OpenLabCluster: Active Learning Based Clustering and Classification of Animal Behaviors in Videos Based on Automatically Extracted Kinematic Body Keypoints

Jingyuan Li$^1$  Moishe Keselman$^2$  Eli Shlizerman$^{1,2}$

$^1$Department of Electrical and Computer Engineering, University of Washington, Seattle, WA

$^2$Department of Applied Mathematics, University of Washington, Seattle, WA

Abstract

Quantifying natural behavior from video recordings is a key component in ethological studies. Markerless pose estimation methods have provided an important step toward that goal by automatically inferring kinematic body keypoints. The next step in behavior quantification is utilization of these features toward organizing and interpreting behavioral segments into states. In this work, we introduce a novel deep learning toolset to address this aim. In particular, we introduce OpenLabCluster which clusters segments into groups according to the similarity of kinematic body keypoints and then employs active learning approach which refines the clusters and classifies them into behavioral states. The active learning approach is an iterative semi-supervised deep learning methodology selecting representative examples of segments to be annotated such that the annotation informs clustering and classification of all segments. With these methodologies, OpenLabCluster contributes to faster and more accurate organization of behavioral segments with only a sparse number of them being annotated. We demonstrate OpenLabCluster performance on four different datasets, which include different animal species exhibiting natural behaviors, and show that it boosts clustering and classification compared to existing methods, even when all segments have been annotated. OpenLabCluster has been developed as an open-source interactive graphic interface which includes all necessary functions to perform clustering and classification, informs the scientist of the outcomes in each step, and incorporates the choices made by the scientist in further steps.

1 Introduction

Analysis and interpretation of animal behavior are essential for a multitude of biological investigations. Behavioral studies extend from ethological studies to behavioral essays as a means to investigate biological mechanisms \[1\,2\,3\,4\,5\,6\,7\]. In these studies, methodologies facilitating robust, uninterrupted, and high-resolution observations are key. Indeed, researchers have been recording animal behaviors for decades with various modalities, such as video, sound, placement of physical markers, and more \[8\,9\,10\,11\,12\,13\,14\]. Recent enhancements in recording technologies have extended the ability for the deployment of recording devices in various environments and for extended periods. The enhancement in the ability to perform longer observations and in the number of modalities, brings forward the need to organize, interpret and associate recordings with an identified repertoire of behaviors, i.e., perform classification of the recordings into behavioral states. Performing these operations manually would typically consume a significant amount of time and would require expertise. For many recordings, manual behavior classification becomes an unattainable task. It is therefore critical to develop methodologies to accelerate classification of behavioral states and require as little involvement from the empiricist as possible \[15\,16\,17\].

For video recordings, deep-learning-based methods classifying behaviors directly from the video frames (image-based) have been proposed \[18\,19\,20\,21\]. While such methods could be effective for specific scenarios, they are prone either lower accuracy. This is due to video footage including noise (e.g. camera artifacts) and extra information unrelated to behavior (e.g. background). To address
these challenges, the methods have been extended to include image transformation techniques during training or have been accompanied with additional modalities, such as audio [22, 23, 14]. However, for general videos, background interference still presents itself as a significant challenge leading to non-robust performance. In addition, image-based methods are computationally expensive since operate with a high-dimensional input of the full video frame or a sequence of frames, and include multiple computational layers to narrow down this information to behavioral classification [21]. Furthermore, one of the most critical drawbacks of direct image-based approaches is that they are typically not generalizable across animal species and experimental setups.

In contrast, methods that focus on movements, utilizing body keypoints or kinematics extracted from video frames, could avoid these challenges [24, 25]. Automatic detection of body keypoints, markerless pose estimation, is especially advantageous since there is no need for physical markers. Indeed, several open-source packages for markerless pose estimation with an admissible accuracy for general videos are readily available. Popular examples include OpenPose [26], an open-source package which performs human pose estimation, DeepLabCut [27], an open-source package for estimation of animal keypoints. DeepLabCut includes an adaption of a pre-trained model to the particular video and keypoints which allows it to perform animal pose estimation across various species. Extensions and improvements of DeepLabCut, such as DeepLabCut 2+, enabling 3D pose estimation, better accuracy, and multi-animal pose estimation have been recently made available and in further development [28, 29]. Furthermore, additional approaches which address animal pose estimation have been introduced and are further developed [30, 31, 32, 33, 34, 35, 36].

Organization of such body keypoints time series into behavioral classes consists of two complementary tasks. The first task is clustering, in which recorded segments are mapped into groups according to the similarity of behavior that the segments represent. In such a map, segments that represent similar behavioral states would be located close to each other while segments that represent distinct behavioral states would be mapped farther from each other [2, 37]. Algorithms for clustering are typically ‘unsupervised’, i.e., they do not rely on the annotation of the segments. Clustering algorithms range from classical approaches such as KMeans to deep-learning-based approaches, such as...
Predict&Cluster \cite{39} or variational deep-learning approaches designed particularly for animal motion embeddings, such as VAME \cite{40}. Beyond clustering, it is desired to obtain semantic meaning and associate a label of a state with each behavioral segment. Such a task is the classification task and is performed by generating probability distribution for each segment with respect to the behavioral states \cite{24,25}. Classification algorithms are typically ‘supervised’ since these algorithms require annotation, i.e., the association of behavioral labels (states) to segments, to be used in training. The accuracy of classification depends on the quality of the annotation and the number of annotated samples. Supervised classification approaches range from classical approaches such as KNN to deep-learning-based approaches \cite{41,42,43}. In general, better classification accuracy would be obtained when more labels are provided. However, in practice, annotation is a time-consuming process and prone to errors since the quality and the interpretation will vary between annotators.

While classification does not require clustering, successful clustering can enhance the classification accuracy and reduce dependence on annotation. Indeed, in the case of a prior grouped segments into well-separated clusters, annotation of a single representative in each cluster would suffice to classify all segments. This is the guiding principle of ‘semi-supervised’ methods which employ both unsupervised and supervised approaches to minimize the annotation efforts while at the same time not undermining accuracy \cite{44}. Semi-supervised methods propose to perform unsupervised pre-training as a way to transfer input data into latent representations which are used in conjunction with selecting a subset of segments for annotation to classify all segments \cite{43}. In practice, the selection of segments can significantly boost or diminish performance, and approaches minimizing the number of annotations and maximizing accuracy through systematic selection are warranted. Approaches for such selection are known as active learning methods and multiple works in various classification applications have shown that active learning can be applied to achieve such balance between accuracy and annotation \cite{46,47,48,49}.

In this work, inspired by DeepLabCut and associated projects in computational ethology, which provide a toolbox along with graphic interface to deep-learning approach applied to animal videos, we extend the semi-supervised methodology to animal behavior classification. In particular, we developed the OpenLabCluster toolset, a semi-supervised platform embedded in a graphic interface for animal behavior classification from body keypoints. The system implements and allows to work with multiple semi-supervised active learning methods. Active learning is performed in an iterative way, where, in each iteration, an automatic selection of a subset of candidate segments is chosen for annotation, which in turn enhance the accuracy of clustering and classification. OpenLabCluster is composed of two components illustrated in Fig. 1A: (1) Unsupervised deep encoder-decoder clustering of action segments generating low-dimensional Cluster Maps which depict the segments as points and show their groupings. (2) Iterative automatic selection of segments for annotation and subsequent generation of Behavior Classification Maps. In each iteration, each point in the Cluster Map is being re-positioned and associated with a behavioral class (colored with a color that corresponds to a particular class). These operations are performed through training of a clustering encoder-decoder (component (1)) along with a deep classifier (component (2)). OpenLabCluster implements these methodologies as an open-source graphical user interface (GUI) to empower scientists with little or no deep-learning expertise to perform animal behavior classification. In addition, OpenLabCluster includes advanced options for experts.

2 Results

Datasets. Behavioral states and their dynamics vary from species to species and from recordings to recordings. We use four different datasets to demonstrate OpenLabCluster applicability to various settings. The datasets include videos of behaviors of four different animal species (Mouse \cite{18}, Zebrafish \cite{2}, C. elegans \cite{50}, Monkey \cite{32}) with three types of motion features (body keypoints, kinematics, segments), as depicted in Fig. 2. Two of the datasets include a-priori annotated behavioral states (ground truth) \cite{Mouse, C. elegans}, while the Zebrafish dataset includes ground truth a-priori predicted by another method, and the Monkey dataset does not include ground truth annotations. Three of the datasets have been segmented into single-action clips (Mouse, Zebrafish, C. elegans), while the Monkey dataset is a continuous recording that requires segmentation into clips. We describe further details about each
Figure 2: Visualization of four animal behavior datasets. (A) Home-Cage mouse dataset (Mouse) (B) *C. elegans* Movement Dataset (*C. elegans*) (C) Zebrafish free swimming dataset (Zebrafish) (D) OpenMonkeyStudio Macaque behaviors dataset (Monkey). The top row shows positions of extracted keypoints for each dataset.

dataset below.

**Home-Cage Mouse.** The dataset includes video segments of 8 identified behavioral states [18]. In particular, it contains videos recorded by front cage cameras when the mouse is moving freely and exhibits natural behaviors, such as drinking, eating, grooming, hanging, micromovement, rearing, walking, resting. Since keypoints have not been provided in this dataset, we use DeepLabCut [27] to automatically mark and track eight body joint keypoints (snout, left-forelimb, right-forelimb, left-hindlimb, right-hindlimb, fore-body, hind-body, and tail) in all recorded videos frames. An example of estimated keypoints overlaid on top of the corresponding video frame is shown in Fig. 2A (top). To reduce the noise that could be induced by the pose estimation procedure, we only use the segments for which DeepLabCut estimation confidence is high enough. We use 8 sessions for training the models of clustering and classification (2856 segments) and test classification accuracy on 4 other sessions (393 segments).

**Zebrafish.** The dataset includes video footage of zebrafish movements and was utilized in [2] for unsupervised behavior clustering using 101 precomputed kinematic features, a procedure that identified 13 clusters which were manually related to 13 behavior prototypes (see Appendix 5.2). In the application of OpenLabCluster to this dataset, we utilize only a small subset of these features (16 features) and examine whether OpenLabCluster is able to generate classes aligned with the unsupervised clustering results obtained on full 101 features (as the ground truth). We use 5294 segments for training and 2781 segments for testing.

**C. elegans.** The dataset is recorded with Worm Tracker 2.0 when the worm is freely moving. The body contour is identified automatically using contrast to background from which kinematic features are calculated and constitute 98 features that correspond to body segments from head to tail in 2D coordinates, see [50] and Fig. 2C. Behavioral states are divided into three classes: moving forward, moving backward, and staying stationary. We use 10 sessions (a subset) to investigate the application of OpenLabCluster to this dataset, where the first 7 sessions (543 segments) are used for training and the remaining 3 sessions (196 segments) are used for testing.

**Monkey.** This dataset is from OpenMonkeyStudio repository [32] and captures freely moving macaques in a large unconstrained environment using 64 cameras encircling an open enclosure. 3D keypoints positions are reconstructed from 2D images by applying deep neural network reconstruction algorithms on the multi-view images. Among the movements, 6 behavioral classes have been identified. In contrast to other datasets, this dataset consists of continuous recordings without segmentation into action clips. We thereby segment the videos by clipping them into fixed duration clips (10 frames with 30 fps rate) which results in 919 segments, where each segment is \( \approx 0.33 \) seconds long. OpenLabCluster receives
the 3D body key points of each segment as an input. Notably, a more advanced technology could be implemented to segment the videos as described in [51]. Here, we focused on examining the ability of OpenLabCluster to work with segments that have not been pre-analyzed and thus used the simplest and most direct segmentation method.

<table>
<thead>
<tr>
<th>Home-Cage Mouse Behavior (8 classes; Keypoints)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Labels (%)</strong></td>
</tr>
<tr>
<td><strong>Labels (#)</strong></td>
</tr>
<tr>
<td>KNN</td>
</tr>
<tr>
<td>SVM</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>OLC</td>
</tr>
<tr>
<td>CS</td>
</tr>
<tr>
<td>Top</td>
</tr>
</tbody>
</table>

Table 1: Classification accuracy of Home-Cage Mouse behaviors for increasing number of annotated segments (reported as percentage (%)). Top: Accuracy of standard classification methods KNN, SVM, and C. Bottom: Accuracy of OpenLabCluster (OLC) using active learning approaches: CS, Top, and MI. Best accuracy is highlighted in boldface and the difference between it and best standard method is shown.

**Evaluation Metrics.** We evaluate the accuracy of OpenLabCluster by computing the percentage of segments in the test set that OpenLabCluster correctly associated with the states given as ground truth, such that 100% accuracy will indicate that OpenLabCluster correctly classified all segments in the test set. Since OpenLabCluster implements the semi-supervised approach to minimize the number of annotations, we compute the accuracy given annotation budgets of overall 5%, 10%, 20% labels to be used over the possible iterations in conjunction with active learning. In particular, we test the accuracy when the Top, CS, and MI active learning methods implemented in OpenLabCluster are used for selections of segments to annotate. In addition, we implement non-active learning methods such as K-Nearest Neighbour (KNN) [41], support vector machine (SVM) [52], Deep Classifier containing the encoder and the classifier components of OpenLabCluster without the decoder (C). The accuracy of these methods is used to OpenLabCluster against them.

<table>
<thead>
<tr>
<th>Zebrafish (13 classes; Kinematics)</th>
<th>C. elegans (3 classes; Segments)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Labels (%)</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Labels (#)</strong></td>
<td>265</td>
</tr>
<tr>
<td>C</td>
<td>72.5</td>
</tr>
<tr>
<td>OLC</td>
<td></td>
</tr>
<tr>
<td>CS</td>
<td>71.9</td>
</tr>
<tr>
<td>Top</td>
<td>72.0</td>
</tr>
<tr>
<td>MI</td>
<td><strong>74.2</strong> (+1.7)</td>
</tr>
</tbody>
</table>

Table 2: Classification accuracy of Zebrafish and C. elegans behaviors for increasing number of annotated segments (reported as percentage (%)). Top: Accuracy of standard classifier network, C. Bottom: Accuracy of OpenLabCluster (OLC) using various active learning approaches: CS, Top, and MI. Best accuracy is highlighted in boldface and the difference between it and best standard method is shown.

**Outcomes.** The results of evaluation are shown in Tables 1 and 2 and further analysed in Fig. 3 and Fig. 4. We summarize the main outcomes of the evaluation and their interpretation below.

**Accuracy of Classification.** We observe that the accuracy of classification of OpenLabCluster across datasets is consistently higher than standard supervised classification methods for almost any budget of annotation. Specifically, for the Home-Cage Mouse Behavior dataset Table 1, OpenLabCluster (OLC) achieves accuracy of 65.3% when just 143 (5% of 2856) segments have been annotated.
improves along with the increase in the number of annotated segments, i.e., accuracy is 76.3% when 10% of segments are annotated, and 82.2% when 20% of segments are annotated. Overall, when compared to supervised approaches, the increase in accuracy is of ≈ 16% on average. We observe that among different active learning methods, CS and MI, achieve comparable accuracy and both exceed the Top selection. Notably, while active learning is expected to be especially effective in sparse annotation scenarios, when all segments are annotated (fully supervised scenario) the accuracy of the OpenLabCluster MI approach exceeds supervised classification approaches by 12.7% (rightmost column in Table 1). This reflects the effectiveness of the targeted selection of candidates for annotation and the use of clustering latent representation to enhance the overall organization of the segments.

For Zebrafish and C. elegans datasets, OpenLabCluster evaluation on sparse annotation scenarios (5% - 20%) exceeds the accuracy of a standard classifier. The improvements in accuracy are in the range of 0.3% – 9.1% and are not as extreme as for the Mouse dataset. These could be associated with not having manually identified ground truth behavior states for Zebrafish and having only 3 classes for the C. elegans dataset. Having such a small number of classes is relatively a simpler semantic task that does not challenge classifiers. We can indeed observe that when all annotations are considered in C. elegans dataset, all approaches perform well (above 92%) and a standard classifier achieves the best accuracy.

The Amount of Required Annotations. Since accuracy varies across datasets and depends on the number of classes and other aspects, we examine the relationship between accuracy and the number of required annotations. In Fig. 3A, we compute the necessary number of annotations required to achieve 80% of classification accuracy with OpenLabCluster vs. standard deep-classifier (C) on the Mouse dataset. We observe that active learning methods require only 15-20% of annotated segments while standard deep classifier requires 9 times more annotated segments than the most optimal active learning approach. Classical non-deep classifiers like SVM and KNN require even more annotations to achieve the same accuracy. We also visualize the relationship between the number of annotated samples and the accuracy that OpenLabCluster can achieve for the three datasets in Fig. 3B. We observe that for most cases, OpenLabCluster methods lead to higher accuracy for given number of annotations than the counterpart supervised standard classifier. The curves indicating the accuracy of various OpenLabCluster active
Figure 4: 2D tSNE projection of behavioral segments. Left, Top: 2D tSNE projection of keypoints compared with 2D tSNE projection of Latent Representation (Cluster Map). Left, Bottom: CHI and DBI metrics computed for each projection for increasing number of clusters. Right: Behavior Classification Map (2D tSNE projection of Latent Representation with associated behavioral states colors) along with example video frames from segments associated with each class.

learning methods (red, green, blue) have a clear gap between them and the supervised curve (gray), especially in mid-range of the number of annotations. Their relative placement to each other shows that all methods could potentially contribute to effective classification. In Fig. 3C we further examine class-wise confusion matrices for the Zebrafish dataset on 4 annotation budgets (5%, 10%, 20%, 100%). From visual inspection, it appears that the matrix that corresponds to 20% annotations is close to the matrix that corresponds to 100% annotations. This proximity suggests that annotation of the full dataset might be redundant. Indeed, further inspection of Fig. 3C, indicates that samples annotated as LCS and BS classes (y-axis) by the unsupervised learning method are likely to be predicted as the LLC (x-axis) by OpenLabCluster. One possibility for the discrepancy could be annotation errors of prior clustering method which are taken as the ground truth. Re-examination of the dynamics of some of the features (e.g. tail angle) further supports this hypothesis and demonstrates that the methods in OpenLabCluster can potentially identify the outlier segments which annotation settles the organization of the Behavior Classification Map (for more details see in Appendix 5.2).

Organization of the Latent Representation. Our results indicate that the Latent Representation captured by the OpenLabCluster encoder-decoder and the classifier is able to better organize behavioral segments in comparison to direct embeddings of body keypoints. We quantitatively investigate such an organization with the Monkey dataset, for which ground truth annotations and segmentation are unavailable. Specifically, we obtain the Cluster Map of the segments with OpenLabCluster and then annotate 5% of segments through active learning methods which further train the encoder-decoder and the classifier generating a revised Cluster Map. We then depict the 2D tSNE projection of the Latent Representation and compare it with the 2D tSNE projection of body keypoints in Fig. 4. Indeed, it can be observed that within the Cluster Map, segments are grouped into more distinct and enhanced clusters. To measure the clustering properties of each embedding, we apply clustering metrics of Calinski-Harabasz (CHI) and Davies-Bouldin (DBI). CHI measures the ratio of inter- and intra-cluster dispersion, with larger values indicating better clustering. DBI measures the ratio of inter-cluster distance to intra-cluster distance, with lower values indicating better clustering. CHI and DBI are shown in the bottom left of Fig. 4 considering the various number of clusters (2, 4, 6, 8, 10, 12). The comparison shows that the CHI index is higher for the Cluster Map than the embedding of the keypoints regardless of the number of clusters being considered and is monotonically increasing with the number of clusters. The DBI index for the Cluster Map is significantly lower than the index of the...
embedding of the keypoints for up to 10 clusters with minima at 6 and 10 clusters. This is consistent
with the expectation that clustering quality will be consistent with the number of behavioral types.
Indeed in these behaviors, there are 6 major behavioral classes that can be identified and possibly
several transitional ones. We show in Fig. 4 (right) the Behavioral Classification Map generated by
OpenLabCluster along with frames from representative segments of each class.

3 Discussion

In this paper, we introduce OpenLabCluster, a novel toolset for quantitative studies of animal behavior
from video recordings in terms of automatic grouping and depiction of behavioral segments into clusters
and their association with behavioral classes. OpenLabCluster works with body keypoints which
describe the pose of the animal in each frame, and across frames reflecting the kinematic information
of the movement that is being exhibited in the segment. The advancement and the availability of
automatic tools for markerless pose estimation in recent years allows the employment of such tools in
conjunction with OpenLabCluster for performing almost automatic organization and interpretation of
a variety of ethological experiments.

Efficacy of OpenLabCluster is attributed to two major components; (i) Unsupervised pre-training
process which groups segments with similar movement patterns and disperses segments with dissimilar
movement patterns (Clustering); (ii) Automatic selection of samples of segments for association with
behavioral classes (active learning) through which all segments class labels are associated (classification)
and the clustering representation is being refined.

We evaluate OpenLabCluster performance on various datasets of recorded animal species freely
behaving, such as Home-Cage Mouse, Zebrafish, C. elegans and Monkey datasets. For the datasets for
which ground-truth labels have been annotated, we show that OpenLabCluster classification accuracy
exceeds the accuracy of a direct deep-classifier for any annotation budget even when all segments in
the training set have been annotated. The underlying reason for the efficacy of OpenLabCluster is
the unsupervised pre-training stage of the encoder-decoder which establishes similarities and clusters
segments with the Latent Representation of the encoder-decoder. Such a representation turns out to
be useful in informing which segments could add semantic meaning of the groupings and refine the
representation further.

In practice, we observe that even a sparse annotation of a few segments (5%-20% of the training set)
chosen with appropriate active learning methods would boost clustering and classification significantly.
Classification accuracy continues to improve when more annotations are performed, however, we also
observe that the increase in accuracy is primarily in the initial annotation steps, which demonstrates
the importance of employing clustering in conjunction with active learning selection in these critical
steps. Indeed, our results demonstrate that among different active learning approaches, more direct
approaches such as Top, are not as effective as others considering the need to include more metrics
quantifying uncertainty and similarity of the segments.

As we describe in the Methodology section, OpenLabCluster includes advanced techniques of
unsupervised and semi-supervised neural network training through active learning (Appendix 5.1).
Inspired by DeepLabCut project, we implement these techniques jointly with a graphic user interface
to enable scientists to use the methodology to analyze various ethological experiments with no deep
learning technical expertise. In addition, OpenLabCluster is an open-source project and is designed such
that further methodologies and extensions would be seamlessly integrated into the project by developers.
Beyond ease of use, the graphic interface is an essential part of OpenLabCluster functionality, since it
visually informs the scientists of the outcomes in each iteration step. This provides the possibility to
inspect the outcomes and assist with additional information "on-the-go". Specifically, OpenLabCluster
allows for pointing at points (segments) in the maps, inspecting their associated videos, adding or
excluding segments to be annotated, working with different low-dimensional embeddings (2D or 3D),
switching between active learning methods, annotating the segments within the same interface, and
more.
4 Materials and Methods

Existing approaches for behavior classification from keypoints are supervised and require annotation of extensive datasets before training [24, 25]. The requirement limits the generalization of classification from one subject to another, from animal to animal, from a set of keypoints to another, and from one set of behaviors to another due to the need for re-annotation when such variations are introduced.

In contrast, grouping behavioral segments into similarity groups (clustering) typically does not require annotation and could be done by finding an alternative representation of behavioral segments reflecting the differences and the similarities among segments. Both classical and deep-learning approaches address such groupings [2, 37]. Notably, clustering is a ‘weaker’ task than classification since does not provide the semantic association of groups with behavioral classes, however, could be used as a preliminary stage for classification. If leveraged effectively, as a preliminary stage, clustering can direct annotation to minimize the number of segments that need to be annotated and at the same time to boost classification accuracy.

OpenLabCluster, that is primarily based on this concept, first infers a Cluster Map and then leverages it for automatic selection of sparse segments for annotation (active learning) that will both inform behavior classification and enhance clustering. It iteratively converges to a detailed Behavior Classification Map where segments are grouped into similarity classes and each class is homogeneously representing a behavioral state. Below we describe the components.

**Clustering** The inputs into OpenLabCluster denoted as $\mathcal{X}$, are sets of keypoints coordinates (2D or 3D) or kinematics features for each time segment along with the video footage (image frames that correspond to these keypoints). Effectively, each input segment of keypoints is a matrix with the row dimension indicating the keypoints coordinate, e.g. the first row will indicate the x-coordinate of the first keypoint and the second row will indicate the y-coordinate of the first keypoint and so on.

The first stage of OpenLabCluster is to employ a Recurrent Neural Network (RNN) encoder-decoder architecture that will learn a Latent Representation for the segments as shown in Fig. 5. The encoder is composed of $m$ bi-directional gated recurrent units (bi-GRU) sequentially encoding time-series input into a Latent Representation (latent vector in $\mathbb{R}^m$ space). Thus each segment is represented as a point in the Latent Representation $\mathbb{R}^m$. The decoder is composed of uni-directional GRUs that receive as input the latent vector and decode (reconstruct) the same keypoints from the latent vector. Training optimizes encoder-decoder connectivity weights such that the reconstruction loss, the distance between the segment keypoints reconstructed by the decoder and the input segment, is minimized, see

![Figure 5: Latent Representation is learned by performing the reconstruction task using an encoder-decoder structure. Latent vectors (last state of the encoder) are projected onto low dimensional space with various dimension reduction techniques to visualize the Latent Representation which constitute the Cluster Map.](image)
the Appendix for further details (Section 5.1). This process reshapes the latent vector points in the Latent Representation space to better represent the segments similarities and distinctions.

To visualize the relative locations of segments in the Latent Representation, OpenLabCluster implements various dimension reductions (from $\mathbb{R}^m \rightarrow \mathbb{R}^2$ or $\mathbb{R}^m \rightarrow \mathbb{R}^3$), such as PCA, tSNE, UMAP, to obtain Cluster Maps, see Fig. 5-bottom. Thus each point in the Cluster Map is a reduced-dimensional Latent Representation of an input segment. From inspection of the Cluster Map on multiple examples and benchmarks, it can be observed that the Latent Representation clusters segments that represent similar movements into same clusters, to a certain extent, typically more effectively than an application of dimension reduction techniques directly to the keypoints segments [39, 56].

![Behavior Classification with Active Learning](image)

**Figure 6:** Behavior Classification Map is generated by a fully connected classifier network (green rectangle) which receives the latent vector transformed by the encoder-decoder as input and classifies them into behavior classes (example shown: 8 classes in Home-Cage Mouse Behavior dataset). Behavioral Classification Map is generated from the Cluster Map and indicates the predicted classes of all segments.

**Classification.** To classify behavioral segments that have been clustered, we append a classifier, a fully connected network, to the encoder. The training of the classifier is based on segments that have been annotated and minimizes the error between the predicted behavioral states and the behavioral states given by the annotation (cross-entropy loss). When the annotated segments well represent the states and the clusters, the learned knowledge is transferable to other unlabeled segments. Active Learning methods such as Cluster Center (Top), Core-Set (CS) and Marginal Index (MI) aim to select such representative segments by analyzing the Latent Representation. Top selects representative segments which are located at the centers of the clusters (obtained by Kmeans [57]) in the Latent Representation space. This approach is effective at the initial stage. CS selects samples that cover the remaining samples with minimal distance [58]. MI is an uncertainty-based selection method, selecting samples that the network is most uncertain about. See Appendix 5.1 for further details regarding these methods. Once segments for annotation are chosen by the active learning method, OpenLabCluster highlights the points in the Cluster Map that represent them and their associated video segments, such that they can be annotated within the graphic interface of OpenLabCluster (choosing the most related behavioral class). When the annotations are set, the full network of encoder-decoder with appended classifier is re-trained to perform classification and predict the labels of all segments. The outcome of this process is the Behavior Classification Map which depicts both the points representing segments in clusters and associated states labels with each point (color) as illustrated in Fig. 6. In this process, each time that a new set of samples is selected for annotation, the parameters of the encoder-decoder and the classifier are being tuned to generate more distinctive clusters and more
accurate behavioral states classification. The process of annotation and tuning is repeated, typically until the number of annotations reaches the maximum amount of the annotation budget, or when clustering and classification appear to converge to a steady-state.

References


[10] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.


5 Appendix

5.1 Methods

The inputs of OpenLabCluster are multi-dimensional time series representing the coordinates of animal body keypoints or other kinematic features tracked during movement along with video segments frames.
from these were extracted. We denote the times-series as $X = \{X_u \cup X_l\}$, with $X_u$ representing
the sequences in the unlabeled set and $X_l$ in the labeled set, and assume that the segments are unlabeled,
$X = X_u$, in the beginning. A segment $x_i \in X$ is represented as a sequence $x_i = [x_1, x_2, \ldots, x_T]$, where
$x_t$ is the vector of features for movement at time $t$, $x_t \in \mathbb{R}^p$. For the examples we consider
here, the number of features is 16, 98, 16, 39 for Home-Cage Mouse, C. elegans, Zebrafish and Monkey
datasets, respectively. OpenLabCluster includes three main components: (i) An encoder-decoder
network that learns to reconstruct sequences and forms Latent Representations. (ii) A classifier network
which performs behavior classification. (iii) Active Learning (AL) selection which integrates Latent
Representation information with the classifier output to optimize classification and annotation.

(i) **Encoder-Decoder:** We adopt the Predict & Cluster encoder-decoder framework which has
been shown to achieve self-organizing latent representations [39, 56]. The encoder uses bidirectional
Gated Recurrent Units (GRU) and receives $x_i \in X$ as input. The vector $h_i^T$ is the latent representation
which is the hidden state of the encoder GRU at the last time step $T$. It encodes the dynamic properties
of the whole sequence $x_i$ and lies in the latent space $V$, where $V = \{h_i^T | h_i^T = \text{encoder}(x_i), x_i \in X\}$,
i.e., the space spanned by the latent codes of all sequences. The unidirectional GRU-based decoder receives $h_i^T$
and generates $\hat{x}_i$ - the reconstruction of the original input sequences. The encoder-decoder network is trained by minimizing the reconstruction loss

$$L_{re} = |\hat{x}_i - x_i|.$$  \hspace{1cm} (1)

(ii) **Classification:** The classifier is a one-layer fully connected network appended to the encoder.
It takes Latent Representation as input and generates the probabilities that a sample belongs to each
behavioral states. During training, the classifier is learned to maximize the probability of the annotated
behavior state. In other words, with the annotated samples given the classifier output, the classification
loss is computed as

$$L_{cla}^i = \sum_{l=1}^C -y_l^i \log(p_l^i(x_i)), \hspace{1cm} (2)$$

where $y_l^i = 1$ if $x_i$ belongs to class $l$, and $y_l^i = 0$ otherwise. The complete loss for each sample $x_i$, is then
composed from the reconstruction loss $L_{re}^i$ and the classification loss $L_{cla}^i$ for the annotated samples.

$$L = \sum_{x_i \in X_l} L_{cla}^i + \frac{1}{|X|} \sum_{x_i \in X} L_{re}^i, \hspace{1cm} (3)$$

where $|X|$ is the total number of samples in the dataset. This includes all labeled samples annotated in
current and earlier iteration.

(iii) **Active Learning:** There are three Active Learning (AL) methods embedded in OpenLab-
Cluster: Cluster Center (Top), Uncertainty with Marginal Index (MI), Core-Set (CS).

**Cluster Center (Top)** leverages clusters information in the latent space to enhance coverage and
effectiveness of selected segments (samples). Specifically, $K$-Means clustering is used to transform the
latent representation into a collection of clusters $\mathcal{K}$. The number of clusters $k$

$$k = \frac{1}{N_{iter}} \times \text{percentage} \times |X|, \hspace{1cm} (4)$$

is chosen based on the total number of selection iterations $N_{iter}$ and the percentage of data would be
annotated in total. $k$ is fixed across selection iterations such that $k$ is the number of samples to be
annotated and each is located in a different cluster.

**Marginal Index (MI)** is based on the classifier output and measures the difference between the
classifier output, evaluating top two difference of $p$. The probability prediction $p$ of each class $l$ is

$$p^l(x_i) = p^l(y_i = l | W_{\theta}, W_{\delta}) = \mathcal{C}(x_i),$$

where $p^l$ denotes the probability of a sample to belong to a class $l$ among $C$ classes ($l \in [1, C]$) predicted
by the classifier $\mathcal{C}$. $\mathcal{C}$ indicates the transformation with the classifier. MI is computed as the measure of
the confidence, difference between the most probable class and the second most probable class \cite{59, 60}, i.e.,
\[
MI = \max_{l \in [1:C]} \left( p_l \right) - \max_{l \in [1:C] \setminus \{ l^* \}} \left( p_l \right),
\]
where \( l^* = \arg \max_{l \in [1:C]} \left( p_l \right) \).

Core-Set (CS) aims to discover a set of samples that can cover unlabeled samples with a certain radius \cite{58}. The algorithm finds a set of samples such that the radius is minimal.

With the labeled samples, the classifier is trained until the classification accuracy on these labeled samples converges. Annotation iteration repeats until reaching the annotation budget.

![Prototype Dynamics](image)

**Examples of disparity between OLC and GT**

Figure 7: Top: Prototypical dynamics of three behavioral states (LLC, BS, LCS), envelop (blue). Bottom: Three examples of disparity between OpenLabCluster (OLC) and Ground Truth (GT) classification.

5.2 Zebrafish Behaviors and Dynamics

Zebrafish behavior is subdivided into 13 classes: “approach swims” (ASs), slow types 1 (S1), slow types 2 (S2), short and long capture swim (SCS and LCS), burst type forward swim (BS), J-turns, high-angle turn (HAT), C-start escape swims (SLC), long latency C-starts (LLC), O-bends, and routine turns (RTs), spot avoidance turn (SAT).

As shown in Fig 3C, OpenLabCluster could “misclassify" samples to belong to LLC class, although they have been annotated by previous analysis as BS or LCS. Since the annotation is obtained by a prior unsupervised learning method and not a case-by-case manual annotation, it could be possible that the annotation is made incorrectly, especially for those segments that are located near the boundaries of clusters. We show several of such conflicting examples in Fig. 4. In the top row, we show the typical dynamics of three ground-truth (GT) classes (BS, LCS, and LLC). Inspecting the prototypical dynamics of the LCS, BS, and LLC, one can find that each profile has unique dynamic profile. For example,
Table 3: Comparison of OpenLabCluster (OLC) against its ablated versions: RC, IRC for 5%, 10% and 20% annotations on Mouse, Zebrafish and C. elegans datasets.

<table>
<thead>
<tr>
<th></th>
<th>Mouse</th>
<th>Zebrafish</th>
<th>C. elegans</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label(%)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>Label(#)</td>
<td>143</td>
<td>286</td>
<td>571</td>
</tr>
<tr>
<td></td>
<td>265</td>
<td>530</td>
<td>1059</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>55</td>
<td>109</td>
</tr>
<tr>
<td>OLC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS</td>
<td>65.3</td>
<td>75.4</td>
<td>82.2</td>
</tr>
<tr>
<td></td>
<td>71.9</td>
<td>76.6</td>
<td>79.6</td>
</tr>
<tr>
<td></td>
<td>73.5</td>
<td>64.3</td>
<td>75.4</td>
</tr>
<tr>
<td>Top</td>
<td>58.5</td>
<td>58.4</td>
<td>76.4</td>
</tr>
<tr>
<td></td>
<td>72.0</td>
<td>77.1</td>
<td>80.1</td>
</tr>
<tr>
<td></td>
<td>76.5</td>
<td>76.5</td>
<td>77.8</td>
</tr>
<tr>
<td>MI</td>
<td>63.4</td>
<td>76.3</td>
<td>78.9</td>
</tr>
<tr>
<td></td>
<td>74.2</td>
<td>79.1</td>
<td>81.1</td>
</tr>
<tr>
<td></td>
<td>76.7</td>
<td>76.8</td>
<td>77.0</td>
</tr>
</tbody>
</table>

LLC profile peaks in the beginning and gradually decays. In the bottom row of Fig. 7, we show three examples of segments classified by OpenLabCluster as LLC but marked differently by prior annotation that we consider as ground-truth. It appears that while these examples are similar to LLC profile (having a peak in the beginning and decay) supporting the decision of OpenLabCluster to classify these segments as LLC, the ground truth annotations made by prior clustering analysis are contradictory.

5.3 Ablation study

The two components of OpenLabCluster of unsupervised clustering and semi-supervised classification are essential to the accuracy. In Table 3, we examined “ablated” versions of OpenLabCluster to demonstrate the impact of each component: C designates a variant that includes a classifier network only. RC designates a variant where pretraining phase of OpenLabCluster is ablated, i.e., all the weights of the encoder-decoder and the classifier are randomly initialized. In this scenario, there is no pre-organized Latent Representation. IRC designates a variant where active learning is not being applied but the encoder-decoder structure as well as the training paradigm including pretraining is kept the same as in OpenLabCluster (with pre-organized Latent Representation). As shown in Table 3, the accuracy of IRC is higher than that of C and RC showing the importance of pre-organized encoder-decoder Latent Representation. OpenLabCluster along with various active learning approaches further enhances the accuracy (over IRC) in most cases, especially on the Home-Cage Mouse and the Zebrafish datasets with an average improvement 5.78%. The reason that improvement of OpenLabCluster is not significant on the C. elegans dataset could be that the diversity is limited on the dataset, i.e., samples are equally informative for behavior states learning.