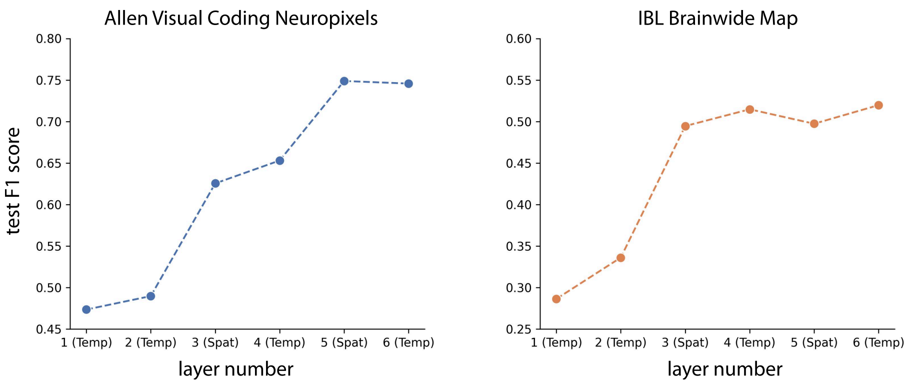

Figure 5: **Classification performance of intermediate layer representations.** Plots present the classification performance at the *output* of all layers in our 6 layer model for a single seed.

A.4 Comparison with POYO+ on Allen VC

POYO+ [5] is a multi-task behavior decoder model that can be trained simultaneously on multiple recordings. It has been shown to perform well on a Cre-line classification task on the Allen Brain Observatory calcium imaging dataset, using the *session-averaged* latent outputs of its encoder. This is possible since each recording (session) in that dataset has neurons with only one cell-type. This analysis method used in the POYO+ manuscript is not directly applicable here since we aim to produce neuron-level embeddings that can identify different types of neurons *within* a given population.

However, POYO+ learns a “unit embedding” for each neuron, which can, in theory, be used for the purpose of decoding neuron-level properties. We use this strategy to compare POYO+ with NuCLR. We train POYO+ on the Allen VC dataset for decoding the following behaviors: Drifting Gratings Orientation, Drifting Gratings Temporal Frequency, Gabor Orientation, Gabor Position, Natural Scenes, Running Speed, and Static Gratings Orientation.

Since POYO+ can only be tested under transductive evaluations, we choose the transductive zero-shot regime for our comparison, and find that NuCLR outperforms the learned unit embeddings of POYO+ by a strong margin (Table 5).

Table 5: **Comparing the unit embeddings in POYO+ with `NuCLR` and other methods.** Evaluation done in transductive zero-shot setting on the Allen VC dataset, with result being macro-F1 scores represented as mean ± std. dev. over 5 seeds.

POYO+	NuCLR	NeuPRINT	NEMO
0.3521 ± 0.0233	0.7218 ± 0.0113	0.3999 ± 0.0312	0.4256 ± 0.0114

Pretraining POYO+ required an average of 7.5 hours on 4 NVidia B200s to converge (about 100 epochs), while `NuCLR` requires only about 1.5 hours on the same hardware. We used the main training configuration from the example provided by the authors[8]. We also report the behavior decoding performance achieved by POYO+ after training across five random seeds:

- Drifting Gratings Orientation Accuracy: $93.07\% \pm 1.76\%$ (Chance: 12.5%)
- Drifting Gratings Temporal Frequency Accuracy: $93.40\% \pm 1.36\%$ (Chance: 20%)
- Gabor Orientation Accuracy: $56.12\% \pm 1.18\%$ (Chance: 25.0%)
- Gabor Position (2D) R^2: 0.6888 ± 0.0487
- Natural Scenes Accuracy: $53.23\% \pm 10.18\%$ (Chance: 0.84%)
- Running Speed R^2: 0.7681 ± 0.0115
- Static Gratings Orientation Accuracy: $75.99\% \pm 0.97\%$ (Chance: 12.5%)

A.5 Asymmetric temperature application

The original SimCLR [9] and DCL [53] losses contain only one temperature hyperparameter. However, we found that our method performs slightly better if we break the symmetry between within-view and across-view cosine similarity. This changes the loss expression in Equation (7) to be like

$$\mathcal{L}(\mathcal{F}, g \mid \tilde{\mathcal{X}}^1, \tilde{\mathcal{X}}^2) = \frac{1}{|\mathcal{M}|} \sum_{(n,m)\in\mathcal{M}} -\log\Bigg(\frac{\exp\left(\langle \mathbf{p}_n^1, \mathbf{p}_m^2 \rangle / \tau_{\text{across}}\right)}{\sum\limits_{n'\neq n} \exp\left(\langle \mathbf{p}_n^1, \mathbf{p}_{n'}^1 \rangle / \tau_{\text{within}}\right) + \sum\limits_{(n,k)\notin\mathcal{M}} \exp\left(\langle \mathbf{p}_n^1, \mathbf{p}_k^2 \rangle / \tau_{\text{across}}\right)} \Bigg)$$
$$+ \text{ symmetric term for view 2 to 1,} \tag{9}$$

where τ_{within} and τ_{across} are the separated temperature hyperparameters. We use $\tau_{\text{across}} = 0.2$ and $\tau_{\text{within}} = 1.0$ for all our models, and present results from an ablation study in Table 6. Note that the results are only marginally better at the cost of introducing another hyperparameter. So, for most practical settings, we suggest simply using the same temperature values for both cosine similarity terms.

Table 6: **Ablation study for separate temperatures.** Values are reported as mean ± std. dev. across 5 seeds.

Hyperparameter Variant	Allen VC	IBL
$\tau_{\text{across}} = 0.2, \tau_{\text{within}} = 1.0$ (main model)	0.7200 ± 0.0267	0.5295 ± 0.0040
$\tau_{\text{across}} = 0.2, \tau_{\text{within}} = 0.2$	0.7033 ± 0.0223	0.5275 ± 0.0059

A.6 Ablating temporal-only attention layers

To assess the importance of dedicated temporal-only attention layers, we replace them with an equivalent number of spatio-temporal layers. Specifically, the first two temporal-only layers in the original model were substituted with one spatio-temporal layer to maintain a similar model capacity.

As shown in Table 7 the inclusion of dedicated temporal-only layers appears to be a relatively inconsequential design decision, as performance does not change significantly between the two model configurations. However, a benefit of retaining the temporal-only layers is computational efficiency. Given that the number of temporal patches (~ 10) is typically much smaller than the number of neurons (~ 100), a temporal layer has less computationally expensive than a using separate temporal layers reduces the overall computational complexity, as attention mechanisms scale quadratically with the number of tokens in a sequence.

[8] `https://github.com/neuro-galaxy/torch_brain/tree/main/examples/poyo_plus`

Table 7: **Ablation study for temporal-only layers.** Values are reported as mean ± std. dev. across 5 seeds for full `NuCLR` , and 3 seeds for the ablated model.

Model Variant	**Allen VC**	**IBL**
Full NuCLR	0.7200 ± 0.0267	0.5295 ± 0.0040
w/o temporal attention layers	0.7184 ± 0.0124	0.5259 ± 0.0089

A.7 Confusion matrices

We present the confusion matrices achieved by `NuCLR` in Figure 6 for all datasets evaluated in the inductive zero-shot setting.

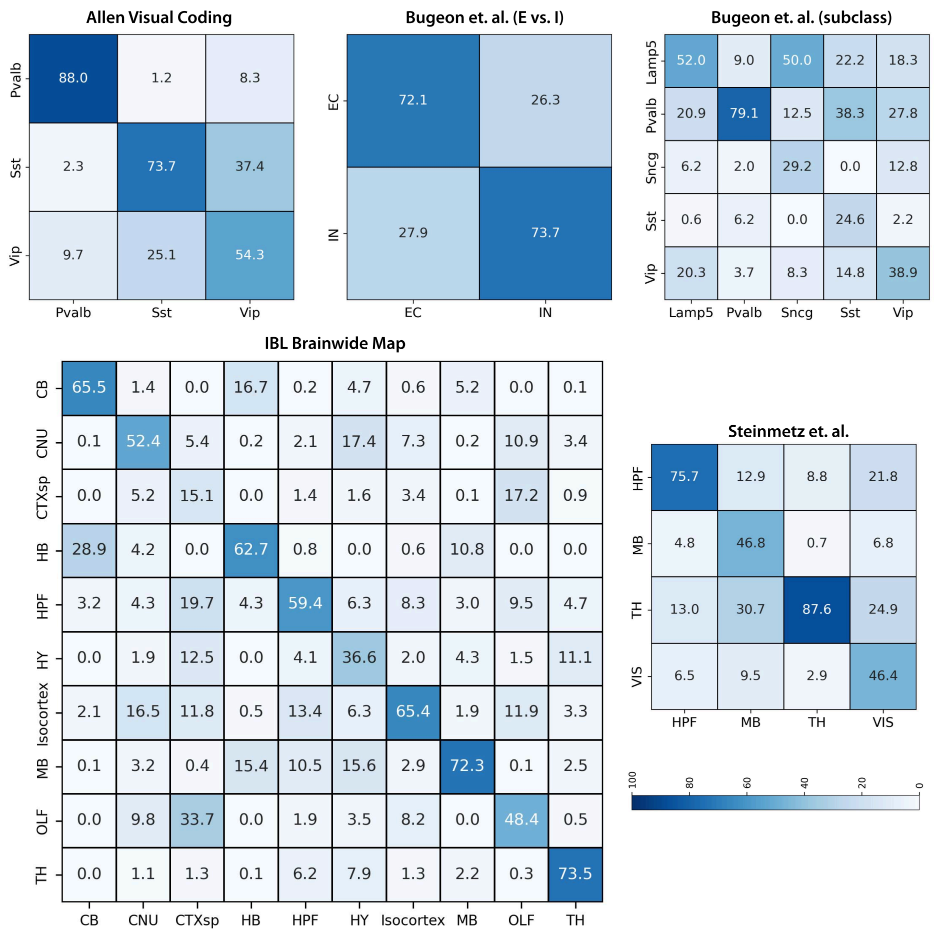

Figure 6: **Confusion matrices achieved by `NuCLR` on inductive zero-shot evaluation.**

A.8 Additional results on Bugeon et. al. dataset

We evaluate the representations of `NuCLR` and baseline models on an 11-class label set of the Bugeon et. al. dataset. This label set consists of the classes: Lamp5-Chrna7, Lamp5-Npy, Lamp5-Tmem182, Pvalb-Tac1, Pvalb-Vipr2, Sncg-Pdzrn3, Sncg-Vip, Sst-Reln, Sst-Tac1, Vip-Cp, Vip-Reln. As seen in Table 8, `NuCLR` outperforms both NeuPRINT and LOLCAT baselines on this labelling of neurons.

Table 8: **Macro F1-score for cell type classification on the 11-class label set of Bugeon et. al.** Reported as mean ± std. dev. across 5 training seeds. N/A indicates the method cannot operate in that evaluation setting.

Dataset	# Classes	Setting	NuCLR	NeuPRINT	LOLCAT
Bugeon et. al. (11-class)	11	Transductive	0.4056 ± 0.0425	0.2748 ± 0.0131	0.1559 ± 0.0318
		Transductive zero-shot	0.2333 ± 0.0129	0.1825 ± 0.0158	N/A
		Inductive zero-shot	0.1990 ± 0.0591	N/A	0.1482 ± 0.0256

B Additional method details

B.1 View-pair sampling

We sample view pairs during pretraining in three steps:

1. **First view.** The first view is sampled using the `RandomFixedWindowSampler` found in `torch_brain`[9]. This sampler divides each recording into non-overlapping windows of length T_{ctx} and randomly samples one window (without replacement throughout an epoch). Across epochs, a shared random jitter to the window start times at the beginning of every epoch. This emulates uniform sampling while ensuring complete data coverage each epoch.
2. **Second view.** The second view is sampled relative to the first. Its start time is drawn uniformly from a range constrained by the ΔT_{max} setting, ensuring the two views come from nearby temporal contexts.
3. **Neuron dropout.** To increase robustness to partial observations, we apply neuron dropout independently to each view. Given a view with N neurons, we first sample the number of neurons to drop from a uniform distribution: $N_{\text{drop}} \sim \mathcal{U}(0, 0.5N)$. We then randomly select N_{drop} neurons and exclude them from the view during that training step.

B.2 Model implementation details

Transformer. We use standard scaled-dot-product attention as implemented in xformers [25], pre-normalization using LayerNorm. Our feedforward network (FFN) uses the GEGLU activation function, with its hidden dimension being 4x the dimension of the tokens (D). In the temporal transformer layers, we use Rotary Embeddings to incorporate timing information, as described below.

Rotary Time Embeddings. We use rotary time embeddings following the formulation in Section A.1 of [3], which includes the use of value rotation in addition to query-key rotation. The only difference in our implementation lies in the choice of temporal scaling parameters. Because our input tokens are uniformly spaced with a stride of T_{patch}, we set $T_{\text{min}} = T_{\text{patch}}$ and $T_{\text{max}} = 8 \times T_{\text{ctx}}$.

B.3 Adapting to Calcium Imaging Data

To adapt `NuCLR`'s spatio-temporal transformer to calcium imaging data, we replace the binning-and-patching step used for spike trains with a temporal patching operation applied directly to the calcium fluorescence time series (i.e. the $\Delta F/F$ signal). This modification affects only the input stage of the model; the architecture and training procedure remain unchanged. Unlike electrophysiology, calcium imaging does not involve physical probes or insertions. Therefore, we treat all simultaneously recorded neurons within a session as a single population—both for the spatial transformer layers and for contrastive loss computation.

B.4 Hyperparameters

We use the AdamW [28] optimizer with a linear learning rate warm-up over the first epoch, followed by cosine decay until end of training. All relevant hyperparameters for training `NuCLR` are listed in Table 9. These values were mainly selected via manual line searches on the IBL development set (for ephys data, Appendix D.2) and the Bugeon et al. development set (for calcium imaging data, Appendix D.4). Across all datsets, we pretrain the model for 50,000 steps and use the `bfloat16`

[9]`https://github.com/neuro-galaxy/torch_brain`

number format throughout. Pretraining takes approximately 3 hours on a machine with 4 $\times$ NVidia H100 GPUs.

Table 9: **Key hyperparameters for `NuCLR`.**

Parameter	**Value for Ephys.**	**Value for Ca^{+2}** (diff. only)
T_{ctx}	10s	30s
T_{patch}	1s	
ΔT_{max}	30s	240s
Bin size	20ms	N/A
L_{T}	2	
L_{ST}	2	
D	256	
Num. attention heads	4	
Linear dropout	0.2	
Attention dropout	0.0	
Max. neuron dropout	50%	
Num. training steps	50,000	
Batch size	128	16
Max learning rate	5×10^{-4}	1.25×10^{-4}
Weight decay	0.01 (default)	
β_1	0.9 (default)	
β_2	0.999 (default)	

C Embedding Visualizations

We present UMAP-based visualizations of `NuCLR`'s output representations in Figure 7. For the Allen VC dataset, which includes a large number of subjects, the embeddings exhibit clear density modes that align most strongly with brain-region information.

In contrast, the Bugeon et al. and Steinmetz et al. datasets are relatively small, containing only 4 and 10 subjects respectively, with limited total number of recordings. For these datasets, `NuCLR`'s embeddings cluster primarily by subject or session identity. However, *within* these clusters, we still observe meaningful density modes corresponding to cell-type and brain-region structure. Embeddings for the IBL dataset, shown in Figure 2A, reveal strong region-based organization without noticeable clustering by subject or session.

The subject- and session-specific clustering observed in smaller datasets (Bugeon et al. and Steinmetz et al.) may hinder data-driven discovery, as it suggests entanglement with recording-specific factors. While this effect appears only in small datasets, mitigating it remains an important direction for future work.

D Details on datasets and evaluation methodology

This section contains description of each dataset, and details our train-test splits for all evaluation settings. Additionally, our code-base[10] includes all preprocessing, split preparation, and evaluation scripts.

D.1 Allen Visual Coding

The Allen Visual Coding (VC) dataset consists of 58 Neuropixels recordings, each from a unique mouse. Each recording spans approximately two hours, during which various visual stimuli are presented. Of these recordings, 16 contain optotagged neurons,[11] with each labeled recording containing neurons from only one of the three inhibitory cell types: Pvalb, Vip, or Sst. Since reliable cell-type labels are only available for neurons in the visual cortex, we restrict all models—including `NuCLR` and baselines—to use only neurons from the VIS region during pretraining.

[10] `https://github.com/nerdslab/nuclr`

[11] We use the same set of labeled neurons as in the original LOLCAT publication [38].

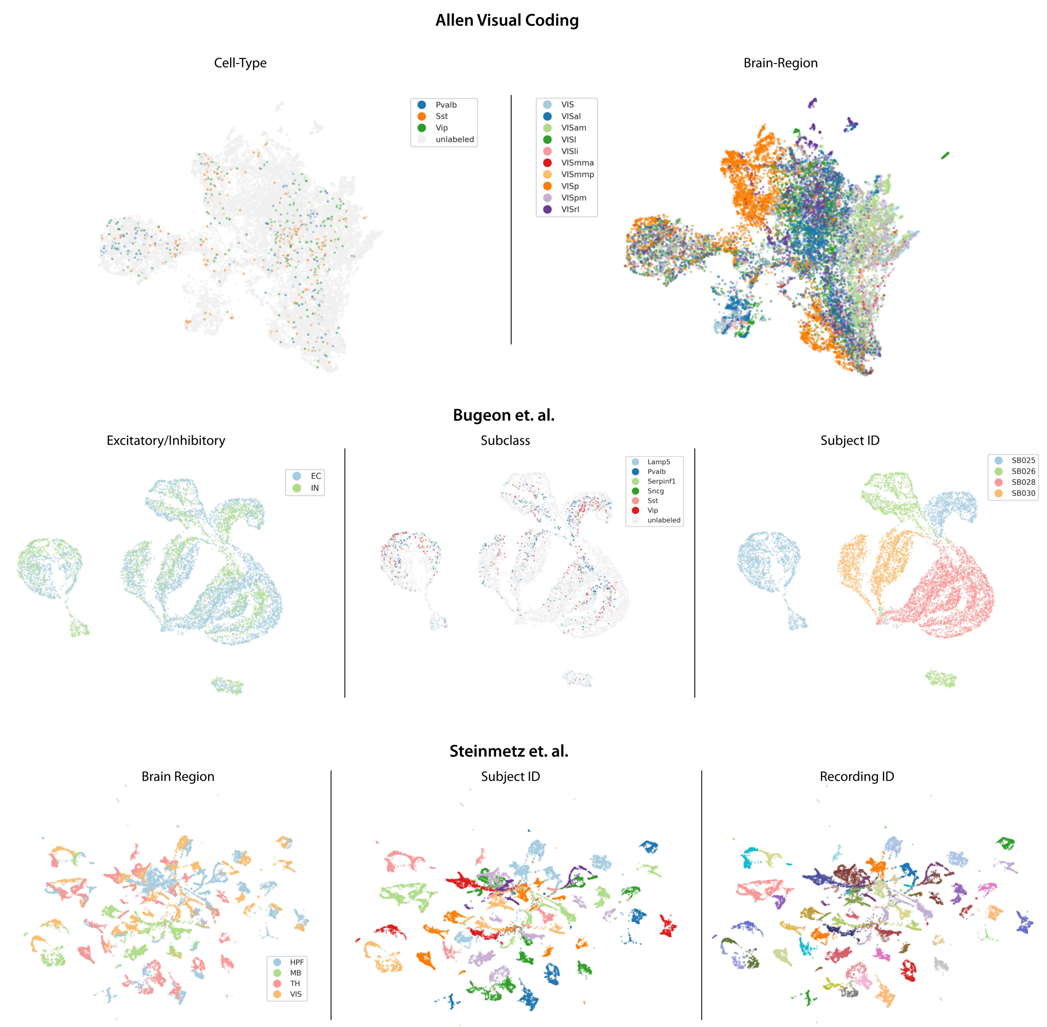

Figure 7: **UMAP visualizations** of `NuCLR`'s embeddings colored by various properties.

For zero-shot classifier evaluation, we follow a leave-one-subject-out strategy. Specifically, there are 16 test folds corresponding to the 16 labeled subjects. In each fold, one subject is held out for testing, and the remaining 15 are used for training. Within the training set, we perform 4-fold cross-validation (subject-wise) to select the best training epoch. The final test score is reported from a classifier trained on all 15 subjects using the selected epoch. We did not test classifier performance on this dataset in the transductive setting as a "perfect" classifier only has to infer which of the previously seen sessions does the neuron belong to, and the cell-type corresponding to that session.

D.2 IBL Brainwide Map

The IBL Brainwide Map dataset consists of 439 recordings from 139 unique mice, with each recording performed using one or two Neuropixels insertions. While performing zero-shot evaluations (both inductive and transductive), we construct train–test splits at the recording level, ensuring that no co-recorded neuronal populations appear in both sets. In total, 93 recordings are designated for the test set, and the remaining 346 are used for training and validation. Of the training set, 91 recordings are used as a development set for tuning hyperparameters specific to electrophysiology data. During classification, these same 91 recordings also serve as a validation set for selecting the best training epoch. Final performance metrics are reported using a model trained on all 346 training recordings, evaluated on the held-out test set.

For non-zero-shot evaluation, we create a neuron-wise stratified test-train split with a test size of 20%. The validation set for finding the best epoch is created from the train fold with a size of 20%.

D.3 Steinmetz et al.

The Steinmetz et al. dataset consists of 39 Neuropixels recordings from 10 unique mice, with each recording comprising 2 or 3 probe insertions. For non-zero-shot evaluation, we create train–test splits following the same procedure as described for the IBL dataset (Appendix D.2).

For transductive zero-shot evaluation, we use a 10-fold leave-one-subject-out strategy. In each fold, one subject is held out for testing, and the remaining nine are used for training. We select the best training epoch by performing a stratified 80/20 train–validation split within the training set. For inductive zero-shot evaluation, we hold out 3 subjects for testing and pretrain on the remaining 7. To select the best classifier epoch, we perform 4-fold subject-wise cross-validation within the training subjects.

D.4 Bugeon et al.

The Bugeon et al. dataset contains 17 spatial transcriptomic calcium imaging recordings from 4 unique mice. We train on all stimulus-specific sub-recordings within these sessions. Since this is the only optophysiology dataset in our evaluation, we adjusted several hyperparameters specifically for calcium activity. During development, we held out 4 recordings—one from each subject—as a validation set for tuning model hyperparameters. These recordings are never included in the test set when reporting final performance.

For non-zero-shot evaluation, we create neuron-wise train–test splits following the same protocol as in the IBL dataset (Appendix D.2). For transductive zero-shot evaluation, we adopt a 4-fold leave-one-subject-out strategy. In each fold, we perform 3-fold subject-wise cross-validation within the training subjects to select the best epoch. For inductive zero-shot evaluation, we hold out one subject (SB028) for testing and pretrain on the remaining three. As in the transductive case, we perform 3-fold subject-wise cross-validation on the training set to select the best model epoch.

E Implementation details for baseline models

E.1 NeuPRINT

Calcium Imaging (Bugeon et al.). For the Bugeon dataset, we used the publicly available NeuPRINT implementation and followed a transductive evaluation setup consistent with the original codebase. The model was first pretrained on the full dataset. For evaluation, the embedding table was reinitialized and optimized again using the self-supervised loss, while keeping the backbone encoder frozen. The resulting neuron embeddings were then used for downstream classification.

Electrophysiology Datasets. For electrophysiology datasets, we modified NeuPRINT's training loop and epoch structure to accommodate the larger scale and greater diversity of these datasets. In the original implementation, there is no true notion of an epoch—batches are drawn from individual sessions without ensuring full dataset coverage, and the sampling strategy is manually biased toward neurons with labeled cell types. This setup does not translate well to larger datasets such as Steinmetz et. al., IBL, and Allen VC.

We instead adopted the standard definition of an epoch, which is a full pass through all training data, and trained for 300 epochs. We also replaced the session-specific sampling strategy in the original implementation with uniform sampling across all sessions to ensure unbiased data coverage.

Furthermore, because these electrophysiology datasets are substantially larger than Bugeon et al. we did not re-initialize and relearn the embedding table during evaluation. The original NeuPRINT protocol is tailored for the Bugeon et al. dataset, where such re-initialization is computationally feasible. However, for large-scale datasets like Steinmetz, IBL, and Allen VC, this procedure becomes too computationally expensive. Therefore, we adopted a more scalable and simpler protocol that just uses learned embeddings from pretraining for downstream evaluation.

All hyperparameters were retained from the original NeuPRINT setup, except for the learning rate, which we set to 1×10^{-2}, and the backward context window, which was set to 10.

NeuPRINT relies on behavior features being provided along with the mean and standard deviations for the population activity. The behavioral features used for electrophysiology dataset were as follows:

- **Steinmetz et al.**: face motion, pupil area, and wheel velocity
- **IBL**: wheel velocity
- **Allen VC**: running speed

E.2 NEMO

We evaluated NEMO using the official implementation provided by the original authors upon request. When adapting the method to new datasets, we retained all hyperparameters from the original paper, with the exception of the waveform template dimensionality, which we adjusted to match the characteristics of each dataset.

E.3 LOLCAT

We evaluated LOLCAT using the official implementation provided by the original authors upon request. For all experiments (unless otherwise stated below), we used a batch size of 64, a learning rate of 1×10^{-3}, a weight decay of 1×10^{-5}, and a dropout rate of 0.5. The MLP hidden dimensions were set to [64, 32, 16], and 4 attention heads were used. The snippet dropout rate during training was 0.45. We trained the models for 300 epochs with the optimizer outlined in the paper. These default parameters were chosen based on manual exploration of the "reduced" hyperparameter search range and explicit prescriptions in the LOLCAT paper.

- **Allen VC**: We used the same final hyperparameter selection as the paper with new results on our split of the data.
- **IBL**: We used the same hyperparameters as Allen VC.
- **Steinmetz transductive**: Batch size was increased to 1024, and the epochs were set to 1000.
- **Steinmetz inductive**: The minimum factor was set to 0.01, and we initialized the classes undersampling factors to [8 (HPF), 8 (MB), 0.01 (TH), 0.01 (VIS)].
- **Bugeon subclass**: The classes undersampling factors were initialized to [0.01 (Lamp5), 0.01 (Pvalb), 8 (Sncg), 8 (Sst), 8 (Vip)] and the minimum factor to 0.01.
- **Bugeon E vs. I**: Class undersampling factors were initialized to [0.01 (E), 8 (I)] and the minimum factor to 0.01.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: All claims made have been confirmed in the results section Section 3, to the best of our ability.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: No new theoretical results have been introduced in this work.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have described our model architecture and training methodology clearly and fully in Section 2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will release our code along with rest of the supplementary material for this submission.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe all the training and hyperparamter details in Appendix B, and describe datasets and train-test split methodologies in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We use SEM as our error reporting method, and mention it in all our results tables.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the details of computational resources required for reproducing the our experimets in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read through the guidelines and confirm that our research conforms to those guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work is aimed at developing a fundamental tool for neuroscience research, advancing the field. While many potential and futuristic downstream societal consequences of advancing the field of neuroscience, we feel these do not need to be specifically highlighted in this work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper is about the design of a neuroscience research tool, which do not pose any such risks directly.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: The code provided in supplementary material is original work, and does not use existing assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Attached code is well documented.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No studies with human subjects were involved in this research.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No studies with human subjects were conducted.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs were used only for writing, editing, or formatting purposes.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.