
Self-supervised pretraining of vision transformers for animal behavioral analysis and neural encoding

Yuchen Wang*
Columbia University
New York
yw4503@columbia.edu

Han Yu*
Columbia University
New York
hy2562@columbia.edu

Ari Blau
Columbia University
New York
bsb2144@columbia.edu

Yizi Zhang
Columbia University
New York
yz4123@columbia.edu

The International Brain Laboratory
The International Brain Laboratory

Liam Paninski
Columbia University
New York
lmp2107@columbia.edu

Cole Hurwitz
Columbia University
New York
ch3676@columbia.edu

Matthew R Whiteway
Columbia University
New York
m.whiteway@columbia.edu

Abstract

The brain can only be fully understood through the lens of the behavior it generates—a guiding principle in modern neuroscience research that nevertheless presents significant technical challenges. Many studies capture behavior with cameras, but video analysis approaches typically rely on specialized models requiring extensive labeled data. We address this limitation with BEAST (**B**Ehavioral Analysis via Self-supervised pretraining of **T**ransformers), a novel and scalable framework that pretrains experiment-specific vision transformers for diverse neuro-behavior analyses. BEAST combines masked autoencoding with temporal contrastive learning to effectively leverage unlabeled video data. Through comprehensive evaluation across multiple species, we demonstrate improved performance in three critical neuro-behavioral tasks: extracting behavioral features that correlate with neural activity, and pose estimation and action segmentation in both the single- and multi-animal settings. Our method establishes a powerful and versatile backbone model that accelerates behavioral analysis in scenarios where labeled data remains scarce.

1 Introduction

Understanding the relationship between brain and behavior is a fundamental challenge across a wide range of medical and scientific disciplines [25, 10, 41]. Precise methods for extracting meaningful information from behavioral videos are essential for advancing these fields [49]. Self-supervised learning has revolutionized image and video understanding through large-scale foundation models [9, 7, 53, 15], offering powerful tools that are beginning to transform scientific analyses [19, 31, 83]. However, these models have yet to be effectively translated to specialized domains like animal behavior analysis, creating a significant opportunity for methods that bridge cutting-edge machine learning with the specific demands of neuroscience and behavioral research.

Animal behavior videos present unique characteristics and challenges distinct from general video understanding. Controlled experiments generate large quantities of videos with static backgrounds and

*These authors contributed equally

consistent camera angles, where the primary variation arises from animal movements and interactions. Behavioral videos enable numerous downstream analyses, and here we focus on three fundamentally different applications that collectively address a large proportion of behavioral neuroscience use cases: (1) neural encoding, which requires extracting behavioral features that correlate with simultaneously recorded brain activity [10, 49, 66]; (2) pose estimation, which tracks specific anatomical landmarks for quantitative analysis of movement patterns [41, 49]; and (3) action segmentation, which classifies distinct behavioral states like grooming, rearing, or social interactions on every frame [10, 49]. Each task demands different representations of the same underlying behavioral data, yet current approaches typically require task-specific models and extensive labeled datasets.

The primary barrier to sophisticated behavioral analysis in most laboratories is not the conceptual challenge of creating experiment-specific datasets, but rather the practical constraints of scale and technical expertise [67]. Individual labs typically lack sufficient data volume to train high-quality models from scratch and face computational limitations that make large-scale model development infeasible [49, 67]. This creates substantial adoption barriers across the scientific community. While recent approaches attempt to reduce labeled data requirements [40, 48, 57, 60, 3, 14, 63] or eliminate them entirely through unsupervised methods [2, 18, 61, 39, 73, 68], each comes with significant limitations. Supervised methods still require tedious and potentially biased human annotation, while unsupervised approaches often require complex post-processing to yield interpretable results suitable for scientific inquiry. Both approaches are typically task-specific and therefore lack versatility.

Our work addresses these challenges through a novel self-supervised pretraining framework for vision transformers that serves as a robust backbone for multiple downstream tasks. BEAST (**B**Ehavioral **A**nalysis via **S**elf-supervised pretraining of **T**ransformers) leverages the unique properties of experimental videos by combining masked autoencoding [15] to capture rich frame-level appearance information with temporal contrastive learning [9] to model behavioral dynamics. We introduce a novel frame sampling strategy for the contrastive loss, specifically designed to focus on learning representations of animal behavior against static experimental backgrounds. BEAST trains on videos from a single experimental setup, creating tailored, versatile models that can be fine-tuned for multiple analytical needs specific to that experimental context. We demonstrate the value of this approach through comprehensive evaluation on three downstream tasks: (1) neural activity prediction (or “neural encoding”) in three mouse datasets; (2) pose estimation across four datasets spanning two species and single- and multi-camera setups; and (3) action segmentation in both single- and multi-animal experiments. BEAST achieves competitive or superior performance across these tasks while eliminating the need for pose estimation or other preprocessing steps typically required by existing neural encoding and action segmentation methods. These results establish BEAST as a simple yet powerful foundation that can accelerate behavioral understanding across disciplines where fine-grained analysis is essential. Code is available at <https://github.com/paninski-lab/beast>.

2 Related Work

Neural encoding models. Neural encoding measures how observable signals predict neural activity, and provide a quantitative framework for interrogating neural representations. Earlier approaches applied generalized linear models to single neurons using controlled stimuli such as visual or auditory inputs [46, 65, 50, 42]. More recently, deep learning methods have shown great promise in predicting neural population responses to sensory stimuli, including visual [76, 56, 69], auditory [23, 35], and tactile [84] inputs. The widespread adoption of video monitoring during experiments has demonstrated that video-based behavioral covariates explain significant neural variability in both spontaneous [58, 63] and task-driven behaviors [44, 29, 71, 8, 82]. For example, Musall et al. [44] showed that uninstructed movements explain a substantial fraction of cortical population variance, and the International Brain Lab’s Brainwide Map [29] leveraged large-scale, region-resolved encoding analyses to chart the distribution of task-related information across the brain. However, extracting rich spatiotemporal information from video remains challenging. Most studies rely on either a small set of keypoints [63, 29, 71, 8] or latent dimensions using PCA [58, 44] or autoencoders [1, 71, 8], with limited efforts to predict neural activity directly from raw video [71].

Large-scale models for behavioral video analysis. Large-scale models for animal behavior analysis have predominantly focused on single tasks. For pose estimation, methods differ in how they balance flexibility and labeling requirements. DeepLabCut [40] leverages ImageNet-pretrained backbones for fine-tuning on experiment-specific labeled datasets, offering flexibility but requiring more manual labeling. In contrast, several specialized pose estimation tools provide tailored solutions for common

experimental setups, such as top-down views of freely moving mice (SuperAnimal [77]) and facial analysis of head-fixed rodents (Facemap [63]), significantly reducing labeling requirements. Similarly, in action segmentation, specialized systems developed for resident-intruder assays [57, 14] achieve high performance but remain limited to a specific experimental paradigm. While VideoPrism [83] offers a general foundation model supporting multiple behavioral tasks [62], it relies on a frozen backbone trained on generic internet data rather than domain-specific content, and remains unavailable to the public. Despite these advances, a critical gap remains: there are currently no accessible solutions for creating general behavior analysis models that leverage abundant unlabeled data to improve performance across multiple tasks. Our work, BEAST, addresses this need by introducing a pretraining strategy that enables individual labs to develop experiment-specific models supporting diverse downstream tasks using their own unlabeled videos. This approach allows researchers to extract maximum value from their experimental data without extensive manual labeling.

Self-supervised learning for images and videos. Contrastive learning has emerged as a powerful self-supervised representation learning framework; among its many predecessors and variants, SimCLR [9] popularized a simple and effective recipe that maximizes agreement between differently augmented views of the same sample via a contrastive loss in latent space. The contrastive method has also been extended to the video domain [52, 54]. Masked modeling frameworks are a complementary approach that have demonstrated remarkable success, particularly masked autoencoding (MAE) [15], which revolutionized visual self-supervised learning by adapting BERT-style masked prediction to images using Vision Transformers [12]. This approach has been extended to video data by VideoMAE [64] and BEVT [70], which leverage spatiotemporal dependencies. Various works have combined contrastive and MAE objectives as a more efficient alternative for capturing spatiotemporal dependencies [43, 20, 38, 17, 34]. Of note is ViC-MAE [17], which uses local (patch-based) features for a masked image modeling loss. The local features are also pooled into a global feature vector which is used with a contrastive loss computed across frames from multiple videos. We further describe this method in the next section.

3 Methods

BEAST uses a combination of image-based masked autoencoding (MAE) and contrastive objectives (Fig. 1A). The MAE loss excels at capturing per-frame appearance details, while the temporal contrastive loss captures dependencies across frames. This integration enables a single backbone to excel across diverse downstream tasks, from precise keypoint localization to predicting complex structure in neural activity (Fig. 1B).

BEAST builds upon ViC-MAE [17], which combines masked autoencoding and contrastive losses, but introduces key adaptations for neuroscience applications. The most significant modification is how frames are sampled for the contrastive loss. ViC-MAE allows any two frames from the same video to be a positive pair, and frames from different videos are negative pairs [75]. While this may be appropriate for benchmark datasets with short clips, animal behavior experiments generate continuous, long-duration recordings where behaviors repeat across time. Our approach instead defines positive frames within a narrow temporal window around the anchor (± 1 frame), while allowing negative frames to be either distant and dissimilar frames in the same video, or from entirely different videos. Crucially, we find this strategy outperforms that of ViC-MAE (Table 5). See Appendix B for more details on our frame selection strategy and additional training and architecture simplifications of ViC-MAE.

Vision transformer (ViT). The standard image ViT [12] data pipeline starts with a 2D image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ (where H, W, C are height, width, channels) and splits it into 2D patches, each with shape $(P \times P \times C)$, where the patch size P is typically 16. Each patch is reshaped to a vector of length P^2C , and all patches are concatenated into a sequence of the N flattened 2D patches $\mathbf{x}_p \in \mathbb{R}^{N \times (P^2C)}$, where N is the total number of patches. Each flattened patch is mapped with a trainable linear projection to a “patch token,” a vector of size $D = 768$ (the standard embedding size for the ViT-B/16 architecture). We add standard 1D position embeddings to the patch tokens to retain information about the patch location. We prepend a learnable CLS token to the sequence of patch tokens, which serves as a summary representation for the image. The resulting patch tokens augmented with position embeddings ($\mathbf{t} \in \mathbb{R}^{N \times D}$) and the concatenated CLS token serve as the input to the standard ViT encoder architecture.

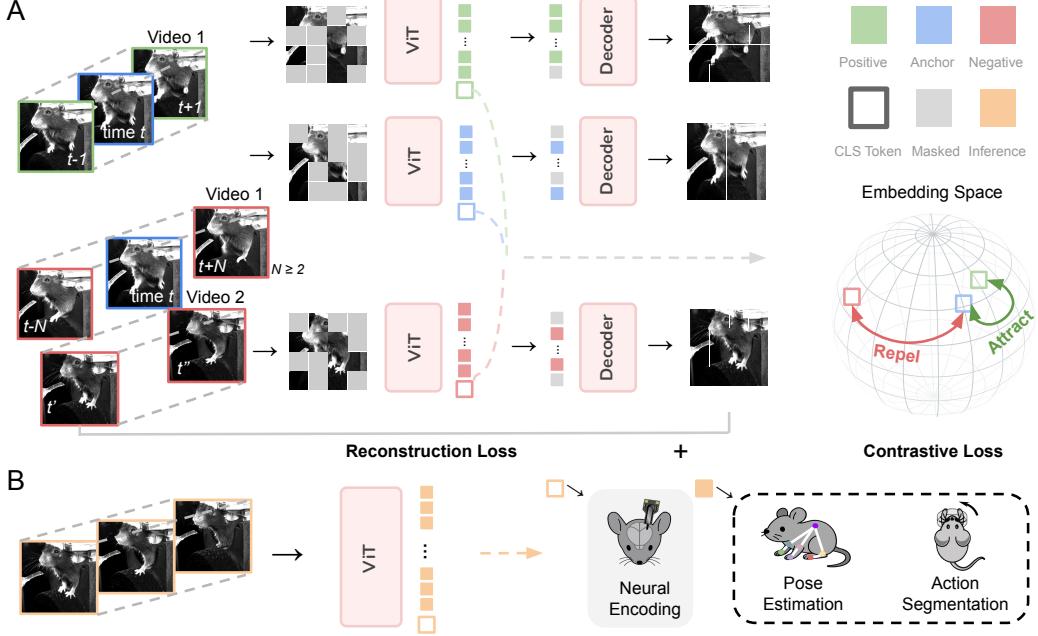


Figure 1: **BEAST framework.** **A:** Our self-supervised pretraining framework BEAST combines masked autoencoding [15] with temporal contrastive learning [9]. An anchor frame at time t is paired with a positive frame from $t \pm 1$, while more distant frames from the same video, or frames from other videos, serve as negative examples. Frames are divided into patches, with most patches randomly masked. A Vision Transformer (ViT) processes the remaining patches, which must reconstruct all patches. The ViT’s CLS tokens, which serve as a global representation of each frame, are nonlinearly projected into a new space where the contrastive loss pulls anchor-positive pairs together while pushing anchor-negative pairs apart. **B:** BEAST supports various downstream neuro-behavioral tasks including neural encoding, pose estimation, and action segmentation.

Masked autoencoding loss. The masked autoencoding loss randomly masks out a high proportion of the patch tokens (here, 0.75 [15]). We call the resulting unmasked tokens $\mathbf{t}_{um} \in \mathbb{R}^{L \times D}$, where $L = 0.25 \times N$ is the number of unmasked tokens. The unmasked tokens are then processed by the ViT, resulting in embeddings $\mathbf{z}_{um} = \text{ViT}(\mathbf{t}_{um})$. The masked embeddings $\mathbf{t}_m \in \mathbb{R}^{(N-L) \times D}$ (consisting of all zeros) are then combined with the unmasked embeddings processed by ViT to reform a complete patch sequence $\mathbf{z} \in \mathbb{R}^{N \times D}$. These embeddings are passed through a transformer decoder to produce a reconstruction $\hat{\mathbf{x}}_p \in \mathbb{R}^{N \times P^2 C}$ trained via mean square error (MSE): $\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{p=1}^N (\mathbf{x}_p - \hat{\mathbf{x}}_p)^2$. We refer to the model trained only with this masked autoencoding loss as ViT-M.

Temporal contrastive loss. The masked autoencoding loss is sufficient for reconstructing low-level features on individual frames. To imbue our embeddings with temporal information (which may be required for certain downstream tasks), we employ a contrastive loss that produces similar embeddings for frames close in time, and distinct embeddings for frames far apart in time or from different videos. To achieve this, each batch with B samples contains $B/2$ anchor frames, and each anchor frame \mathbf{x}_i^v (from time t and video v) has a randomly chosen positive frame from $\mathbf{x}_{i \pm 1}^v$. All remaining $B - 2$ frames are treated as negative frames. Note this approach differs from other temporal contrastive losses that allow any frame from the anchor frame’s video to be a positive frame [75, 17], which does not perform well with temporally-extended behavioral videos (Table 5). To improve the robustness of this approach, we select the initial set of anchor frames from a given video to be as visually distinct from each other as possible (Table 4). We utilize the InfoNCE loss [45] computed on nonlinear projections of the CLS embeddings that are output by the ViT. The projector outputs $\{\mathbf{z}_b^p\}$ are used for the contrastive learning, calculated as $\mathcal{L}_{\text{InfoNCE}} = -\frac{2}{B} \sum_{i \in \mathcal{A}} \log \frac{\exp(\mathbf{z}_i^p \cdot \mathbf{z}_{i'}^p)}{\sum_{j \neq i} \exp(\mathbf{z}_i^p \cdot \mathbf{z}_j^p)}$, where i' is the positive example associated with i and \mathcal{A} is the set of $B/2$ anchor frames. We refer to the model trained with both the masked autoencoding and contrastive losses as BEAST.

Training and finetuning. We initialize our models with pretrained ImageNet weights [11, 15]. Details of the training dataset, data augmentations, and batch construction are provided in Appendix B. We

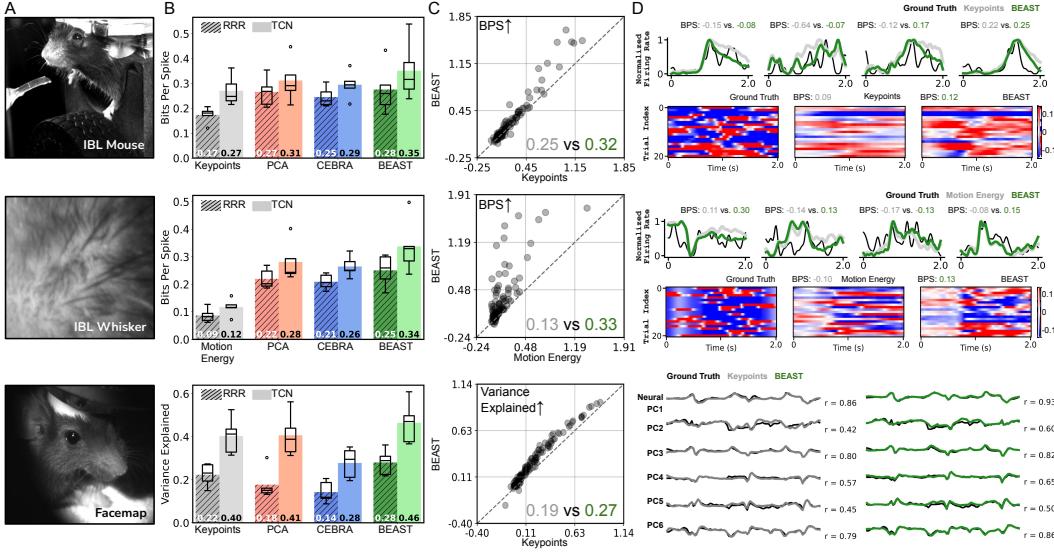


Figure 2: BEAST improves neural encoding. **A:** Example video frame from each dataset. **B:** Encoding performance is evaluated across multiple baseline features with both linear models (hatched bars; reduced rank regression, or RRR [51]) and nonlinear models (solid bars; temporal convolution network, or TCN [63]). CEBRA [55] uses a contrastive (but not masked autoencoding) loss to embed video frames in a latent feature space. “Motion energy” for the IBL-whisker dataset is a 1D estimate of movement calculated as the sum of the absolute pixel differences between successive frames. BEAST features outperform all baselines in both linear and nonlinear regimes. Boxplot showing variability across five test sessions. **C:** Scatterplot comparison of BEAST vs keypoint-based model performance in an example session. Each dot corresponds to an individual neuron. The values in the bottom-right corner represent the session-averaged BPS. **D:** *Top, middle:* comparison of the predicted trial-averaged firing rates for BEAST and keypoints (lines) and single-trial variability obtained by subtracting the neuron’s average firing rate on each trial (heatmaps). *Bottom:* comparison of predicted neural principal components for the Facemap dataset.

define the loss as $\mathcal{L}_{\text{MSE}} + \lambda \cdot \mathcal{L}_{\text{InfoNCE}}$, where λ is a hyperparameter that balances the two losses and is set between 0.01 and 0.03. We select λ using the validation sets of the various datasets. Models are trained for 800 epochs using the AdamW optimizer [37] with a cosine annealing learning rate scheduler [36], taking approximately 25 hours on 8 Nvidia A40 GPUs.

4 Results

We demonstrate the versatility of BEAST through comprehensive evaluation across three downstream neuro-behavioral tasks: (1) neural encoding, which challenges the model to extract spatiotemporal features that can predict patterns in neural activity; (2) pose estimation, which assesses the model’s ability to extract fine-grained appearance details; and (3) action segmentation, which evaluates the model’s capacity to extract spatiotemporal information required for understanding behavioral sequences. Throughout these evaluations, we present systematic ablation experiments that demonstrate the critical importance of our combined loss functions.

4.1 Neural encoding

Predicting neural activity from behavior videos represents a significant challenge with promising implications for understanding the relationship between brain and behavior [44, 58, 71]. Traditional approaches often rely on keypoints [28, 63], potentially missing critical behavioral features that are not included in tracking or are obscured by fur or feathers. While several studies have employed Principal Component Analysis (PCA) [44, 58], this linear technique may inadequately capture subtle behavioral nuances. Vision transformer embeddings offer a compelling alternative, potentially outperforming linear approaches without being constrained by predefined keypoints, thereby providing richer behavioral representations that could reveal previously undetectable neuro-behavioral correlations.

Datasets We present results on three high-quality neuro-behavioral datasets employing diverse neural recording technologies (Fig. 2). The first dataset is a head-fixed mouse performing a decision-making task from the International Brain Laboratory (IBL) Reproducible Electrophysiology study [28]. This

dataset, “IBL,” features simultaneous behavioral video and neural activity monitoring at single-cell, single-spike resolution using Neuropixels probes [21] spanning multiple brain regions (average of 168 neurons per session). The second dataset, “IBL-whisker,” reuses the same sessions as “IBL” but utilizes a cropped area around the whisker pad in the video, a particularly salient behavioral feature for predicting neural activity [58, 74, 63]. The third dataset comes from the Facemap study [63], where neural activity is captured through two-photon calcium imaging, a technique capable of resolving a large number of individual cells, but unable to detect individual spikes. Following the authors’ approach, we predict the principal components of the neural activity to capture predominant variance patterns across the large recorded neural populations.

Models We first describe various feature representations used for neural encoding, followed by two models (linear and nonlinear) that we fit to each representation. The first representation for IBL and Facemap are keypoints tracked across the face and body (11 for IBL, 12 for Facemap). For the IBL-whisker dataset, which lacks keypoints, we instead utilize a 1D estimate of whisker pad motion energy (Appendix C). The second representation utilizes PCA applied to raw video frames, while the third leverages CEBRA [55] (which employs a contrastive loss to embed inputs in a latent feature space) applied to raw video frames. Finally, we present results (using the CLS token) from BEAST, which is initialized with ImageNet weights then fine-tuned separately on each test session. We train two encoders: linear encoders, which reveal how directly accessible information is within the features, and nonlinear encoders, which better determine the upper bounds of information content in the features. The linear encoder is a reduced rank regression model trained across multiple sessions [81]. The nonlinear encoder is the temporal convolution network proposed in the Facemap study [63].

Evaluation All model hyperparameters are tuned to ensure robust baseline performance (Appendix C). To evaluate our neural encoding approaches, we utilize the Bits Per Spike (BPS) metric [47] on the spike-resolved IBL dataset (higher values indicate better performance), and the R^2 metric on the neural principal components in the Facemap dataset. All models are evaluated on five test sessions.

Results We find that nonlinear encoders consistently outperform their linear counterparts across all datasets and feature representations (Fig. 2 and Table 11). Notably, non-keypoint representations surpass keypoint-based approaches in both IBL and Facemap datasets, confirming our hypothesis that behavior videos contain richer information than what pose estimation typically captures. BEAST shows consistent improvements in neural encoding quality across all datasets and a range of dimensionalities (Fig. 6), indicating that BEAST’s exceptional performance is not limited to high-dimensional embedding spaces. Interestingly, the comparable BPS values for BEAST in both IBL and IBL-whisker datasets suggest that a substantial portion of the neurally-relevant behavior information is captured by the whisker pad activity, at least in the recorded brain regions.

The ViT-based models in Fig. 2 are fine-tuned individually for each session to enable direct comparison with baseline approaches. We investigated whether pretraining provides additional benefits (Table 1). Strikingly, models pretrained on ImageNet using only the MAE loss, “ViT-M (IN)”, outperform baselines without any fine-tuning. Further pretraining on 77 IBL sessions, “ViT-M (IN+PT)”, improves performance on both IBL datasets, validating the importance of domain-specific pretraining. By incorporating the contrastive objective, BEAST achieves superior zero-shot performance: BEAST (IN+PT) outperforms the MAE-only variant. Session-specific fine-tuning, “BEAST (IN+PT+FT)”, provides additional significant gains, reaching performance levels comparable to models fine-tuned directly from ImageNet weights (Table 11). Notably, even without fine-tuning, domain-specific pre-trained models remain highly competitive, offering researchers a practical option when computational resources for fine-tuning are limited. Finally, we experimented with using the patch embeddings as input to the neural encoder, but found superior performance with the CLS tokens (Table 10).

4.2 Pose estimation

Pose estimation is a fundamental technique in animal behavior analysis [49], enabling precise quantification of posture and movement. Unlike human pose estimation, which benefits from extensive labeled datasets and standardized anatomy, animal pose estimation presents unique challenges such as scarcity of large annotated datasets and significant morphological diversity across species. Pretraining

Method (TCN)	IBL	IBL-whisker
ViT-M (IN)	0.325 ± 0.091	0.307 ± 0.068
ViT-M (IN+PT)	0.334 ± 0.098	0.316 ± 0.073
BEAST (IN+PT)	0.337 ± 0.103	0.317 ± 0.083
BEAST (IN+PT+FT)	0.352 ± 0.106	0.335 ± 0.079

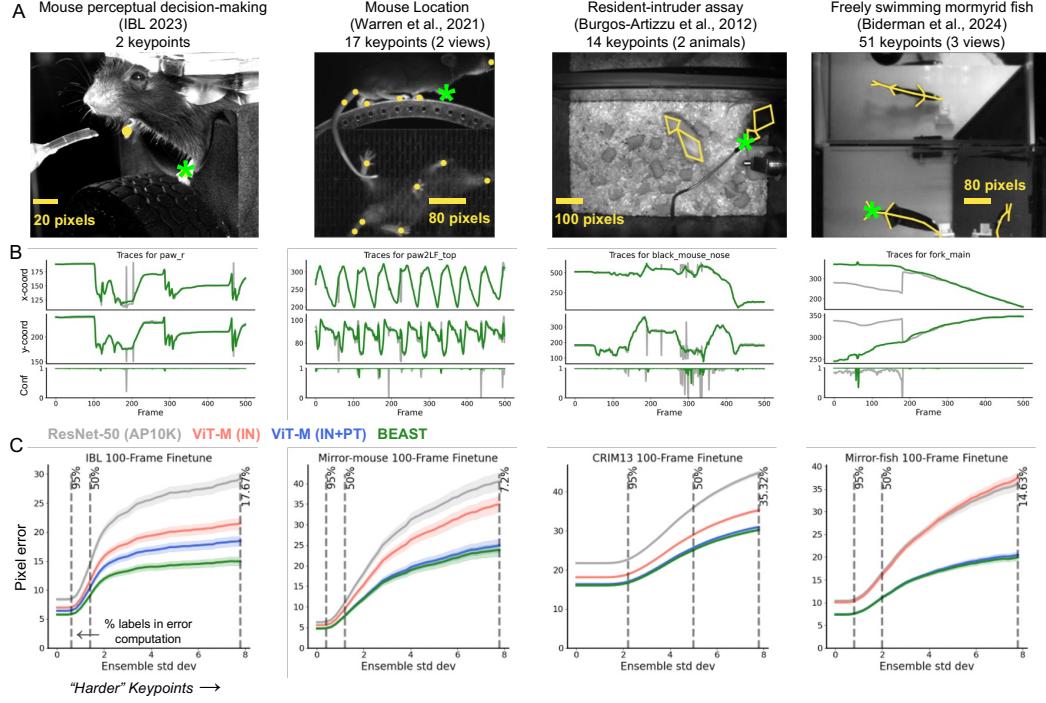


Figure 3: BEAST improves pose estimation. **A:** Example frame from each dataset overlaid with ground truth annotations. Green stars indicate the highlighted keypoint in panel B. **B:** Example traces from the ResNet-50 (gray) and BEAST (green) models for a single keypoint in a held-out video. BEAST traces evolve more smoothly in time and do not contain erroneous jumps like the ResNet-50 baseline. **C:** Pixel error as a function of keypoint difficulty (see main text; smaller is better): left-hand side shows performance across all keypoints; moving to the right drops the easier keypoints defined by inter-seed and -model prediction variance. Vertical dashed lines indicate the percentage of data used for the pixel error computation. ViT-M (IN) is a ViT backbone pretrained on ImageNet with a masked autoencoding loss; ViT-M (IN+PT) uses the same architecture and loss but is initialized with ImageNet-pretrained weights then further pretrained on experiment-specific unlabeled frames.

vision transformers on large volumes of unlabeled behavior videos can potentially reduce the labeled data requirements for accurate keypoint localization in various experimental paradigms.

Datasets We present results on four distinct datasets (Fig. 3): (1) a head-fixed mouse performing a decision-making task from the IBL [29]; (2) a head-fixed mouse running on a treadmill, seen from two views [72]; (3) the Caltech Resident-Intruder Mouse (CRIM13) dataset, consisting of two freely interacting mice [6]; and (4) a freely moving weakly electric fish, seen from three views [3].

Models We implemented our pose estimation models using the Lightning Pose framework [3]. We established a strong baseline utilizing a ResNet-50 backbone pretrained on the AP-10K dataset [79], which previous research has shown outperforms ImageNet-pretrained alternatives for animal pose estimation tasks [3]. To evaluate architectural advantages, we incorporated a Vision Transformer (ViT-B/16) pretrained on ImageNet as our second baseline [15], enabling assessment of potential improvements when transitioning from convolutional- to transformer-based architectures. Our own ViT-based models utilize this same architecture. For consistency across all model variants, we employ an identical pose estimation head that transforms backbone features into keypoint heatmaps.

Evaluation To rigorously evaluate our pose estimation models, we designed a challenging limited-data scenario with only 100 labeled training frames, a realistic constraint for many specialized research settings where extensive annotation is impractical. We measured pixel error between predicted keypoints and ground truth on a test set of novel subjects. For each backbone, we fit three models on different 100-frame subsets. Results are presented as pixel error relative to ensemble standard deviation (e.s.d.) across all seeds and backbones following [3], with error curves showing performance at varying difficulty thresholds. Each point corresponds to keypoints with e.s.d. exceeding the threshold value, with the leftmost portion showing error across all keypoints and rightward movement including only increasingly challenging keypoints (those with higher inter-model variability).

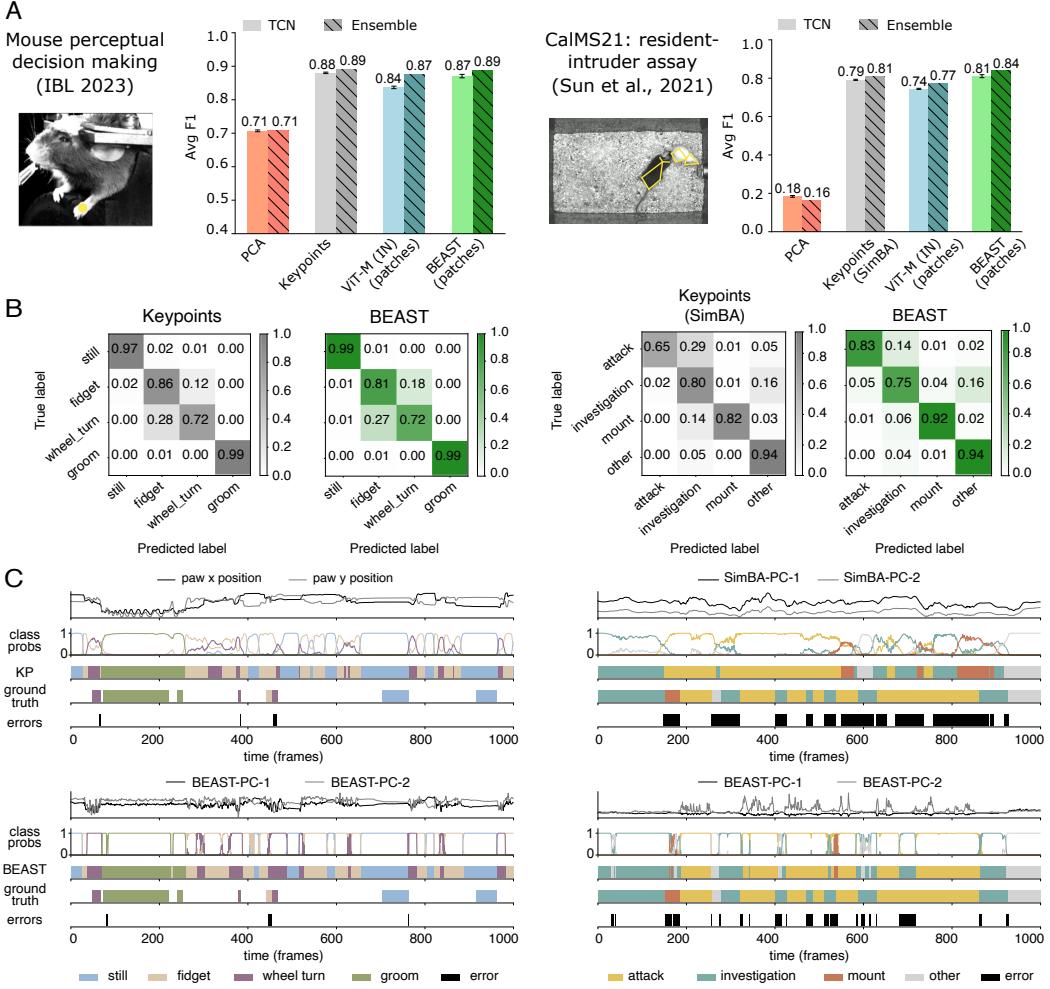


Figure 4: BEAST improves action segmentation. **A:** Example frame from each dataset; action segmentation performance evaluated across multiple baseline features with both TCN (solid) and ensembled (hatched) models. BEAST single-model patch embeddings outperform all baselines in the resident-intruder assay, and perform about as well as keypoints in IBL. With ensembling, BEAST matches or exceeds all baselines. Error bars on single models represent standard error of the mean across five random initializations. **B:** Confusion matrices for the single TCN models based on keypoints and BEAST patch embeddings. **C:** Example behavior sequences with feature traces (single seed shown for BEAST models), ensemble probabilities, ensembled model-predicted ethograms, ground truth ethograms, and error frames. PCs of SimBA and BEAST features are shown for illustration, but the models utilize the full feature set.

Results We find robust improvements in pose estimation quality across all datasets when utilizing BEAST (Fig. 3). The ImageNet-pretrained ViT outperforms the AP-10K-pretrained ResNet-50 on all datasets except the fish, demonstrating the effectiveness of transformers even with limited labels. Notably, pretraining the transformer with the MAE objective on experiment-specific data yields substantial performance gains across all datasets, including the challenging fish dataset. Augmenting MAE with the contrastive objective (BEAST) produces additional performance improvements for some datasets, with particularly pronounced benefits observed in the IBL dataset. We also find these performance advantages persist when scaling to larger training datasets (Fig. 7).

4.3 Action segmentation

Action segmentation classifies discrete behaviors using spatiotemporal video features [49]. Similar to pose estimation, a central challenge in animal action segmentation is the lack of large annotated datasets, as behaviors of interest often vary across species and experimental contexts. Many current approaches rely on keypoints [5, 22, 57, 13, 14], requiring an initial labor-intensive and error-prone

preprocessing step. Vision transformer embeddings eliminate this preprocessing requirement and provide an attractive alternative if they match or exceed the performance of keypoint-based methods.

Datasets We present results on two datasets (Fig. 4): (1) the “IBL” dataset [29], which contains four behavior classes for the paw closest to the camera: still, fidget, wheel-turn, and groom; and (2) the Caltech Mouse Social Interactions (CalMS21) dataset [59], a resident–intruder assay of interacting mouse pairs that contains four social behavior classes: attack, investigation, mount, and other.

Models We implemented action segmentation models using two types of embeddings from frozen ViT models: (1) CLS token embeddings and (2) per-patch embeddings. For CLS embeddings, we tested both a linear model and a temporal convolution network (TCN) [32], enriching the input with inter-frame differences [4]. We used a sliding window over this feature sequence to predict the action class of the central frame. For patch embeddings, we applied multi-head attention pooling [33, 80, 62] to integrate information across patches, then concatenated the resulting frame-level embeddings with their inter-frame differences before processing them through a TCN (Fig. 8).

For the IBL dataset, we compared against three baseline features: (1) a single paw keypoint, obtained using five pose estimation networks (each trained with 7,000 labeled frames) post-processed with an Ensemble Kalman Smoother [3]; (2) principal components of the video frames; and (3) the CLS and patch embeddings extracted from a frozen-weight ImageNet-pretrained ViT. For CalMS21, we compared against four baseline features: (1) Trajectory Embedding for Behavior Analysis (TREBA) [60], a self-supervised feature extraction method for keypoint trajectories (seven keypoints on each mouse, Fig. 4A); (2) Simple Behavioral Analysis (SimBA) [14], which manually extracts hundreds of features based on spatiotemporal relationships in trajectories of those same keypoints; (3) principal components of video frames; and (4) the CLS and patch embeddings from a frozen-weight ImageNet-pretrained ViT. The pose estimator used for TREBA and SimBA was trained with 15,000 labeled frames [59]. For all baselines except SimBA, we also concatenated inter-frame differences.

Evaluation All model hyperparameters are tuned to ensure robust baseline performance (Appendix E). We evaluate model performance on a held-out set of animals using the macro-averaged F1 score. For the CalMS21 dataset, following [59], we focused our evaluation on the attack, investigation, and mount classes, averaging the F1 score over these three behaviors. For all feature types and models we train five networks using different random weight initializations. We also report the results of the TCN ensemble by averaging unnormalized logits across seeds before applying softmax.

Results BEAST demonstrates strong action segmentation performance across all datasets (Fig. 4). Remarkably, ImageNet-pretrained ViT-M patch embeddings nearly match keypoint-based methods despite utilizing a frozen, general-purpose backbone. This establishes a competitive baseline without requiring the thousands of labeled frames needed to train pose estimation networks. On the IBL dataset, BEAST improves upon ImageNet baselines. The keypoint-based model excels here due to action classes corresponding to single paw movements that are easily captured by pose estimation, but this advantage disappears with ensembling: BEAST ensemble performance matches the keypoint ensemble. The CalMS21 dataset better demonstrates BEAST’s capabilities, where our method surpasses the SimBA baseline and substantially outperforms the TREBA baseline (Table 2). Additional experiments confirm domain-specific pretraining benefits: BEAST CLS tokens consistently outperform their ImageNet-pretrained counterparts (Table 2), though patch-based models perform significantly better due to their enhanced spatial resolution and multi-headed attention pooling. Across all experiments, nonlinear models consistently outperform their linear counterparts, except for PCA features on CalMS21 (Table 13). All evaluations use frozen backbones with only linear/TCN heads fine-tuned, suggesting further gains may be possible through full backbone fine-tuning.

BEAST’s advantages extend beyond absolute F1 improvements. Pose estimation-based approaches require both extensive labeling and iterative training and validation of pose estimation models before action segmentation, often a months- or even years-long process [30]. BEAST eliminates this entire pipeline, achieving competitive or superior performance using only unlabeled video for pretraining.

Table 2: Action segmentation performance ($F1 \pm S.E.M.$).

Method (TCN)	IBL	CalMS21
TREBA	–	0.72 ± 0.01
ViT-M (IN) (CLS)	0.79 ± 0.00	0.60 ± 0.00
ViT-M (IN) (patch)	0.84 ± 0.00	0.74 ± 0.00
BEAST (IN+PT) (CLS)	0.81 ± 0.00	0.63 ± 0.00
BEAST (IN+PT) (patch)	0.87 ± 0.01	0.81 ± 0.01

5 Discussion

This work introduces BEAST, a framework for self-supervised vision transformer pretraining leveraging domain-specific video data. We demonstrated BEAST’s significant benefits across neural encoding, pose estimation, and action segmentation tasks. While BEAST requires substantial computational resources (hundreds of GPU hours for large-scale pretraining on a single dataset), this initial investment creates a versatile backbone applicable to multiple downstream tasks. For many laboratories, this computational cost is far outweighed by the dramatic reduction in human labeling time and the improved accuracy achieved without iterative model refinement. The black-box nature of ViT embeddings presents interpretability challenges in scientific contexts where transparent representations like pose estimates are often preferred. Future work will focus on model compression strategies to reduce training time without performance degradation, and further exploring the transformer attention mechanism to better understand how BEAST functions in different task contexts.

Acknowledgments and Disclosure of Funding

This work was supported by the following: Gatsby Charitable Foundation GAT3708, NIH U19NS123716, NSF 1707398, the National Science Foundation and DoD OUSD (R&E) under Cooperative Agreement PHY-2229929 (The NSF AI Institute for Artificial and Natural Intelligence), Simons Foundation, Wellcome Trust 216324, and funds from Zuckerman Institute Team Science.

References

- [1] Eleanor Batty, Matthew Whiteway, Shreya Saxena, Dan Biderman, Taiga Abe, Simon Musall, Winthrop Gillis, Jeffrey Markowitz, Anne Churchland, John P Cunningham, et al. Behavenet: nonlinear embedding and bayesian neural decoding of behavioral videos. *Advances in neural information processing systems*, 32, 2019.
- [2] Gordon J Berman, Daniel M Choi, William Bialek, and Joshua W Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99):20140672, 2014.
- [3] Dan Biderman, Matthew R Whiteway, Cole Hurwitz, Nicholas Greenspan, Robert S Lee, Ankit Vishnubhotla, Richard Warren, Federico Pedraja, Dillon Noone, Michael M Schartner, et al. Lightning pose: improved animal pose estimation via semi-supervised learning, bayesian ensembling and cloud-native open-source tools. *Nature Methods*, 21(7):1316–1328, 2024.
- [4] Ari Blau, Evan S Schaffer, Neeli Mishra, Nathaniel J Miska, International Brain Laboratory, Liam Paninski, and Matthew R Whiteway. A study of animal action segmentation algorithms across supervised, unsupervised, and semi-supervised learning paradigms. *ArXiv*, pages arXiv–2407, 2024.
- [5] Kristin Branson, Alice A Robie, John Bender, Pietro Perona, and Michael H Dickinson. High-throughput ethomics in large groups of drosophila. *Nature methods*, 6(6):451–457, 2009.
- [6] 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.
- [7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.
- [8] Susu Chen, Yi Liu, Ziyue Aiden Wang, Jennifer Colonell, Liu D Liu, Han Hou, Nai-Wen Tien, Tim Wang, Timothy Harris, Shaul Druckmann, et al. Brain-wide neural activity underlying memory-guided movement. *Cell*, 187(3):676–691, 2024.
- [9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.
- [10] 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.
- [11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [13] Christopher J Gabriel, Zachary Zeidler, Benita Jin, Changliang Guo, Caitlin M Goodpaster, Adrienne Q Kashay, Anna Wu, Molly Delaney, Jovian Cheung, Lauren E DiFazio, et al. Behaviordepot is a simple, flexible tool for automated behavioral detection based on markerless pose tracking. *Elife*, 11:e74314, 2022.
- [14] Nastacia L Goodwin, Jia J Choong, Sophia Hwang, Kayla Pitts, Liana Bloom, Aasiya Islam, Yizhe Y Zhang, Eric R Szelenyi, Xiaoyu Tong, Emily L Newman, et al. Simple behavioral analysis (simba) as a platform for explainable machine learning in behavioral neuroscience. *Nature neuroscience*, 27(7):1411–1424, 2024.

- [15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.
- [16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [17] Jefferson Hernandez, Ruben Villegas, and Vicente Ordonez. Vic-mae: Self-supervised representation learning from images and video with contrastive masked autoencoders. In *European Conference on Computer Vision*, pages 444–463. Springer, 2024.
- [18] Alexander I Hsu and Eric A Yttri. B-soid, an open-source unsupervised algorithm for identification and fast prediction of behaviors. *Nature communications*, 12(1):5188, 2021.
- [19] Shih-Cheng Huang, Anuj Pareek, Malte Jensen, Matthew P Lungren, Serena Yeung, and Akshay S Chaudhari. Self-supervised learning for medical image classification: a systematic review and implementation guidelines. *NPJ Digital Medicine*, 6(1):74, 2023.
- [20] Zhicheng Huang, Xiaojie Jin, Chengze Lu, Qibin Hou, Ming-Ming Cheng, Dongmei Fu, Xiaohui Shen, and Jiashi Feng. Contrastive masked autoencoders are stronger vision learners. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(4):2506–2517, 2023.
- [21] James J Jun, Nicholas A Steinmetz, Joshua H Siegle, Daniel J Denman, Marius Bauza, Brian Barbarits, Albert K Lee, Costas A Anastassiou, Alexandru Andrei, Çağatay Aydin, et al. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, 2017.
- [22] 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.
- [23] Alexander JE Kell, Daniel LK Yamins, Erica N Shook, Sam V Norman-Haignere, and Josh H McDermott. A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy. *Neuron*, 98(3):630–644, 2018.
- [24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [25] John W Krakauer, Asif A Ghazanfar, Alex Gomez-Marin, Malcolm A MacIver, and David Poeppel. Neuroscience needs behavior: correcting a reductionist bias. *Neuron*, 93(3):480–490, 2017.
- [26] International Brain Laboratory. Data release - Brainwide map - Q4 2022. *Figshare*, 1 2023.
- [27] International Brain Laboratory, Valeria Aguillon-Rodriguez, Dora Angelaki, Hannah Bayer, Niccolo Bonacchi, Matteo Carandini, Fanny Cazettes, Gaelle Chapuis, Anne K Churchland, Yang Dan, et al. Standardized and reproducible measurement of decision-making in mice. *Elife*, 10:e63711, 2021.
- [28] International Brain Laboratory, Kush Banga, Julius Benson, Jai Bhagat, Dan Biderman, Daniel Birman, Niccolò Bonacchi, Sebastian A Bruijns, Kelly Buchanan, Robert AA Campbell, et al. Reproducibility of in vivo electrophysiological measurements in mice. *Elife*, 13:RP100840, 2025.
- [29] International Brain Laboratory, Brandon Benson, Julius Benson, Daniel Birman, Niccolò Bonacchi, Matteo Carandini, Joana A Catarino, Gaelle A Chapuis, Anne K Churchland, Yang Dan, et al. A brain-wide map of neural activity during complex behaviour. *bioRxiv*, pages 2023–07, 2023.
- [30] International Brain Laboratory, Dan Birman, Niccolò Bonacchi, Kelly Buchanan, Gaelle Chapuis, Julia Huntenburg, Guido Meijer, Liam Paninski, Michael Schartner, Karel Svoboda, Matthew Whiteway, Miles Wells, and Olivier Winter. Video hardware and software for the international brain laboratory. *figshare*, 2022.

- [31] Erica Lastufka, Mariia Drozdova, Vitaliy Kinakh, Davide Piras, and S Voloshynovskyy. Vision foundation models: can they be applied to astrophysics data? *arXiv preprint arXiv:2409.11175*, 2024.
- [32] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. In *Computer vision–ECCV 2016 workshops: Amsterdam, the Netherlands, October 8–10 and 15–16, 2016, proceedings, part III 14*, pages 47–54. Springer, 2016.
- [33] 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.
- [34] Johannes Lehner, Benedikt Alkin, Andreas Fürst, Elisabeth Rumetshofer, Lukas Miklautz, and Sepp Hochreiter. Contrastive tuning: A little help to make masked autoencoders forget. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 2965–2973, 2024.
- [35] Yuanning Li, Gopala K Anumanchipalli, Abdelrahman Mohamed, Peili Chen, Laurel H Carney, Junfeng Lu, Jinsong Wu, and Edward F Chang. Dissecting neural computations in the human auditory pathway using deep neural networks for speech. *Nature Neuroscience*, 26(12):2213–2225, 2023.
- [36] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.
- [37] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [38] Cheng-Ze Lu, Xiaojie Jin, Zhicheng Huang, Qibin Hou, Ming-Ming Cheng, and Jiashi Feng. Cmae-v: contrastive masked autoencoders for video action recognition. *arXiv preprint arXiv:2301.06018*, 2023.
- [39] Kevin Luxem, Petra Mocellin, Falko Fuhrmann, Johannes Kürsch, Stephanie R Miller, Jorge J Palop, Stefan Remy, and Pavol Bauer. Identifying behavioral structure from deep variational embeddings of animal motion. *Communications Biology*, 5(1):1267, 2022.
- [40] 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.
- [41] Mackenzie Weygandt Mathis and Alexander Mathis. Deep learning tools for the measurement of animal behavior in neuroscience. *Current opinion in neurobiology*, 60:1–11, 2020.
- [42] James M McFarland, Yuwei Cui, and Daniel A Butts. Inferring nonlinear neuronal computation based on physiologically plausible inputs. *PLoS computational biology*, 9(7):e1003143, 2013.
- [43] Shlok Mishra, Joshua Robinson, Huiwen Chang, David Jacobs, Aaron Sarna, Aaron Maschinot, and Dilip Krishnan. A simple, efficient and scalable contrastive masked autoencoder for learning visual representations. *arXiv preprint arXiv:2210.16870*, 2022.
- [44] Simon Musall, Matthew T Kaufman, Ashley L Juavinett, Steven Gluf, and Anne K Churchland. Single-trial neural dynamics are dominated by richly varied movements. *Nature neuroscience*, 22(10):1677–1686, 2019.
- [45] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [46] Liam Paninski. Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, 15(4):243, 2004.
- [47] Felix Pei, Joel Ye, David Zoltowski, Anqi Wu, Raeed H Chowdhury, Hansem Sohn, Joseph E O’Doherty, Krishna V Shenoy, Matthew T Kaufman, Mark Churchland, et al. Neural latents benchmark’21: Evaluating latent variable models of neural population activity. *arXiv preprint arXiv:2109.04463*, 2021.

- [48] Talmo D Pereira, Diego E Aldarondo, Lindsay Willmore, Mikhail Kislin, Samuel S-H Wang, Mala Murthy, and Joshua W Shaevitz. Fast animal pose estimation using deep neural networks. *Nature methods*, 16(1):117–125, 2019.
- [49] Talmo D Pereira, Joshua W Shaevitz, and Mala Murthy. Quantifying behavior to understand the brain. *Nature neuroscience*, 23(12):1537–1549, 2020.
- [50] Jonathan W Pillow, Jonathon Shlens, Liam Paninski, Alexander Sher, Alan M Litke, EJ Chichilnisky, and Eero P Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, 2008.
- [51] Lorenzo Posani, Shuqi Wang, Samuel P Muscinelli, Liam Paninski, and Stefano Fusi. Rarely categorical, always high-dimensional: how the neural code changes along the cortical hierarchy. *bioRxiv*, pages 2024–11, 2025.
- [52] Rui Qian, Tianjian Meng, Boqing Gong, Ming-Hsuan Yang, Huisheng Wang, Serge Belongie, and Yin Cui. Spatiotemporal contrastive video representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6964–6974, 2021.
- [53] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [54] Adria Recasens, Pauline Luc, Jean-Baptiste Alayrac, Luyu Wang, Florian Strub, Corentin Tallec, Mateusz Malinowski, Viorica Pătrăucean, Florent Altché, Michal Valko, et al. Broaden your views for self-supervised video learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1255–1265, 2021.
- [55] Steffen Schneider, Jin Hwa Lee, and Mackenzie Weygandt Mathis. Learnable latent embeddings for joint behavioural and neural analysis. *Nature*, 617(7960):360–368, 2023.
- [56] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Franziska Geiger, et al. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv*, page 407007, 2018.
- [57] 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.
- [58] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Charu Bai Reddy, Matteo Carandini, and Kenneth D Harris. Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437):eaav7893, 2019.
- [59] 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. *Advances in neural information processing systems*, 2021(DB1):1, 2021.
- [60] 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, 2021.
- [61] Jennifer J Sun, Serim Ryou, Roni H Goldshmid, Brandon Weissbourd, John O Dabiri, David J Anderson, Ann Kennedy, Yisong Yue, and Pietro Perona. Self-supervised keypoint discovery in behavioral videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2171–2180, 2022.
- [62] 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*, pages 2024–07, 2024.

- [63] Atika Syeda, Lin Zhong, Renee Tung, Will Long, Marius Pachitariu, and Carsen Stringer. Facemap: a framework for modeling neural activity based on orofacial tracking. *Nature neuroscience*, 27(1):187–195, 2024.
- [64] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.
- [65] Wilson Truccolo, Uri T Eden, Matthew R Fellows, John P Donoghue, and Emery N Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of neurophysiology*, 93(2):1074–1089, 2005.
- [66] Anne E Urai, Brent Doiron, Andrew M Leifer, and Anne K Churchland. Large-scale neural recordings call for new insights to link brain and behavior. *Nature neuroscience*, 25(1):11–19, 2022.
- [67] Lukas von Ziegler, Oliver Sturman, and Johannes Bohacek. Big behavior: challenges and opportunities in a new era of deep behavior profiling. *Neuropsychopharmacology*, 46(1):33–44, 2021.
- [68] Lukas M von Ziegler, Fabienne K Roessler, Oliver Sturman, Rebecca Waag, Mattia Privitera, Sian N Duss, Eoin C O’Connor, and Johannes Bohacek. Analysis of behavioral flow resolves latent phenotypes. *Nature Methods*, pages 1–12, 2024.
- [69] Eric Y Wang, Paul G Fahey, Zhuokun Ding, Stelios Papadopoulos, Kayla Ponder, Marissa A Weis, Andersen Chang, Taliah Muhammad, Saumil Patel, Zhiwei Ding, et al. Foundation model of neural activity predicts response to new stimulus types. *Nature*, 640(8058):470–477, 2025.
- [70] Rui Wang, Dongdong Chen, Zuxuan Wu, Yinpeng Chen, Xiyang Dai, Mengchen Liu, Yu-Gang Jiang, Luowei Zhou, and Lu Yuan. Bevt: Bert pretraining of video transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14733–14743, 2022.
- [71] Ziyue Aiden Wang, Susu Chen, Yi Liu, Dave Liu, Karel Svoboda, Nuo Li, and Shaul Druckmann. Not everything, not everywhere, not all at once: a study of brain-wide encoding of movement. *bioRxiv*, pages 2023–06, 2023.
- [72] Richard A Warren, Qianyun Zhang, Judah R Hoffman, Edward Y Li, Y Kate Hong, Randy M Bruno, and Nathaniel B Sawtell. A rapid whisker-based decision underlying skilled locomotion in mice. *Elife*, 10:e63596, 2021.
- [73] Caleb Weinreb, Jonah E Pearl, Sherry Lin, Mohammed Abdal Monium Osman, Libby Zhang, Sidharth 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.
- [74] Matthew R Whiteway, Dan Biderman, Yoni Friedman, Mario Dipoppa, E Kelly Buchanan, Anqi Wu, John Zhou, Niccolò Bonacchi, Nathaniel J Miska, Jean-Paul Noel, et al. Partitioning variability in animal behavioral videos using semi-supervised variational autoencoders. *PLoS computational biology*, 17(9):e1009439, 2021.
- [75] Jiarui Xu and Xiaolong Wang. Rethinking self-supervised correspondence learning: A video frame-level similarity perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10075–10085, 2021.
- [76] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the national academy of sciences*, 111(23):8619–8624, 2014.
- [77] Shaokai Ye, Anastasiia Filippova, Jessy Lauer, Steffen Schneider, Maxime Vidal, Tian Qiu, Alexander Mathis, and Mackenzie Weygandt Mathis. Superanimal pretrained pose estimation models for behavioral analysis. *Nature communications*, 15(1):5165, 2024.

- [78] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [79] Hang Yu, Yufei Xu, Jing Zhang, Wei Zhao, Ziyu Guan, and Dacheng Tao. Ap-10k: A benchmark for animal pose estimation in the wild. *arXiv preprint arXiv:2108.12617*, 2021.
- [80] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.
- [81] Yizi Zhang, Hanrui Lyu, Cole Hurwitz, Shuqi Wang, Charles Findling, Felix Hubert, Alexandre Pouget, International Brain Laboratory, Erdem Varol, and Liam Paninski. Exploiting correlations across trials and behavioral sessions to improve neural decoding. *bioRxiv*, 2024.
- [82] Yizi Zhang, Yanchen Wang, Mehdi Azabou, Alexandre Andre, Zixuan Wang, Hanrui Lyu, The International Brain Laboratory, Eva Dyer, Liam Paninski, and Cole Hurwitz. Neural encoding and decoding at scale. *arXiv preprint arXiv:2504.08201*, 2025.
- [83] 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.
- [84] Chengxu Zhuang, Jonas Kubilius, Mitra J Hartmann, and Daniel L Yamins. Toward goal-driven neural network models for the rodent whisker-trigeminal system. *Advances in Neural Information Processing Systems*, 30, 2017.

Appendix

A Datasets

All datasets used for this study were collected in compliance with the relevant ethical regulations (see the references for each dataset).

A.1 IBL

This dataset [26] from the International Brain Lab (IBL) and consists of head-fixed mice performing a decision-making task [27, 28, 29]. Two cameras—‘left’ (60 Hz) and ‘right’ (150 Hz)—capture roughly orthogonal side views of the mouse’s face and upper trunk during each session. Frames are down-sampled to 256×320 pixels for labeling and video storage. We accessed the raw videos and neural activity under the CC-BY 4.0 license using these instructions: https://int-brain-lab.github.io/iblenv/notebooks_external/data_release_brainwidemap.html. The data can also be visualized through a browser at <https://viz.internationalbrainlab.org>.

Pose estimation Frames were reshaped during training to 256×256 pixels. Two keypoints were labeled per view, one for each paw. We accessed the initial pose estimation labels from the public repository at https://ibl-brain-wide-map-public.s3.amazonaws.com/aggregates/Tags/2023_Q1_Biderman_Whiteway_et_al._ibl_videoTracking.trainingDataPaw.7e79e865-f2fc-4709-b203-77dbdac6461f.zip under the CC-BY 4.0 license. This dataset contains 6,071 labeled train frames from 35 animals and 1,446 labeled test frames from 10 animals, all at 128×102 pixel resolution.

Despite the large number of labeled frames, we observed poor performance in sessions with bright lights or other unusual distractors. Additionally, the low resolution often obscured fine details, for example making it difficult to visually distinguish individual paws when they are close together. To address these limitations, we retrieved the full-resolution frames (1280×1024 for left view, 640×512 for right) from the raw videos and downsampled them to 320×256 pixels. This higher resolution revealed occasional labeling errors, which we manually corrected. We then added 1,437 newly labeled frames from 15 additional animals, creating an expanded training set of 7,609 frames from 50 animals. We will publicly release these updated labels under a CC-BY 4.0 license upon acceptance of this manuscript.

Action segmentation Four action classes are labeled for the paw closest to the camera: (1) still; (2) grooming; (3) turning the wheel; and (4) fidget (any movement that is not grooming or wheel turning). We accessed the initial action segmentation labels from <https://doi.org/10.6084/m9.figshare.27479760.v1> under the CC-BY 4.0 license. This dataset contains 1,000,000 frames from 10 animals, of which 14,107 are labeled.

We expanded this dataset as the original study [4] only used five animals each for training and testing. First, we trained an ensemble of five TCN-based action segmentation models on all 10 existing animals. We applied these models to a new batch of 53 sessions and calculated the variance in predicted probabilities across all models for each frame. The 19 sessions with the highest average ensemble variance (indicating where models disagreed most) were selected for further labeling. We then labeled an additional 36,009 frames from these sessions. For the analyses in this paper, we selected the subset of all labeled sessions that are included in the BEAST pretraining sessions, and split these into train (32,521 frames from 18 animals) and test sets (7,786 frames from 5 animals). We will publicly release these updated labels under a CC-BY 4.0 license upon acceptance of this manuscript.

Neural encoding For the neural analysis we use a subset of the IBL repeated site dataset [28]. This dataset consists of Neuropixels recordings collected from 10 labs with standardized experimental pipelines. The recordings target the same five brain regions across all mice: VISa (primary visual cortex), CA1 and DG (hippocampus), and LP and PO (thalamic nuclei). We evaluate neural encoding models on five randomly selected sessions. Moreover, we used the trial-aligned neural activity data, taking 2 seconds of activity aligned to wheel movement onset. We binned the spikes every 20 ms to get a total of 100 bins per trial. We also filtered out the low firing rate neurons by setting a minimum threshold of 2 Hz.

A.2 IBL-whisker

We localize the whisker pad using anchor keypoints on the nose and eye, following the procedure in [30]. We use the same sessions and neural activity as the “IBL” dataset.

A.3 Mirror-mouse

Head-fixed mice ran on a circular treadmill while avoiding a moving obstacle [72]. The treadmill had a transparent floor and a mirror mounted inside at 45°, allowing a single camera to capture two roughly orthogonal views (side view and bottom view via the mirror) at 250 Hz. The camera was positioned at a large distance from the subject (~1.1 m) to minimize perspective distortion. Frames are 406 × 396 pixels and reshaped during pose estimation training to 256 × 256 pixels. Seventeen keypoints were labeled across the two views including seven keypoints on the mouse’s body per view, plus three keypoints on the moving obstacle. The full training dataset consists of 789 labeled frames across 10 animals; the test dataset consists of 253 labeled frames across three animals. We accessed the labeled pose estimation dataset from <https://doi.org/10.6084/m9.figshare.24993315.v1> under the CC-BY 4.0 license.

A.4 Mirror-fish

Mormyrid fish of the species *Gnathonemus petersii* swam freely in and out of an experimental tank, capturing worms from a well [3]. The tank had a side mirror and a top mirror, both at 45°, providing three different views seen from a single camera at 300 Hz. The camera was placed ~1.7 m away from the center of the fish tank to reduce distortions. Frame are 384 × 512 pixels and reshaped during training to 256 × 384 pixels. Seventeen body parts were labeled across each of three views for a total of 51 keypoints. The full training dataset consists of 373 frames across three animals; the test dataset consists of 94 frames across three animals. We accessed the labeled pose estimation dataset from <https://doi.org/10.6084/m9.figshare.24993363.v1> under the CC-BY 4.0 license.

A.5 CRIM13

The Caltech Resident-Intruder Mouse (CRIM13) dataset [6] consists of two mice interacting in an enclosed arena, captured by top and side-view cameras at 30 Hz. We only used the top view. Frames are 480 × 640 pixels and reshaped during training to 256 × 256 pixels. Seven keypoints were labeled on each mouse for a total of 14 keypoints [57]. The full training dataset consists of 3,986 frames across four resident mice; the test dataset consists of 1,274 frames across the same four resident mice but a different set of intruder mice. The original dataset is available at <https://data.caltech.edu/records/4emt5-b0t10>. We accessed the labeled pose estimation dataset from <https://doi.org/10.6084/m9.figshare.24993384.v1> under the CC-BY 4.0 license.

A.6 CalMS21

The Caltech Mouse Social Interactions (CalMS21) dataset [59], like CRIM13, consists of two mice interacting in an enclosed arena, captured by a top-view camera at 30 Hz. The dataset consists of many videos with tracked poses and corresponding frame-level behavior annotations. Four behavior classes are labeled: attack, investigation, mount, and other (i.e., none of the above). The full training dataset consists of 506,668 frames across 68 videos; the test dataset consists of 262,107 frames across 19 videos. We accessed the pose estimates, TREBA features [60], and behavior annotations from <https://doi.org/10.22002/D1.1991> under the CC-BY 4.0 license.

A.7 Facemap

Head-fixed mice were free to run on an air-floating ball in darkness [63]. A single infrared camera captured one of several side or front views of the mouse’s face and upper trunk during each session at 50 Hz. Fifteen keypoints were labeled across the face (mouse, nose, whiskers, eyes) and paw. Neural activity was recorded across visual and sensorimotor areas using two-photon calcium imaging. Approximately 30,000 to 50,000 cells were recorded in a given session. In our encoding task, we predict the 128 neural principal components following [63]. We evaluate neural encoding models on five randomly selected sessions. The publicly available data did not contain additional videos, so we only fine-tuned neural encoding models with this dataset. We accessed the raw videos, pose estimates, and neural activity from <https://doi.org/10.25378/janelia.23712957> under the CC-BY-NC 4.0 license.

Table 3: Number of training/validation/test frames utilized across tasks, with number of source videos in parentheses. Pretraining frames are unlabeled. Pose estimation and action segmentation frames are labeled; these models are trained across multiple videos. Neural encoding frames have matched neural activity (output) for each time point of behavior (input); these models are trained on single videos, since the neural populations change from one session to the next.

Pretraining	Pose estimation		Action segmentation		Neural encoding	
	train/val	test	train/val	test	train/val	test
IBL [29]	138,600 (77)	7,609 (128)	1,446 (19)	35,521 (18)	7,786 (5)	338,760 (5)
Mirror-mouse [72]	94,252 (17)	789 (17)	253 (5)	-	-	-
Mirror-fish [3]	47,921 (28)	373 (28)	94 (10)	-	-	-
CRIM13 [6]	99,914 (37)	3,986 (37)	1,274 (19)	-	-	-
CalMS21 [59]	103,544 (37)	-	-	506,668 (68)	262,107 (19)	-
Facemap [63]	-	-	-	-	-	1,790,200 (5)
						447,550 (5)

B BEAST implementation

BEAST utilizes a standard ViT-B/16 architecture [12], and combines a masked autoencoding and temporal contrastive learning loss. This approach is also taken by ViC-MAE [17], and we introduce key adaptations and simplifications to make BEAST suitable for applications in behavioral neuroscience, which we elaborate on more in the following sections.

B.1 Architecture

We selected the “base” ViT-B/16 architecture over other ViT variants for two reasons: it is expressive enough to capture rich frame-level information, while remaining computationally efficient for training and inference on long videos.

Our architecture differs from the standard ViT in its use of a nonlinear projector for the contrastive loss. While ViC-MAE employs a pooled attention layer to transform patch embeddings into a 768-dimensional vector for their contrastive loss, we take a different approach. We use the standard CLS token as our global image representation rather than pooled patch embeddings. This CLS token passes through a nonlinear projector with four components: a linear layer, Batch Norm (necessary for stable training, Fig. 5), ReLU activation, and a final linear layer.

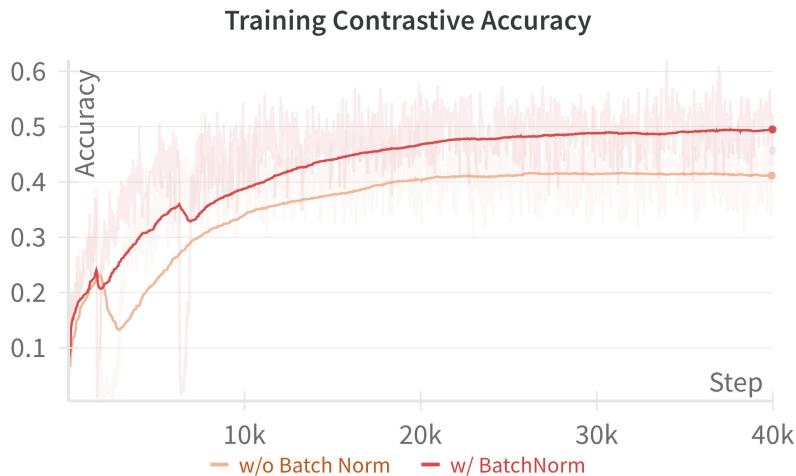


Figure 5: **Effect of Batch Normalization on contrastive training accuracy.** Training contrastive accuracy improves significantly with the use of Batch Normalization (BatchNorm) in the nonlinear projection head. Models trained with BatchNorm exhibit smoother learning curves and achieve higher final accuracy compared to those without BatchNorm. “Accuracy” is defined as the fraction of anchor frames in a batch where the corresponding positive frame has a logit score higher than that of all other negative frames.

B.2 Training

We discuss frame selection and sampling strategies, and data augmentations below. We apply the MAE loss uniformly across all frame types (anchor, positive, negative), whereas ViC-MAE only applies the MAE loss to anchor frames. The global batch size is set to 2048, distributed across 8 Nvidia A40 GPUs. We use the AdamW optimizer with a weight decay of 0.05. The learning rate is scheduled using PyTorch’s `OneCycleLR` scheduler, with a base learning rate of 5×10^{-5} . The maximum learning rate is computed as $\text{max_lr} = \text{base_lr} \times \frac{\text{global_batch_size}}{256}$, with `pct_start` set to 0.15 and `div_factor` set to 10. We train all models for 800 epochs.

Frame selection strategy Animal behavior videos often contain extended periods of inactivity or repetitive behaviors. Pretraining ViT models on all available frames would capture redundant information and increase computation time unnecessarily. Instead, we focus on extracting diverse frames that showcase distinct poses (to optimize the masked autoencoding loss) while preserving meaningful temporal relationships in local neighborhoods (to leverage the temporal contrastive loss). Our approach begins by downsampling all video frames to 32×32 pixels and calculating motion energy, defined as the absolute pixel-wise differences between consecutive frames. We eliminate frames in the bottom 50th percentile of motion energy, retaining only those with significant movement. We then apply k-means clustering to the remaining downsampled frames, with the number of clusters matching our target number of anchor frames per video (e.g., 600). For each cluster, we select the frame closest to the cluster center, along with its immediate predecessor and successor in time, which serve as positive examples for the contrastive loss (for a total of, e.g., 1800 frames per video). This methodology ensures a high-quality, diverse dataset for efficient pretraining. We find this method outperforms a random frame selection strategy in the neural encoding task (Table 4).

Table 4: Frame selection strategy ablation. We pretrain models using either a random frame selection strategy (“Random”) or the PCA+k-means strategy described above (“Selected”). We evaluate the representations of these models using zero-shot performance on the neural encoding task using the bits per spike (BPS) metric. We report the mean and standard deviation of BPS across five test sessions.

Method	IBL		IBL-whisker	
	TCN	RRR	TCN	RRR
ViT-M (Random)	0.311 ± 0.107	0.168 ± 0.091	0.319 ± 0.084	0.147 ± 0.026
BEAST (Random)	0.319 ± 0.104	0.176 ± 0.094	0.319 ± 0.081	0.138 ± 0.035
BEAST (Selected)	0.337 ± 0.103	0.177 ± 0.076	0.317 ± 0.083	0.138 ± 0.029

Frame sampling strategy during training Once we have a diverse set of training frames, we must construct batches during training. As stated in Sec. 3, for a batch of size B we randomly select $B/2$ anchor frames, which can originate from any and all videos. For BEAST, each anchor frame x_t^v is paired with a positive frame randomly selected from $x_{t \pm 1}^v$. All other frames serve as negative frames, including frames from the same video. Due to the frame selection strategy described above, even frames from the same video will be visually distinct and not interfere with the contrastive loss. This batch construction procedure is distinct from ViC-MAE, which allows any two frames from the same video to be a positive pair, while only frames from different videos are negative pairs. We find our approach outperforms the ViC-MAE approach in the neural encoding task (Table 5). This sampling strategy only applies to BEAST; the ViT-M models do not contain the contrastive loss, and we only train them with the anchor frames.

Table 5: Frame sampling strategy ablation. We pretrain models using either the ViC-MAE or BEAST frame sampling strategy; both models use the superior “Selected” frame selection strategy. We evaluate the representations of these models using zero-shot performance on the neural encoding task using the bits per spike (BPS) metric. We report the mean and standard deviation of BPS across five test sessions.

Method	IBL		IBL-whisker	
	TCN	RRR	TCN	RRR
ViC-MAE (IN+PT)	0.331 ± 0.103	0.141 ± 0.080	0.289 ± 0.055	0.127 ± 0.033
BEAST (IN+PT)	0.337 ± 0.103	0.177 ± 0.076	0.317 ± 0.083	0.138 ± 0.029

Data augmentation The default data augmentation procedure [15] applies a random resized crop to 244×244 pixels with a crop ratio between 0.2 and 1.0, followed by a random horizontal flip with probability 50%. We also explore an extended augmentation strategy that adds further random transformations: rotation up to 45 degrees and color jittering (brightness=0.4, contrast=0.4, saturation=0.4, hue=0.1), in addition to the crop and flip. We compare the performance of BEAST using both the default and the extended augmentation strategy. The additional augmentations achieve performance similar to the default setting (Table 6), so we use the default augmentation throughout the paper.

Table 6: Data augmentation ablation. We pretrain models using either default or extended data augmentations. We evaluate the representations of these models using zero-shot performance on the neural encoding task using the bits per spike (BPS) metric. We report the mean and standard deviation of BPS across five test sessions.

Method	IBL		IBL-whisker	
	TCN	RRR	TCN	RRR
BEAST (default data aug)	0.337 ± 0.103	0.177 ± 0.076	0.317 ± 0.083	0.138 ± 0.029
BEAST (extend data aug)	0.328 ± 0.102	0.163 ± 0.081	0.314 ± 0.077	0.150 ± 0.042

B.3 Hyperparameter details

The BEAST objective combines two losses: the reconstruction loss and the contrastive loss. During the early stages of training, it is important for the model to focus on accurately reconstructing the input, so the reconstruction loss should dominate. As training progresses and the model learns to capture low-level pixel structure, the contrastive loss gradually becomes more important. It acts as a regularizer, encouraging the model to learn higher-level temporal representations rather than overfitting to local pixel patterns.

The weighting factor for the contrastive loss, λ , plays a crucial role during pretraining; too large and the model will not reconstruct the input well; too small and the model does not reap its regularizing benefits. Through hyperparameter tuning based on neural encoding performance (on the validation set), we set $\lambda = 0.03$ for all datasets, except CRIM13, where we set $\lambda = 0.01$. This choice ensures the reconstruction loss is emphasized in the early phases of training, while the contrastive loss naturally takes over as reconstruction error decreases toward the end, eliminating the need for an annealing schedule.

The masking strategy for the reconstruction loss also significantly impacts performance. We found that an aggressive mask ratio of 0.75 works effectively across various tasks, for both ViT-M and BEAST. When fine-tuning BEAST for neural encoding models (IN+FT or IN+PT+FT), we tested mask ratios of 0.75 and 0.9, with 0.9 performing better on validation data. These 0.9-ratio models are used for the final fine-tuning results presented in our tables and figures.

C Neural encoding

C.1 Feature representations

Keypoints For the IBL dataset we use 11 keypoints in the publicly available dataset: left and right paws, two edges of the tongue, two edges of the lick spout, nose, and four edges of the pupil. For the Facemap dataset we use 12 keypoints in the publicly available dataset: three whiskers, four points on the nose, four corners of the eye, and the one visible paw.

Whisker pad motion energy We localize the whisker pad in the IBL dataset using anchor keypoints on the nose and eye. We then compute the motion energy of the whisker pad as the absolute pixel-wise differences between consecutive frames, resulting in a one-dimensional representation at each time point [30].

Principal Component Analysis We compute PCA on a per-session basis using all frames in the video. A subset of the resulting PCs are used for neural encoding. See Sec. C.6 for information on our dimensionality experiment.

CEBRA CEBRA [55] is a contrastive learning approach that provides a baseline which is complementary to the ViT-M models pretrained on ImageNet with a masked autoencoding objective. Similar to our PCA approach, we train an individual unsupervised CEBRA model for each session using the convolutional neural network option and the default offset10-model.

ViT-M variants For the ViT-M models, we extract the CLS embedding for each frame using a frozen pretrained backbone, and use these as input to the encoding models. Alternative approaches could include: (1) using patch embeddings with a multi-head attention pooling layer (as in our action segmentation work), or (2) fine-tuning the backbone itself while using either CLS or patch embeddings (similar to our approach for pose estimation). We expect these alternative approaches would improve performance and plan to explore them in future work.

- ViT-M (IN): ViT-M model pretrained on ImageNet using a masked autoencoding loss. We did not pretrain this model ourselves, but rather used the model checkpoint available at <https://huggingface.co/facebook/vit-mae-base>.
- ViT-M (IN+PT): Initialized with the ImageNet-pretrained weights, then further pretrained on dataset-specific frames.
- ViT-M (IN+PT+FT): Initialized with the dataset-specific pretrained weights (IN+PT), and then further fine-tuned on a single session.

All training (except ViT-M (IN)) is performed as described in Appendix B.

BEAST variants The BEAST model variants follow the same naming pattern as ViT-M, with one exception: there is no “BEAST (IN)” variant, as BEAST requires video frames rather than just static images for pretraining.

C.2 Reduced rank regression

IBL For the IBL dataset, we followed the Reduced Rank Regression (RRR) setup described in [51]. We trained all models using the L-BFGS optimizer and set the rank constraint to 3. To denoise the neural signals, we applied a 1-dimensional smoothing filter to the neural activity. The hyperparameter search (Sec. C.4) was conducted over the ranges specified in Table 7.

Table 7: RRR model hyperparameters for IBL dataset.

Hyperparameter	Value Range
Output Dim	Number of Neurons
Rank	3
Optimizer	L-BFGS
Learning Rate	Log-Uniform(0.1, 2)
L2	100

Facemap For the Facemap dataset, we followed the setup described in [63], using the implementation provided in the official Facemap repository. We adopted the default rank of 32 as used in the original implementation. The Lambda parameter refers to the regularization strength, which we set to a relatively low value to avoid over-penalizing the weights. The output dimensionality was set to 128, corresponding to the number of neural principal components used in the model. Model parameters are estimated via a closed-form least squares approach. The hyperparameters we used are specified in Table 8.

Table 8: RRR model hyperparameters for Facemap dataset.

Hyperparameter	Value Range
Output Dim	128
Rank	32
Lambda	1e-6

C.3 Temporal convolution network

We used the same implementation of the Temporal Convolution Network (TCN) to process frame embeddings for both the IBL and Facemap datasets, based on the official Facemap repository. The convolutional kernel operates along the temporal dimension of the input data. The TCN model was trained for 300 epochs using the AdamW optimizer, with learning rate decimation (multiplied by 0.1) applied at epochs 120 and 200. The hyperparameter search (Sec. C.4) was conducted over the ranges specified in Table 9.

Table 9: TCN model hyperparameters; *IBL dataset; **Facemap dataset

Hyperparameter	Value Range
Output Dim	Number of Neurons*, 128**
Learning Rate	Log-Uniform(5e-5, 2e-3)
Optimizer	AdamW
Weight Decay	1e-4

C.4 Hyperparameter selection

For the IBL data, we divided the trials into train (80%), validation (10%), and test (10%) sets. For the Facemap data we followed the experimental setup as described in the original study [63]: the session is split into ten blocks; the first 75% of each block is assigned to the training set; the following 3 seconds are excluded to remove data leakage due to autocorrelation in behavior and neural activity; and the final set of frames from the block are assigned to the test set. There is no validation set. For both the RRR and TCN models, we conducted hyperparameter searches separately for each feature type to identify the best-performing configurations. Specifically, we performed 30 runs of randomly selected hyperparameters per model type, evaluating performance on the validation set (IBL) or test set (Facemap) using an evaluation metric specific to each dataset: bits per spike (BPS) for IBL and variance explained for Facemap. The only exception was the RRR model for Facemap, where the parameters were fixed according to the original implementation. We select the model with the best performance on the validation (IBL) or test (Facemap) set, and report results on the test set.

C.5 Patch embeddings

The output of the ViT encoder consists of a CLS token embedding $z_{\text{CLS}} \in \mathbb{R}^D$ and patch token embeddings $z \in \mathbb{R}^{N \times D}$, where $D = 768$ is the embedding dimension and N is the total number of image patches ($N = 196$ for a 224×224 input frame). We compare the CLS token and patch embeddings as inputs for neural encoding (see implementation details in E.4), and find that the CLS token outperforms the patch embeddings (Table 10). Given its lower dimensionality and superior results, we adopt the CLS token representation for all subsequent encoding tasks.

Table 10: Comparison of CLS and patch embeddings with a TCN encoder.

Method	IBL	IBL-whisker
BEAST (CLS)	0.337 ± 0.103	0.317 ± 0.083
BEAST (Patch)	0.278 ± 0.065	0.288 ± 0.070

C.6 Dimensionality ablation experiments

One of the most important hyperparameters for the PCA, CEBRA, and ViT models is the latent/embedding dimensionality. To thoroughly explore performance across this parameter, we tested these models using various dimensionality values. For a given dimensionality k , we used different approaches: (1) for PCA, we selected the top k principal components; (2) for CEBRA, we retrained the model with k latent dimensions; and (3) for ViT models, due to computational constraints, we first trained the full 768-dimensional models, then applied PCA to the embedding space and selected the top k ViT principal components. For each feature, model type, and dimensionality k , we fit downstream neural encoding models using the complete hyperparameter search described previously. Figure 2 reports the best result for each model, though notably BEAST outperforms all baselines across all dimensionality values (Fig. 6).

C.7 Extended neural encoding results

We collect all neural encoding results in Table 11. The values for PCA and CEBRA correspond to the 100-dimensional results in Fig. 6; the values for ViT-M and BEAST correspond to the full 768-dimensional models.

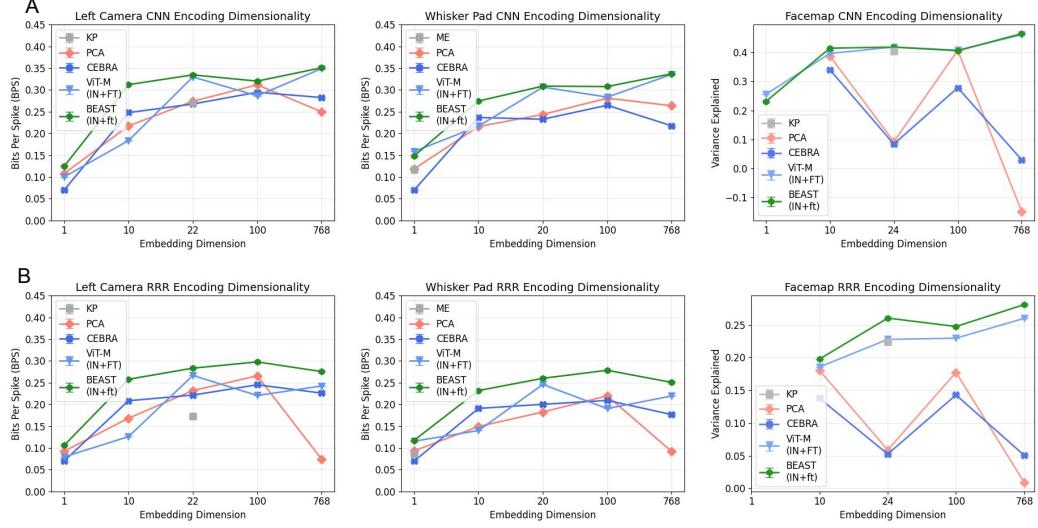


Figure 6: Encoding performance as a function of embedding dimension. BEAST outperforms all other baselines over embedding dimensions spanning several orders of magnitude, demonstrating the superiority of its representations for any given dimensionality. Results for Keypoints and Motion Energy are included at their respective dimensionalities for each dataset. The Facemap encoding results for 1-dimensional data performed poorly and included NaN values in some sessions, so we excluded them from the figure.

Table 11: Neural encoding results across feature types and models. All ViT-based models use a frozen backbone. “IN” refers to a model pretrained with ImageNet weights; “IN+PT” refers to models that are initialized with ImageNet-pretrained weights then further pretrained on experiment-specific data; “+FT” refers to models that are initialized with pretrained weights based on what comes before “+” then fine-tuned on individual sessions.

Features	IBL		IBL-whisker		Facemap	
	RRR	TCN	RRR	TCN	RRR	TCN
Keypoints	0.173 ± 0.029	0.271 ± 0.054	—	—	0.224 ± 0.047	0.403 ± 0.077
Motion energy	—	—	0.087 ± 0.023	0.117 ± 0.028	—	—
PCA	0.266 ± 0.054	0.312 ± 0.078	0.220 ± 0.038	0.281 ± 0.065	0.177 ± 0.064	0.407 ± 0.090
CEBRA	0.245 ± 0.036	0.295 ± 0.049	0.209 ± 0.265	0.265 ± 0.034	0.143 ± 0.046	0.278 ± 0.064
ViT-M (IN)	0.201 ± 0.070	0.325 ± 0.091	0.142 ± 0.051	0.307 ± 0.068	0.254 ± 0.061	$0.446 \pm 0.100s$
ViT-M (IN+PT)	0.182 ± 0.071	0.334 ± 0.098	0.156 ± 0.032	0.316 ± 0.073	—	—
ViT-M (IN+FT)	0.242 ± 0.089	0.349 ± 0.106	0.219 ± 0.048	0.336 ± 0.075	0.260 ± 0.051	0.461 ± 0.099
ViT-M (IN+PT+FT)	0.293 ± 0.082	0.351 ± 0.106	0.244 ± 0.042	0.335 ± 0.092	—	—
BEAST (IN+PT)	0.277 ± 0.076	0.337 ± 0.103	0.138 ± 0.029	0.317 ± 0.083	—	—
BEAST (IN+FT)	0.276 ± 0.088	0.351 ± 0.106	0.251 ± 0.051	0.337 ± 0.088	0.281 ± 0.054	0.464 ± 0.089
BEAST (IN+PT+FT)	0.291 ± 0.087	0.352 ± 0.106	0.243 ± 0.048	0.335 ± 0.079	—	—

D Pose estimation

D.1 Models

The pose estimation models consist of a backbone and a head. The backbone is either a ResNet-50 [16] or a ViT-B/16 [12], both producing feature maps of shape $[N, H, W]$ for a given image, where N denotes the feature dimension and H, W denote the height and width of the feature maps. All models employ an identical linear upsampling head that begins with a PixelShuffle layer, reshaping the feature maps to $[N/4, 2H, 2W]$. These reshaped features then pass through two consecutive 2D convolutional transpose layers with kernel size $(3, 3)$ and stride $(2, 2)$, doubling the spatial resolution after each layer. The head architecture omits batch normalization and nonlinearities between these layers. The output passes through a 2D softmax function, generating a normalized heatmap for each keypoint.

D.2 Training

We divided the labeled data into training (95%) and validation (5%) sets, with test frames coming from entirely held-out videos. We used a batch size of eight frames. Data augmentations include random crops, rotations, motion blur and histogram equalization. Models were trained for 300 epochs, with validation loss recorded every five epochs. For final evaluation, we selected the model with the lowest validation loss. Training utilized an Adam optimizer [24] with an initial learning rate of 0.001, which was halved at epochs 150, 200, and 250. To facilitate feature learning, we kept the backbone frozen during the first 20 epochs of training before allowing for full end-to-end optimization. The loss function is the mean square error between each predicted heatmap and a ground truth heatmap constructed from labeled data.

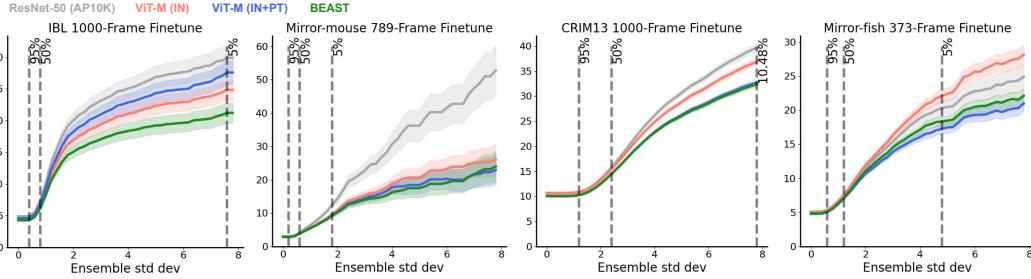


Figure 7: Pose estimation performance with more training frames. The results in Fig. 3 demonstrate pose estimation performance of various models using just 100 labeled frames. To ensure those comparisons hold on larger datasets we trained models (three random seeds per backbone) using the maximum number of frames in the training dataset or 1000 frames, whichever is smaller. We still see consistent gains for both the transformer architecture pretrained on ImageNet (red) and our pretraining strategy (blue, green).

E Action segmentation

For action segmentation we consider a variety of input feature types and modeling approaches. We first consider a range input features: keypoints, PCA on the raw video frames (fit on the same frames as ViT pretraining; we use 768 PCs for downstream models), and ViT-based CLS tokens. For each of these feature types we fit both linear and nonlinear (temporal convolution network, TCN) models. We also train a TCN model on the ViT patch embeddings, with a multi-head attention pooling layer to reduce the dimensionality of the features before entering the TCN.

E.1 Models

Linear model The linear action segmentation model uses a 1D temporal convolution layer, followed by a linear layer that maps from the number of features the the number of action classes, followed by a softmax. There are no other forms of nonlinearity in the model.

Temporal convolution network The nonlinear action segmentation model is a dilated TCN [32] with 2 dilation blocks. Each dilation block consists of a sequence of 2 sub-blocks (1D convolution layer → leaky ReLU nonlinearity → dropout with probability=0.10), as well as a residual connection between the input and output of the dilation block. The dilation of the convolutional filters starts with 1 for the first dilation block, then increases by a factor of 2 for each additional dilation block. This results in a larger temporal receptive field as the model gets deeper, allowing for learning of longer range dependencies [78].

Both models utilize a weighted cross entropy loss function, with class weights inversely proportional to the class frequency in the training data.

E.2 Training

Each video is split into sequences of 500 (IBL) or 100 (CalMS21) frames. We divide the data into training (90%) and validation (10%) sets, with test frames coming from entirely held-out videos. We use a batch size of 16 sequences. Models were trained for 500 epochs using the Adam optimizer [24].

E.3 Hyperparameter details

For each model type—linear and nonlinear—and each feature type, we run a hyperparameter search across all combinations of parameters in Table 12 using three random weight initializations. The hyperparameter combination with the best F1 score on the validation data, averaged across the three seeds, is selected for evaluation on the test set. For this hyperparameter combination, we train with two additional seeds and report results in the figures and tables averaged across five seeds.

Table 12: Action segmentation hyperparameters. *TCN only

Hyperparameter	Value Range
Learning rate	1e-3, 1e-4, 1e-5
Dropout	0.1
Temporal filter length	9, 17, 33
Number of hidden units*	16, 32, 64, 96
Number of hidden layers*	2

E.4 Temporal convolution network with multi-head attention pooling

We aggregate the patch embeddings from the BEAST encoder using a multi-head attention pooling layer [33], which produces a single pooled embedding per frame as input to the TCN. This layer uses a learnable query $S \in \mathbb{R}^{1 \times D}$ with patch embeddings $z \in \mathbb{R}^{N \times D}$ as keys and values. To capture motion-related features, we further concatenate the frame-to-frame difference of the pooled embeddings as additional input to the TCN (Fig. 8). We fixed the number of attention heads in the pooling layer to 8 for all models.

Our previous model utilized CLS embeddings, which allowed for an efficient workflow: we processed videos through the transformer backbone and saved the CLS embedding from each frame as a separate file. These pre-computed embeddings could then be directly loaded to train the downstream TCN classifier without requiring video reading during training. However, patch embeddings present significantly larger memory requirements, making disk storage infeasible. To address this challenge, we developed a data loading pipeline that performs end-to-end processing: it loads video frames, passes them through the transformer backbone, and feeds the resulting patch embeddings directly to the TCN model within the same training loop. This integrated approach, while computationally more intensive, eliminates the need for intermediate storage. Due to these increased computational demands, we modified the training procedure for the multi-head attention pooling models as follows.

The pooling layer and TCN classifier were trained jointly for 200 epochs on CalMS21 and 100 epochs on IBL using the Adam optimizer [24] with an initial learning rate of $1e-3$. The epoch counts were determined through a separate experiment that withheld a subset of validation videos and monitored the validation F1 score until convergence. Training followed a cosine-annealing schedule with warm restarts [36] configured with $T_0 = 34$, $T_{mult} = 2$ and $\eta_{min} = 5e-5$. We used 6 or 8 NVIDIA A40 GPUs for the training, each with a batch size of 2, giving an effective batch size of 12 or 16. The sequence length was fixed at 500. Due to the increased compute required to train these models, we fixed the TCN hyperparameters to be those found for the CLS-based model for each dataset.

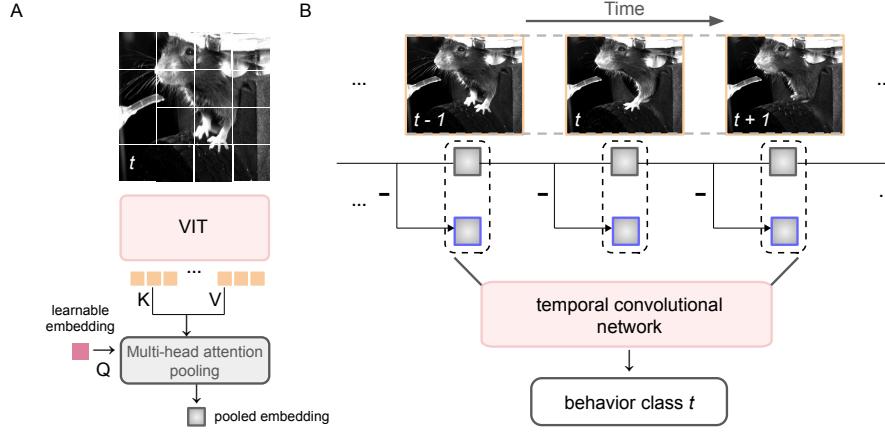


Figure 8: Multi-head attention pooling TCN head for action segmentation. **A:** Per-patch embeddings from the BEAST encoder are pooled using a multi-head attention layer, where a learnable query token attends to the patch embeddings to produce a single pooled embedding for each frame. **B:** Temporal differences between consecutive pooled embeddings are concatenated as additional input to the TCN, which predicts the behavior class of the center frame in the sliding window. A window size of 3 is shown for illustration; the actual window size was tuned as a hyperparameter.

Table 13: Action segmentation results on IBL and CalMS21 datasets across feature types and models. All ViT-based models use frozen a frozen backbone; “CLS” indicates models trained on the global CLS embeddings, while “patch” indicates models trained with a multi-head attention pooling layer applied to the patch embeddings. “IN” refers to a model pretrained with ImageNet weights, “IN+PT” refers to models that are initialized with ImageNet-pretrained weights then further pretrained on experiment-specific data.

Dataset	Features	Linear		TCN	
		Features	Features, Δ Features	Features	Features, Δ Features
IBL	Keypoints	$0.54 \pm 1.4e-3$	$0.55 \pm 1.5e-3$	$0.86 \pm 1.4e-3$	$0.88 \pm 2.2e-3$
	PCA	$0.54 \pm 4.6e-3$	$0.55 \pm 6.3e-3$	$0.64 \pm 1.0e-2$	$0.71 \pm 2.8e-3$
	VIT-M (IN) (CLS)	$0.68 \pm 1.4e-3$	$0.68 \pm 7.0e-4$	$0.78 \pm 7.4e-3$	$0.79 \pm 2.7e-3$
	VIT-M (IN+PT) (CLS)	$0.74 \pm 2.5e-3$	$0.72 \pm 2.0e-3$	$0.78 \pm 4.0e-3$	$0.80 \pm 4.6e-3$
	BEAST (IN+PT) (CLS)	$0.70 \pm 7.9e-3$	$0.69 \pm 2.5e-3$	$0.80 \pm 3.0e-3$	$0.81 \pm 6.9e-4$
	VIT-M (IN) (patch)	-	-	-	$0.84 \pm 3.7e-3$
	VIT-M (IN+PT) (patch)	-	-	-	$0.85 \pm 3.6e-3$
	BEAST (IN+PT) (patch)	-	-	-	$0.87 \pm 5.1e-3$
CalMS21	SimBA [14]	$0.75 \pm 5.1e-4$	$0.53 \pm 8.1e-3$	$0.78 \pm 3.8e-3$	$0.79 \pm 2.9e-3$
	TREBA [60]	$0.29 \pm 1.4e-3$	$0.30 \pm 1.5e-3$	$0.70 \pm 4.6e-3$	$0.72 \pm 7.4e-3$
	PCA	$0.10 \pm 3.1e-3$	$0.10 \pm 3.2e-3$	$0.16 \pm 5.1e-3$	$0.18 \pm 4.5e-3$
	VIT-M (IN) (CLS)	$0.50 \pm 2.2e-2$	$0.52 \pm 1.3e-3$	$0.53 \pm 1.2e-2$	$0.60 \pm 2.8e-3$
	VIT-M (IN+PT) (CLS)	$0.60 \pm 5.5e-3$	$0.60 \pm 1.1e-3$	$0.60 \pm 9.1e-3$	$0.65 \pm 2.2e-3$
	BEAST (IN+PT) (CLS)	$0.53 \pm 4.6e-3$	$0.51 \pm 4.0e-2$	$0.58 \pm 5.2e-3$	$0.63 \pm 2.7e-3$
	VIT-M (IN) (patch)	-	-	-	$0.74 \pm 2.9e-3$
	VIT-M (IN+PT) (patch)	-	-	-	$0.82 \pm 9.5e-3$
	BEAST (IN+PT) (patch)	-	-	-	$0.81 \pm 7.7e-3$

F Broader Impacts

The BEAST framework enables more efficient extraction of meaningful information from video data, potentially accelerating behavioral neuroscience research with several beneficial outcomes. By reducing the need for extensive human labeling while improving accuracy, BEAST can democratize advanced video analysis capabilities for laboratories with limited resources. This efficiency could accelerate basic science discoveries that underlie advances in biomedical applications, neurological disorder treatments, and improved understanding of brain function.

While BEAST is developed primarily for behavioral neuroscience studies using animal subjects, the underlying technology could potentially be repurposed for human video analysis, raising several concerns:

- Surveillance capabilities: The improved ability to track and categorize behaviors could enhance surveillance technologies, potentially infringing on privacy rights if deployed without appropriate oversight.
- Bias and fairness: As with any AI system trained on specific datasets, BEAST-derived models may perform differently across demographic groups if applied to human subjects, potentially perpetuating biases in downstream applications.
- Resource inequality: While a pretrained BEAST model can improve the efficiency of downstream tasks, the computational requirements for pretraining itself may limit access to this technology for under-resourced institutions, potentially widening existing disparities in research capabilities.