MICE: Multi-animal Interaction Context Encoder — A Hierarchical Foundation Model for Mouse Behavior

Yen-Shuo Su **Weihan Li** **Anqi Wu**
Georgia Institute of Technology
Atlanta, GA, USA
{ericsu, weihanli, anqiwu}@gatech.edu

Abstract

Quantifying mouse behavior from pose is a foundational tool in neuroscience, ethology, drug discovery, and animal welfare. Self-supervised foundation models are a natural fit, but a foundation model for mouse behavior must handle social behavior — many behaviors that matter (aggression, courtship, mounting, allogrooming) are inherently multi-animal. Existing self-supervised pose models fall short: they encode each animal independently and discard interaction signal, hard-code the animal count, and pretrain one dataset at a time. We introduce **MICE** (Multi-animal Interaction Context Encoder), a hierarchical foundation model for mouse behavior that pretrains jointly on six mouse-pose corpora spanning one to four mice and 7 to 27 keypoints. MICE combines a *hierarchical individual encoder* capturing per-mouse kinematics at multiple temporal scales with a *Perceiver-style social encoder* whose learned latent codebook decouples parameter count from animal count, so a single checkpoint serves any group size without retraining. Across four evaluation protocols spanning six datasets, MICE matches or exceeds prior keypoint-based baselines, and layer-wise probing shows the social stage incrementally improves discriminability beyond the individual encoder. Code, weights, and data are available at `https://github.com/eric0708/MICE`.

1 Introduction

Automated quantification of mouse behavior from video is increasingly central to neuroscience, ethology, drug discovery, and animal welfare [Anderson and Perona, 2014, Datta et al., 2019, Pereira et al., 2020], with pose-estimation toolkits [Mathis et al., 2018, Pereira et al., 2022, Segalin et al., 2021] now producing keypoint trajectories at scale. The bottleneck has shifted from *tracking* to *interpretation*: turning (x, y) keypoint sequences into ethologically meaningful annotations. Self-supervised foundation models are a natural way to bridge this gap, but a foundation model for mouse behavior must handle more than per-animal kinematics — many of the most important behaviors (aggression, courtship, mounting, allogrooming) are intrinsically *social*. Existing self-supervised pose models [Sun et al., 2021b, Azabou et al., 2023b, Stoffl et al., 2024] were not designed to scale into a foundation-model regime, for three reasons. **(1) Pooling destroys interaction structure** [Azabou et al., 2023b, Qi et al., 2017]: per-animal representations collapsed via mean- or max-pooling lose the relational signal defining dyadic behaviors. **(2) Fixed-ordering or fixed-N inputs** [Sun et al., 2021b, Stoffl et al., 2024]: concatenating multi-animal keypoints hard-codes both the animal count N and the ordering at training time. **(3) Skeletal heterogeneity**: different labs annotate different keypoint sets, so a model trained on one skeleton cannot accept input from another without explicit alignment.

We introduce **MICE** (Multi-animal Interaction Context Encoder), a hierarchical self-supervised foundation model for mouse behavior that pretrains jointly across heterogeneous mouse-pose datasets with variable animal counts. The architecture makes no mouse-specific assumptions and extends to other tracked species; we focus on mice because they are the dominant model organism in

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

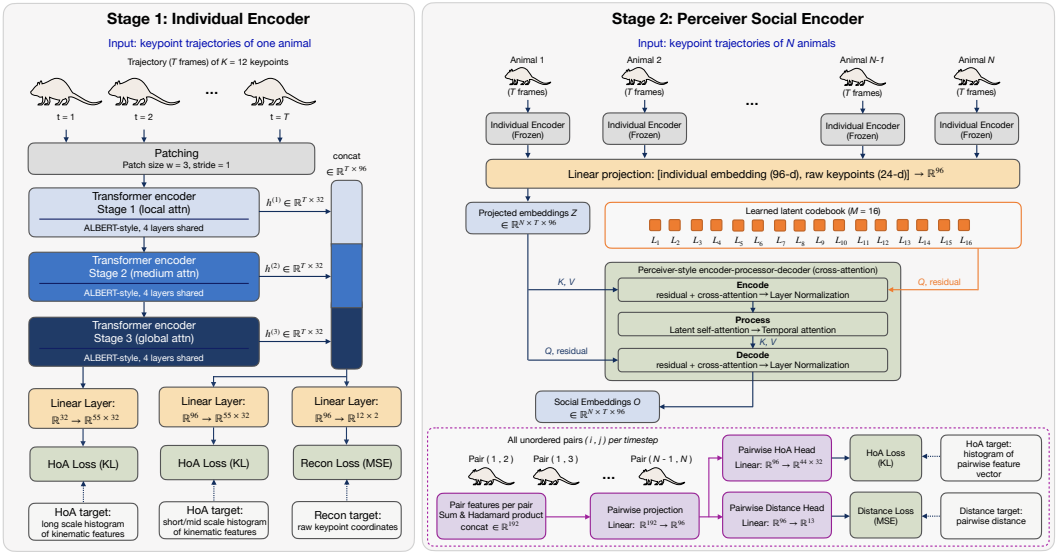

Figure 1: The MICE architecture. **Stage 1 (left, individual encoder):** a per-animal pose sequence is tokenized by a patching layer, then processed through three ALBERT-style transformer stages of increasing attention scope (local, medium, global) with weights shared within each stage. Per-stage outputs are concatenated and supervised by three losses: a long-scale Histogram-of-Activities (HoA) loss on Stage 3's output, a short/mid-scale HoA loss on the concatenated output, and a span-masked keypoint reconstruction loss. **Stage 2 (right, social encoder):** the frozen individual encoder is applied per mouse; per-animal embeddings are fused with raw keypoints and passed to a Perceiver with a learned latent codebook. The latents cross-attend over animals (encode), refine across time (process), and are read back to each animal (decode), producing per-animal social embeddings. Animal pairs are formed by an order-invariant projection and supervised by pairwise HoA and pairwise distance losses.

neuroscience and the only species with the public keypoint datasets and ethologically-validated labels needed for rigorous evaluation. Our contributions are: (i) a **mouse-behavior foundation model** pretrained jointly across six corpora — four published datasets (MABe22 [Sun et al., 2023], CalMS21 [Sun et al., 2021a], CRIM13 [Burgos-Artizzu et al., 2012], MouseLockbox [Reiske et al., 2025]) and two codenamed mouse tracks from the MABe 2025 challenge [Sun et al., 2025] (AdaptableSnail, NiftyGoldfinch) — aligned through a universal 12-keypoint vocabulary, a regime no prior keypoint-based behavior model has demonstrated; (ii) a **hierarchical individual encoder** with widening attention windows, capturing per-mouse kinematics at multiple temporal scales; (iii) a **Perceiver-style social encoder** that processes an arbitrary number of mice through a learned latent codebook, letting a single checkpoint serve any group size without retraining; and (iv) a **comprehensive evaluation** across all six datasets and four evaluation protocols, where MICE matches or exceeds prior keypoint-based baselines, with layer-wise probing showing that the social stage incrementally improves discriminability beyond the individual encoder, plus controlled tests of what the social stage contributes (§4.6).

2 Related Work

Pose-based behavior analysis. Classical pipelines hand-engineer features (velocity, distance, angles) for Random Forests or HMMs [Kabra et al., 2013, Hong et al., 2015]. Deep approaches [Nilsson et al., 2020, Segalin et al., 2021] learn from keypoints with supervised CNNs/RNNs but require large labeled corpora.

Self-supervised pose representation. BAMS [Azabou et al., 2023b] predicts a histogram of binned action features (HoA) from a dual-branch dilated TCN, the closest precursor to our individual-stage objective. TBERT [Sun et al., 2023] applies BERT-style masked modeling to keypoints; hBehave-MAE [Stoffl et al., 2024] introduces a hierarchical masked autoencoder over (time×mice×feature)

tensors but fixes both the skeleton and the animal count; PointNet [Qi et al., 2017] treats keypoints as an unordered point cloud. Unlike these, MICE's two-stage approach explicitly separates intra-animal kinematics from inter-animal interaction.

Video foundation models for mouse behavior. A parallel line of work pursues behavioral foundation models from raw video. BEAST [Wang et al., 2026] uses masked autoencoding plus temporal contrastive learning; Sun et al. [2024] apply VideoPrism [Zhao et al., 2024] to mouse behavior datasets; generative video models have also been used to relate neural dynamics to behavior [Wang et al., 2024]. Video preserves appearance but is lab-specific (lighting, camera, arena), while keypoints abstract these confounds and are far cheaper at training and inference. MICE focuses on the keypoint regime, where this abstraction enables transfer across labs with different recording conditions.

Social interaction modeling and foundation models for behavior. Existing social models hard-code animal count N [Eyjolfsdottir et al., 2016, Sun et al., 2023] or rely on $\mathcal{O}(N^2)$ GNN attention [Kipf et al., 2018]. The Perceiver [Jaegle et al., 2021] and Set Transformer [Lee et al., 2019] use learned latent queries to handle variable-length input; MICE is, to our knowledge, the first application of Perceiver-style bottlenecks to multi-animal social behavior. Foundation models exist for DNA [Nguyen et al., 2023], proteins [Lin et al., 2023], and brain activity [Azabou et al., 2023a]. MABe22 [Sun et al., 2023] and MABe 2025 [Sun et al., 2025] introduced benchmarks with baselines evaluated on their own datasets but not across labs. MICE is the first behavior foundation model to pretrain jointly across multiple mouse-pose datasets with variable group sizes.

3 Method

MICE has two stages (Figure 1): an *individual encoder* processes each mouse independently for per-animal kinematic representations at multiple temporal scales, and a *social encoder* operates on these to model inter-animal interactions. Each stage is pretrained with self-supervision matched to its structure — kinematics and dyadic relations respectively. We describe the universal keypoint vocabulary (§3.1), then each stage (§3.2, §3.3).

3.1 Universal Mouse-Keypoint Space

Joint pretraining across multiple datasets requires a shared input vocabulary, but different labs annotate different keypoint sets. We define a universal mouse-pose vocabulary of $K_{\max}{=}12$ keypoints following the MABe22 layout (nose, left/right ear, neck, left/right forepaw, center back, left/right hindpaw, tail base, tail middle, tail tip). Each dataset's native skeleton is mapped to this space by named-synonym matching, with missing keypoints zero-padded and masked from any feature that depends on them. Zero-padding is benign: keypoints have anatomical correspondence across labs, and the per-animal flat dimension $K_{\max}{\cdot}2{=}24$ is small enough that even CalMS21's 7-keypoint skeleton loses no more than half its slots to padding. This mapping is used only for combined pretraining; single-dataset models use each dataset's native keypoints.

Variable-N handling. The animal count is qualitatively different from keypoint count. Mouse anatomy bounds $K_{\max}{=}12$, but $N_{\max}$ has no such ceiling: group living, colony recordings, or cross-species extensions can push it beyond the dyadic regime, and a model trained at $N_{\max}{=}4$ cannot run on $N{=}6$ without retraining. MICE's Perceiver social encoder (§3.3) cross-attends over whichever animals are present, so a single checkpoint runs on any N. Fixed-input baselines (TBERT [Sun et al., 2023], hBehaveMAE [Stoffl et al., 2024]) commit to $N_{\max}$ at training time; pooling-based baselines (BAMS [Azabou et al., 2023b], PointNet [Qi et al., 2017]) handle variable N but lose interaction signal.

3.2 Individual Encoder: Hierarchical Transformer

The individual encoder produces per-mouse kinematic representations (Figure 1, left). It takes a per-animal pose sequence (T frames of $K{=}12$ keypoints in 2D) and returns a per-token feature vector at multiple temporal scales. Self-attention is bidirectional throughout (adapted from ALBERT [Lan et al., 2020]): behavior analysis is post-hoc, so each token attends to past and future frames within its window.

Hierarchical design with continuous window schedule. Mouse behavior unfolds at very different temporal scales: a paw step lasts tens of milliseconds, a sniff bout about a second, an attack episode several seconds. Prior designs commit to a small set of scales (hBehaveMAE [Stoffl et al., 2024] uses discrete pooling levels; BAMS [Azabou et al., 2023b] uses a short- and long-term branch). Our design instead lets the attention scope grow continuously with depth (readout and supervision still use a few scales, Eqs. 2–3): across the 12 transformer layers ($i \in \{0, \ldots, 11\}$), the attention window $\omega(i)$ grows linearly,

$$\omega(i) \;=\; \left\lfloor \omega_{\text{short}} + \frac{i}{11}\left(\omega_{\text{long}} - \omega_{\text{short}}\right) \right\rfloor, \tag{1}$$

with ω_{short}=3 and ω_{long}=T tokens, then rounded to the nearest odd integer so windows are symmetric around each query token. The full set of layers covers a continuum from short to long-range.

Stage organization. The 12 layers are grouped into three stages of 4, indexed $g \in \{1, 2, 3\}$, for two reasons. First, it enables ALBERT-style weight sharing [Lan et al., 2020]: layers within a stage share parameters, striking a middle ground between full sharing (one block handling every scale) and 12 independent layers (quadrupling the parameter count). Second, the stages define *readout points* from which we extract one feature map per stage and concatenate them (Figure 1, "concat"):

$$\mathbf{H}_{\text{ind}} \;=\; \text{concat}\Big[\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}\Big] \;\in\; \mathbb{R}^{T\times 96}, \qquad D_{\text{ind}} = 3D = 96, \tag{2}$$

where $\mathbf{h}^{(g)} \in \mathbb{R}^{T\times D}$ is stage g's last-layer output and D=32. Three readouts are the minimum that gives a short/mid/long-range summary — a single readout would force one representation to capture every scale, while per-layer probing would inflate the downstream parameter cost.

Self-supervised objectives. The encoder is pretrained with three losses. Two are *Histogram-of-Activities (HoA) prediction* losses, inspired by BAMS [Azabou et al., 2023b]: per-token histograms of a kinematic feature vector are predicted at two readout heads, yielding a short/mid-scale loss $\mathcal{L}_{\text{HoA}}^{\text{short/mid}}$ on $\mathbf{H}_{\text{ind}}$ and a long-scale loss $\mathcal{L}_{\text{HoA}}^{\text{long}}$ on $\mathbf{h}^{(3)}$ alone. The third is *span-masked keypoint reconstruction* $\mathcal{L}_{\text{recon}}$ on $\mathbf{H}_{\text{ind}}$, anchoring the encoder to raw (x, y) positions that HoA alone could leave unconstrained. The total objective combines all three:

$$\mathcal{L}_{\text{ind}} \;=\; \mathcal{L}_{\text{HoA}}^{\text{short/mid}} + \mathcal{L}_{\text{HoA}}^{\text{long}} + \mathcal{L}_{\text{recon}}, \tag{3}$$

where each $\mathcal{L}_{\text{HoA}}$ is a per-token KL divergence between predicted and target histograms and $\mathcal{L}_{\text{recon}}$ is the per-token MSE. Loss equations and target-window sizes are in Appendix A, and feature definitions in Appendix C.

3.3 Social Encoder: Perceiver-based Model

The social encoder operates on top of the per-animal representations from §3.2 and produces an interaction-aware embedding for each mouse (Figure 1, right). It runs on multi-animal pose with N animals across T frames.

Why a Perceiver. Two requirements shape the design: handle a variable number of mice N so a single trained model serves datasets with different group sizes, and preserve per-animal identity (pooling all animals into one vector would lose relational structure). A natural design satisfying both is direct N-to-N cross-attention between animals, but its outputs are still built on animal representations — the architecture cannot learn codes that specialize to relationships between animals (chasing, face-to-face confrontation, parallel locomotion). We instead use a Perceiver [Jaegle et al., 2021]: a small set of learned latent codes (M=16 slots, the orange row in Figure 1) reads from the N animals, gets refined over time, and is passed back to each animal. Each latent is free to specialize to a behavioral motif, and at decode time each animal looks up which motifs it participates in.

Inputs. The frozen individual encoder is applied per mouse, yielding hidden representations $\mathbf{H}_{\text{ind}}^{(n)} \in \mathbb{R}^{T\times 96}$. A linear projection (Figure 1, "Linear projection") fuses each per-animal representation with raw keypoints into a per-token embedding $\mathbf{Z}_{n,t} \in \mathbb{R}^{96}$. During pretraining, contiguous temporal spans of $\mathbf{Z}$ are replaced with a mask token. Details in Appendix B.

Encode–process–decode. The Perceiver applies three operations per timestep (Figure 1, green block). The learned latent codebook $\mathbf{L}_0 \in \mathbb{R}^{M\times 96}$ has $M{=}16$ slots; $\mathbf{Z}_{:,t}$ stacks the N animal embeddings at timestep t. Using multi-head cross-attention $\mathrm{CrossAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$, latent self-attention $\mathrm{SelfAttn}$ (within a timestep), temporal self-attention $\mathrm{TempAttn}$ (across T time tokens, per latent slot), and layer normalization $\mathrm{LayerNorm}$:

$$\textbf{Encode:} \quad \mathbf{L}'_t = \mathrm{LayerNorm}\Big(\mathbf{L}_0 + \mathrm{CrossAttn}\big(\mathbf{Q}{=}\mathbf{L}_0,\, \mathbf{K}{=}\mathbf{V}{=}\mathbf{Z}_{:,t}\big)\Big), \tag{4}$$

$$\textbf{Process:} \quad \mathbf{L}''_t = \mathrm{TempAttn}\Big(\mathrm{SelfAttn}\big(\mathbf{L}'_t\big)\Big), \tag{5}$$

$$\textbf{Decode:} \quad \mathbf{O}_{n,t} = \mathrm{LayerNorm}\Big(\mathbf{Z}_{n,t} + \mathrm{CrossAttn}\big(\mathbf{Q}{=}\mathbf{Z}_{n,t},\, \mathbf{K}{=}\mathbf{V}{=}\mathbf{L}''_t\big)\Big). \tag{6}$$

Encode reads animal information into the latents; Process refines them within timestep then across time; Decode reads the refined latents back to each animal. $\mathbf{O}_{n,t} \in \mathbb{R}^{96}$ is the per-animal, per-timestep social embedding.

Pairwise projection. Since social self-supervision is defined over pairs, we project each animal pair's social embeddings into a single representation. For each unordered pair (i, j) with both animals valid at timestep t (Figure 1, dashed pink box):

$$\mathbf{p}_{ij,t} \;=\; \mathrm{Linear}([\,\mathbf{O}_{i,t} + \mathbf{O}_{j,t};\; \mathbf{O}_{i,t} \odot \mathbf{O}_{j,t}\,]) \;\in\; \mathbb{R}^{96}, \tag{7}$$

where $\mathrm{Linear} : \mathbb{R}^{192} \to \mathbb{R}^{96}$ applies to the concatenation of the elementwise sum and Hadamard product. Both operations are swap-invariant, so $\mathbf{p}_{ij,t}{=}\mathbf{p}_{ji,t}$ by construction; we choose order-invariance because directionality ("i chases j" vs. "j chases i") is recoverable from $\mathbf{O}_i$ and $\mathbf{O}_j$, while our pretraining targets are largely symmetric in (i, j).

Self-supervised objectives. The social encoder is pretrained with two pairwise losses. The primary loss is *pairwise HoA prediction* — the dyadic analog of the individual HoA loss — predicting histograms of a pairwise feature vector capturing velocity differences, heading geometry, and other dyadic kinematics (Appendix C). The auxiliary loss is *pairwise distance regression* on the 13 per-keypoint and centroid distances, anchoring the pair representation to physical proximity. Because input spans are masked during pretraining, it also acts as a masked reconstruction of inter-animal distances; it targets relative positions because the frozen individual encoder already captures absolute ones. The objective is

$$\mathcal{L}_{\text{soc}} = \mathcal{L}_{\text{HoA-pair}} + \mathcal{L}_{\text{dist}}, \tag{8}$$

where $\mathcal{L}_{\text{HoA-pair}}$ is a per-pair-timestep KL between predicted and target histograms and $\mathcal{L}_{\text{dist}}$ is per-pair-timestep MSE. Loss equations and validity-mask definition are in Appendix B.

4 Experiments

We evaluate MICE under four protocols on six mouse-pose datasets: linear probing (§4.1), unsupervised behavioral segmentation (§4.2), cross-dataset transfer (§4.3), and few-shot label efficiency (§4.4), supplemented by layer-wise probing (§4.5), a social-stage ablation (§4.6), and leave-one-dataset-out transfer (§4.7). Variable-N and embedding-variant ablations appear in Appendix I.

Datasets. The six datasets span $N{\in}\{1, 2, 3, 4\}$ mice and 7–27 native keypoints, all mapped to the universal 12-keypoint vocabulary (§3.1): MABe22 [Sun et al., 2023], CalMS21 [Sun et al., 2021a], CRIM13 [Burgos-Artizzu et al., 2012], MouseLockbox [Reiske et al., 2025], AdaptableSnail, and NiftyGoldfinch (the last two are codenamed mouse tracks from MABe 2025 [Sun et al., 2025]). Per-dataset details are in Appendix D.

Protocol. We pretrain MICE on all six datasets jointly, then evaluate frozen embeddings via linear probing per-dataset (an MLP on frozen features); for comparison we also report *single-dataset* pretraining. The headline MICE configuration is `ind+soc_cat`; other pooling combinations appear as ablations in Tables 1 and 12. Probe details are in Appendix E.

Table 1: Linear-probing macro-F1 (%) across the six datasets under two pretraining regimes: *single* (pretrain and probe on the same dataset) and *combined* (joint pretrain on all six, probe per dataset). Mean ± standard deviation over 5 random seeds. Within each regime, **bold** = best method and underline = best baseline; * and † additionally mark these under the *single* regime. "—": social variants do not apply to single-animal MouseLockbox. AdaptSnail = AdaptableSnail; Goldfinch = NiftyGoldfinch; Lockbox = MouseLockbox.

Model	Regime	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch	Lockbox
PointNet [Qi et al., 2017]	single	$36.54_{\pm 0.7}$	$62.24_{\pm 0.5}$	$21.48_{\pm 0.7}$	$7.91_{\pm 0.5}$	$21.73_{\pm 0.6}$	$13.73_{\pm 0.4}$
	combined	$40.75_{\pm 0.6}$	$65.93_{\pm 0.4}$	$26.49_{\pm 0.6}$	$\underline{19.59}_{\pm 0.6}$	$19.61_{\pm 0.5}$	$13.42_{\pm 0.4}$
TBERT [Sun et al., 2023]	single	$20.18_{\pm 0.8}$	$42.23_{\pm 0.7}$	$19.01_{\pm 0.7}$	$\underline{11.12}^{\dagger}_{\pm 0.6}$	$13.69_{\pm 0.6}$	$14.20_{\pm 0.4}$
	combined	$25.45_{\pm 0.7}$	$56.40_{\pm 0.5}$	$23.35_{\pm 0.6}$	$13.72_{\pm 0.5}$	$12.59_{\pm 0.5}$	$\underline{13.55}_{\pm 0.4}$
BAMS [Azabou et al., 2023b]	single	$\underline{43.90}^{\dagger}_{\pm 0.6}$	$74.98_{\pm 0.4}$	$\underline{41.41}^{\dagger}_{\pm 0.7}$	$8.95_{\pm 0.5}$	$14.86_{\pm 0.6}$	$13.30_{\pm 0.4}$
	combined	$46.17_{\pm 0.5}$	$59.08_{\pm 0.5}$	$29.65_{\pm 0.6}$	$18.67_{\pm 0.6}$	$17.39_{\pm 0.5}$	$12.92_{\pm 0.4}$
hBehaveMAE [Stoffl et al., 2024]	single	$42.72_{\pm 0.6}$	$\underline{76.05}^{\dagger}_{\pm 0.4}$	$38.30_{\pm 0.7}$	$9.03_{\pm 0.5}$	$\underline{26.02}^{\dagger}_{\pm 0.6}$	$\underline{14.58}^{\dagger}_{\pm 0.4}$
	combined	$\underline{47.82}_{\pm 0.5}$	$\underline{77.68}_{\pm 0.3}$	$\underline{40.35}_{\pm 0.6}$	$12.13_{\pm 0.5}$	$\underline{29.21}_{\pm 0.5}$	$13.21_{\pm 0.3}$
Progressive (ours, ablation)	single	$53.93_{\pm 0.6}$	$54.93_{\pm 0.6}$	$18.71_{\pm 0.8}$	$14.17_{\pm 0.6}$	$9.50_{\pm 0.5}$	$15.78_{\pm 0.4}$
	combined	$61.87_{\pm 0.4}$	$66.97_{\pm 0.4}$	$27.98_{\pm 0.6}$	$23.63_{\pm 0.6}$	$13.54_{\pm 0.5}$	$13.54_{\pm 0.4}$
Multiscale (ours, ablation)	single	$52.28_{\pm 0.6}$	$64.03_{\pm 0.5}$	$24.54_{\pm 0.7}$	$14.61_{\pm 0.6}$	$17.43_{\pm 0.5}$	$15.72_{\pm 0.4}$
	combined	$63.68_{\pm 0.4}$	$74.24_{\pm 0.4}$	$31.40_{\pm 0.6}$	$20.92_{\pm 0.6}$	$15.39_{\pm 0.5}$	$13.04_{\pm 0.4}$
MICE (`ind_cat`)	single	$65.58_{\pm 0.5}$	$79.76_{\pm 0.3}$	$39.12_{\pm 0.6}$	$20.91_{\pm 0.6}$	$28.55_{\pm 0.5}$	$\mathbf{15.93}^{\star}_{\pm 0.4}$
	combined	$67.85_{\pm 0.4}$	$81.22_{\pm 0.3}$	$44.71_{\pm 0.5}$	$22.66_{\pm 0.5}$	$29.12_{\pm 0.5}$	$\mathbf{13.68}_{\pm 0.4}$
MICE (`soc_cat`)	single	$67.13_{\pm 0.5}$	$79.34_{\pm 0.3}$	$40.92_{\pm 0.5}$	$21.34_{\pm 0.5}$	$29.42_{\pm 0.5}$	—
	combined	$68.88_{\pm 0.4}$	$81.86_{\pm 0.3}$	$44.09_{\pm 0.5}$	$25.15_{\pm 0.4}$	$30.35_{\pm 0.6}$	—
MICE (`ind+soc_cat`)	single	$\mathbf{67.60}^{\star}_{\pm 0.5}$	$\mathbf{83.28}^{\star}_{\pm 0.4}$	$\mathbf{44.61}^{\star}_{\pm 0.7}$	$\mathbf{21.93}^{\star}_{\pm 1.0}$	$\mathbf{31.08}^{\star}_{\pm 0.7}$	—
	combined	$\mathbf{69.26}_{\pm 0.3}$	$\mathbf{84.56}_{\pm 0.3}$	$\mathbf{46.38}_{\pm 0.6}$	$\mathbf{26.76}_{\pm 0.8}$	$\mathbf{31.90}_{\pm 0.7}$	—

Baselines. We compare to four methods (BAMS [Azabou et al., 2023b], PointNet [Qi et al., 2017], TBERT [Sun et al., 2023], hBehaveMAE [Stoffl et al., 2024]) and two MICE ablations: **Progressive** (individual encoder with fixed-$N_{\max}$ padding) isolates the variable-N contribution; **Multiscale** (multi-resolution supervision without widening attention windows) isolates the widening-window design. All methods share the universal keypoint mapping and identical downstream probes. MICE is the smallest competitive model (583K encoder parameters, about 7× fewer than hBehaveMAE; Table 21). Implementation details in Appendix G.

4.1 Linear-Probe Behavior Classification

Table 1 reports macro-F1 under two pretraining regimes: *single* (pretrain and probe on the same dataset) and *combined* (joint pretrain on all six). Three findings stand out.

MICE matches or exceeds the strongest baseline on every dataset under both regimes. The largest gaps appear on the datasets with the most mice per recording — MABe22 (N=3) and AdaptableSnail ($N \in \{3, 4\}$) — where variable-N handling and joint social/individual modeling matter most. Fixed-ordering baselines such as hBehaveMAE trail MICE by a wider margin on these multi-animal datasets than on dyadic ones, supporting the variable-N argument in §3.1. Appendices J.6 and J.7 compare MICE with a vanilla transformer and with a supervised reference.

The social stage adds consistent value on multi-animal datasets. MICE (`ind_cat`) vs. MICE (`ind+soc_cat`) isolates the social encoder's contribution, positive on every multi-animal dataset under combined pretraining. The Progressive ablation, which stacks all N animals into a single fixed-size input, lags MICE (`ind_cat`) on most multi-animal datasets, showing that handling each mouse separately matters.

Combined pretraining helps on most datasets. On the five datasets with shared multi-animal structure (MABe22, CalMS21, CRIM13, AdaptableSnail, NiftyGoldfinch), combined pretraining matches or exceeds single-dataset pretraining for all MICE variants — direct evidence for the

Table 2: Unsupervised behavioral segmentation via K-means clustering (K=50) on frozen embeddings, mean $\pm$ standard deviation across the four datasets with frame-level labels (CalMS21, CRIM13, AdaptableSnail, NiftyGoldfinch) over 5 random seeds. All metrics reported as %. **Bold** = best, underline = best baseline. MICE leads ARI and NMI by sizable margins and matches hBehaveMAE on Boundary-F1 (within seed variation).

Model	ARI (%) ↑	NMI (%) ↑	Boundary-F1 (%) ↑
PointNet [Qi et al., 2017]	$\underline{33.73}_{\pm 0.24}$	$\underline{27.50}_{\pm 0.18}$	$26.19_{\pm 0.31}$
TBERT [Sun et al., 2023]	$24.59_{\pm 0.27}$	$18.64_{\pm 0.21}$	$29.06_{\pm 0.28}$
BAMS [Azabou et al., 2023b]	$9.28_{\pm 0.34}$	$8.03_{\pm 0.28}$	$16.45_{\pm 0.36}$
hBehaveMAE [Stoffl et al., 2024]	$31.32_{\pm 0.21}$	$25.44_{\pm 0.16}$	$\underline{31.37}_{\pm 0.24}$
Progressive (ours, ablation)	$9.33_{\pm 0.32}$	$8.13_{\pm 0.26}$	$23.93_{\pm 0.29}$
Multiscale (ours, ablation)	$26.13_{\pm 0.23}$	$21.41_{\pm 0.19}$	$30.33_{\pm 0.25}$
MICE (`ind+soc_cat`)	$\mathbf{43.33}_{\pm 0.17}$	$\mathbf{35.63}_{\pm 0.13}$	$\mathbf{31.57}_{\pm 0.21}$

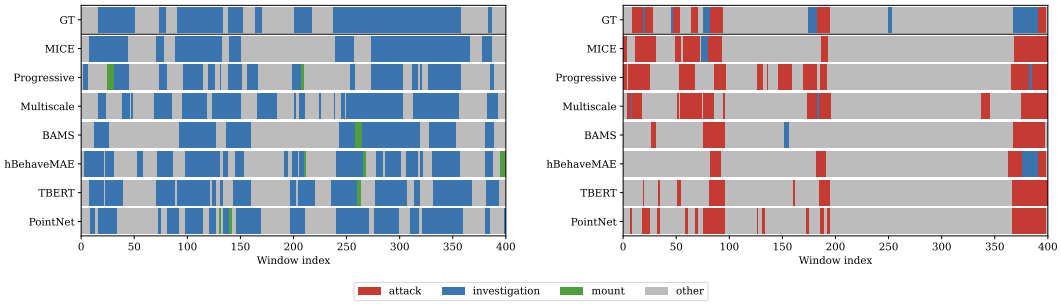

Figure 2: Qualitative segmentation timelines from CalMS21, investigation episodes (left) and attack episodes (right). Each row shows one method's hard cluster assignments mapped to ground-truth classes. MICE matches ground-truth boundaries most closely; baselines either fragment episodes and produce false positives (e.g., Progressive and Multiscale) or miss real episodes (e.g., BAMS and hBehaveMAE on the early attack clusters).

foundation-model framing. On MouseLockbox it slightly hurts (single-animal-only structure limits transfer). Combined-regime MICE remains the best method under combined pretraining.

4.2 Unsupervised Behavioral Segmentation

We test whether frozen embeddings recover behavior *without labels* by clustering them with K-means and scoring the segmentation against frame-level annotations. Per-window embeddings are partitioned into K=50 clusters; consecutive frames sharing a cluster are grouped into segments and mapped to ground-truth classes by majority vote. Table 2 reports ARI, NMI, and Boundary-F1 across the four datasets with frame-level labels. MICE leads ARI and NMI by sizable margins ($\sim$10 and $\sim$8 points over the strongest baseline) and matches hBehaveMAE on Boundary-F1; the three clustering metrics confirm MICE's embeddings align with behavior categories best among all methods. Protocol details, per-dataset results, and a comparison with Keypoint-MoSeq [Weinreb et al., 2024] are in Appendix F.

4.3 Cross-Dataset Transfer

Using the six single-dataset MICE checkpoints from §4.1, we linear-probe each on every dataset. As expected, the diagonal is best within each column among single-dataset checkpoints. Table 3 reports the transfer matrix; the bottom row reproduces the combined-pretraining result. Two patterns emerge.

Combined pretraining matches or exceeds single-dataset pretraining on five of six targets — direct evidence that joint pretraining transfers positive information across datasets. The exception is MouseLockbox, where combined pretraining trails by 2.3 points; we attribute this to label heterogeneity, since MouseLockbox's fine-grained puzzle-box behaviors differ from the social and locomotor labels dominating the other five datasets.

Table 3: Cross-dataset transfer: macro-F1 (%) when MICE is pretrained on the row dataset and linear-probed on the column. Mean ± standard deviation over 5 random seeds. Diagonal cells correspond to single-regime MICE results in Table 1 and the bottom row to the combined-regime results; MouseLockbox uses `ind_cat`. **Bold** = best per column. AdaptSnail = AdaptableSnail; Goldfinch = NiftyGoldfinch; Lockbox = MouseLockbox.

	Probe dataset					
Pretrain dataset	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch	Lockbox
MABe22	$67.60_{\pm 0.45}$	$72.65_{\pm 0.51}$	$38.45_{\pm 0.78}$	$20.85_{\pm 0.94}$	$19.34_{\pm 0.71}$	$13.41_{\pm 0.36}$
CalMS21	$58.74_{\pm 0.62}$	$83.28_{\pm 0.39}$	$38.34_{\pm 0.85}$	$20.74_{\pm 0.97}$	$26.59_{\pm 0.83}$	$13.13_{\pm 0.41}$
CRIM13	$53.18_{\pm 0.71}$	$72.05_{\pm 0.48}$	$44.61_{\pm 0.72}$	$20.82_{\pm 0.96}$	$24.90_{\pm 0.81}$	$15.55_{\pm 0.42}$
AdaptSnail	$62.45_{\pm 0.66}$	$75.59_{\pm 0.52}$	$27.64_{\pm 0.85}$	$21.93_{\pm 1.04}$	$22.17_{\pm 0.79}$	$13.64_{\pm 0.39}$
Goldfinch	$47.93_{\pm 0.94}$	$73.98_{\pm 0.55}$	$32.91_{\pm 0.87}$	$19.77_{\pm 0.91}$	$31.08_{\pm 0.74}$	$15.39_{\pm 0.45}$
Lockbox	$43.27_{\pm 1.08}$	$74.25_{\pm 0.59}$	$30.14_{\pm 0.92}$	$20.75_{\pm 0.87}$	$14.11_{\pm 0.62}$	$\mathbf{15.93}_{\pm 0.43}$
Combined	$\mathbf{69.26}_{\pm 0.34}$	$\mathbf{84.56}_{\pm 0.31}$	$\mathbf{46.38}_{\pm 0.55}$	$\mathbf{26.76}_{\pm 0.78}$	$\mathbf{31.90}_{\pm 0.72}$	$13.68_{\pm 0.37}$

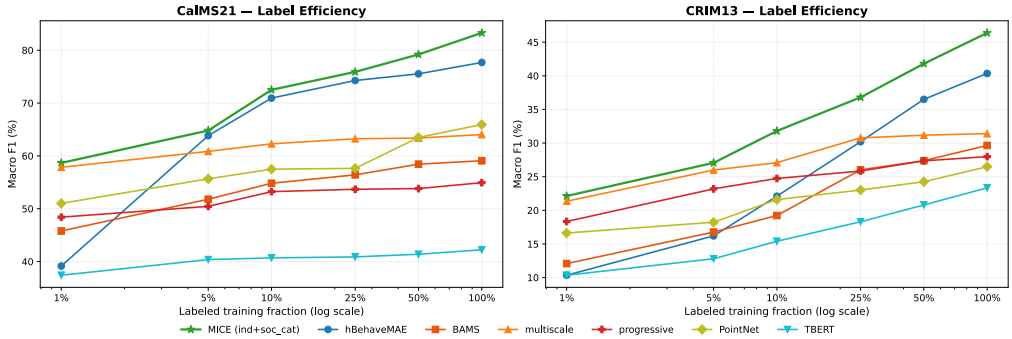

Figure 3: Label efficiency on CalMS21 (left) and CRIM13 (right). Each method's frozen encoder feeds the same downstream MLP probe, retrained at six labeled-data fractions (log x-axis). Each curve is the mean macro-F1 over 5 random seeds on the official test split; standard deviations are within $\pm 1.5\%$ at every fraction (largest at the 1% point, where training-set variance dominates) and are omitted from the plot for visual clarity. MICE leads at every label fraction; the gap over hBehaveMAE is largest at the lowest fractions.

Single-animal pretraining transfers poorly to multi-animal targets. The MouseLockbox row produces the lowest off-diagonal scores on MABe22 and NiftyGoldfinch, matching the variable-N argument from §3.1: a model never exposed to multi-animal interaction during pretraining cannot encode the social context downstream probes need. Appendix J.5 gives the same matrices for hBehaveMAE and PointNet.

4.4 Few-Shot Label Efficiency

We evaluate label efficiency on CalMS21 and CRIM13, two established mouse social-behavior benchmarks covering complementary difficulty regimes (CalMS21 moderately balanced; CRIM13 heavily imbalanced). We sweep the labeled training fraction across $\{1, 5, 10, 25, 50, 100\}\%$ and retrain only the MLP probe with the encoder frozen.

MICE leads on both datasets at every label fraction (Figure 3), and its margin over hBehaveMAE is largest at low fractions. With only 1% of CalMS21 labels, MICE already matches BAMS's and exceeds TBERT's full-label performance, and substantially outperforms hBehaveMAE, whose representations degrade sharply when labels are scarce. With 5% labels, MICE on CalMS21 nearly matches PointNet's full-label performance. A similar pattern holds on the harder CRIM13.

4.5 Layer-wise Probing Analysis

To localize *where* discriminative information emerges, we extract embeddings at every intermediate layer with forward hooks and train an MLP probe per layer (encoder frozen). For MICE, this gives

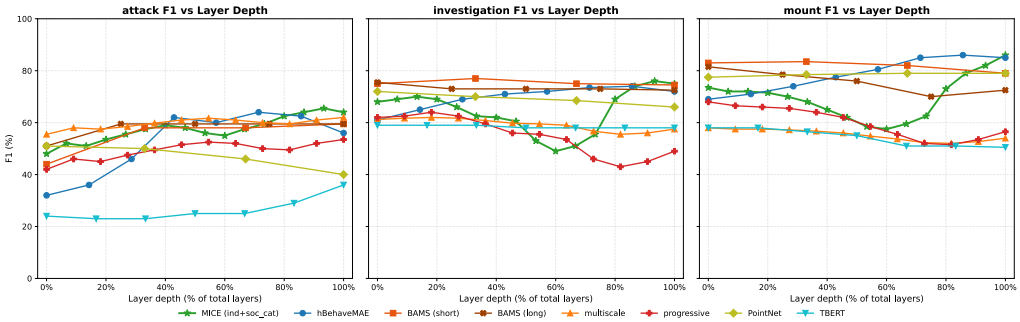

Figure 4: Layer-wise linear-probe F1 on CalMS21 for three social behavior classes. x-axis: probe depth as percentage of total encoder depth. Each curve is the mean over 5 random seeds; per-layer standard deviations are within $\pm 1.0\%$ across all methods and are omitted from the plot for visual clarity. For MICE, 0–75% is the individual encoder and 75–100% is the social encoder. BAMS (short)/(long) = BAMS's two parallel branches; the long branch targets longer clips (Appendix G).

Table 4: Train-time social ablation: macro-F1 (%) with the social stage retrained once per condition from one frozen individual encoder with correct, mis-paired, or removed partners, and probed likewise; mean ± standard deviation over 5 random probe seeds. First and last rows: combined-regime MICE (`ind+soc_cat`, `ind_cat`) from Table 1. MouseLockbox (one mouse, no social stage) is omitted. **Bold** = best per column.

Partners (train and probe)	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch
correct (default)	$\mathbf{69.26}_{\pm 0.34}$	$\mathbf{84.56}_{\pm 0.31}$	$\mathbf{46.38}_{\pm 0.55}$	$\mathbf{26.76}_{\pm 0.78}$	$\mathbf{31.90}_{\pm 0.72}$
mis-paired (shuffled)	$67.31_{\pm 0.54}$	$81.19_{\pm 0.48}$	$44.17_{\pm 0.71}$	$22.79_{\pm 0.86}$	$29.33_{\pm 0.91}$
removed	$66.92_{\pm 0.50}$	$81.00_{\pm 0.54}$	$44.08_{\pm 0.66}$	$21.83_{\pm 0.92}$	$29.02_{\pm 0.83}$
`ind_cat` only	$67.85_{\pm 0.42}$	$81.22_{\pm 0.28}$	$44.71_{\pm 0.47}$	$22.66_{\pm 0.53}$	$29.12_{\pm 0.48}$

16 probe points: 12 in the individual encoder and 4 in the social encoder. Probe-point details in Appendix H. Figure 4 shows three patterns.

Baselines show no depth-wise organization of behavior. For TBERT, PointNet, and each of BAMS's branches, early and late layers decode behavior about equally well, and no behavior shows up at a particular depth.

MICE's individual encoder dips through Stage 3 on investigation and mount. F1 dips from Stage 1 to 3 before recovering; Progressive shows the same pattern. The likely cause is what each stage represents: early layers (short attention windows) emphasize positional information, while later layers (long attention windows) emphasize velocity and motion integrated over time. Investigation and mount are pose-configurational (nose-to-flank, vertical posture), aligning with early-layer positional features; attack is motion-defined (lunging, fast directional change), aligning with later-layer velocity features and climbing gradually with depth.

The social stage recovers MICE's F1 sharply; Progressive, lacking it, does not. Starting at 75% depth, MICE's F1 climbs steeply on investigation and mount, peaking near the final layer. Progressive dips at the same point but only partially recovers, ending well below MICE. This gap at 100% is the empirical contribution of the Perceiver social encoder, enabled by combining each mouse's kinematic embedding (kept via a residual connection) with context about nearby animals.

4.6 Social-Stage Ablation and Analysis

To test whether the social stage's gain comes from modeling the other animals rather than from capacity, keypoint fusion, the residual path, or the pairwise objective, we retrain it with partners removed or mis-paired (real mice from other clips), keeping every parameter and loss, and probe each model under its training condition (Table 4). Two findings stand out.

Without correct partner information, the social stage's gain disappears. Both altered models fall back to about the level of the individual readout despite keeping the same capacity, keypoint fusion,

Table 5: Leave-one-dataset-out transfer: macro-F1 (%) when pretraining on all six datasets, the other five, or the target alone; mean ± standard deviation over 5 random probe seeds. Rows 1 and 3: combined and single-dataset MICE (Table 3); MouseLockbox uses `ind_cat`. **Bold** = best per column.

Pretraining	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch	Lockbox
all six datasets	$\mathbf{69.26}_{\pm 0.34}$	$\mathbf{84.56}_{\pm 0.31}$	$\mathbf{46.38}_{\pm 0.55}$	$\mathbf{26.76}_{\pm 0.78}$	$\mathbf{31.90}_{\pm 0.72}$	$13.68_{\pm 0.37}$
target left out (unseen)	$64.28_{\pm 0.52}$	$84.05_{\pm 0.38}$	$43.97_{\pm 0.72}$	$20.45_{\pm 0.69}$	$28.33_{\pm 0.82}$	$12.92_{\pm 0.53}$
target dataset only	$67.60_{\pm 0.45}$	$83.28_{\pm 0.39}$	$44.61_{\pm 0.72}$	$21.93_{\pm 1.04}$	$31.08_{\pm 0.74}$	$\mathbf{15.93}_{\pm 0.43}$

and residual path; since mis-pairing keeps the pairwise objective non-trivial, extra supervision does not explain the gain either.

The social embedding encodes how the animals are arranged relative to each other. Linear probes show that it captures inter-animal distance and relative body orientation far better than the individual embedding, while adding nothing about the focal mouse's own movement (Appendix J.3): the social stage contributes relational information that per-animal kinematics cannot express. Direct animal-to-animal attention scores slightly lower on every dataset (Appendix J.1) and, unlike the latents, scales with N.

4.7 Leave-One-Dataset-Out Transfer

We pretrain MICE on five datasets and probe it on the unseen sixth (Table 5). Two patterns emerge.

MICE transfers to unseen datasets at a modest, dataset-dependent cost. Leaving out CalMS21 or MouseLockbox barely matters, whereas leaving out AdaptableSnail or MABe22 costs more: transfer works best when other datasets cover the target's behaviors.

A model that never saw the target nearly matches one pretrained on it. Pretrained only on the other five datasets, MICE scores within 3.3 points of a model pretrained on the target alone, making a checkpoint pretrained on other datasets a reasonable starting point for a new dataset.

5 Discussion and Conclusion

Why does the social channel complement the individual channel? The individual and social pretraining objectives are non-overlapping (intra-animal kinematics vs. inter-animal pairwise quantities), so the two signals are not redundant: a fast-moving mouse next to another fast-moving mouse can be a chase or two parallel exploring mice, and only relative-heading disambiguates.

Limitations. (i) Synonym alignment is manually curated and must be extended by hand for new skeletons, though it needs no training data from a new lab. (ii) Datasets are mouse-only and filmed from above; extending to other species would require re-curating the universal vocabulary. Bottom-up and 3D data remain untested. (iii) The pretraining corpus, while spanning six datasets and $N \in \{1, 2, 3, 4\}$, remains small relative to vision/language foundation models; scaling-law studies are left for future work. The range $N \leq 4$ is limited by available labeled datasets. Hyperparameters were not tuned beyond one loss-weight sweep (Appendix J.2). (iv) Behavior label sets are coarse; ethological subtleties below current annotation resolution will require new benchmarks.

Broader impact. MICE could reduce annotation cost in behavioral neuroscience, accelerating drug discovery and welfare assessment. Adaptation for industrial-livestock surveillance carries dual-use risk; we recommend practitioners assess use-case ethics before deployment.

Conclusion. MICE is a self-supervised foundation model for mouse social behavior combining a universal 12-keypoint vocabulary that aligns six heterogeneous datasets, a hierarchical individual encoder with linearly-widening attention windows, a Perceiver-based social encoder that natively handles variable animal counts, and pairwise pretraining objectives grounded in ethological features. Across the six datasets and four protocols, MICE achieves state-of-the-art results and combined multi-dataset pretraining matches or exceeds dataset-specific pretraining on most datasets — evidence that a single checkpoint can serve diverse mouse-behavior tasks without architectural change.

Acknowledgments and Disclosure of Funding

This material is based upon work supported by the National Science Foundation under Award No. IIS-2336055. Research reported in this publication was supported by the National Institute of Mental Health of the National Institutes of Health under award number RF1MH140429. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

David J Anderson and Pietro Perona. Toward a science of computational ethology. *Neuron*, 84(1):18–31, 2014.

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

Mehdi Azabou, Michael Mendelson, Nauman Ahad, Maks Sorokin, Shantanu Thakoor, Carolina Urzay, and Eva Dyer. Relax, it doesn't matter how you get there: A new self-supervised approach for multi-timescale behavior analysis. *Advances in Neural Information Processing Systems*, 36:28491–28509, 2023b.

Xavier P Burgos-Artizzu, Piotr Dollár, Dayu Lin, David J Anderson, and Pietro Perona. Social behavior recognition in continuous video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1322–1329. IEEE, 2012.

Sandeep Robert Datta, David J Anderson, Kristin Branson, Pietro Perona, and Andrew Leifer. Computational neuroethology: a call to action. *Neuron*, 104(1):11–24, 2019.

Eyrun Eyjolfsdottir, Kristin Branson, Yisong Yue, and Pietro Perona. Learning recurrent representations for hierarchical behavior modeling. *arXiv preprint arXiv:1611.00094*, 2016.

Weizhe Hong, Ann Kennedy, Xavier P Burgos-Artizzu, Moriel Zelikowsky, Santiago G Navonne, Pietro Perona, and David J Anderson. Automated measurement of mouse social behaviors using depth sensing, video tracking, and machine learning. *Proceedings of the National Academy of Sciences*, 112(38):E5351–E5360, 2015.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and João Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pages 4651–4664. PMLR, 2021.

Mayank Kabra, Alice A Robie, Marta Rivera-Alba, Steven Branson, and Kristin Branson. JAABA: interactive machine learning for automatic annotation of animal behavior. *Nature Methods*, 10(1):64–67, 2013.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, pages 2688–2697. PMLR, 2018.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753. PMLR, 2019.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9):1281–1289, 2018.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, et al. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution. *Advances in Neural Information Processing Systems*, 36:43177–43201, 2023.

Simon RO Nilsson, Nastacia L Goodwin, Jia Jie Choong, Sophia Hwang, Hayden R Wright, Zane C Norville, Xiaoyu Tong, Dayu Lin, Brandon S Bentzley, Neir Eshel, et al. Simple Behavioral Analysis (SimBA)–an open source toolkit for computer classification of complex social behaviors in experimental animals. *bioRxiv*, 2020.

Talmo D Pereira, Joshua W Shaevitz, and Mala Murthy. Quantifying behavior to understand the brain. *Nature Neuroscience*, 23(12):1537–1549, 2020.

Talmo D Pereira, Nathaniel Tabris, Arie Matsliah, David M Turner, Junyu Li, Shruthi Ravindranath, Eleni S Papadoyannis, Edna Normand, David S Deutsch, Z Yan Wang, et al. SLEAP: A deep learning system for multi-animal pose tracking. *Nature Methods*, 19(4):486–495, 2022.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

Patrik Reiske, Marcus N. Boon, Niek Andresen, Sole Traverso, Katharina Hohlbaum, Lars Lewejohann, Christa Thöne-Reineke, Olaf Hellwich, and Henning Sprekeler. Mouse Lockbox Dataset: Behavior recognition for mice solving lockboxes. In *CVPR Workshop on Computer Vision for Animal Behavior Tracking and Modeling (CV4Animals)*, 2025. arXiv:2505.15408.

Cristina Segalin, Jalani Williams, Tomomi Karigo, May Hui, Moriel Zelikowsky, Jennifer J Sun, Pietro Perona, David J Anderson, and Ann Kennedy. The Mouse Action Recognition System (MARS) software pipeline for automated analysis of social behaviors in mice. *eLife*, 10:e63720, 2021.

Lucas Stoffl, Andy Bonnetto, Stéphane d'Ascoli, and Alexander Mathis. Elucidating the hierarchical nature of behavior with masked autoencoders. In *European Conference on Computer Vision*, pages 106–125. Springer, 2024.

Jennifer J Sun, Tomomi Karigo, Dipam Chakraborty, Sharada P Mohanty, Benjamin Wild, Quan Sun, Chen Chen, David J Anderson, Pietro Perona, Yisong Yue, et al. The multi-agent behavior dataset: Mouse dyadic social interactions. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021a.

Jennifer J Sun, Ann Kennedy, Eric Zhan, David J Anderson, Yisong Yue, and Pietro Perona. Task programming: Learning data efficient behavior representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2876–2885, 2021b.

Jennifer J Sun, Markus Marks, Andrew Wesley Ulmer, Dipam Chakraborty, Brian Geuther, Edward Hayes, Heng Jia, Vivek Kumar, Sebastian Oleszko, Zachary Partridge, et al. MABe22: A multi-species multi-task benchmark for learned representations of behavior. In *International Conference on Machine Learning*, pages 32936–32990. PMLR, 2023.

Jennifer J Sun, Hao Zhou, Long Zhao, Liangzhe Yuan, Bryan Seybold, David Hendon, Florian Schroff, David A Ross, Hartwig Adam, Bo Hu, et al. Video foundation models for animal behavior analysis. *bioRxiv*, 2024.

Jennifer J. Sun, Markus Marks, Sam Golden, Talmo Pereira, Ann Kennedy, Sohier Dane, Addison Howard, and Ashley Chow. MABe challenge – social action recognition in mice. `https://kaggle.com/competitions/MABe-mouse-behavior-detection`, 2025. Kaggle.

Yanchen Wang, Han Yu, Ari Blau, Yizi Zhang, Liam Paninski, Cole Lincoln Hurwitz, Matthew R Whiteway, et al. Animal behavioral analysis and neural encoding with transformer-based self-supervised pretraining. In *The Fourteenth International Conference on Learning Representations*, 2026.

Yule Wang, Chengrui Li, Weihan Li, and Anqi Wu. Exploring behavior-relevant and disentangled neural dynamics with generative diffusion models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Caleb Weinreb, Jonah E Pearl, Sherry Lin, Mohammed Abdal Monium Osman, Libby Zhang, Sidharta Annapragada, Eli Conlin, Red Hoffmann, Sofia Makowska, Winthrop F Gillis, et al. Keypoint-MoSeq: parsing behavior by linking point tracking to pose dynamics. *Nature Methods*, 21(7):1329–1339, 2024.

Long Zhao, Nitesh B Gundavarapu, Liangzhe Yuan, Hao Zhou, Shen Yan, Jennifer J Sun, Luke Friedman, Rui Qian, Tobias Weyand, Yue Zhao, et al. VideoPrism: A foundational visual encoder for video understanding. *arXiv preprint arXiv:2402.13217*, 2024.

A Individual Encoder: Implementation Details

This section provides reproduction-level details for the individual encoder described in §3.2.

Tokenization. The patching layer groups every $w{=}3$ consecutive frames at stride $s{=}1$ and projects each group via a single linear layer to a hidden state of dimension $D{=}32$, producing T tokens per input pose sequence.

Within-stage layer iteration. Denoting a single ALBERT block of stage g as f_g and the input to layer ℓ within stage g as $\mathbf{h}_\ell^{(g)}$, the layer iteration is

$$\mathbf{h}_{\ell+1}^{(g)} \;=\; f_g\Big(\mathbf{h}_\ell^{(g)};\, \omega(i)\Big), \qquad \ell = 0, 1, \ldots, L_g - 1,$$

where $\omega(i)$ is the attention window size at the global layer index $i = (g-1)\cdot L_g + \ell$. The input to stage $g{=}1$ is the patched token sequence; the input to stage $g{>}1$ is the output of the previous stage's last layer. Within-stage weight sharing means f_g is the same block applied L_g times.

HoA losses. The encoder predicts a soft histogram of the 55-dim kinematic feature vector $\boldsymbol{\phi}_t$ (Appendix C) over $B_{\text{HoA}}{=}32$ bins per feature. Predicting a soft histogram — rather than the features themselves — pushes the encoder toward short-window behavioral statistics rather than instantaneous values, closer to how ethologists characterize behavior. We apply HoA at two readout heads. The *short/mid-scale head* reads off the concatenated output $\mathbf{H}_{\text{ind}} \in \mathbb{R}^{T\times 96}$ with histogram targets computed over a 7-frame window centered on each token; this supervises representations from earlier-stage layers (small attention windows). The *long-scale head* reads off stage 3's output $\mathbf{h}^{(3)} \in \mathbb{R}^{T\times 32}$ with histogram targets computed over the full sequence of T frames; this supervises the long-range representation alone. With $\mathcal{T}{=}\{1,\ldots,T\}$ indexing tokens, $p^s_{\text{pred}}, p^l_{\text{pred}}$ the predicted histograms at the two readouts, and $p^s_{\text{target}}, p^l_{\text{target}}$ the corresponding 7-frame and T-frame target histograms,

$$\mathcal{L}_{\text{HoA}}^{\text{short/mid}} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\sum_{c=1}^{55} \mathrm{KL}\big(p^s_{\text{target}}(c,t)\,\big\|\,p^s_{\text{pred}}(c,t)\big)\,, \tag{9}$$

$$\mathcal{L}_{\text{HoA}}^{\text{long}} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\sum_{c=1}^{55} \mathrm{KL}\big(p^l_{\text{target}}(c,t)\,\big\|\,p^l_{\text{pred}}(c,t)\big)\,. \tag{10}$$

Span-masked reconstruction loss. Contiguous frame spans of length sampled uniformly from $[4, 12]$ frames are masked in the input by zero-filling them in input keypoint space before patching. A linear layer then reads the predicted keypoint configuration $\hat{\mathbf{x}}_t \in \mathbb{R}^{K\times 2}$ off $\mathbf{H}_{\text{ind}}$ at each token's center frame:

$$\mathcal{L}_{\text{recon}} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \big\|\hat{\mathbf{x}}_t - \mathbf{x}_t\big\|_2^2, \tag{11}$$

where $\mathbf{x}_t$ is the ground-truth keypoint configuration. Span-masking (rather than per-frame masking) prevents the model from solving the task by linearly interpolating between adjacent unmasked frames: mouse motion is locally smooth at single-frame scales, but multi-frame spans cover direction changes that linear interpolation cannot recover.

Total objective. $\mathcal{L}_{\text{ind}} = \mathcal{L}_{\text{HoA}}^{\text{short/mid}} + \mathcal{L}_{\text{HoA}}^{\text{long}} + \mathcal{L}_{\text{recon}}$, computed at all token positions for denser supervision than masked-only objectives.

B Social Encoder: Implementation Details

This section provides reproduction-level details for the social encoder described in §3.3.

Input projection. The frozen individual encoder produces per-animal hidden representations $\mathbf{H}_{\text{ind}}^{(n)} \in \mathbb{R}^{T \times D_{\text{ind}}}$ for $n = 1, \ldots, N$. A linear projection fuses each per-animal embedding with the raw keypoints at the corresponding token-center frame:

$$\mathbf{Z}_{n,t} \;=\; \text{Linear}\Big([\, \mathbf{H}_{\text{ind}}^{(n)}[t];\; \mathbf{x}_{n,t_c} \,] \Big) \;\in\; \mathbb{R}^{d_{\text{model}}}, \qquad d_{\text{model}} = D_{\text{ind}} = 96,$$

where t_c is the center frame of token t (the middle of the w=3-frame patching window) and $[\cdot;\cdot]$ denotes concatenation. The projection is a single linear layer $\mathbb{R}^{D_{\text{ind}}+2K} \to \mathbb{R}^{d_{\text{model}}}$ with no nonlinearity.

Process operation details. SelfAttn in Equation 5 denotes L_{soc}=2 stacked self-attention layers applied to the latent codebook within each timestep; TempAttn applies temporal attention across the T time tokens independently for each of the M=16 latent slots.

Pairwise HoA loss. For each unordered pair (i, j) at each timestep t, the pairwise HoA head predicts a soft histogram of the 44-dim pairwise feature vector $\boldsymbol{\psi}_{ij}(t)$ (defined in Appendix C; $44 = 3K + 8$ for K=12, comprising per-keypoint distance and velocity-difference components plus eight relational features) over B_{HoA}=32 bins per feature. Let $\mathcal{P}$ denote the set of valid (pair, timestep) tuples, where validity requires both animals to have observed keypoints at t:

$$\mathcal{L}_{\text{HoA-pair}} = \frac{1}{|\mathcal{P}|} \sum_{(i,j,t)\in\mathcal{P}} \sum_{c=1}^{44} \text{KL}\big(p_{\text{target}}(c,i,j,t) \,\big\|\, p_{\text{pred}}(c,i,j,t)\big)\,, \tag{12}$$

where $p_{\text{pred}}(c, i, j, t)$ is the predicted histogram for pair (i, j) at timestep t and feature channel c, and $p_{\text{target}}(c, i, j, t)$ is the target histogram computed from $\boldsymbol{\psi}_{ij}$ over a 7-frame window centered on t.

Pairwise distance loss. The pairwise distance head reads off each pair representation $\mathbf{p}_{ij,t}$ to predict the 13-dim distance vector $\hat{\mathbf{d}}_{ij}(t) \in \mathbb{R}^{13}$ comprising the 12 per-keypoint Euclidean distances between animal i's and animal j's corresponding keypoints, plus the centroid-to-centroid distance:

$$\mathcal{L}_{\text{dist}} = \frac{1}{|\mathcal{P}|} \sum_{(i,j,t)\in\mathcal{P}} \big\|\hat{\mathbf{d}}_{ij}(t) - \mathbf{d}_{ij}(t)\big\|_2^2, \tag{13}$$

where $\mathbf{d}_{ij}(t) \in \mathbb{R}^{13}$ is the corresponding ground-truth distance vector computed directly from the keypoint coordinates.

Validity mask. The set $\mathcal{P}$ excludes (pair, timestep) tuples where either animal has no valid keypoints at timestep t (e.g., out-of-frame or tracking failure). For datasets where validity is not annotated per-frame, all (pair, timestep) tuples are treated as valid. Sequences with no valid pair at any timestep are removed, so every pretraining batch contains valid pairs.

Total objective. $\mathcal{L}_{\text{soc}} = \mathcal{L}_{\text{HoA-pair}} + \mathcal{L}_{\text{dist}}$, computed at all valid (pair, timestep) tuples. The two terms are weighted equally; we did not tune this weighting, and unit weighting keeps the objective interpretable.

C Self-Supervision Feature Definitions

Individual-stage feature vector $\boldsymbol{\phi}_t$ (55-dim). For each token t, the kinematic feature vector consumed by the HoA loss comprises:

- **Per-keypoint velocity ($2K$=24 dims):** x- and y-velocity for each of the K=12 keypoints.
- **Per-keypoint centroid-relative position ($2K$=24 dims):** the offset $(x_k - x_{\text{ctr}}, y_k - y_{\text{ctr}})$ of each keypoint from the body centroid.
- **Body-length deviation (1 dim):** current body length minus the sequence-mean body length.
- **Forepaw–ear distance (1 dim):** a proxy for grooming, which involves the forepaws being brought near the head.
- **Heading direction (2 dims):** $(\cos h_t, \sin h_t)$ where h_t is the angle of the tail-base–to–nose vector.

- **Turn direction (2 dims):** $(\cos \dot{h}_t, \sin \dot{h}_t)$ where $\dot{h}_t$ is the angular velocity of the heading.
- **Tail extension (1 dim):** a proxy for arousal, computed as the tail-base–to–tail-tip distance.

Total: $24 + 24 + 1 + 1 + 2 + 2 + 1 = 55$ dimensions. All features are normalized by sequence-mean body length for scale invariance. Histogram targets use B_{HoA}=32 bins per feature, computed over a 7-frame sliding window for the short/mid-scale head and over the full sequence for the long-scale head (Appendix A).

Pairwise feature vector $\psi_{ij}(t)$ (44-dim). For each unordered pair of valid animals (i, j) at token t:

- **Per-keypoint inter-animal distance (K=12 dims):** $\|\mathbf{x}_i^{(k)} - \mathbf{x}_j^{(k)}\|_2/\bar{\ell}$ for each keypoint k, normalized by the mean body length $\bar{\ell}$.
- **Per-keypoint velocity difference ($2K$=24 dims):** per-axis absolute velocity difference $|\mathbf{v}_{i,a}^{(k)} - \mathbf{v}_{j,a}^{(k)}|$ for each keypoint k and axis $a \in \{x, y\}$, body-length normalized.
- **Centroid-centroid distance (1 dim):** $\|\mathbf{c}_i - \mathbf{c}_j\|_2/\bar{\ell}$.
- **Heading similarity (1 dim):** $(1 + \cos(h_j - h_i))/2$, close to 1 when both animals face the same direction (parallel motion) and close to 0 when anti-parallel.
- **Heading orthogonality (1 dim):** $|\sin(h_j - h_i)|$, close to 1 when headings are perpendicular and close to 0 when collinear.
- **Face-to-face geometry score (1 dim):** 0 when one animal's nose points at the other's tail (chasing); 1 when noses point at each other (face-to-face confrontation).
- **Body-size asymmetry (1 dim):** $|\ell_i - \ell_j|/\max(\ell_i, \ell_j)$, indicating size differences relevant to dominance.
- **Forepaw–ear (grooming) asymmetry (1 dim):** the difference in forepaw–ear distance between the two animals.
- **Tail-extension (arousal) asymmetry (1 dim):** the difference in tail extension between the two animals.
- **Turn-rate similarity (1 dim):** angular distance between the two animals' instantaneous turning rates.

Total: $12 + 24 + 8 = 44$ dimensions, where the 12 + 24 = 36 per-keypoint dimensions plus 8 global-pair scalars give the $3K + 8$ form referenced in Appendix B.

D Dataset Details

This section provides per-dataset specifics referenced from §4. All datasets are top-down mouse-pose video keypoints; each native skeleton is mapped to the universal 12-keypoint vocabulary by named-synonym matching (§3.1), with missing slots zero-padded and masked from any feature that depends on them.

MABe22 [Sun et al., 2023]. N=3 animals per sequence, 12 keypoints per mouse; its skeleton defines the universal vocabulary used by all other datasets. Behavior annotations are multi-label per window and cover a broad set of social behaviors.

CalMS21 [Sun et al., 2021a]. N=2 animals per sequence, 7 keypoints per mouse mapped to 7 of the 12 universal slots. Four mutually exclusive class labels: *attack*, *investigation*, *mount*, *other*. 70 training / 19 test sequences. Primary benchmark for dyadic social-interaction classification.

CRIM13 [Burgos-Artizzu et al., 2012]. N=2 animals per sequence, 7 keypoints per mouse mapped to 7 of the 12 universal slots. Eight mutually exclusive class labels: *other*, *approach*, *attack*, *disengage*, *mount*, *rear*, *selfgroom*, *sniff*, with the *other* class dominating (∼80% of frames). Also used for the few-shot evaluation in §4.4.

AdaptableSnail (MABe 2025 challenge). Variable group size, $N \in \{3, 4\}$ mice per sequence, 10 keypoints per mouse mapped to 10 of the 12 universal slots. Used to evaluate the social encoder under within-dataset N variation and to stress-test the universal mapping under a non-standard skeleton.

NiftyGoldfinch (MABe 2025 challenge). N=2 mice per sequence. The set of keypoints differs between sequences (each file lists its own body-part names), so the mapping to the universal vocabulary is applied per sequence rather than once per dataset. This makes it the most stringent test of the universal mapping.

MouseLockbox [Reiske et al., 2025]. N=1, 27 native DLC SuperAnimal keypoints mapped onto the 12 universal slots; the remaining 15 native keypoints are dropped. It covers fine-grained puzzle-box behaviors and is the only single-animal dataset in our evaluation.

Dataset sizes and sampling. Table 6 lists training-split sizes. Batches are drawn from the concatenated corpus (proportional sampling, no reweighting). The corpus is imbalanced by $5.4\times$ (largest to smallest), not by orders of magnitude; for substantially more heterogeneous corpora we expect temperature-based sampling to become necessary.

Table 6: Training-split sizes (frames) and group sizes.

Dataset	Training frames	N
AdaptableSnail	641,842	3–4
CalMS21	507,738	2
MouseLockbox	427,380	1
NiftyGoldfinch	284,590	2
CRIM13	223,362	2
MABe22	118,560	3

E Linear Probe Details

For all probing experiments ("linear probing" in the main text, following common usage for frozen-encoder evaluation), frozen embeddings are passed to a single-hidden-layer MLP with 256 ReLU units. Training uses Adam with weighted cross-entropy (inverse-frequency class weights), batch size 256, learning rate 1×10^{-3}, and 200 epochs with early stopping on macro-F1 over a held-out validation split. We report macro-F1 on the official test splits where available, otherwise on a 70/15/15 train/val/test split sampled at the recording level. Each result is the mean over 5 random seeds.

Embedding variants. We evaluate six embedding variants of MICE: `ind_mean`, `ind_cat`, `soc_mean`, `soc_cat`, `ind+soc_mean`, `ind+soc_cat`. The prefix (`ind`, `soc`, or `ind+soc`) says which output is used: the individual encoder's, the social encoder's, or both concatenated. The suffix says how the T=64 tokens are combined: `mean` averages them into one 96- or 192-dim vector, and `cat` flattens them into a $T{\cdot}96$- or $T{\cdot}192$-dim vector. This is done per animal, and the vectors of all animals in the clip are then concatenated. The headline MICE result in Table 1 uses `ind+soc_cat`, and `ind_cat` and `soc_cat` appear in the same table as ablations; all six variants, including the mean-pooled ones, are compared on CalMS21 in Appendix I.2. Table 7 summarizes them.

Table 7: MICE embedding readouts. $\mathbf{H}_{\text{ind}}$: concatenated individual-encoder output (Eq. 2); $\mathbf{O}$: per-animal social output (Eq. 6); T=64 tokens. Each readout concatenates the embeddings of all animals in the clip, so `ind_cat` already contains the partners' individual embeddings.

Readout	Source	Pooling over T tokens	Dim. per animal
`ind_mean` / `ind_cat`	$\mathbf{H}_{\text{ind}} \in \mathbb{R}^{T\times96}$	mean-pool / flatten	96 / $T{\cdot}96$
`soc_mean` / `soc_cat`	$\mathbf{O} \in \mathbb{R}^{T\times96}$	mean-pool / flatten	96 / $T{\cdot}96$
`ind+soc_mean` / `ind+soc_cat`	$[\mathbf{H}_{\text{ind}}; \mathbf{O}] \in \mathbb{R}^{T\times192}$	mean-pool / flatten	192 / $T{\cdot}192$

The social readout also contains individual information. The Decode step (Eq. 6) adds the focal mouse's own projected embedding back in, so `soc` is not purely social. This is why `soc_cat` alone

is usually only modestly better than `ind_cat`, and why `ind+soc_cat`, which also keeps an unmixed copy of the individual embedding, does best.

F Unsupervised Segmentation: Protocol and Qualitative Details

Protocol. For each method, we extract per-window embeddings on the test sequences with the most balanced mix of behavior labels. We run K-means with $K{=}50$ on the embeddings to produce hard cluster assignments. Consecutive frames with the same cluster assignment form a segment; each segment is mapped to the ground-truth class that appears most often in its frames (majority vote). ARI and NMI are computed at the frame level between predicted and ground-truth class labels. Boundary-F1 treats a predicted boundary as correct if it lies within 2 windows of any ground-truth boundary. All results are averaged over 5 seeds varying K-means initialization and the train/val/test split.

Qualitative observations from Figure 2. The figure shows two test sequences from CalMS21. In the left panel (investigation-rich), ground truth contains a long sustained investigation block at windows $\sim$240–360. MICE recovers most of it as two long segments separated by a short gap, with outer boundaries close to ground truth, while Progressive fragments the same block into several shorter segments. In the right panel (attack-rich), ground truth contains several attack clusters near the start (windows $\sim$10–95), one near window 190, and one near the end. MICE marks attack in both regions, whereas BAMS and hBehaveMAE miss most of the early clusters. Progressive and Multiscale also predict attack in windows where ground truth has none (e.g., $\sim$120–180 for Progressive), consistent with their tendency toward false positives.

Per-dataset results. Table 8 breaks the averages of Table 2 down by dataset. MICE has the highest ARI and NMI on every dataset; on Boundary-F1 it leads on CalMS21, while hBehaveMAE leads on CRIM13 and NiftyGoldfinch and Progressive on AdaptableSnail.

Table 8: Per-dataset unsupervised segmentation results (%, mean over 5 random seeds). The Avg. column is the average reported in Table 2.

Model	CalMS21	CRIM13	AdaptSnail	Goldfinch	Avg.
ARI					
PointNet	57.90	51.49	10.74	14.77	33.73
TBERT	42.18	39.19	9.51	7.48	24.59
BAMS	16.54	13.81	5.04	1.71	9.28
hBehaveMAE	51.19	50.15	12.21	11.72	31.32
Progressive (ours, ablation)	11.78	13.76	8.85	2.92	9.33
Multiscale (ours, ablation)	40.39	43.32	11.32	9.48	26.13
MICE (`ind+soc_cat`)	72.31	58.59	19.82	22.60	43.33
NMI					
PointNet	44.24	40.02	10.68	15.07	27.50
TBERT	29.60	27.61	9.44	7.92	18.64
BAMS	11.78	10.83	6.22	3.27	8.03
hBehaveMAE	38.43	38.52	12.81	12.00	25.44
Progressive (ours, ablation)	9.14	10.59	10.03	2.76	8.13
Multiscale (ours, ablation)	32.61	31.35	12.54	9.12	21.41
MICE (`ind+soc_cat`)	58.64	49.45	14.92	19.51	35.63
Boundary-F1					
PointNet	31.98	27.83	9.26	35.70	26.19
TBERT	27.48	32.37	16.35	40.04	29.06
BAMS	15.96	16.77	16.11	16.95	16.45
hBehaveMAE	33.82	37.08	8.29	46.30	31.37
Progressive (ours, ablation)	17.43	19.69	16.67	41.94	23.93
Multiscale (ours, ablation)	28.72	34.23	13.93	44.43	30.33
MICE (`ind+soc_cat`)	40.69	34.97	14.20	36.42	31.57

Comparison with Keypoint-MoSeq. Keypoint-MoSeq (KPMS) [Weinreb et al., 2024] is a widely used unsupervised method that segments pose into behavioral "syllables". We run it as the package

intends for multi-animal data: each mouse is passed in as its own pose sequence with the same skeleton, and one model is fit over all of them, so KPMS learns a single syllable vocabulary and returns one syllable sequence per mouse. (Stacking the mice into one pose would place the body center between them, so the syllables would describe the pair's configuration rather than either animal.) We use two protocols. In *train-fit*, one model is fit on all training recordings of a dataset and applied to each test recording, giving KPMS the same train/test separation as MICE. In *per test recording*, a separate model is fit on each test recording; this is the easier setting for KPMS, since nothing has to transfer. Scoring follows our segmentation protocol: syllables are pooled into the same windows by majority vote, each syllable is mapped to the label it most often coincides with, and we report ARI, NMI and Boundary-F1 on CalMS21, CRIM13 and AdaptableSnail. Because KPMS returns one sequence per animal while the labels describe the whole scene, we score every animal and report the best one per recording, the most generous reading for KPMS (Table 9). KPMS has no term that couples animals: it assigns syllables from one egocentrically aligned pose at a time, whereas most labels in these datasets (e.g., attack, mount, sniff, approach) involve two animals. MICE leads in all nine cells. Fitting KPMS on each test recording barely changes its ARI and NMI, so the gap is not due to transfer from training to test data; it is widest on CRIM13, the dataset with the most interaction-defined labels, where KPMS's ARI and NMI are close to zero.

Table 9: Unsupervised segmentation: Keypoint-MoSeq (KPMS) vs. MICE (`ind+soc_cat`). Each cell gives ARI / NMI / Boundary-F1 (%).

Dataset	KPMS (train-fit)	KPMS (per test recording)	MICE
CalMS21	16.50 / 16.34 / 32.82	17.14 / 17.11 / 28.16	72.31 / 58.64 / 40.69
CRIM13	0.75 / 0.91 / 0.27	0.74 / 1.45 / 1.95	58.59 / 49.45 / 34.97
AdaptableSnail	3.21 / 3.17 / 11.58	3.25 / 3.23 / 13.15	19.82 / 14.92 / 14.20

G Baseline Implementation Details

All baselines share the universal keypoint mapping (§3.1) and raw-keypoint inputs, and feed their frozen embeddings to the same probe described in Appendix E.

BAMS [Azabou et al., 2023b]. Dual-branch dilated Conv1d (short-term + long-term) trained per-mouse with HoA prediction over a $K \times 4$ velocity/shape feature target; mean-pooled across animals at inference. As in MICE's individual encoder, BAMS receives only raw keypoints as input, and the engineered features are used only as HoA prediction targets. This matches MICE's input and prevents the model from simply copying its targets from the input. Note that BAMS's long-term branch looks back about 1,365 frames, far longer than our 64-frame clips, so most of its input here is padding; its results should not be taken as representative of that branch.

PointNet [Qi et al., 2017]. Shared per-animal MLP with global max-pooling across the animal dimension yielding a permutation-invariant frame embedding; pretrained with temporal contrastive learning (positives = adjacent frames, negatives = different sequences).

TBERT [Sun et al., 2023]. BERT-style transformer over flattened multi-animal keypoints (fixed animal ordering), pretrained with masked-frame modeling ($\sim$40% frame masking, MSE reconstruction).

hBehaveMAE [Stoffl et al., 2024]. Hierarchical masked autoencoder built on Hiera with $(2, 1, n_{\text{kp}} \times 2)$ patching, two pooling stages ($4\times$ temporal then animal-dimension pooling), three transformer stages (9 blocks total, embed dims 64/128/256), and multi-scale Conv3d fusion.

Progressive (ours, ablation). The same architecture as MICE's individual encoder, but applied to all animals stacked into one multi-animal input with fixed $N_{\max}{=}4$ padding, and with no social stage; the multi-animal input makes it larger than MICE's per-animal individual encoder (Table 21). It isolates the contribution of per-animal encoding with a variable-N social stage.

Multiscale (ours, ablation). A single-transformer variant with simultaneous multi-resolution HoA heads at intermediate layers and no widening attention windows, in contrast to Progressive's staged hierarchy. It isolates MICE's staged, widening-window design from a flat multi-resolution alternative.

H Layer-wise Probing: Probe Locations

For the layer-wise probing analysis (§4.5), we probe MICE at 16 sequential depth points: 12 within the individual encoder and 4 within the social encoder. The mapping from probe index to architectural stage is:

Idx	Component	Operation
1–4	Individual: Stage 1	ALBERT layer iterations 1–4 (shared weights)
5–8	Individual: Stage 2	ALBERT layer iterations 1–4 (shared weights)
9–12	Individual: Stage 3	ALBERT layer iterations 1–4 (shared weights)
13	Social: Encode	After cross-attention with N animals + LayerNorm
14	Social: Process 1	After first latent self-attention layer
15	Social: Process 2	After second latent self-attention layer
16	Social: Temporal	After temporal attention across time tokens

To read out probe points 13–16, we pass the current latent state through the social encoder's Decode cross-attention, including its residual connection (§3.3). Each social probe therefore gives the per-animal output the encoder would produce if it stopped at that point. The input projection $\mathbf{Z}_{n,t}$ is not probed separately.

Baselines are probed at architecturally analogous points: BAMS's short and long branches are each probed at their successive layers and plotted as two separate curves; hBehaveMAE is probed at each of its hierarchical stage outputs (9 blocks across 3 stages); TBERT and PointNet are probed at all transformer/MLP layers respectively. To plot all methods on a common x-axis, each method's depth points are normalized to $\{0\%, \ldots, 100\%\}$ by total architectural depth.

I Ablations: Tables and Per-Ablation Interpretation

This section contains two ablation studies: a variable-N reorganization of the combined-regime linear-probe results, and an embedding-variant comparison on CalMS21.

I.1 Variable-N Analysis

Table 10 reorganizes Table 1's combined-regime results by per-dataset N, with AdaptableSnail stratified by group size so each N column reflects only recordings at that size. MICE wins every column; its lead over the baselines is negligible at $N{=}1$, modest at $N{=}2$, and substantial at $N{=}3$ and $N{=}4$. The most informative comparison is hBehaveMAE's reasonable $N{=}3$ performance on MABe22 versus its collapse on AdaptableSnail's $N{=}4$ recordings: the same architecture that handles fixed $N{=}3$ fails when N varies within recordings. The MICE-family ablations sharpen the picture: the Progressive-vs-MICE gap is largest at $N{=}2$, where Progressive's fixed $N_{\max}{=}4$ padding leaves half the slots empty, and Multiscale performs comparably to Progressive overall. Because MICE's individual readout alone (`ind_cat`) already outperforms both ablations in almost every column of Table 1, most of MICE's advantage comes from encoding each animal separately, with the social stage adding a further gain.

Progressive details. Progressive has the same architecture as MICE's individual encoder but takes all animals stacked into one input tensor with $N_{\max}{=}4$ padding, and has no social stage. The Progressive-vs-MICE gap is largest at $N{=}2$ (18.1 points), where Progressive's fixed padding wastes capacity on padded slots ($N{=}2$ on a $N_{\max}{=}4$ tensor), and about 6 points at $N{=}3$ and $N{=}4$.

Multiscale details. Multiscale is a variant of the individual encoder that supervises several temporal resolutions at once, without a staged hierarchy. It performs comparably to Progressive on most N columns. Since both ablations trail MICE's individual readout alone in almost every column, the

Table 10: Macro-F1 (%) under combined pretraining, organized by per-dataset animal count. Mean ± standard deviation over 5 random seeds. Datasets per column: $N{=}1$: MouseLockbox; $N{=}2$: CalMS21, CRIM13, NiftyGoldfinch; $N{=}3$: MABe22 + AdaptableSnail-$N{=}3$; $N{=}4$: AdaptableSnail-$N{=}4$. For MICE, $N{=}1$ uses `ind_cat` (social variants not applicable). **Bold** = best per column.

	Macro-F1 (%) at dataset N				Overall
Model	$N{=}1$	$N{=}2$	$N{=}3$	$N{=}4$	avg
TBERT [Sun et al., 2023]	$13.55_{\pm 0.41}$	$30.78_{\pm 0.58}$	$23.62_{\pm 0.62}$	$12.90_{\pm 0.69}$	$20.21_{\pm 0.45}$
PointNet [Qi et al., 2017]	$13.42_{\pm 0.39}$	$37.34_{\pm 0.51}$	$37.40_{\pm 0.55}$	$18.50_{\pm 0.62}$	$26.67_{\pm 0.41}$
BAMS [Azabou et al., 2023b]	$12.92_{\pm 0.40}$	$35.37_{\pm 0.54}$	$41.70_{\pm 0.52}$	$17.90_{\pm 0.65}$	$26.97_{\pm 0.43}$
hBehaveMAE [Stoffl et al., 2024]	$13.21_{\pm 0.33}$	$49.08_{\pm 0.45}$	$41.98_{\pm 0.49}$	$11.40_{\pm 0.58}$	$28.92_{\pm 0.38}$
Progressive (ours, ablation)	$13.54_{\pm 0.39}$	$36.16_{\pm 0.49}$	$55.72_{\pm 0.45}$	$22.30_{\pm 0.61}$	$31.93_{\pm 0.40}$
Multiscale (ours, ablation)	$13.04_{\pm 0.38}$	$40.34_{\pm 0.46}$	$56.72_{\pm 0.43}$	$19.85_{\pm 0.59}$	$32.49_{\pm 0.39}$
MICE	$\mathbf{13.68}_{\pm 0.37}$	$\mathbf{54.28}_{\pm 0.39}$	$\mathbf{61.87}_{\pm 0.37}$	$\mathbf{28.50}_{\pm 0.51}$	$\mathbf{39.58}_{\pm 0.32}$

bulk of MICE's advantage comes from encoding each animal separately rather than from the staged hierarchy, with the social stage adding a further gain.

Per-animal pooling baselines. BAMS and PointNet handle variable N trivially via per-animal pooling but discard interaction structure entirely, reflected in scores that lag MICE by 10–25 points across $N{\geq}2$.

Single-dataset pretraining by group size. Table 11 repeats this analysis with single-dataset pretraining, using the same grouping by N. The benefit of combined pretraining grows with the number of animals and is negative only at $N{=}1$, i.e., on MouseLockbox, where there is no social behavior for the other datasets to inform. Combined pretraining therefore helps most when the target involves the most social behavior.

Table 11: MICE macro-F1 (%) by animal count under single-dataset vs. combined pretraining, grouped as in Table 10. Δ = combined − single.

N	Single-dataset	Combined	Δ
1	15.93	13.68	-2.25
2	52.99	54.28	$+1.29$
3	56.77	61.87	$+5.10$
4	20.38	28.50	$+8.12$

Foundation-model framing. The 10.7-point gap on the overall average between MICE (39.58) and the strongest baseline (hBehaveMAE, 28.92) suggests that the benefits of handling variable N and of jointly modeling individual and social behavior add up across diverse datasets. This is exactly the setting a foundation model is meant for: serving multi-animal datasets with different group sizes from a single checkpoint without retraining.

I.2 Embedding Variant Analysis

Table 12 compares six embedding variants on CalMS21. Two patterns emerge. First, concatenation consistently beats mean-pooling, most of all for the individual channel, where averaging discards the temporal structure the hierarchical encoder is designed to capture. Second, combining the individual and social channels gives a small but consistent gain over either alone, indicating that they carry complementary information.

Embedding-variant details. The largest mean-vs-cat gap is on the individual channel (`ind_mean`: 57.79 vs. `ind_cat`: 81.22, a 23-point difference), where mean-pooling collapses the temporal structure most severely. The smaller gaps for the social and combined channels (12 and 11 points respectively) suggest that the social-stage representations are somewhat more robust to mean-pooling than the individual-stage representations, possibly because the social stage's temporal attention already

Table 12: Embedding-variant comparison on CalMS21 (combined-regime macro-F1, %). Mean ± standard deviation over 5 random seeds. Channel: `ind` / `soc` / `ind+soc` = individual / social / both concatenated. Pooling: `mean` / `cat` = average / concatenation over T. **Bold** = best.

Variant	`ind_mean`	`ind_cat`	`soc_mean`	`soc_cat`	`ind+soc_mean`	`ind+soc_cat`
F1 (%)	$57.79_{\pm 0.42}$	$81.22_{\pm 0.28}$	$70.08_{\pm 0.36}$	$81.86_{\pm 0.28}$	$73.38_{\pm 0.34}$	$\mathbf{84.56}_{\pm 0.31}$

aggregates information over time before pooling. The lift from `soc_cat` (81.86) to `ind+soc_cat` (84.56) is 2.7 points; `ind_cat` alone is 81.22, so the ind+soc combination beats either channel alone but not by much. When embedding size is a constraint, `soc_cat` alone is a reasonable choice at a cost of 2.7 points.

J Additional Analyses

These analyses complement §4.6–4.7.

J.1 Retraining the social stage with different partner information

Design. All runs start from the same frozen individual encoder. Because this encoder sees one animal at a time, it carries no partner information, and the individual readout (`ind_cat`) is the same in every run. The social stage is then retrained from scratch for 500 epochs under one of three conditions:

- *Correct partners*: the default setting.
- *Mis-paired partners*: each partner is replaced by the mouse in the same slot of another clip in the batch. Batches are grouped by animal count, so partners are real mice and only the pairing is wrong.
- *No partners*: partner slots are set to zero.

Only the social encoder's input is changed. All runs are compared at epoch 500. For each run, we train a fresh probe and test it under the same partner condition used in training, so no model sees unfamiliar input. With mis-paired partners, the pairwise loss remains meaningful, because its targets are computed from a real (if wrong) mouse. With no partners, these targets become trivial, so that run shows what happens without any partner information.

Direct animal-to-animal attention. To test what the latent codebook adds over the simplest alternative, we replace the Perceiver latents with direct attention between animals, keeping correct partners and everything else fixed (Table 13). Direct attention scores slightly lower than the Perceiver on all five datasets, by 0.7 to 1.8 points, so at these group sizes the latent codebook is at least as effective as the simplest alternative while also keeping cost independent of N.

Table 13: Perceiver latents vs. direct animal-to-animal attention in the social stage: macro-F1 (%, `ind+soc_cat`). Mean ± standard deviation over 5 random probe seeds. The Perceiver row is the combined-regime MICE result of Table 1. **Bold** = best per column.

Social stage	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch
Perceiver latents (MICE)	$\mathbf{69.26}_{\pm 0.34}$	$\mathbf{84.56}_{\pm 0.31}$	$\mathbf{46.38}_{\pm 0.55}$	$\mathbf{26.76}_{\pm 0.78}$	$\mathbf{31.90}_{\pm 0.72}$
direct animal-to-animal attention	$68.13_{\pm 0.42}$	$83.85_{\pm 0.40}$	$45.11_{\pm 0.57}$	$24.98_{\pm 0.85}$	$30.92_{\pm 0.81}$

J.2 Weighting the two individual-stage losses

Raising the reconstruction weight from 0.1 to 1 improves both datasets (CalMS21 76.46 to 81.22; CRIM13 36.01 to 44.71; Table 14), so we use equal weights. Removing the HoA loss (w_{HoA}=0) costs 10.7 F1 on CRIM13 and 2.4 on CalMS21.

Table 14: Individual-stage loss weights (macro-F1, %, individual-only readout). w_{HoA} and w_{rec} weight the HoA and reconstruction losses; all models are pretrained on all six datasets, and the first row is the combined-regime `ind_cat` result of Table 1. **Bold** = best per column.

w_{HoA}	w_{rec}	CalMS21	CRIM13
1	1.00	**81.22**	**44.71**
1	0.30	78.49	37.97
1	0.10	76.46	36.01
1	0.03	73.78	33.29
1	0.00	74.10	33.92
0	1.00	78.84	34.02

J.3 What the social stage encodes

These analyses use the model pretrained on all six datasets, evaluated on the CalMS21 test set.

Relative position. We fit linear models that predict simple measurements of the two mice from each frozen embedding (Table 15). The social embedding predicts how the mice are positioned relative to each other far better than the individual embedding, and is no better at describing the focal mouse alone.

Table 15: How well a linear model predicts each measurement from the individual vs. social embedding (R^2; higher is better).

Measurement	individual	social	difference
Distance between the mice	0.358	0.975	$+0.617$
Relative body orientation	0.145	0.644	$+0.499$
Approach speed	-0.221	0.018	$+0.239$
Partner speed	-0.059	-0.032	$+0.027$
Focal mouse speed	0.702	0.638	-0.064
Focal mouse body spread	0.975	0.953	-0.022

J.4 Per-latent lift analysis

We also examined whether individual latents respond selectively to particular behaviors.

Method. For every token in the test set, we record the $M{=}16$ latents produced by the social encoder's Encode step, that is, after the latents have attended to the animals. For each latent, we rank all tokens by how strongly the latent is activated, take the top 100, and compute the class distribution among them. Dividing by the class prior gives a lift score, $P(c \mid \text{top 100 of latent } m)/P(c)$. A lift of 1 means the latent is active on that class no more often than chance; a lift of 10 means tokens where the latent is most active are ten times more likely to carry that label than a random token. No labels are used during pretraining; they are used only to read the latents afterwards.

Table 16: Most-enriched class per selected latent. Left: CalMS21 (priors: attack 4.97%, investigation 23.4%, mount 12.1%, other 59.5%). Right: CRIM13 (other prior 70.7% among the analyzed tokens).

CalMS21				
Latent	Class	Lift	$P(c \mid \text{top})$	$P(\text{other} \mid \text{top})$
3	attack	10.86	54%	34%
12	attack	7.04	35%	49%
14	mount	3.05	37%	24%
4	investigation	1.75	41%	59%

CRIM13				
Latent	Class	Lift	$P(c \mid \text{top})$	$P(\text{other} \mid \text{top})$
15	disengage	13.14	53%	36%
12	rear	9.85	18%	70%
9	attack	9.67	25%	51%
1	attack	7.35	19%	50%
7	disengage	6.69	27%	56%
0	approach	5.29	19%	76%

Results. On CalMS21, four of the sixteen latents are most enriched for attack, two of them strongly. Latent 3 is the clearest case: attack makes up 4.97% of tokens overall but 54% of the hundred tokens where latent 3 is most active. Eight latents are most enriched for other, at low lift (1.26–1.68). CRIM13 shows the same structure more sharply, with higher lift because its social classes are rarer: five latents are most enriched for attack and five for disengage, both interaction-defined classes. For CalMS21 latent 3 (attack, 54% vs. 34% other) and CRIM13 latent 15 (disengage, 53% vs. 36%), the named class is the outright majority of the top 100, and for CalMS21 latent 14 (mount, 37% vs. 24% other) it exceeds the background class.

Caveats. Lift cannot exceed $1/P(c)$, so a common class can never reach a high lift: on CRIM13, the maximum lift is 1.41 for other (prior 70.7%) but 38.5 for attack (prior 2.6%). A high lift also does not mean the class dominates: other is still the largest class in the top 100 for twelve of sixteen CRIM13 latents. These latents therefore respond selectively to their class relative to how common it is, rather than responding only to it.

J.5 Cross-dataset transfer for baselines

Table 3 shows how MICE transfers across datasets but gives no reference for whether that behavior is specific to MICE. Tables 17 and 18 repeat the same protocol for hBehaveMAE, the strongest baseline, and PointNet: each model is pretrained on one dataset (rows) and probed on every dataset (columns). Both baselines show the patterns we report for MICE. Within each column, pretraining on the target itself is the best single-dataset choice in 5 of 6 columns for hBehaveMAE and in all 6 for PointNet. Combined pretraining beats every single-dataset checkpoint on 5 of 6 targets for hBehaveMAE and 4 of 6 for PointNet, with MouseLockbox an exception for both. Pretraining on MouseLockbox gives the lowest off-diagonal score in 4 of the 5 multi-animal columns for hBehaveMAE and 2 of 5 for PointNet. Since these patterns appear across architectures that share only the data, they reflect the datasets rather than MICE.

Table 17: Cross-dataset transfer for hBehaveMAE: macro-F1 (%) when pretrained on the row dataset and probed on the column dataset. Mean ± standard deviation over 5 random seeds. Diagonal cells and the bottom row are hBehaveMAE's single- and combined-regime results in Table 1.

Pretrain \ Probe	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch	Lockbox
MABe22	$42.72_{\pm 0.62}$	$69.03_{\pm 0.49}$	$35.22_{\pm 0.72}$	$8.53_{\pm 0.64}$	$20.36_{\pm 0.77}$	$12.48_{\pm 0.56}$
CalMS21	$35.57_{\pm 0.74}$	$76.05_{\pm 0.42}$	$36.75_{\pm 0.78}$	$8.11_{\pm 0.65}$	$18.80_{\pm 0.75}$	$13.67_{\pm 0.50}$
CRIM13	$32.93_{\pm 0.68}$	$72.30_{\pm 0.44}$	$38.30_{\pm 0.68}$	$9.49_{\pm 0.70}$	$19.20_{\pm 0.68}$	$12.91_{\pm 0.51}$
AdaptSnail	$34.26_{\pm 0.81}$	$71.81_{\pm 0.58}$	$26.91_{\pm 0.83}$	$9.03_{\pm 0.53}$	$18.05_{\pm 0.71}$	$12.63_{\pm 0.63}$
Goldfinch	$30.49_{\pm 0.66}$	$72.54_{\pm 0.49}$	$35.86_{\pm 0.76}$	$8.66_{\pm 0.65}$	$26.02_{\pm 0.59}$	$14.35_{\pm 0.54}$
Lockbox	$27.04_{\pm 0.73}$	$64.33_{\pm 0.51}$	$27.85_{\pm 0.80}$	$7.73_{\pm 0.72}$	$12.07_{\pm 0.69}$	$14.58_{\pm 0.44}$
Combined	$47.82_{\pm 0.52}$	$77.68_{\pm 0.34}$	$40.35_{\pm 0.58}$	$12.13_{\pm 0.53}$	$29.21_{\pm 0.50}$	$13.21_{\pm 0.33}$

Table 18: Cross-dataset transfer for PointNet, in the same format as Table 17. Diagonal cells and the bottom row are PointNet's single- and combined-regime results in Table 1.

Pretrain \ Probe	MABe22	CalMS21	CRIM13	AdaptSnail	Goldfinch	Lockbox
MABe22	$36.54_{\pm 0.71}$	$58.36_{\pm 0.63}$	$18.51_{\pm 0.76}$	$7.59_{\pm 0.55}$	$20.74_{\pm 0.63}$	$12.90_{\pm 0.52}$
CalMS21	$30.25_{\pm 0.75}$	$62.24_{\pm 0.54}$	$17.93_{\pm 0.64}$	$7.20_{\pm 0.46}$	$19.10_{\pm 0.68}$	$12.70_{\pm 0.49}$
CRIM13	$32.83_{\pm 0.68}$	$56.13_{\pm 0.67}$	$21.48_{\pm 0.65}$	$6.72_{\pm 0.57}$	$19.52_{\pm 0.55}$	$13.64_{\pm 0.50}$
AdaptSnail	$32.76_{\pm 0.72}$	$61.50_{\pm 0.59}$	$19.94_{\pm 0.69}$	$7.91_{\pm 0.48}$	$20.09_{\pm 0.62}$	$13.00_{\pm 0.42}$
Goldfinch	$28.38_{\pm 0.71}$	$61.24_{\pm 0.59}$	$20.48_{\pm 0.70}$	$6.03_{\pm 0.61}$	$21.73_{\pm 0.60}$	$12.20_{\pm 0.56}$
Lockbox	$26.59_{\pm 0.80}$	$61.18_{\pm 0.65}$	$18.93_{\pm 0.72}$	$7.37_{\pm 0.53}$	$17.06_{\pm 0.62}$	$13.73_{\pm 0.42}$
Combined	$40.75_{\pm 0.64}$	$65.93_{\pm 0.42}$	$26.49_{\pm 0.57}$	$19.59_{\pm 0.58}$	$19.61_{\pm 0.52}$	$13.42_{\pm 0.41}$

J.6 Comparison with a vanilla transformer

To test the individual encoder's design as a whole, we replace it with a vanilla variant: a single stage of 12 layers with full attention in every layer and a single readout at matched width, trained with

the same setup as MICE. This removes both the widening attention windows and the staged readout. Because only the individual encoder differs, we compare the individual readout (`ind_cat`), with MouseLockbox included (Table 19). MICE's individual encoder outperforms the vanilla transformer on all six datasets, by 0.6 to 5.4 points, with the largest gaps on CRIM13 and NiftyGoldfinch, while also using fewer attention operations than full attention.

Table 19: MICE's individual encoder vs. a vanilla transformer (single stage, full attention): macro-F1 (%, `ind_cat`) under combined pretraining. Mean ± standard deviation over 5 random seeds. MICE values are from Table 1. Gap = MICE − vanilla. **Bold** = best per row.

Dataset	Vanilla transformer	MICE	Gap
MABe22	$65.32_{\pm 0.34}$	$\mathbf{67.85}_{\pm 0.42}$	$+2.53$
CalMS21	$80.16_{\pm 0.42}$	$\mathbf{81.22}_{\pm 0.28}$	$+1.06$
CRIM13	$39.33_{\pm 0.67}$	$\mathbf{44.71}_{\pm 0.47}$	$+5.38$
AdaptableSnail	$21.30_{\pm 0.88}$	$\mathbf{22.66}_{\pm 0.53}$	$+1.36$
NiftyGoldfinch	$23.23_{\pm 0.82}$	$\mathbf{29.12}_{\pm 0.48}$	$+5.89$
MouseLockbox	$13.10_{\pm 0.52}$	$\mathbf{13.68}_{\pm 0.37}$	$+0.58$

J.7 Supervised reference

As a reference point rather than a competitor, we take MICE's individual encoder with an identical configuration, initialize it randomly without self-supervised pretraining, attach a classification head, and train both end-to-end with cross-entropy on each dataset's behavior labels (Table 20). Unlike every method in Table 1, this model uses labels to train the encoder itself. Supervised training is ahead on every dataset, as expected for a model that sees labels during training. The gap is widest on AdaptableSnail, where MICE's scores are weakest among the multi-animal datasets, and narrowest on CalMS21, where they are strongest.

Table 20: Supervised end-to-end training of MICE's individual encoder vs. self-supervised MICE (single-dataset regime, `ind+soc_cat`; `ind_cat` for MouseLockbox; Table 3 diagonal): macro-F1 (%). Mean ± standard deviation over 5 random seeds. Gap = supervised − MICE.

Dataset	Supervised (end-to-end)	MICE	Gap
MABe22	$71.86_{\pm 0.62}$	$67.60_{\pm 0.45}$	$+4.26$
CalMS21	$85.86_{\pm 0.78}$	$83.28_{\pm 0.39}$	$+2.58$
CRIM13	$48.22_{\pm 0.74}$	$44.61_{\pm 0.72}$	$+3.61$
AdaptableSnail	$29.87_{\pm 0.75}$	$21.93_{\pm 1.04}$	$+7.94$
NiftyGoldfinch	$35.71_{\pm 0.66}$	$31.08_{\pm 0.74}$	$+4.63$
MouseLockbox	$18.59_{\pm 0.63}$	$15.93_{\pm 0.43}$	$+2.66$

J.8 Model sizes

Table 21: Encoder parameters used at inference (training-only heads excluded). MICE: 159,712 individual + 423,552 social = 583,264; with training-only heads, 772,053.

Model	Parameters	vs. MICE
PointNet	28K	$0.05\times$
MICE	**583K**	$1\times$
Multiscale (ours, ablation)	620K	$1.06\times$
Progressive (ours, ablation)	687K	$1.18\times$
TBERT	2.132M	$3.66\times$
BAMS	2.240M	$3.84\times$
hBehaveMAE	4.061M	$6.96\times$

MICE is small mainly because layers within each stage share weights (ALBERT-style).

K Training Hyperparameters

Architecture. *Individual encoder*: patch size w=3, stride s=1; per-stage hidden dim D=32; three stages each with L_g=4 ALBERT-style layers and h=8 attention heads; concatenated output dim D_{ind}=96; attention windows follow the linear schedule of §3.2 from ω_{short}=3 to ω_{long}=T. *Social encoder*: d_{model}=96, learned latent codebook size M=16, social transformer layers L_{soc}=2, attention heads h=8. *Pretraining heads*: HoA bins B_{HoA}=32 for both individual and pairwise heads. Sequence length T=64.

Span masking. *Individual encoder*: 25% of frames per sequence are masked in contiguous spans of length sampled uniformly from $[4, 12]$ frames (zero-fill in input keypoint space). *Social encoder (Masked Animal Modeling)*: 25% of T temporal tokens per animal are masked in contiguous spans of length sampled uniformly from $[2, 6]$ tokens, replaced by a learned mask token $\mathbf{m} \in \mathbb{R}^{d_{\text{model}}}$. Masking is applied only during training.

Loss weights. The individual objective (Eq. 3) uses unit weights; equal weights worked best among those we tried (Appendix J.2). The social objective also uses unit weights.

Optimization. Adam optimizer with learning rate 1×10^{-4}, weight decay 1×10^{-4}, batch size 256. Mixed-precision (fp16) training with gradient scaling. Pretraining: 500 individual-stage epochs followed by 500 social-stage epochs with the individual encoder frozen. Each individual-stage epoch uses a random 5% subsample of the pretraining corpus. Hardware: a single NVIDIA A40 GPU (48GB VRAM) per run; we ran many experiments in parallel rather than using multiple GPUs per run. The released code also includes a multi-GPU (DistributedDataParallel) training path.

Training and inference time. On one NVIDIA A40, an individual-stage epoch takes about 149 s and a social-stage epoch at most about 3.5 min, so the 500 + 500-epoch schedule takes about 21 h + 29 h. Table 22 reports inference time for a batch of 32 clips of 64 frames, averaged over 200 timed iterations after 20 warm-up iterations. The individual encoder's cost does not depend on the number of animals, while the social stage adds about 1.6 ms per additional animal.

Table 22: Inference time on one NVIDIA A40 (batch of 32 clips, 64 frames each).

Animals N	Individual encoder (ms/batch)	Full model (ms/batch)	Social stage (ms/batch)	Full-model throughput (animal-sequences/s)
1	7.07	12.49	5.43	2,562
2	7.13	14.12	6.98	4,534
3	7.12	15.72	8.60	6,109
4	7.19	17.39	10.20	7,360

Tuning. We did not tune architectural or training hyperparameters: the first configuration we tried was competitive, and we spent our compute on the method itself. Further tuning may improve results. How model size should grow with the amount of pretraining data is, in our view, the most important open question.

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: The abstract and introduction state our four contributions (joint multi-dataset pretraining, hierarchical individual encoder with widening attention, Perceiver-style social encoder, comprehensive evaluation). These are directly supported by the experiments in §4.1–§4.7 and the architectural details in §3.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: Limitations are discussed in the Limitations paragraph of §5, covering manual synonym alignment, mouse-only scope, the modest pretraining-corpus size relative to vision/language foundation models, and the coarseness of available behavior labels.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: The paper presents an empirical contribution (a model architecture and pretraining recipe) and does not include formal theoretical results.

Guidelines:

- The answer [N/A] 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: Architecture, loss functions, and training procedures are fully specified in §3 and Appendices A–B. Self-supervision feature definitions are in Appendix C. Per-dataset specifications, probe-architecture details, segmentation protocol, baseline implementations, and hyperparameters are in Appendices D–K. Code and pretrained weights are available at `https://github.com/eric0708/MICE`.

Guidelines:

- The answer [N/A] 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: Code is provided as supplementary material, including pretraining and evaluation scripts. All datasets used are publicly available: MABe22, CalMS21, CRIM13, and MouseLockbox via their published releases, and AdaptableSnail and NiftyGoldfinch via the MABe 2025 challenge. Pretrained weights, the six reformatted datasets, and the full training and evaluation code are publicly available at `https://github.com/eric0708/MICE`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Pretraining hyperparameters (architecture, span masking, optimizer, learning rate, batch size) are in Appendix K. Linear-probe training details (probe architecture, optimizer, batch size, epochs, train/val/test splits, seeds) are in Appendix E. Baseline implementation details are in Appendix G.

   Guidelines:

   - The answer [N/A] 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: All probe results are averaged over 5 random seeds, with standard deviations reported alongside the means in the relevant tables. For probing, the 5 seeds vary the random initialization of the probe MLP; for segmentation, they vary the K-means initialization and the data split (Appendix F). Pretraining runs are not repeated across seeds. Probe-training procedure details are in Appendix E.

Guidelines:

- The answer [N/A] 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: Hardware and pretraining schedule (epochs) are in Appendix K.

   Guidelines:

   - The answer [N/A] 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: The work uses only public datasets of laboratory mouse pose recordings. No human subjects are involved. We have reviewed the NeurIPS Code of Ethics and find no deviations.

   Guidelines:

   - The answer [N/A] 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: [Yes]

    Justification: Broader impact is discussed in §5, covering positive applications in behavioral neuroscience, drug discovery, and animal welfare assessment, alongside the dual-use risk in industrial-livestock surveillance contexts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The released model is a keypoint-based behavior representation for laboratory mice, and does not pose meaningful misuse risk in the categories specified (no generative outputs, no human data, no scraped media).

    Guidelines:

    - The answer [N/A] 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: [Yes]

    Justification: All datasets used (MABe22, CalMS21, CRIM13, MouseLockbox, AdaptableSnail, NiftyGoldfinch) are properly cited in §4 and Appendix D. Each is used under its public release terms; the MABe 2025 challenge tracks (AdaptableSnail, NiftyGoldfinch) are used per challenge participation terms. Baseline code references are cited in Appendix G.

    Guidelines:

    - The answer [N/A] 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: The MICE model checkpoint is released at `https://github.com/eric0708/MICE` under CC BY 4.0 with documentation including architecture configuration, training procedure, and intended-use guidelines.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects; behavior labels in all datasets used were annotated by the original dataset publishers.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: This work does not involve human subjects. The animal-pose datasets used were collected by the original dataset publishers, who are responsible for their own institutional animal-care approvals.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used as a component of the core methodology. Any use of LLMs in the project was limited to writing, editing, and formatting assistance, which does not require declaration per NeurIPS policy.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.