Disentangled multi-subject and social behavioral representations through a constrained subspace variational autoencoder (CS-VAE)

1 Appendix

A Experimental Methods and Preprocessing for the Multi-Subject Dataset

In our work, we employed a subset of the behavioral dataset detailed in Musall et al., 2019 [1]. Briefly, the task entailed pressing a lever to initiate the task, after which a visual stimulus was displayed towards the left or the right. After a delay period, the spouts come forward, at which time the mouse makes its decision by licking the spout corresponding to the direction of the visual stimulus (left or right). Finally, the mice receive a juice reward if they choose correctly.

We tested the CS-VAE on the behavioral data for the four mice performing a visual task, and randomly chose 388 trials per mouse each of the trial has 189 number of frames. Each frame was pre-processed and resized to have both the length and width being 128. One example trial for each mouse can be found in Supplementary Material 1.

Before inputting the data into the model, we sorted the trials by the amount of variance in the images, and shuffled the first half (high variance) and the second half (low variance) of the dataset separately. This was done to speed up training by training the model on high variance trials first. We tested our model by randomly choosing 4 trials from all trials for each mouse 5 times. The same procedure was applied when training the model on the simulation dataset, i.e., the doctored data for one subject.

B Experimental Methods and Preprocessing for the Freely-Moving Social Behavior Dataset

The dataset consists of a 16-minute video of two adult novel C57BL/6J mice, a female and a male, interacting in a clean cage. Prior to the recording session the mice were briefly socially isolated for 15 minutes to increase interaction time. This dataset was collected by one of the authors. The original data has 24917 number of frames with length and width being 1920 and 1080, respectively. The example fraction of the video can be found in Supplementary Material 6.

The nose, ears, and tail base of each mouse were manually annotated using AlphaTracker. We kept 19659 number of frames that have the labels for precrocessing and training. We perform several preprocessing steps to align and crop the video as well as the labels based on one of the two mice (Mouse 1, female). All of the preprocessing steps were based on the AlphaTracker labels. For each frame, we first rotate it to ensure that the nose and tailbase for Mouse 1 are on the same horizontal line, with the central point for rotation as the left ear. Next, we aligned the frame such that the left ear of Mouse 1 was at the same location across all frames. Finally, we resize the frame to be 128×128 and consequently the AlphaTracker labels. For this dataset, since there was a relatively low number of frames, we obtained the CS-VAE MSE and label R^2 for the entire dataset.

C Methodological details of the Partitioned Subspace VAE

The Partitioned Subspace VAE (PS-VAE) was introduced in [2], and we borrow the notation used in that paper when detailing the CS-VAE. Thus, we include here a full description of the model.

First of all, we define the input frame as x, and the corresponding pose estimation tracking label as y. The reconstructed variables are termed $\hat{x}$ and $\hat{y}$, respectively. The supervised latent space is denoted as z_s,

unsupervised latent as z_u, and the background latent as z_b. In a VAE model, we would like to minimize the distance, typically the KL divergence, between the posterior distribution of the latent variables $p(z|x)$ and a chosen distribution $q(z|x)$. However, since $p(z|x)$ is an unknown distribution, the Evidence Lower Bound (ELBO) is introduced as an alternative method to reduce the KL divergence:

$$\mathcal{L}'_{ELBO} = \mathbb{E}_{q(z|x)}[\log(p(x|z)] - KL[q(z|x)||p(z)] \tag{1}$$

Following [2], if we have a finite dataset $\{x_n\}_{n=1}^N$, and we treat n as a random variable with a uniform distribution $p(n)$ while defining $q(z|n) := q(z_n|x_n)$, we can rewrite the ELBO as:

$$\mathcal{L}_{ELBO} = \mathbb{E}_{p(n)}[\mathbb{E}_{q(z|n)}[\log(p(x|z)]] - \mathbb{E}_{p(n)}[KL[q(z|n)||p(z)]] \tag{2}$$

We define the loss over frames $\mathcal{L}_{frames}$ as the first of the two terms above. In the PS-VAE model, there are two inputs: frames x and labels y. Therefore, in Equation (2), instead of writing the input likelihood as $p(x|z)$, we can now write it as $p(x, y|z)$. A simplifying assumption is made that x and y are conditionally independent given z, and thus we can directly write $\mathcal{L}_{frames+labels}$ as $\mathcal{L}_{frames} + \mathcal{L}_{labels}$, where $\mathcal{L}_{labels}$ is calculated by replacing x with y in $\mathcal{L}_{frames}$.

After assuming the prior $p(z)$ has a factorized form: $p(z) = \prod_i p(z_i)$, the KL term $\mathcal{L}_{KL}$ can be split as the addition of ℓ_{KL-s} and ℓ_{KL-u}, i.e., the KL terms for the supervised and unsupervised latents, respectively. We decompose the KL term for the unsupervised latent as the following [2].

$$\begin{aligned}\mathcal{L}_{KL-u} &= \mathcal{L}_{ICMI} + \mathcal{L}_{TC} + \mathcal{L}_{DWKL} \\ &= KL[q(z_u, n)||q(z_u)p(n)] + KL[q(z_u)||\prod_j(z_{u,j})] + KL[q(z_{u,j})||\prod_j(z_{u,j})]\end{aligned} \tag{3}$$

where j represents the latent dimension, $\mathcal{L}_{ICMI}$ is the index-code mutual information, which measures how well the latent encodes the corresponding input data. The term TC is short for total correlation, which measures the interdependency of each latent dimension. The third term, $\mathcal{L}_{DWKL}$ is the dimension-wise KL, which calculates the KL divergence for each dimension individually. Finally, the resulting subspace is forced to be orthogonal by applying orthogonal weights across all the different latents.

The authors in [2] introduce an extension to PS-VAE for modeling multi-session data. The Multi-Session PS-VAE (MS-PS-VAE) can only work with a labeled set of discrete sessions, as described in the Introduction. The images from each session are labeled, and the session-specific latents are enforced to be static over time, thus capturing the image-related details. To enforce the background latents to be static over time in a particular session, and to maximize the difference in the background latents across different sessions, the triplet loss is introduced in MS-PS-VAE. As described in the Introduction, this loss term artificially places the latents from the same session together while separating the latents from different sessions. The triplet loss is computed as the following.

$$\mathcal{L}_{triplet} = \max\{d(a, p) - d(a, n) + m, 0\} \tag{4}$$

Here, a is the anchor point, p is the positive point, n is the negative point, and m is a margin. The function pulls the point p towards point a, and pushes the point n away from point a. While training, the data from multiple sessions is included in each mini-batch. The data from each session is split in three, and each third from the same session acts as an anchor and positive point, while the data from another session acts as a negative point. Practically, this requires as many sessions as possible in the same mini-batch during the training for accurate results. As the number of sessions increases, this method becomes computationally intractable, and may lead to unsatisfactory reconstruction results. Moreover, this loss does not allow for varying backgrounds across any one session.

In the MS-PS-VAE model, the triplet loss was applied as a supervised manner to pull the data from the same subject being closer while pushing the different subjects away from each other. This method is only useful when the number of sessions is known, and is not applicable in an open-field setting, for example while modeling freely-moving social behavior as in this manuscript.

Therefore, in this manuscript, we introduce a regularization term that can automatically separate different subjects in the background latent space without specifying the number of sessions or labeling each frame as belonging to a specific session.

Table 1: Hyperparameter for different dataset

Dataset	α	β	σ	γ
Various contrast	1000	5	5	500
Multi-subject	1000	5	15	500
Social behavior	1200	N/A	20	200

Table 2: Latent dimensions and the prior distribution for different dataset

Dataset	supervised	unsupervised	constrained	prior distribution
Various contrast	5	2	3	Swiss roll
Multi-subject	5	2	2	circle
Social behavior	2	0	3	hollow cylinder

D Model Architecture and Training

Our computational experiments were carried out using TensorFlow and Keras. The image decoder we use is symmetric to the encoder, with both of them containing 14 convolution layers. We applied the Adam optimizer with learning rate as 10^{-4}. For the multi-subject dataset, we fixed our batch size to be 256 and trained for 50 epochs. For the freely-moving social behavior dataset, we trained for 500 epochs with batch size 128.

E Choice of Hyperparameters

In the multi-subject dataset, four coefficients need to be decided for the objective function as indicated above: $\{\alpha, \beta, \sigma, \gamma\}$. There is a balance between the choice of β and γ: properly choosing the values could separate the latent in the unsupervised space and the latents in both unsupervised and background space as well. A large separation of the background latent may potentially lead to unsatisfactory reconstruction results. The choice of kernel size σ depends on the dataset, and should be larger than the number of distinct groups in our dataset; since in our current experiments, we have at most four groups, we set $\sigma = 15$. Moreover, we set α to 1000, β to 5, γ to 500. We set the dimensionality of the supervised latent space equal to the number of tracked video parts, which is 5 in our case. We set the dimensionality of the unsupervised latent space as 2, while that of the background latent space as 2.

In the social behavior task, we track the nose location as the supervised latent, since the other labels do not have a high variance (due to the alignment process). Additionally, we do not need any unsupervised latents to explain the individual's behavior. The CS latent in this setting has 3 dimensions. Here, α is 1200, γ is 200, and the kernel size is 20.

The hyperparameters chosen for all three datasets are shown in Tables 1 and 2.

F Motif Generation

A switching linear dynamical system (SLDS) consists of discrete latent state $z_t \in \{1, 2, ..K\}$, continuous latent state $x_t \in \mathbb{R}^M$, and the observation state $y_t \in \mathbb{R}^N$. Here, $t = 1, 2, 3, .., T$ is the time step, T is the length of the input signal; K is the number of discrete states; M is the number of latent dimensions; N is the observation dimensions. The discrete latent state z_t follows the Markovian dynamics with the state transition matrix expressed as:

$$Q_{i,j} = P(z_t = j | z_{t-1} = i) \tag{5}$$

The continuous latent state x_t has the following linear dynamical relations that determined by z_t.

$$x_{t+1} = A_{z_{t+1}} x_t + V_{z_{t+1}} u_t + b_{z_{t+1}} + w_t \tag{6}$$

Here, $A_{z_{t+1}}$ is the dynamic matrix at state z_{t+1}; u_t is the input at time t, with $V_{z_{t+1}}$ being the control matrix; $b_{z_{t+1}}$ is the offset vector and w_t being the noise which is generally the zero mean Gaussian. Here, our

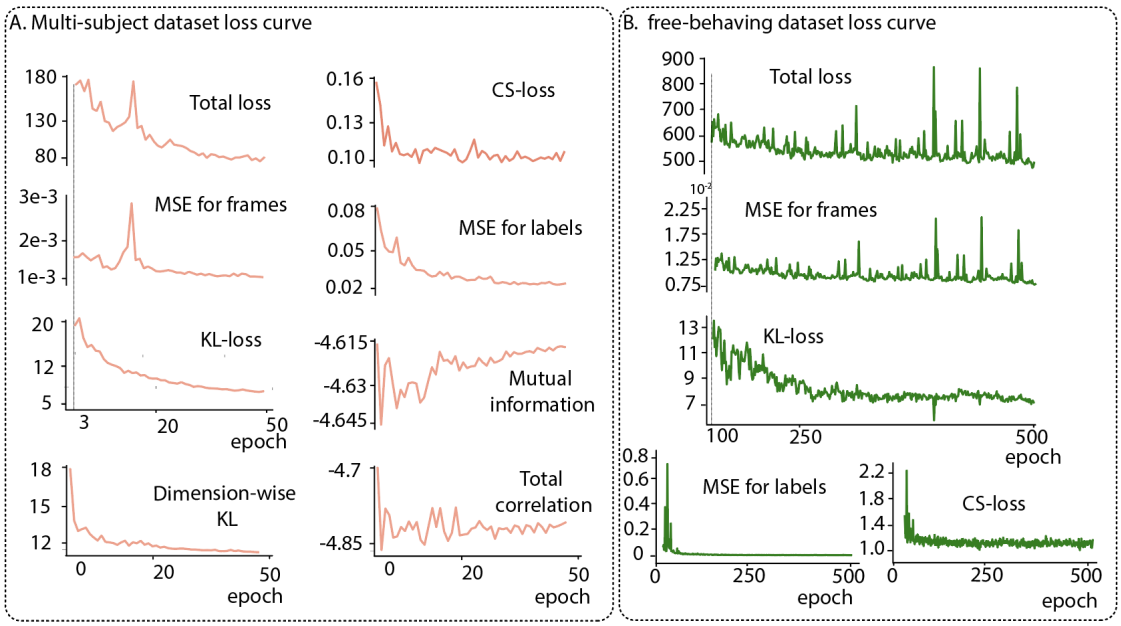

Figure 1: Loss curve for A. training the multi-subject dataset B. training the freely behaving dataset with the specified hyperparameters as in Tables 1 and 2.

observation model is in Gaussian case; therefore, the observation y_t is expressed as:

$$y_t = C_{z_t} x_t + F_{z_t} u_t + d_{z_t} + v_t \tag{7}$$

Here, C_{z_t} is the measurement matrix at state z_t; F_{z_t} is the feedthrough matrix which directly feed the input into the observation; d_{z_t} is the offset vector and v_t is the noise. Here the update was accomplished by the Expectation-Maximization(EM) algorithm. In the E-step, the model updates the hyperparameters. In the M-step, the log-likelihood in Eq.7 is being maximized.

To implement the SLDS, we adopted the open source software from Linderman et al.[3]. We fit the SLDS using different latent dimensions, where the observation dimension was the order of latent dimension and the number of states was determined by visualizing the videos. We use SLDS's to model the motifs in the multi-subject dataset since the behaviors are well separated using their dynamics. We use K-means to model the motifs in the freely-moving social behavior dataset since the behaviors are well separated directly in state space. An autoregressive HMM (a simpler model than an SLDS) applied to the CS latents in the social behavior dataset leads to similar results as the K-means.

G Loss Curves

We show the learning curve for each loss term for both dataset to precisely quantify the model, in Fig. 1. For the multi-sujbect dataset (Fig. 1A), for the unsupervised latents, the final loss for dimension-wise KL, total correlation, and the mutual information are 11.7, -4.8, and -4.6, respectively. The final KL loss for the supervised latents is 5.06 and the final CSD loss for the CS latents is 0.1. For the free behaving dataset, the loss curves for each loss term are shown in Fig. 1B. By the end of the training process, the KL loss for the supervised latents is 7.01 and the CSD loss for the CS latents is 1.15.

H SVM

To further quantify the separation of the latents between different subjects, we applied a supervised classification method to decode the identity of the subject using each latent.

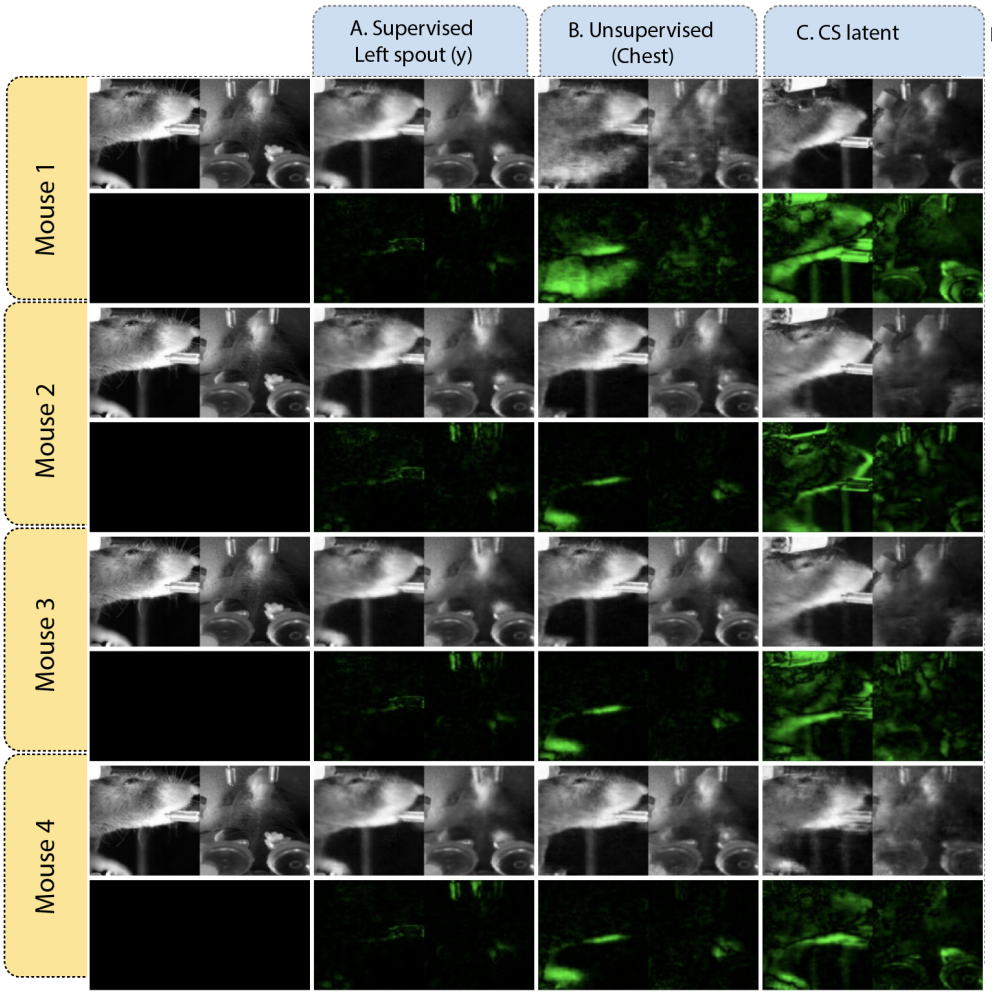

Figure 2: Latent traversals for the multi-subject dataset for the four mice with the same base image A. an example supervised latent, B. an example unsupervised latent, and C. an example CS latent. We see that the same base image (Mouse 3) is transformed into a different mouse each time when changing the CS latent.

After randomly shuffling all the latents, we split all the trials into training trials and test trials, with each mouse having 368 trials in the training set and 20 trials in the test set, and repeated this 5 times with different random seeds.

I Latent traversal

For the multi-subject dataset, we tested the latent traversal with the same base image to validate the results, shown in Figure 2. Here, we randomly chose a frame from a mouse and changed each individual latent within different ranges as detailed in the Methods. For example, in Figure 2, the first row contains the output when the corresponding latent is changed to take on the maximum value from the range of Mouse 1. As in the figures in the main text, the upper images are the latent traversal images while the lower ones are the difference between the upper image and the original image. We see that the base image from Mouse 3 can be flexibly changed to produce a different mouse when changing the CS latent. Moreover, when changing the supervised and unsupervised latents for the different mice, Mouse 3 seems to be flexibly changing with these latents from different mice.

To better visualize the specialization of each latent, we generated the latent traversal videos for each latent with different base images. For different mouse, we first of all find the maximum and the minimum

value for the specific latent. Then, change the latent within that range with 0.5 per step. Finally, concatenate all the latent traversal images into videos. The videos can be found in Supplementary Material 3.

We performed a similar visualization on the freely-moving social behavior dataset for the CS latents. The latent traversal videos can be found in Supplementary Material 4, and some clips from the videos are shown in Fig 3.

J Neural decoding models

The trials were first shuffled and then split into training and testing. Next, we employed the CS-VAE generated latent, and choose one example subject to decoded the behavior at time t using the neural activity recorded between $t - 0.15$s and t. We applied four types of model to compare the performance. A linear model which directly map the neural activities into the behavior. A multilayer perceptron (MLP) with three dense layers to train the decoder. We used the Adam optimizer with learning rate decay from 0.1 with 0.3 decay rate for every 5 step. The batch size was fixed to be 150 and trained for 200 epochs. A LSTM model, which begin with a dense layer followed by a LSTM layer with drop out rate being 0.5 and another dense layer at the end. We applied the same training strategy as in MLP model.

We introduced a model based on transfer learning to perform the decoding test on the previously tested subject. The rest of the three mice were the input to the original training model. The procedures were similar as before, after the trials were being shuffled and split, we decoded the behavior directly with the raw neural activities with the time window being $0.15s$. After that, we implemented three perceptron layers for each of the three mice before the output of which went into a recurrent neural network (RNN). The RNN consisted of one long short-term memory (LSTM) layer with unit number of 64 and a drop out layer with rate being 0.5. We applied the Adam optimizer with learning rate decay from 0.1 with 0.3 decay rate for every 5 steps. The batch size was 150 and we trained for 200 epochs. After we finished training the original network, we transferred the RNN model to the new model which was applied to train for the fourth mouse alone. For the fourth mouse, the trials were split with different training and testing ratio. After applying the same steps to the data, the neural activities then went through a new perceptron layer before went through the pre-trained RNN model. We applied the Adam optimizer with the same learning rate decay procedures as well. We again, trained for 200 epochs with batch size being 128 this time. The trade-off between accuracy and time for different models can be found in Tables 3 and 4.

K Code

The code for training the CS-VAE can be found in Supplementary Material 7. The code can be executed by simply compiling the script 'train.py'.

References

[1] Musall, S., Kaufman, M. T., Juavinett, A. L., Gluf, S. Churchland, A. K. Single-trial neural dynamics are dominated by richly varied movements. Nature neuroscience 22, 1677 – 1686 (2019).

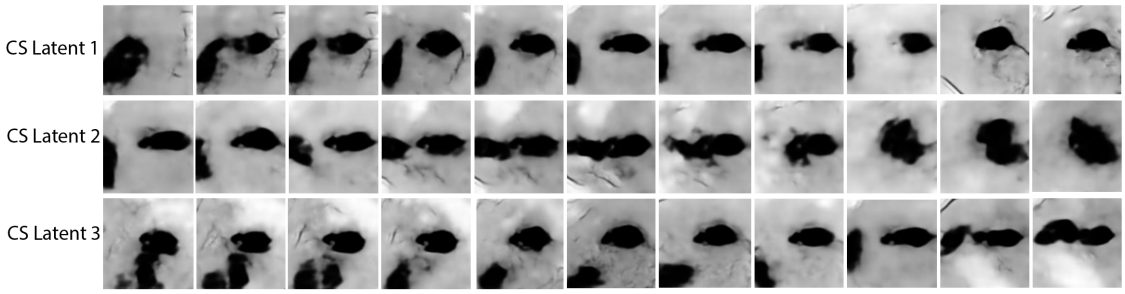

Figure 3: Latent traversals on the CS-latents for the freely-moving social behavior dataset. We see that the latents all encode for social interactions between the two mice.

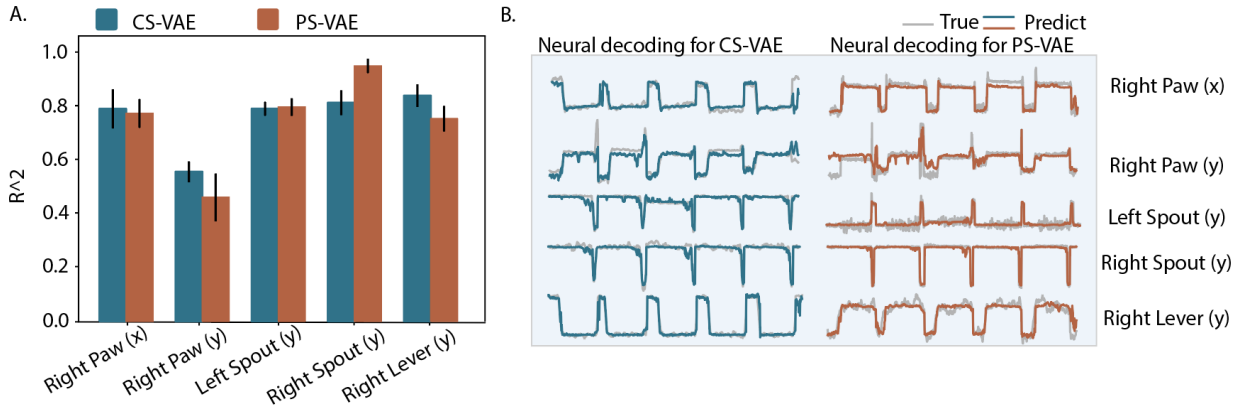

Figure 4: Neural decoding for CS-VAE vs. PS-VAE.

Table 3: Training size vs R^2 value for multi-subject dataset

Training size	Linear model	Dense model	LSTM model	Transfer learning model
67712	0.476 ± 0.048	0.580 ± 0.058	0.610 ± 0.054	0.590 ± 0.050
58512	0.478 ± 0.014	0.560 ± 0.023	0.595 ± 0.022	0.579 ± 0.019
49312	0.483 ± 0.009	0.556 ± 0.014	0.593 ± 0.013	0.576 ± 0.011
40112	0.476 ± 0.013	0.543 ± 0.019	0.576 ± 0.019	0.562 ± 0.015
30912	$0.470 \pm 0.0.011$	0.529 ± 0.015	0.559 ± 0.018	0.552 ± 0.013
21712	0.458 ± 0.010	0.496 ± 0.016	0.524 ± 0.017	0.522 ± 0.013
12512	0.424 ± 0.012	$0.461 \pm 0.0.019$	0.480 ± 0.025	0.485 ± 0.018
3312	0.269 ± 0.030	0.321 ± 0.048	0.325 ± 0.057	0.345 ± 0.043

[2] Whiteway, M. R. et al. Partitioning variability in animal behavioral videos using semi-supervised variational autoencoders. bioRxiv (2021).

[3] Linderman, S. et al. Bayesian Learning and Inference in Re- current Switching Linear Dynamical Systems. In Singh, A. Zhu, J. (eds.) Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, vol. 54 of Proceedings of Machine Learning Research, 914–922 (PMLR, 2017).

Table 4: Training size vs time usage for multi-subject dataset

Training size	Linear model	Dense model	LSTM model	Transfer learning model
67712	1.442 ± 0.282	115.946 ± 1.559	169.801 ± 5.961	169.482 ± 5.041
58512	1.130 ± 0.212	80.428 ± 1.586	146.734 ± 5.063	151.771 ± 4.162
49312	0.937 ± 0.194	68.879 ± 1.257	122.500 ± 2.283	125.240 ± 4.479
40112	0.679 ± 0.114	56.449 ± 1.119	100.336 ± 2.212	102.923 ± 3.391
30912	0.484 ± 0.079	44.427 ± 0.808	78.850 ± 1.771	79.907 ± 2.608
21712	0.309 ± 0.050	31.968 ± 0.574	57.670 ± 1.681	56.032 ± 1.549
12512	0.162 ± 0.008	19.573 ± 0.369	35.557 ± 1.050	33.409 ± 0.741
3312	0.104 ± 0.034	7.292 ± 0.092	13.318 ± 0.342	11.365 ± 0.336