Predicting Mouse Behaviors Using Deep Behavior Mapping

Ankur Aditya
UMass Amherst
aaditya@umass.edu

ByungKwon Moon
UMass Amherst
bmoon@umass.edu

Abstract

Despite substantial progress in classifying mouse behaviors from video data, prior works have been hindered by limitations in classification accuracy. In our study, we address these limitations by introducing modifications to the existing Deep Behaviour Mapping (DBM) model, enhancing its ability to capture intrinsic details of mouse behavior. Additionally, we implement a positional encoding approach during data preprocessing, mapping mouse positions to higher frequencies, enabling the model to effectively capture high-frequency changes in mouse movements. These modifications significantly improve the accuracy of mouse behavior classification to 71.3 %, exceeding the 67% accuracy achieved by previous methods, an improvement greater then 4%.

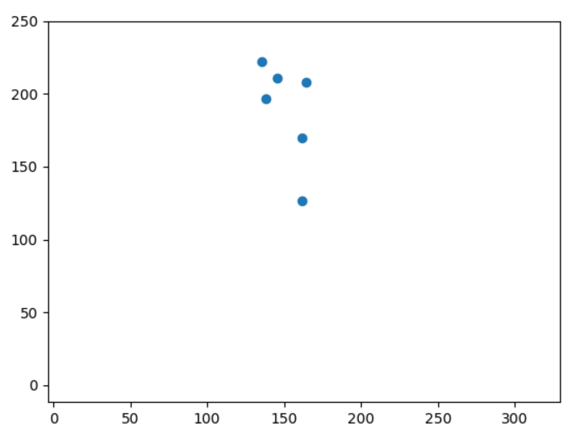
1. Introduction

Recent work has aimed to develop a method to automatically classify mouse behaviors from video data. However, a significant challenge has emerged: the validation accuracy is below 60 percent, indicating that the model struggles to generalize its predictions on unseen data. Improving the accuracy of this model is not just about tweaking an algorithm. An accurate model can revolutionize the way researchers approach behavioral studies, offering a robust, automated, and consistent way to categorize and analyze behaviors without human biases. Using DeepLabCut [1], Mathis et. al. (2018) have managed to track the mouse's location and posture across video frames. This essentially provides us with a "dictionary" of mouse behaviors that can be recognized in the videos. Designed with a 10 node long short-term memory (LSTM) layer, it is trained to predict these pseudo-labels from sequences of pose data. After that the paper used the k-means algorithm to map these trajectories as sequences of discrete states (k =50), with the mouse's behavior in each frame assigned to one of 50 categories , which is termed as behavioral microstates [3].

1.1. Data

In the Lab, we recorded eight different mouse behaviors. Four of them were experimental condition and rest for non-experimental condition. The dataset is recorded for seven days for 23000 frames for each so 56 total data set exist and one of them were corrupted for technical reason.

DeepLabCut (DLC) is used to collect the location of mouse body parts, as shown in Figure 1. We then use the location of the mouse's head (x,y coordinates) to assign an initial behavior. Additionally, we use the distance between the mouse's center of mass and other body parts to assign



(a) Body parts location of one frame.

Head	Head	Head	Nose	Nose	Nose	Left ear	Left ear	Left ear	Right ear	Right ear	Right ear	Center of	Center of	Center of	Tailbase	Tailbase	Tailbase
x	y	likelihood	x	y	likelihood	x	y	likelihood	x	y	likelihood	x	y	likelihood	x	y	likelihood
233.6737	43.36726	1	237.8582	42.96671	0.00134	223.749	40.60463	1	244.6521	50.63865	1	217.6663	88.74942	0.999997	199.129	139.9482	1
232.1967	36.44796	1	237.183	31.02007	0.007503	221.0603	33.42765	1	243.3826	41.83778	1	222.067	81.60242	1	206.4631	131.7275	1
231.2734	35.67641	1	236.3033	30.33004	0.006124	220.433	33.46015	1	242.6452	41.07092	1	224.493	78.58734	0.999999	207.7619	128.4173	1
231.7781	34.47485	1	235.4363	27.00855	0.066505	220.6944	33.08802	1	242.9703	39.38559	1	224.5112	78.40561	0.999998	206.9863	128.0318	1
231.4099	34.35284	1	237.5706	24.87619	0.054983	220.224	33.26252	1	242.9401	40.1764	1	224.2405	78.34667	0.999999	205.6584	127.5222	1
230.0934	34.13422	1	236.9086	24.58852	0.159744	219.7461	34.61254	1	240.8034	39.46803	1	223.5772	78.53316	0.999999	204.1238	128.1947	1
228.6009	32.55224	1	234.0027	26.89422	0.036797	217.2017	33.12959	1	238.6422	38.24269	1	222.8058	78.43415	0.999998	203.4404	128.4446	1
228.4021	32.41175	1	233.6858	27.13868	0.03392	217.1797	33.28362	1	238.365	38.24689	1	222.4875	78.4619	0.999998	201.9576	130.0009	1
226.7146	34.967	1	231.4902	28.09414	0.017517	216.4929	35.09201	1	238.219	39.83576	1	221.3642	79.21812	0.999999	201.4431	130.5961	1
229.4267	36.08565	1	230.9728	29.89046	0.00387	219.477	34.79948	1	240.5739	40.29022	1	222.0612	78.84844	0.999999	201.5104	130.7732	1
229.848	35.03149	1	230.2702	29.41021	0.001632	219.1539	34.25795	1	240.5473	38.31464	1	222.7325	78.80419	0.999997	201.2411	129.1903	1
228.3416	33.63583	1	224.9954	28.80995	0.000526	217.2389	34.1032	1	239.4558	35.30107	1	223.7862	78.4025	0.999997	201.7424	126.462	1
225.6126	32.3667	1	222.6632	26.78432	0.000875	215.5042	32.70481	1	237.331	33.29757	1	223.368	77.31106	0.999999	203.8368	125.8029	1
221.8317	29.73437	1	215.7634	23.25275	0.000713	210.3784	29.43254	1	233.4283	32.80556	1	220.6496	75.01112	1	205.3496	124.8523	1
214.807	29.34178	1	209.135	127.9168	0.000676	202.6727	29.59697	1	227.6225	32.46965	1	218.6721	70.97392	1	208.962	123.2305	1
213.0919	21.5609	1	225.8913	5.104633	0.000773	200.1136	23.23681	1	225.955	25.62348	1	215.8474	60.99737	1	221.6716	111.9927	1
214.7003	19.43302	1	224.4762	114.9728	0.000247	203.872	17.81729	1	225.4737	24.41438	1	215.9373	57.9045	1	223.7249	111.6179	1
214.5103	18.89282	0.999999	223.8939	114.9418	0.000259	204.537	13.85677	1	224.3707	24.22013	0.999999	214.9347	56.76284	0.999999	222.7782	111.4754	1
214.492	19.65031	0.999999	228.9683	3.657188	0.000386	203.7591	15.08663	0.999999	224.5175	25.94126	0.999999	212.2812	56.35325	0.999999	219.1024	111.6344	1
214.0042	20.53403	0.999999	229.504	4.872018	0.000549	203.8314	15.12409	1	224.4944	28.5331	0.999999	210.9511	57.84443	1	217.056	110.997	1
215.0201	20.82885	0.999993	213.6046	113.8591	0.000452	204.1151	15.89112	0.999997	225.2894	29.336	0.999999	210.5617	57.12236	0.999999	214.6093	108.5215	1
219.034	21.36564	0.999994	231.7052	6.248014	0.000369	206.3218	16.29331	0.999989	229.7771	31.72697	1	210.9341	53.5141	1	211.8196	101.5677	1

(b) Result of DLC.

Figure 1. Data set of mouse based on each frames

behaviors such as "Locomotion" and "Rest." Finally, we combine this data to create a deep behavior map.

We next trained an artificial neural network with a 10-node long short-term memory (LSTM) layer. This neural network was able to learn the temporal patterns in the mouse's movements. We then used the k-means algorithm to group the mouse's movements into 50 different categories, which we call behavioral microstates.

2. Problem Statement

This section discusses the deep behavior mapping (DBM) architecture used in the neuron paper by Zhang et al. (2022) and its shortcomings. It also presents our problem statement and how we plan to address it.

The prelimbic cortex (PrL) is a brain region that is involved in the organization and execution of operant behavior. However, the relationship between PrL and operant behaviour is not well understood. In the Neuron paper [3], authors used a technique called deep behaviour mapping (DBM) to extract the detailed representation of behaviour from mice performing an operant task. DBM is a machine learning algorithm that can identify patterns in complex datasets. In this context DBM extracts a detailed representation of mouse behaviour from video recordings. The authors then use this information to map the activity of PrL neurons onto behavioral states.

The authors designed a deep neural network architecture to map these different behaviours. The architecture has variety of layers.

- **Convolutional Layers** extracts spatial features from the high-dimensional calcium imaging data from the PrL neurons.
- **The LSTM layer** is used to learn the long term dependencies in the calcium imaging data. This is an essential step as the behavioural states often depend on the mouse's past behaviour.
- **Fully Connected Layers** combine the features learned by the previous layers to produce a representation of behavioural states of the mouse at a given point.

Although the paper discusses the relationship between PrL and operant behavior in mice, but the DBM architecture used in the study achieved relatively low accuracy in predicting behavioral states, with a mean accuracy of only 65%. This is due to several factors, as described below:

1. Complexity of the relationship between the high-dimensional video data and behavioural states.
2. Limited size of the training dataset.
3. Noise in the calcium imaging data.
4. Choice of layers while designing DBM architecture.

For this project we are re-implementing the DBM architecture from Zhang et al. (2022), using our dataset described in section 1.1. The results of this experiment are as shown in section 4.

To improve the reliability of the DBM architecture, we plan to address the shortcomings of Zhang et al. (2022) by redesigning the architecture and including better data preprocessing steps to handle high-dimensional imaging data. This will help us increase the prediction accuracy of behavioral states of mice. We discuss our approach in detail in section 3.

3. Technical Approach

This section describes our approach for increasing the accuracy of the DBM model (Fig. 3). To achieve this, we are planning to use three new approaches, which we explain in detail in 3.1, 3.2 and 3.3.

3.1. Data Preprocessing

The DBM model is trained on multiple frames of mouse video, where each frame contains the location ((x,y)) of five body parts. The model uses this information to predict one of ten different mouse behaviors. As the location data is captured in multiple frames, it introduces high frequency variation in mouse position in different frames, such details are regarded as high-frequency in nature. Problem is that "Standard" coordinate-based neural network architecture cannot represent high frequency functions [2]. To address this we first pass input coordinates through a high frequency mapping, also know as positional encoding (Fig. 2) in context of transformers.

$$\gamma(p) = (\sin(2^0 \pi p), \cos(2^0 \pi p), \ldots, \\ \sin(2^{L-1} \pi p), \cos(2^{L-1} \pi p))$$

The positional encoding function, denoted by $\gamma(.)$, is applied independently to the x and y coordinates of each mouse body part. In our context, the data consists of (x, y) coordinates representing the locations of the mouse. Therefore, the positional encoding is represented by the tuple, $(\gamma(x), \gamma(y))$. Results of positional encoding are discussed in detail in section 5.3.

3.2. Neural Network Layers

The paper uses four main layers as part of DBM architecture. Including, fully connected, and LSTM layers We have

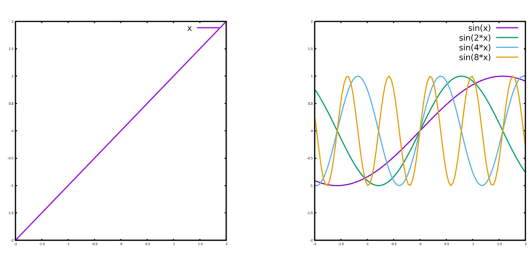


Figure 2. Positional Encoding

planned to change (add or delete) layers so as to achieve better accuracy. Results of modified DBM architecture are discussed in detail in section 5.1.

3.3. Hyperparameter Tuning

We have planned to tune the hyper-parameters to select the best model with high validation accuracy. Results of hyper-parameter tuning are discussed in detail in section 5.2.

4. Intermediate Results

This section discusses about intermediate results which we got when we trained the DBM model on our real dataset 1.1.

4.1. Result of Training Process

When using the DBM model shown in Fig. 3, we observed a low validation accuracy of 67.77%. A potential reason for this is the model's limited ability to differentiate between distinct classification states. This could be attributed to its failure to effectively capture the intrinsic movement patterns of the mouse.

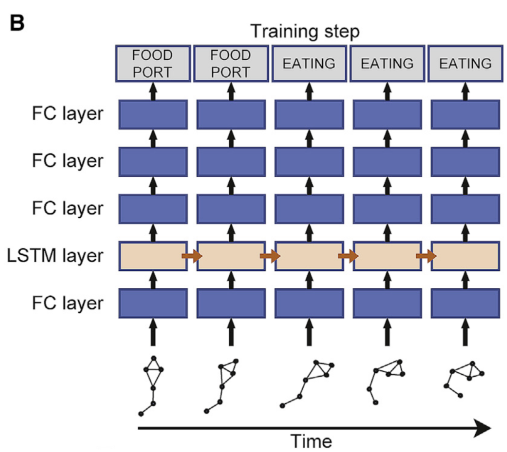


Figure 3. Schematic of the deep behavior mapping (DBM) model

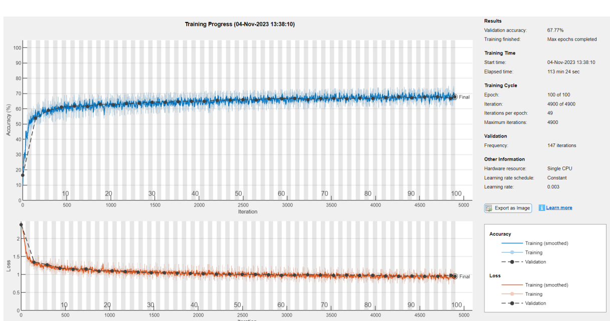
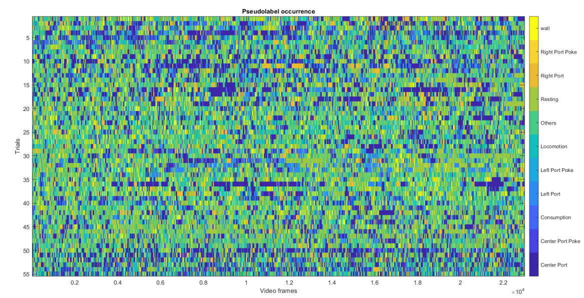
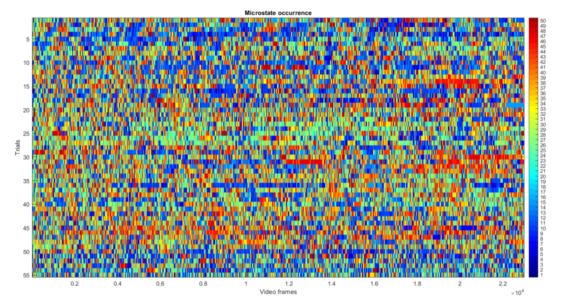


Figure 4. Initial training progress.



(a) initial input behaviors result



(b) Microstates result

Figure 5. (a) Original experiment-defined training labels, and (b) Extracted microstates.

4.2. Result of MicroStates

Figure 5 are the results from the original DBM Output. As seen from the Figure 5b, row represents each video frames and column represent microstates in each video frame. From the results, we conclude that it is hard to visualize and analyze with these tools so we created different visualization method shown in Fig. 6.

4.3. Scatter and Heat Map based on Microstates

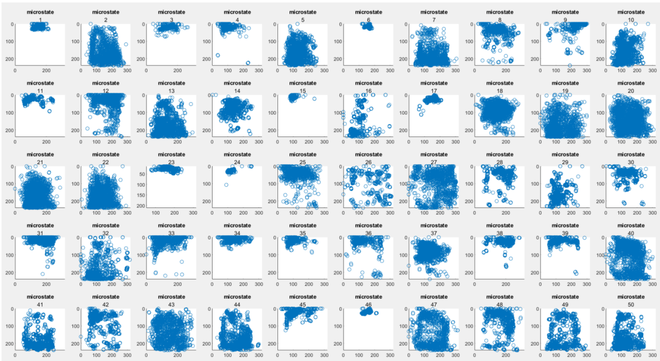
The 6b suggests that our results may be accurate. However, 6a shows that some of the microstates do not match the behaviors that we originally assigned, leading to achieve less accuracy on the DBM model.

5. Initial Experiments on Sample Datasets

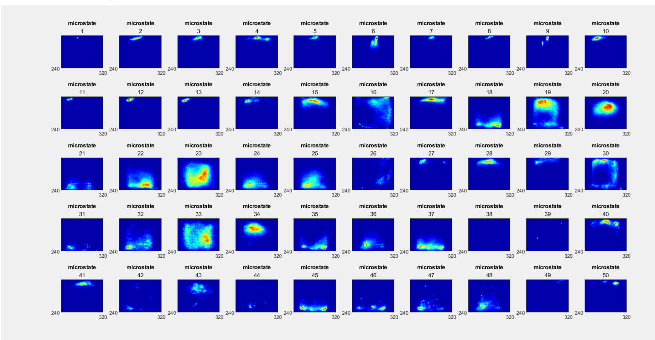
Since our original data has larger dataset, we used sample dataset from Deep behavior mapping. We have made changes following the technical approach described in section 3.1 - 3.3.

5.1. Modified DBM Model

The initial DBM model presented in Fig. 3 employed a three-layered fully connected network and an LSTM layer. To enhance the model's ability to capture temporal features across multiple video frames, we expanded the number of



(a) (x,y) head coordinates of the mouse based on the microstates



(b) Heatmap based on the scatter plot.

Figure 6. (a) Scatter plot , and (b) Heat map based on the microstates.

LSTM layers to two. Additionally, we expanded the architecture by incorporating dropout layers subsequent to each fully connected layer and a combined dropout and BatchNorm layer following the LSTM layers. The design of new DBM model are shown in Fig. 7 and the results are shown in the Fig. 8.

5.2. Hyper Parameter Tuning

With the initial learning rate we ran into over-fitting issue so we tuned the learning rate and from experiments we see that 1e-2 resolves the over-fitting and gives better accuracy.

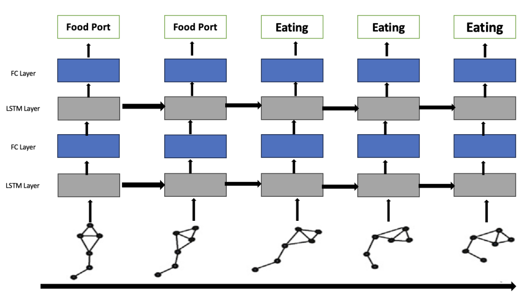


Figure 7. Modified DBM Model

The overfitting result is shown in Fig. 9.

5.3. Data Preprocessing

To capture the high-frequency information embedded in mouse movements, we implemented positional encoding as a data preprocessing step. This technique transforms the raw coordinates into higher-frequency representations, enabling the model to capture rapid changes in mouse position. This is described in details in section 3.1. The effectiveness of positional encoding is demonstrated in Fig. 10, where it yielded a notable accuracy improvement of 75.42%. This indicates that the model is now capable of effectively capturing the mouse's movements.

Furthermore, we integrated positional encoding into the modified DBM model and retrained the model. The results, presented in Fig. 11, showcase the combined advantages of positional encoding and the modified DBM model.

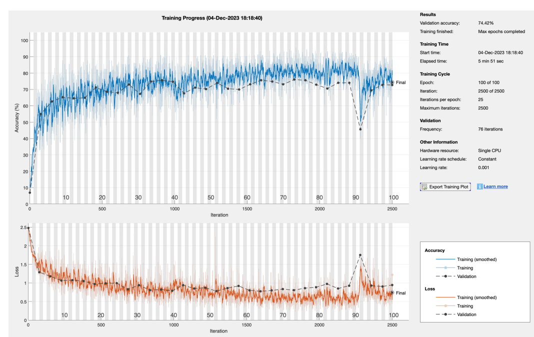


Figure 8. Training with Modified DBM Model

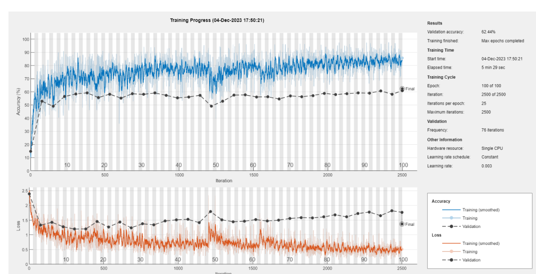


Figure 9. Overfitting

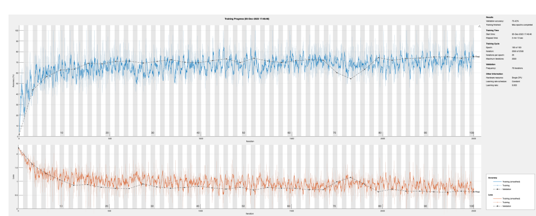


Figure 10. Positional Encoding Only

6. Applying into Real Dataset

This section discusses about results of modified DBM model on our dataset. In this section we train our model based on the modified DBM model shown in Fig. 7. The dataset used in this section is same as used for the intermediate results described in section 4.

6.1. Compare to Intermediate Result

The highest validation accuracy for original model was 67.7 %. Following the technical approach described in the section 3. First with learning rate 0.001 12a , we got 68.3 % validation accuracy which is not significantly different compared to original model. Thus, we change the learning rate 0.004 12b and the validation accuracy for modified DBM model is 71.3% 12c, which is a significant improvement over original model.

6.2. Analysis of the result

In Fig. 13b, we can see microstates of each videos frames by frames. Since each videos are 30 minutes it is hard to analyze exact behavior. Thus, we grouped each mouse and combine microstates occurence based on figure 13a. Since we only care about left-port(action needs to get rewards), center-port(rewards) and right-port(fake-port), we colored them for better visualization as shown in figure 13c. Based on this, we can observe that mouse learned the rewards(blue-bar) were located into the center-port and they wanted to stay in center-port more in 30 minutes after first day is passed.

In figure 14a and 14b, we can observe that new model detected well on each microstates with mouse location. So, in microstates 1-11 mouse are located into the center-port which is top-center position, in microstates 12-15 mouse are located into the left-port which is top-left corner position and microstates 49-50 mouse are located into right-port which is top-right corner position.

However, in the Fig. 13a, we can observe that some of the initial behaviors are neglecting by algorithm. The reason is that, these behaviors data sets are so small compare to whole data-set. For the future, we could apply

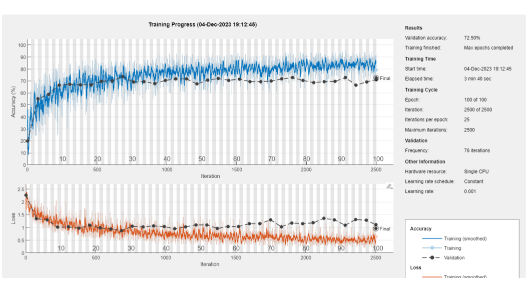
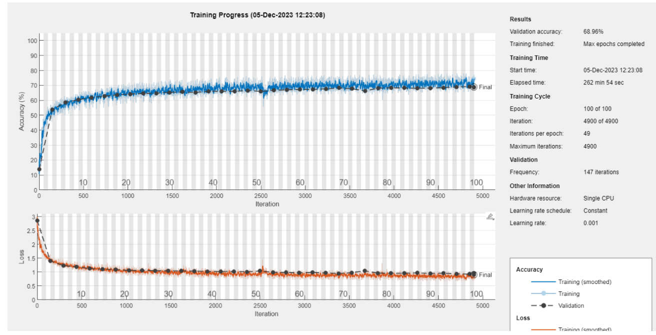
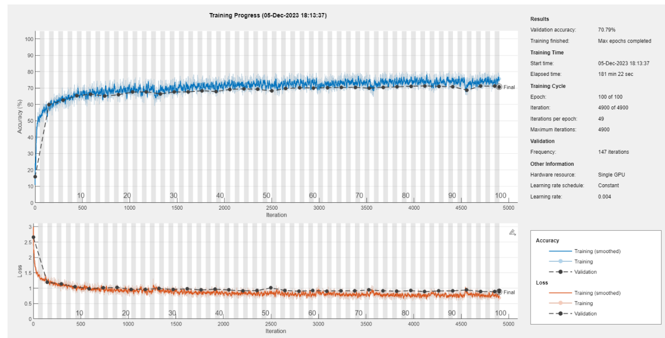


Figure 11. Positional Encoding on modified DBM Model



(a) Learning graph of new model with learning rate 0.001



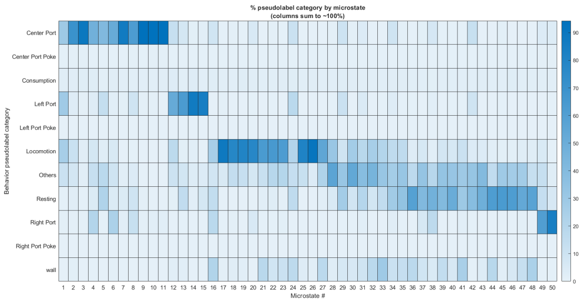
(b) Learning graph of new model with learning rate 0.004

Epoch	Iteration	Time Elapsed (hh:mm:ss)	Mini-batch Accuracy	Validation Accuracy	Mini-batch Loss	Validation Loss	Base Learning Rate
1	1	00:00:15	10.89%	15.77%	2.9903	2.6530	0.0040
2	50	00:01:33	50.71%		1.4737		0.0040
3	100	00:02:52	53.97%		1.3127		0.0040
3	147	00:04:12	59.38%	59.83%	1.2273	1.1912	0.0040
78	3822	02:19:43	69.62%	70.71%	0.8893	0.8986	0.0040
79	3850	02:20:51	71.08%		0.8406		0.0040
80	3900	02:22:41	78.30%		0.6313		0.0040
81	3950	02:24:38	71.49%		0.8612		0.0040
81	3969	02:25:31	78.21%	71.10%	0.6216	0.9181	0.0040
82	4000	02:26:46	71.02%		0.8999		0.0040
83	4050	02:28:18	74.38%		0.7818		0.0040
84	4100	02:30:16	69.43%		0.8288		0.0040
84	4116	02:30:47	74.07%	71.30%	0.7932	0.8866	0.0040
85	4150	02:31:44	76.65%		0.7132		0.0040
86	4200	02:33:26	71.00%		0.8345		0.0040
87	4250	02:35:29	72.43%		0.8179		0.0040
87	4263	02:36:11	74.42%	70.99%	0.7077	0.9084	0.0040
88	4300	02:37:39	72.30%		0.8126		0.0040
89	4350	02:39:28	75.30%		0.7924		0.0040
90	4400	02:41:07	75.90%		0.7661		0.0040
90	4410	02:41:30	71.78%	70.80%	0.8119	0.9112	0.0040
91	4450	02:43:05	78.63%		0.6072		0.0040
92	4500	02:45:01	78.23%		0.6718		0.0040
93	4550	02:46:59	70.59%		0.9002		0.0040
93	4557	02:47:23	75.10%	68.77%	0.7638	0.9418	0.0040
94	4600	02:49:09	74.02%		0.7914		0.0040
95	4650	02:51:09	70.22%		0.8939		0.0040
96	4700	02:53:11	76.28%		0.7226		0.0040
96	4704	02:53:27	73.47%	71.13%	0.7774	0.8906	0.0040
97	4750	02:55:17	77.03%		0.7295		0.0040
98	4800	02:57:17	74.55%		0.7144		0.0040
99	4850	02:59:14	75.41%		0.7207		0.0040
99	4851	02:59:23	78.11%	71.24%	0.6727	0.8860	0.0040

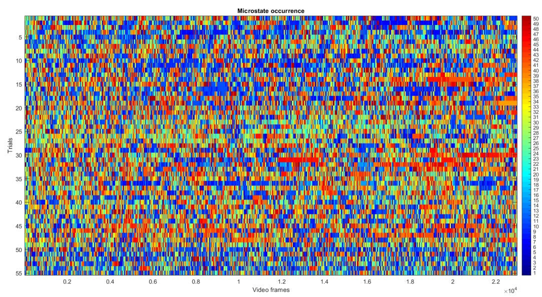
(c) Actual training result for learning rate 0.004

Figure 12. (a) the maximum validation accuracy with new model + learning rate = 0.001 is 68.70% which is not siginificant compare to original model. (b) the maximum validation accuracy with new model + learning rate = 0.004 is 71.30% which is siginificant compared to original model.(c) Actual learning outcome for new model with learning rate 0.004

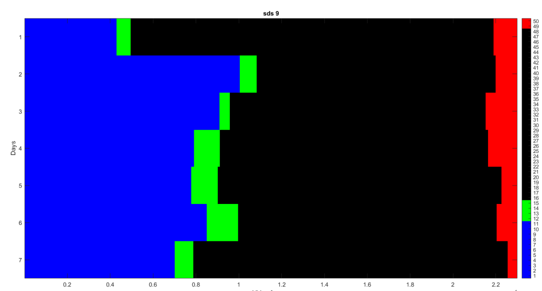
fair-learning method such as reweighting to get fair results. The code of our DBM architecture can be found on https://github.com/ankuraditya98/682_Project.git



(a) percent pseudo label



(b) Microstates with new model

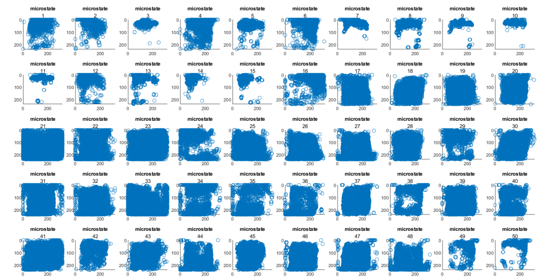


(c) Microstates of one mouse

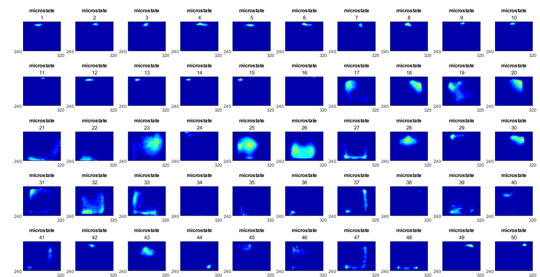
Figure 13. (a) It represent that which microstates is close to the original behavior. For example, microstates 1-11 is close to right-port behavior. (b) Extracted microstates, and (c) Blue represents Center-port, Green represents left-port, red represent right-port and black represent other behaviors. Y-axis are days of experiments.

7. Conclusion and Future Work

Our enhanced model achieved an impressive 71.3% accuracy on real-world data, outperforming the previous benchmark of 67.7%. This significant improvement highlights the effectiveness of our modifications to the model. Further advancements in accuracy could be attained by expanding the dataset to encompass a broader range of mice and reward conditions. Moreover, refining the network architecture could also allow the model to capture even finer details in mouse behavior, leading to even deeper insights into their movements and behaviour classification. Moreover, as



(a) Scatter plot of new model



(b) Heatmap of new model

Figure 14. (a) It represent that which microstates is close to the original behavior. For example, microstate 1-11 is close to right-port behavior. (b) Extracted microstates, and (c) Blue represents Center-port, Green represents left-port, red represent right-port and black represent other behaviors. Y-axis are days of experiments.

figure 13a highlights, our model struggles to detect small sized behaviors. Thus, our future work is going to be applying fairness method such as reweighing method, to ensure fairness of our results.

References

[1] Alexander Mathis and Richard A. Warren. On the inference speed and video-compression robustness of deeplabcut, 2018. 1

[2] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[3] Yan Zhang, Alexander J. Denman, Bo Liang, Craig T. Werner, Nicholas J. Beacher, Rong Chen, Yun Li, Yavin Shaham, Giovanni Barbera, and Da-Ting Lin. Detailed mapping of behavior reveals the formation of prelimbic neural ensembles across operant learning. *Neuron*, 110(4):674–685.e6, 2022. 1, 2