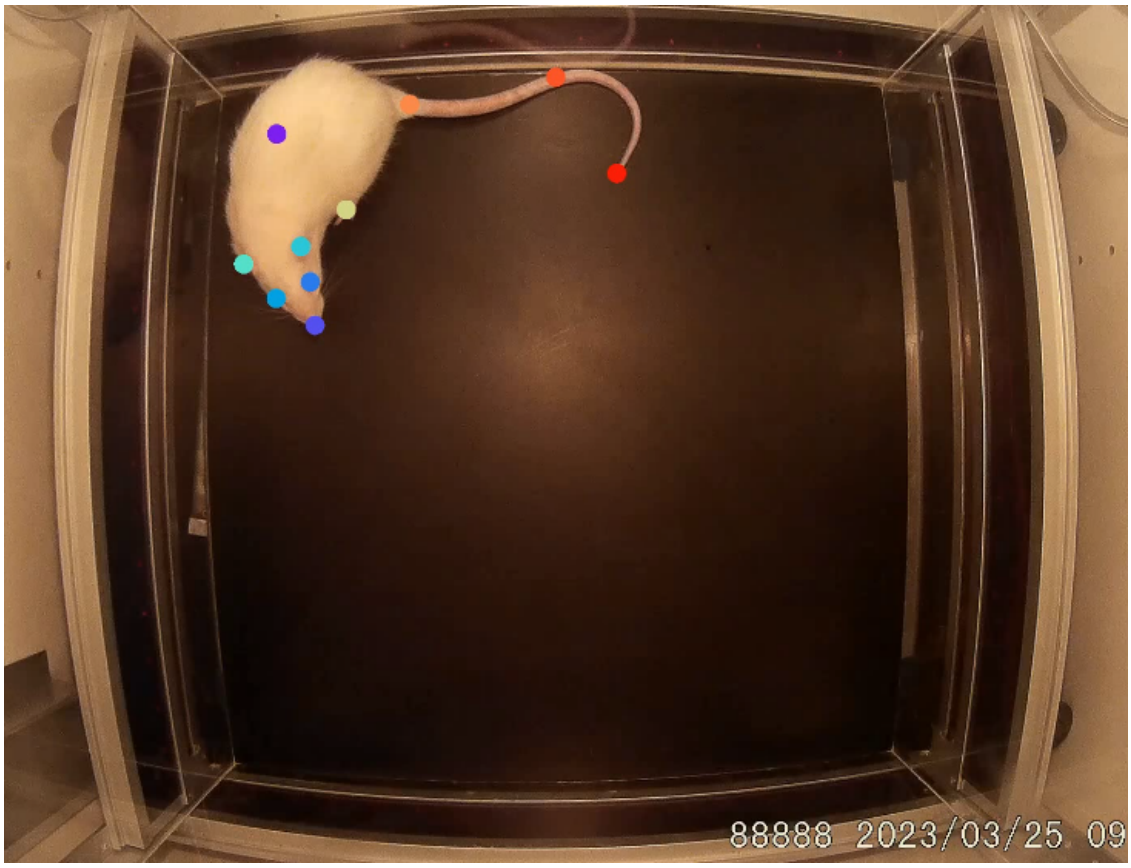
CHALMERS
UNIVERSITY OF TECHNOLOGY



Deep learning methods for identification of animal behavioral patterns

Master's thesis in Engineering Mathematics and Computational Science

EMILIA ROOS

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

Master's thesis 2022

Deep learning methods for identification of animal behavioral patterns

EMILIA ROOS

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Signal processing and Biomedical engineering
Chalmers University of Technology
Gothenburg, Sweden 2022

Deep learning methods for identification of animal behavioral patterns
EMILIA ROOS

© EMILIA ROOS, 2022.

Supervisor: Fredrik Wallner, Irlab Therapeutics AB
Examiner: Ida Häggström, Department of Electrical Engineering

Master's Thesis 2022
Department of Electrical Engineering
Signal processing and Biomedical engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Visualization of pose estimations in Irlab's video.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2022

Deep learning methods for identification of animal behavioral patterns
EMILIA ROOS
Department of Electrical Engineering
Chalmers University of Technology

Abstract

When performing animal testing to investigate how central nervous system drugs and diseases affect the organism, quantification of the animal's behavior can give important insights. An unsupervised approach offers the advantage of not needing to determine the behaviors to capture beforehand which is useful should an unexpected behavior arise. One such unsupervised approach is through the use of a Variational Autoencoder (VAE) that takes sequences of body part coordinates as input and reduces the dimensions by encoding the input through a recurrent neural network (RNN) into a latent vector. The latent vector is then clustered in order to separate behavioral patterns.

In this project, identification of behavior through a VAE was performed on four different datasets based on videos captured from different angles with different camera quality. The pose estimations of the videos were encoded using Bidirectional RNNs and Dilated RNNs and the latent space was clustered using k-means. The performance of the model was then evaluated by computing scores that compared the clustering with annotated data.

The model performs well on some videos but falls short on others, filmed from above and below alike. Some behaviors are identified better than others depending on the angle from which the video has been captured, for example, the model identifies grooming pattern better from above and walking pattern better from below. Moreover, it is concluded that one factor on which the behavioral identification model hinges is the quality of the video. However, due to the black box nature of the model and difficulties to accurately evaluate performance, the reason for varying performances for different datasets is challenging to determine. Similarly, it is difficult to determine which camera angle that will allow for the best results. Nonetheless, the results from this project indicate a promising method for identification of behavioral patterns in animals filmed from above as well as below and is a solid ground for further work on the implementation of robust behavior detection.

Acknowledgements

I would like to thank my supervisor Fredrik Wallner at Irlab Therapeutics for his advice and enthusiasm throughout the project, providing all the support I could have asked for. Furthermore, I would like to thank everyone at Irlab Therapeutics, especially Peder Svensson, Susanna Waters and Johan Kullingsjö, for showing encouragement and offering insightful thoughts and feedback which I have valued a lot.

Finally, I want to thank my friends and family for their reliable and amazing support.

Emilia Roos, Gothenburg, June 2022

List of Abbreviations

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AE	Autoencoder
Bidir	Bidirectional
CNS	Central nervous system
fps	Frames per second
GRU	Gated recurrent unit
ISP	Integrative screeing process
KL	Kullback-Leibler
LSTM	Long short-term memory
NMI	Normalized mutual information
PD	Parkinson's disease
ResNet	Residual neural network
RNN	Recurrent neural network
UMAP	Uniform manifold approximation and projection
VAE	Variational autoencoder

Contents

List of Figures

List of Tables

1
Introduction

Parkinson's disease (PD) is the second most common neurodegenerative disorder after Alzheimer's disease. PD is a lifelong disease and it grows worse over time with both motor and non-motor symptoms. The most prominent cause behind PD is when nerve cells in the basal ganglia, an area of the brain that controls movement, die or become impaired. This results in a decreased production of the neurotransmitter dopamine which causes severe movement problems. The production of norepinephrine is also decreased as a result of PD due to the loss of nerve endings. The decrease in norepinephrine explains some of the non-motor symptons of PD, for example fatigue and irregular blood pressure [1]. The estimated number of people affected by the disease is over six million; a number that is expected to have more than doubled by the end of 2040[2]. There is therefore a need for treatments that can slow down or halt the disease's progression.

Irlab Therapeutics is a pharmaceutical company that aims to discover and develop treatment of diseases that affect the central nervous system (CNS), mainly PD. Their research involves an Integrative Screening Process (ISP) which is constructed around a database of CNS compounds that is analyzed and processed using machine learning. Molecules that are produced in Irlab's labs are injected into rats and the effects of the compound are measured through, among other things, neurochemical analysis, gene expression, and movement patterns. The movement patterns are currently captured through the use of infrared (IR) sensors that measure the rat's movements while in containment. In addition to the IR sensor measurements, the rats are observed and their behaviors at certain time intervals are written down by animal behavior experts. Videos of rats have also been collected, however, these have not been used for analysis and it would therefore be valuable for Irlab if these videos were explored through deep learning methods to see if a video format data source provides useful information. This would allow for an additional source of data that is non-intrusive for the rats in containment.

Understanding and quantifying behavior and movement is a fundamental problem in many academic fields. It can offer valuable insight when, for example, investigating the effects of drugs or diseases in animal studies. Behavioral quantification can be of particular use in cases concerning the CNS since compounds that affect the brain could result in behavioral effects. Moreover, the brain is one of the most complex organs in the body, a full understanding of which has not yet been achieved[2]. Therefore, the effect of various compounds on the CNS might not be known beforehand. This means that using a supervised approach, where the behavioral patterns

to identify are defined beforehand, is not optimal when researching how compounds affect the CNS and thereby behavior. Rather, an unsupervised approach that can find the different behavioral patterns displayed by a rat is preferred.

1.1 Background

This project centers around videos of rats collected by Irlab with the purpose of using unsupervised deep learning methods to find behavioral patterns, such as grooming or walking. Prior to the commencement of this project, a previous project has provided a good jumping off point by changing the format of some of the videos into pose estimations. The idea is to transform the rest of the videos in addition to videos from other sources and then perform unsupervised behavioral pattern identification of these pose estimations. If successful, the model created in this project will offer additional data to use when analyzing the effect of CNS compounds. This would in turn help in Irlab's research purpose and it could also lead to a smaller need for animal testing in the future.

Unsupervised multivariate behavioral classification on videos of rodents has been performed before [3][4] and one particular existing model created by K. Luxem et. al. [3] demonstrates successful behavioral pattern identification using pose estimation of a video of a mouse. However, the video used in that project was filmed from below, unlike Irlab's videos which are filmed from above.

1.2 Objective

The aim of this project is to implement the existing model created by K. Luxem et. al. [3] on Irlab's videos as well as on other datasets in order to identify behavioral patterns in addition to getting an understanding of the model's capabilities and limitations. Furthermore, the goal is to improve the existing model further by implementing different neural networks in its framework.

1.3 Method Outline

The first step of the project is to transform all video data into pose estimations where different bodyparts in the rodents are tracked. The pose data takes the form of x and y coordinates at different time frames in the videos. The pose estimation is performed through the tool DeepLabCut[5]. A sliding window then isolates sequences of poses which results in a highly dimensional dataset which needs to be reduced.

Secondly, a Variational Autoencoder (VAE) is trained using the sequences of poses that are isolated by the sliding window. Two different neural networks are implemented in the encoder of the VAE, bidirectional RNN and dilated RNN. After training the VAE, the trained encoder encodes the entire dataset and thereby re-

duces its dimensions.

After reducing the dataset into a set of latent vectors, the vectors are clustered using k-means. Subsequently, the timeframes in the different clusters are observed in order to see the different behavioral patterns. The observed behaviors are used to annotate a section of each video. The annotated data is then compared with the clustering in order to evaluate the model's performance.

1.4 Scope and limitations

Unsupervised behavioral pattern identification on rodents has been performed before [4][3][6] and this project is inspired by a project that used variational embeddings of animal motion to identify behavior[3]. This is explored further by changing some aspects of the methodology to give new insights and potentially improve the identification process. For example, in this project, the pose data is acquired from a camera aimed at the animal's containment from above, as opposed to below, which results in the use of different pose data when training the networks. In addition to the the data captured from above, videos captured from below other than the data used in the previous project allows for analysis of how the model performs on different datasets.

Furthermore, it is suggested in the discussion of the aforementioned project's documentation that a possible improvement would be to implement dilated RNNs[7] in the encoder instead of bidirectional RNNs. This is implemented in the project. Other neural networks beside the bidirectional and dilated RNNs are not explored or implemented. Furthermore, the Variational framework or the input format are not changed.

Hopefully, this project enables analysis of behavioral pattern changes in animal testing in drug research. However, this project focuses on producing and implementing a pattern identification model that can find behavior, it does not do further analysis on the behavior that it identifies.

2
Theory

This chapter introduces the background of the different tools and concepts used in the project. Section 2.1 explains the functionality of the pose estimation tool that is used to obtain coordinates for different body parts in the videos. Furthermore, section 2.2 presents the Variational framework that is used in the project and section 2.3 explains the neural networks in the Variational framework. Finally, section 2.4 shows the clustering algorithm that is used and section 2.5 explains a visualization technique.

2.1 Pose estimation using DeepLabCut

Pose estimation is the process of tracking and predicting the location of a person or object, enabling extraction of particular aspects in videos. Capturing motion can be done by using reflective markers on the objects that are to be tracked, however, pose estimation can also be performed markerless[8]. DeepLabCut is a tool that performs markerless pose estimation using transfer learning and deep neural networks[5].

DeepLabCut consists of a deep convolutional network that combines pretrained residual neural networks (ResNet) and deconvolutional layers. A ResNet is a type of neural network that utilizes skip connections to jump over some layers and are often used in algorithms for object recognition[5]. The output of the ResNet is up-sampled using deconvolutional layers. The deconvolutional layers also produce spatial probability densities that indicate a bodypart's location. The weights of the network are trained on labeled data which is provided in the form of frames where the tracking points have been annotated. The pose estimation results from using DeepLabCut consist of the x and y coordinates of the different objects that have been tracked. In addition to the coordinates of each object, a likelihood parameter is given that indicates the probability of a tracking point being placed correctly.

2.2 Variational Autoencoder

Dimensionality reduction refers to techniques used when reducing the number of features in a dataset. Data with high dimensionality can be difficult to use when creating predictive models and a model with too many degrees of freedom is likely to overfit the training dataset[9]. Dimensionality reduction leads to a more interpretable representation of the data and can be performed in several different ways. One way to reduce the dimensionality of data is through the use of an Autoencoder.

An Autoencoder, in a nutshell, consists of two parts; an encoder and a decoder. The encoder takes input data, $\boldsymbol{x}$, and reduces its dimensionality, turning it into a latent representation, $\boldsymbol{z}$. The decoder decodes the latent representation into an approximation of the original data, $\boldsymbol{x'}$. If the Autoencoder is able to get a close approximation, then the latent vector is representative of the input data and the encoder can be used to reduce the dimensionality of data. However, while an Autoencoder is convenient to use when compressing data, it is less useful when creating a latent space from which to generate random samples. The latent space created by an Autoencoder has regions with a generative capacity but the latent space typically also contains regions that will generate useless output[10]. This is called a non-regularized latent space. The Variational Autoencoder (VAE)[11] solves this problem and creates a regularized latent space, the entirety of which has generative capabilities.

The VAE, as opposed to the standard Autoencoder, encodes a mean vector μ_x and a standard deviation vector σ_x and the latent vector, $\boldsymbol{z}$, is sampled from a Gaussian distribution with mean μ_x and standard deviation σ_x. The random sampling process of $\boldsymbol{z}$ from the Gaussian distribution can be written using the reparametrization trick as

$$\boldsymbol{z} = \mu_x + \sigma_x \epsilon, \tag{2.1}$$

where $\epsilon \backsim \mathcal{N}(0, 1)$ is an auxiliary noise variable.

It is suggested that the performance of the VAE improves by using a composite model[12]. This means that in addition to the decoder reconstructing the latent representation, another decoder predicts the evolution of time frames, $\tilde{\mathbf{x}}'$. In a composite VAE, the networks are trained using the following mappings where f_{enc} represents encoding and f^1_{dec} and f^2_{dec} represent the two decoders,

$$f_{enc} : \boldsymbol{x} \rightarrow \mu_x, \sigma_x \tag{2.2}$$
$$\text{Sampling} : (\mu_x + \sigma_x \epsilon) \rightarrow \boldsymbol{z} \tag{2.3}$$
$$f^1_{dec} : \boldsymbol{z} \rightarrow \boldsymbol{x'} \tag{2.4}$$
$$f^2_{dec} : \boldsymbol{z} \rightarrow \tilde{\boldsymbol{x}}'. \tag{2.5}$$

A visualization of the composite VAE is presented in figure 2.1.

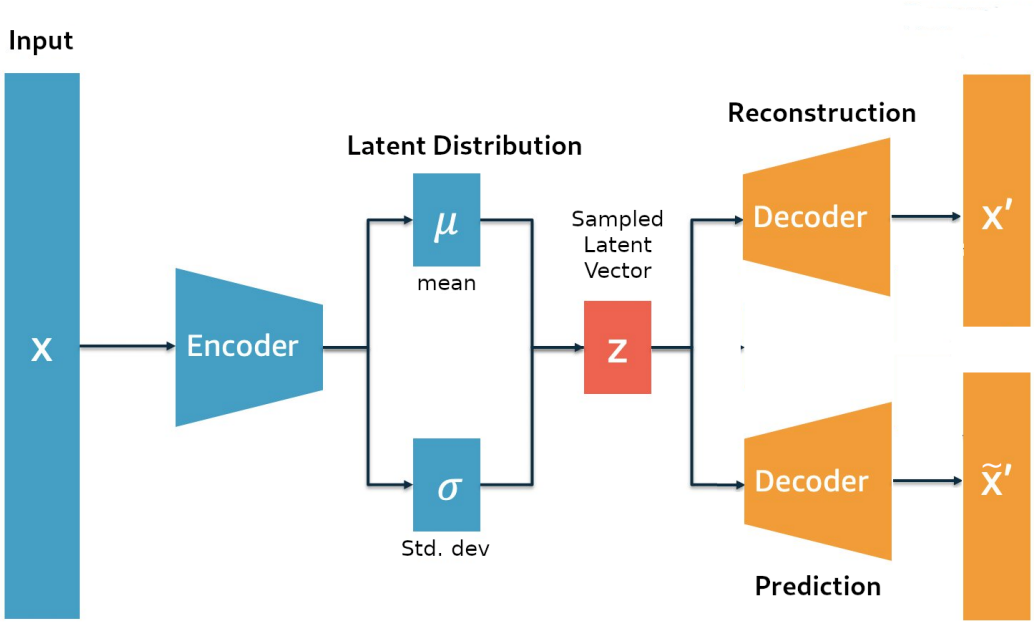


Figure 2.1: Visualization of a composite Variational Autoencoder

2.2.1 Loss function of a composite VAE

When training the VAE, the goal is to minimize its loss function which consists of several parts: The reconstruction loss, which compares the input and its reconstruction and the latent loss (regularization loss) which arises due to the divergence between the latent space distribution and Gaussian distribution. Furthermore, due to the use of a composite VAE, a loss function representing the prediction is also required. Additional loss components can be taken into account in order to favor certain results.

2.2.1.1 Reconstruction and prediction losses

The reconstruction loss compares the input, $\boldsymbol{x}$, with the reconstructed input approximation, $\boldsymbol{x'}$, and computes a similarity measure. A typical similarity measure is mean square error (MSE):

$$\mathcal{L}_{Reconstruction} = ||\boldsymbol{x} - \boldsymbol{x'}||^2 = ||\boldsymbol{x} - f^1_{dec}(\boldsymbol{z})||^2 = ||\boldsymbol{x} - f^1_{dec}(\mu_x + \sigma_x \epsilon)||^2, \tag{2.6}$$

where f^1_{dec} is the reconstruction decoder that creates the approximation $\boldsymbol{x'}$ from the latent representation $\boldsymbol{z}$ which in turn is sampled using equation 2.1. Similarly, the prediction loss is also represented using MSE. However, instead of comparing the input and its reconstruction, the output from the prediction decoder is compared with the actual evolution of the input:

$$\mathcal{L}_{Prediction} = ||\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}'||^2 = ||\tilde{\boldsymbol{x}} - f^2_{dec}(\boldsymbol{z})||^2 = ||\tilde{\boldsymbol{x}} - f^2_{dec}(\mu_x + \sigma_x \epsilon)||^2, \tag{2.7}$$

where $\tilde{\boldsymbol{x}}$ denotes the actual evolution of the input, f^2_{dec} is the prediction decoder and $\tilde{\boldsymbol{x}}'$ denotes the estimated prediction.

2.2.1.2 Regularization loss

The regularization loss is represented by the Kullback-Leibler (KL) divergence between the latent distribution and the Gaussian distribution. It encourages the approximate posterior to be close to the prior and thereby regularizes the latent space. The KL divergence loss can, according to Kingma & Welling [11], be approximated as

$$\mathcal{L}_{KL} = \frac{1}{2}\sum_{j=1}^{J}\left(1 + \log(\sigma_j^2) - \mu_j^2 - \sigma_j^2\right), \tag{2.8}$$

2.2.1.3 Additional loss component

An additional loss function can be taken into account in order to improve the clustering ability of the latent space[13], specifically a k-means clustering objective. The minimization aspect of the k-means algorithm can, according to H. Zha et. al. [14], be reformulated as a trace maximization problem. The solution to the trace maximization problem provides a lower bound solution that can be used as a loss component.

Given a data matrix $\boldsymbol{z}$, the k-means objective has the form,

$$\mathcal{L}_{k-means} = Tr(\boldsymbol{z}^T\boldsymbol{z}) - Tr(\boldsymbol{A}^T\boldsymbol{z}^T\boldsymbol{z}\boldsymbol{A}), \tag{2.9}$$

where Tr stands for the matrix trace which is defined as the sum of elements on the main diagonal of a matrix. Moreover, $\boldsymbol{A}$ denotes the so called cluster indicator matrix which is set to an arbitrary orthogonal matrix. The expression above represents the sum-of-squares cost function which is to be minimized when performing k-means. Said minimization problem is equivalent to the following trace maximization problem.

$$\max_{A} Tr(\boldsymbol{A}^T\boldsymbol{z}^T\boldsymbol{z}\boldsymbol{A}), \quad s.t. \quad \boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}. \tag{2.10}$$

This trace maximization problem has a closed form solution and according to the Ky Fan theorem[15], computing the sum of the largest k eigenvectors of the Gram matrix, $\boldsymbol{A}^T\boldsymbol{A}$, gives a lower bound for the minimum of the k-means objective presented in equation 2.9.

2.3 Recurrent Neural networks

A recurrent neural network (RNN) is a type of artificial neural network that uses sequential or time series data and it can be trained to hold a memory of the past. Image 2.2 presents a compressed recurrent network and a network unfolded in time where h_t is the hidden state, x_t is the input and o_t is the output at time t. To avoid vanishing and exploding gradients in a RNN, gated recurrent units (GRUs) can be implemented in the hidden states h_t. The vanishing-exploding gradients problem is the result of gradients growing exponentially when training the network and which results in large updates in the neural network weights. This means that the model

is unstable and ineffective. GRUs avoid this issue through the use of "gates" that control the flow of information.

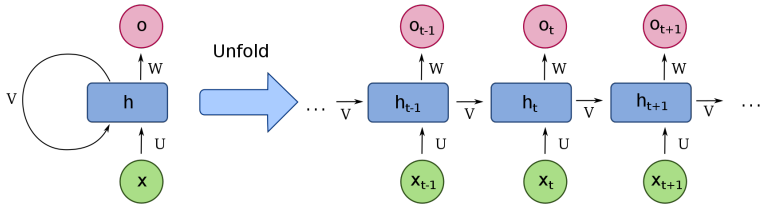


Figure 2.2: Visualization of a recurrent neural network. *Source:* [16]

2.3.1 Bidirectional RNNs

In bidirectional RNNs, the future time steps in a sequence are used to improve the accuracy of the network. A bidirectional RNN therefore consists of a forward layer and a backward layer. Figure 2.3 presents a visualization of a bidirectional RNN.

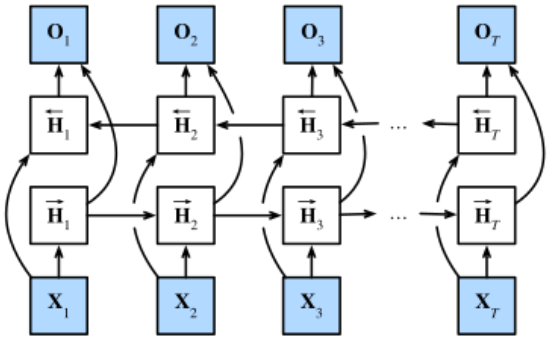


Figure 2.3: Visualization of a bidirectional recurrent neural network. *Source:* [17]

2.3.2 Dilated RNNs

In dilated RNNs a dilated recurrent skip connection has been introduced to alleviate gradient problems and extend the range of temporal dependencies[7]. A network layer has a skip length $s^{(l)}$ that is defined as the dilation of layer l. Figure 2.4 presents two different RNN layers with skip length $s^{(l)} = 4$.

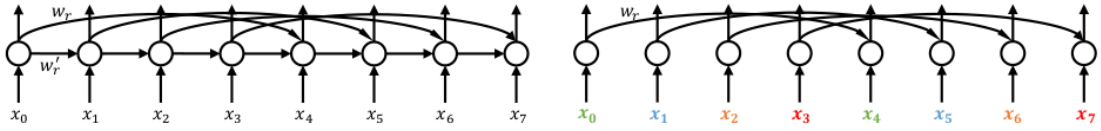


Figure 2.4: A single layer RNN with recurrent skip connections (left) and dilated recurrent skip connections (right). *Source:* [7]

As can be seen in the figure, the dilated (d) and regular (r) skip connections in layer

l at time t can be represented as, respectively,

$$c_{d,t}^{(l)} = f(x_t^{(l)}, c_{d,t-s^{(l)}}^{(l)}), \tag{2.11}$$

$$c_{r,t}^{(l)} = f(x_t^{(l)}, c_{r,t-1}^{(l)}, c_{r,t-s^{(l)}}^{(l)}). \tag{2.12}$$

The transition function f denotes a RNN cell and its output and is typically modeled as long short-term memory (LSTM) or a gated recurrent unit (GRU) [3]. The obvious difference between the two skip connections is that the dilated skip connection does not depend on $c_{t-1}^{(l)}$. The documentation for Dilated RNNs suggests an exponentially increasing dilation throughout the network where the starting dilation affects the performance of the network. This type of architecture is presented in figure 2.5.

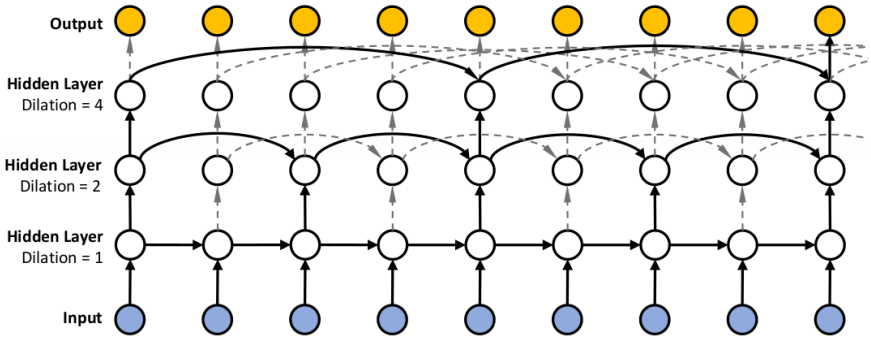


Figure 2.5: Visualization of dilated RNN with multiple layers and exponentially increasing dilation. *Source:* [7]

2.4 k-means clustering

The k-means clustering algorithm aims to partition data into k clusters where each data point belongs to the cluster with nearest mean. More specifically, the algorithm can be described as:

1. Choose the number of clusters (k)
2. Place the centroids c_1, ... , c_k randomly.
3. Repeat steps 4 and 5 until convergence
4. **for** each data point x_i **do**
   Find the closest centroid to x_i and assign the point to that cluster
   end for
5. **for** each cluster j=1, ..., k **do**
   Calculate the new centroid which is the mean of all points assigned to the cluster.
   end for

The k-means clustering algorithm requires the number of clusters, k, to be set before-hand.

2.5 Uniform Manifold Approximation and Projection

Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualization purposes. The parameters that affect the UMAP embedding are number of neighbors and minimum distance. The number of neighbors parameter balances the trade-off between global and local structures in the embedding by determining the size of the local structures that the UMAP will consider when learning the structure of the data. The minimum distance parameter determines how closely points are allowed to be packed.

3
Data and Methods

3.1 Datasets

For the main part of this project, two datasets were used. However, further along into the project, two more datasets were introduced which allowed for comparison between different types of data. The different datasets are presented and described below and images from each dataset are presented in figure 3.1.

3.1.1 VAME data

The VAME data was used in the previous project that created the model that is implemented and altered in this project[18]. In addition to the video, the pose estimations were available where the bodyparts that were tracked were nose, all paws and beginning of tail. Since it was known that the model performs well on the VAME data, the VAME data results were used as a benchmark when comparing with the results from the other datasets. Moreover, it allowed for comparison of results when the model was altered.

The VAME video is approximately 20 minutes long and filmed from below with 25 fps. The contrasts in the video are such that the paws are clearly highlighted which makes them visible for the entirety of the video except for when the rat is standing on its hindlegs or occasionally when the rat is grooming.

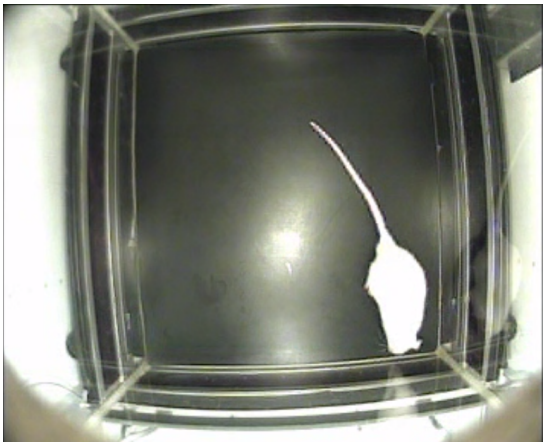
3.1.2 Irlab data I

This dataset consists of 18 one hour videos of varying quality, all filmed from above with 25 fps. The pose estimations for these videos were produced before the commencement of the project. The rats are more or less completely still during the the last half of most of the videos and the tracking points in Irlab data I are center, nose, all paws, and start-, mid-, and endpoint tail.

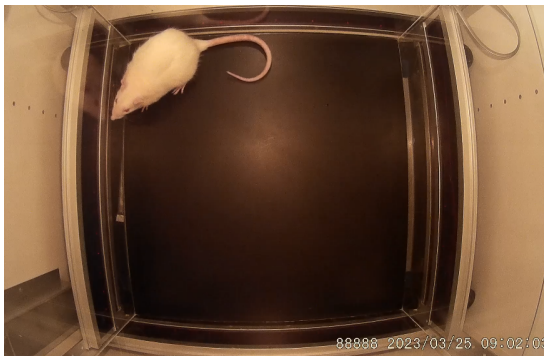
3.1.3 Irlab data II

Irlab data II consists of one video which was collected about halfway through the duration of this project. Much like Irlab data I, it was filmed from above. However, unlike Irlab data I, it was filmed with 30 fps and the quality of the video is significantly better, as can be seen in the images in figure 3.1. Moreover, much like in the

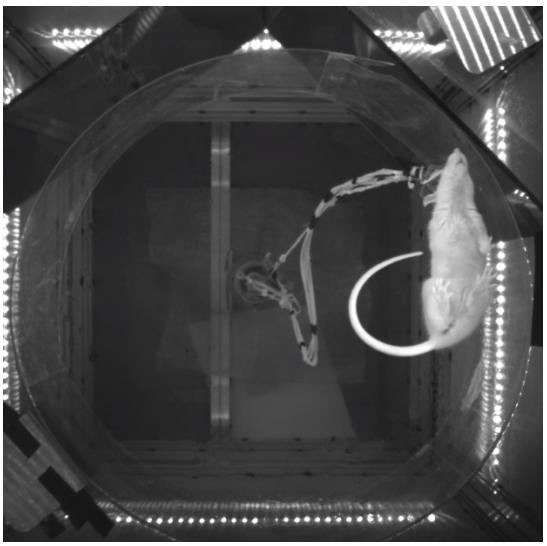
(a) VAME data



(b) Irlab I data



(c) Irlab II data



(d) Lund data

Figure 3.1: Images from each dataset used in the project

videos in the Irlab I dataset, the rat is inactive the second half of the video. The points to be tracked are nose, all paws, and start-, mid-, and endpoint tail.

3.1.4 Lund data

Two videos were shared by researchers in Lund who also perform animal testing on rats. These videos are, like the VAME video, filmed from below, however, these rats have a smaller containment than the mouse in the VAME video. Moreover, the rats in these videos have electrodes attached to their brains which are connected with tubing that is fed through a hole at the top of the containment. The rats are inactivate for some time in both videos. The tracking points are nose, all paws, and start-, mid-, and endpoint tail.

3.2 Pose estimation

The pose estimation for the different datasets was performed using DeepLabCut. The extent of labeled frames that were used when training DeepLabCut for the VAME data is unknown. For Irlab data I, 10 frames were labeled from each video, in other words 180 labeled frames in total. For Irlab data II, 250 frames were labeled and for the Lund data, 100 frames were labeled from each video, therefore 200 frames in total.

The pose estimation data consists of the x and y coordinates of the different body parts at each time frame in the video. Moreover, each body part also has a likelihood parameter which gives the probability of the tracking point being placed correctly at each timeframe.

3.3 Identifying behavioral patterns

The idea behind the pattern identification process is to look at sequences of poses by isolating a number of poses using a sliding window. The time window moves one time frame at a time and thereby isolates a new sequence of poses. The sequences of poses result in high dimensional data and in order to cluster different behavioral patterns, this data needs to be reduced. By first training a Variational Autoencoder, the data could be encoded into a latent representation with reduced dimensions which was then clustered into behavioral patterns using k-means clustering.

3.3.1 Data pre-processing

In the classification process, the position of the rat in the containment and which way it was facing was not of interest. Therefore, the pose data was aligned egocentrically which means that the centerpoint of the rodent is origo of the coordinate system. Moreover, the vector between the snout and start of tail was always in the same direction. This was performed by computing a rotation matrix and rotating the resulting frame around the center between nose and start of tail.

The likelihood parameter that the pose estimation tool outputs was used to determine which values in the pose data were wrong, either due to the point being misplaced or the body part not being visible. The points with a likelihood value below a certain threshold were linearly interpolated or given a set value. Linear interpolation is a method of curve fitting between two points and is useful in tracking when the values between two points are unknown. If the known points given by the coordinates (x_0, y_0) and (x_1, y_1), the linear interpolant is the straight line between these points. For a value x in the interval (x_0, x_1), the y value is given from the equation

$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}. \tag{3.1}$$

Rearranging this equation gives the following relationship between y and x,

$$y = \frac{y_0(x_1 - x) + y_1(x - x_0)}{x_1 - x_0}. \tag{3.2}$$

The coordinates for the body parts around which the coordinates were egocentrically aligned were interpolated before the alignment. The other body part coordinates were interpolated after alignment. The threshold used to determine visibility was 90%.

Before the data was used to train the Variational Autoencoder, unit standardization was performed. Moreover, 10% of the data was reserved for testing.

3.3.2 Composite Variational Autoencoder

The input data consisted of pose estimation sequences obtained through the use of a sliding window. Using a composite VAE, the data was encoded into latent vectors with reduced dimensions. In other words, the input data consisted of a set of n egocenctrically aligned multivariate time series $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$. Each time series consisted of $2m \times T$ real values where m is the number of body parts and T is the size of the sliding window. The encoder in the VAE was a neural network that encoded a mean vector $\boldsymbol{\mu}$ and a standard deviation vector $\boldsymbol{\sigma}$ and the latent representation, $\mathbf{z}$, was sampled from a Gaussian distribution with mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$. The reconstruction decoder decoded reconstructions of the input and the prediction decoder predicted the evolution of time frames. The size of the prediction window was chosen to be half of the sliding window. The size of the sliding window varied but was typically between 1 and 2 seconds long. The number of bodyparts typically ranged between 4 and 10.

Since the input data is temporally dependent, Recurrent Neural Networks (RNNs) were used as encoders and decoders. The decoders were bidirectional RNNs and two different networks were implemented in the encoders; a bidirectional RNN and a dilated RNN. These were implemented separately and the performance of the the model for the different encoders was compared. The bidirectional RNN allowed for usage of future time steps in a sequence when estimating output. The dilated RNN

could lead to an improved treatment the data on multiple time scales, allowing for detection of longer time dependencies.

The bidirectional RNNs had two layers with a hidden layer size of 256 and the dilated RNN consisted of 9 layers with exponentially increasing dilation and hidden layer size 50. The architecture of the bidirectional RNNs was the same as the one used in the model's previous implementation and has been shown to give good results in this task. The architecture of the dilated RNN encoder was determined from its documentation where the same architecture proved successful in other tasks. Gated recurrent units (GRUs) were used as transition functions in the RNNs in the decoders and encoders.

K. Luxem et. al.[3] used latent dimension of 30 and the sliding window size was set to 30. These parameter values were used as a jumping-off point. However, the parameters were tweaked in order to find the best values for the new datasets.

3.3.3 Training the VAE

When training the VAE, the aim was to minimize its loss function which consisted of the following parts,

$$\mathcal{L}_{total} = \mathcal{L}_{reconstruction} + \mathcal{L}_{prediction} + \mathcal{L}_{KL} + \mathcal{L}_{clustering}. \tag{3.3}$$

The reconstruction loss, $\mathcal{L}_{reconstruction}$ and the prediction loss $\mathcal{L}_{prediction}$ are the MSE losses of the two decoders. Moreover, the KL loss, $\mathcal{L}_{KL}$, is the Kullback-Leibler divergence between the latent distribution and Gaussian distribution. Finally, the clustering loss, $\mathcal{L}_{clustering}$, is a k-means objective which improves the clustering ability of the latent space. For each epoch, the model was tested on the test data and the model was saved if the loss value had improved. If there was no improvement for 50 epochs, the training process was considered to have converged and the training stopped. During training, the prediction loss was only used when estimating the total loss of the model on the training dataset. The prediction loss was not used when estimating the total loss of the model on the testset. This was done because the purpose of the prediction loss was to steer the training and regularize the latent space. However, the goal was not to find the best prediction model and therefore the prediction loss was not used when estimating the test loss during the training.

The training of the model was done using the Adam optimizer [19] with a fixed learning rate of 0.0005 and all computing was done in PyTorch.

3.3.4 Clustering

After training the VAE, the datasets were encoded into latent vectors using the encoder. The latent vectors can be clustered in order to separate different behaviors[20]. This was done by performing k-means clustering.

It could be helpful to use a hierarchical method that shows which clusters are closely

related and therefore likely clustering the same behavior. This was done by using the probabilities of transition between clusters. The transition probabilities between different clusters can be modeled as a discrete-time Markov chain where the probability of transition into a future cluster is only dependent on the current cluster. This results in a transition probability matrix, $\mathcal{T}$ with dimensions $K \times K$ where K is the number of clusters. Given the set of clusters $B = \{b_1, ..., b_k\}$ resulting from the k-means clustering, the elements of $\mathcal{T}$,

$$\mathcal{T}_{lk} = P(b_k|b_l), \tag{3.4}$$

are transition probabilities from one cluster $b_l \in B$ to another cluster $b_k \in B$. The hierarchical representation of the clustering is obtained by visualizing the Markov chain presented in equation 3.4 as a directed graph, $\mathbb{G}$, consisting of nodes $v, ..., v_K$ connected by edges with transition probability T_{lk}. $\mathbb{G}$ is transformed into a binary tree by iteratively merging two nodes (v_i, v_j) until only the root node v_R remains. Selecting i and j in reach merging step is done by computing the cost function,

$$C_R = \min_{i,j} \left(\sum_{i,j} \frac{U_i + U_j}{\mathcal{T}_{ij} + \mathcal{T}_{ji}} \right), \tag{3.5}$$

where U_i is the probability occurrence for the ith cluster.

K. Luxem et. al.[3] compared the results for different number of clusters, $K = (15, 30, 45)$. When the number of clusters was increased from 15 to 30, the performance increased significantly and when the number increased from 30 to 45, performance increased slightly. Due to the increase in performance from 30 to 45 being small, the number of clusters used in this project was 30 because that is a more manageable number when analyzing the results.

The latent vectors were visualized using UMAP embedding with the parameters $n\_neighbors = 200$ and $min\_dist = 0.1$.

3.3.5 Behavioral classification

Ideally, annotation is performed post-hoc, choosing which behaviors to annotate by looking at the behaviors that the model has separated. However, in the case where the model was unable to cluster different behaviors, pre-determined behaviors were annotated in order to estimate and compare performance. The pre-determined behaviors to annotate were Stationary Exploration, Moving, Still, Rear, Lean on wall and Grooming. The ethogram for these behaviors is presented in table 3.1. Roughly 5-10 minutes of the different videos were annotated. The behaviors of the different clusters were classified as the annotated label that takes up the majority of the annotated cluster.

3.4 Analysis

The performance of the behavioral pattern identification was evaluated by comparing the clustering with the annotated data and computing different scores. Moreover,

Label	Description
Moving	Walking, walk and bend, walk and sniff
Lean on wall	Standing on hindlegs with paw(s) against wall
Still	Completely still, not doing anything
Stationary exploration	Stationary but sniffing or looking around
Rear	Standing on hindlegs without wall support
Groom	Grooming or scratching

Table 3.1: Ethogram of the pre-determined behaviors that were used to annotate some videos

the performance of the VAE was measured by looking at its encoding and decoding ability.

3.4.1 Encoding and decoding ability

The VAE's ability to compress and then decode the input into an approximation was estimated by taking the input and reconstructions from 5 random time windows and plotting these against each other. This allowed for visualization of whether the model had captured the changes in the data.

3.4.2 Comparison with annotation

To evaluate performance, the cluster labels produced by the VAE were compared with the annotation. In particular, two scores were estimated. Firstly, the purity score of the clustering which looks at the extent of data points that do not belong in a cluster. In other words it estimates the extent to which an annotated behavior makes up a cluster. The formula for purity score for a cluster c_j is

$$\text{purity}(c_j) = \frac{1}{n_j} \max_k |\omega_k \cap c_j|, \tag{3.6}$$

where $\Omega = (\omega_1, ..., \omega_K)$ is the set of of manually annotated labels, $\mathbb{C} = (c_1, ..., c_J)$ is the set of labels produced by the VAE method and n_j is the number of annotated frames in cluster j. The purity score is a number between 0 and 1 and it is desirable to have a purity score close to 1.

The second evaluation metric is Normalized Mutual Information (MNI) which looks at the shared information between clusters and is defined as:

$$\text{MNI}(\Omega, \mathbb{C}) = \frac{I(\Omega, \mathbb{C})}{[H(\Omega) + H(\mathbb{C})]/2}. \tag{3.7}$$

Mutual information, I, between Ω and $\mathbb{C}$ is defined as:

$$I = \sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P(\omega_k \cap c_j)}{P(\omega_k)P(c_j)} \tag{3.8}$$

$$= \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N|\omega_k \cap c_j|}{|\omega_k||c_j|}, \tag{3.9}$$

where $P(\omega_k)$, $P(c_j)$ and $P(\omega_k \cap c_j)$ are the probabilities of a data having label ω_k, having label c_j and being in the intersection of ω_k and c_j respectively. Maximum likelihood estimates for probabilities have been used to obtain the second expression. H is entropy and is defined as:

$$H(\Omega) = -\sum_k P(\omega_k)\log P(\omega_k) \tag{3.10}$$

$$= -\sum_k \frac{|\omega_k|}{N}\log\frac{|\omega_k|}{N}. \tag{3.11}$$

3.4.3 Subjective evaluation

Finally, there was a subjective evaluation component which consisted of observation of the timeframes in the different clusters. This allowed for discovery of new behaviors that had not been pre-determined. Moreover, observation was also used by looking at frames that had not been annotated in the different clusters and observing whether the behaviors correlated with the annotation.

4
Results and Discussion

The visibility rate of the different body parts for the different videos is presented in table 4.1. Only the VAME video was used in its entirety. The videos in the other datasets were shortened due to inactive behavior taking up a large part of the videos' duration. The rates presented in tables 4.1 show the visibility rates of the shortened videos in these cases. Moreover, only in the VAME datasets were all bodyparts in the data used when training the model. For the other datasets, the composition of body parts in the training data was changed. The body parts used in the data for the final results are indicated in bold text in the visibility rate tables.

The Irlab datasets were annotated using the pre-determined behaviors presented in table 3.1. The VAME and Lund datasets were annotated post-hoc; these behaviors and their ethograms are presented in tables 4.2 and 4.3. When determining which behaviors to annotate, some of the timeframes in each clusters were visualized and the behavioral pattern that was shown, if any, was used as a label when annotating.

Label	Description
Moving	Walking, walk and bend, walk and sniff
Stationary exploration	Stationary but sniffing or looking around
Paus	A short break in another behavior
Rear	Standing on hindlegs, with or without wall support

Table 4.2: Ethogram of the behaviors in the VAME results

Label	Description
Moving	Walking, walk and bend, walk and sniff
Stationary exploration	Stationary but sniffing or looking around
Still	Completely still, not doing anything
Groom	Grooming

Table 4.3: Ethogram of the behaviors in the Lund results

4.1 VAME data

A barplot comparing the clustering of the VAME results with the annotations is presented in figure 4.1 and the hierarchical tree of the different clusters is presented

Body part	Visibility (%)
Nose	89.4
Front left paw	93.3
Front right paw	94.5
Hind left paw	99.6
Hind right paw	99.9
Start tail	99.9

(a) VAME data visibility rate

Body part	Visibility (%)
Nose	96.6
Front left paw	79.0
Front right paw	79.1
Hind left paw	96.0
Hind right paw	92.0
Start tail	98.5
Mid tail	77.0
End tail	81.3

(b) Lund data visibility rate

Body part	Visibility (%)
Center	100.0
Nose	72.8
Left ear	92.7
Right ear	80.1
Front left paw	14.5
Front right paw	43.5
Hind left paw	14.8
Hind right paw	20.9
Start tail	99.5
Mid tail	93.3
End tail	97.1

(c) Irlab I visibility

Body part	Visibility (%)
Center	90.4
Nose	69.8
Left eye	64.2
Right eye	59.9
Left ear	91.0
Right ear	91.2
Front left paw	3.8
Front right paw	12.1
Hind left paw	52.0
Hind right paw	0
Start tail	42.4
Mid tail	57.8
End tail	84.2

(d) Irlab II visibility

Table 4.1: The visibility rate of different body parts in the different datasets. The body parts written in bold indicate the bodyparts that were included in the training data when obtaining the best solution.

in figure 4.2. Note that the nodes in the tree are in the same order from left to right as the bars in the barplot above it. The tables presenting the purity and NMI scores are presented in table 4.4. The behavior purity scores in table 4.5 were estimated by taking the average purity score of the clusters where each behavior was in majority.

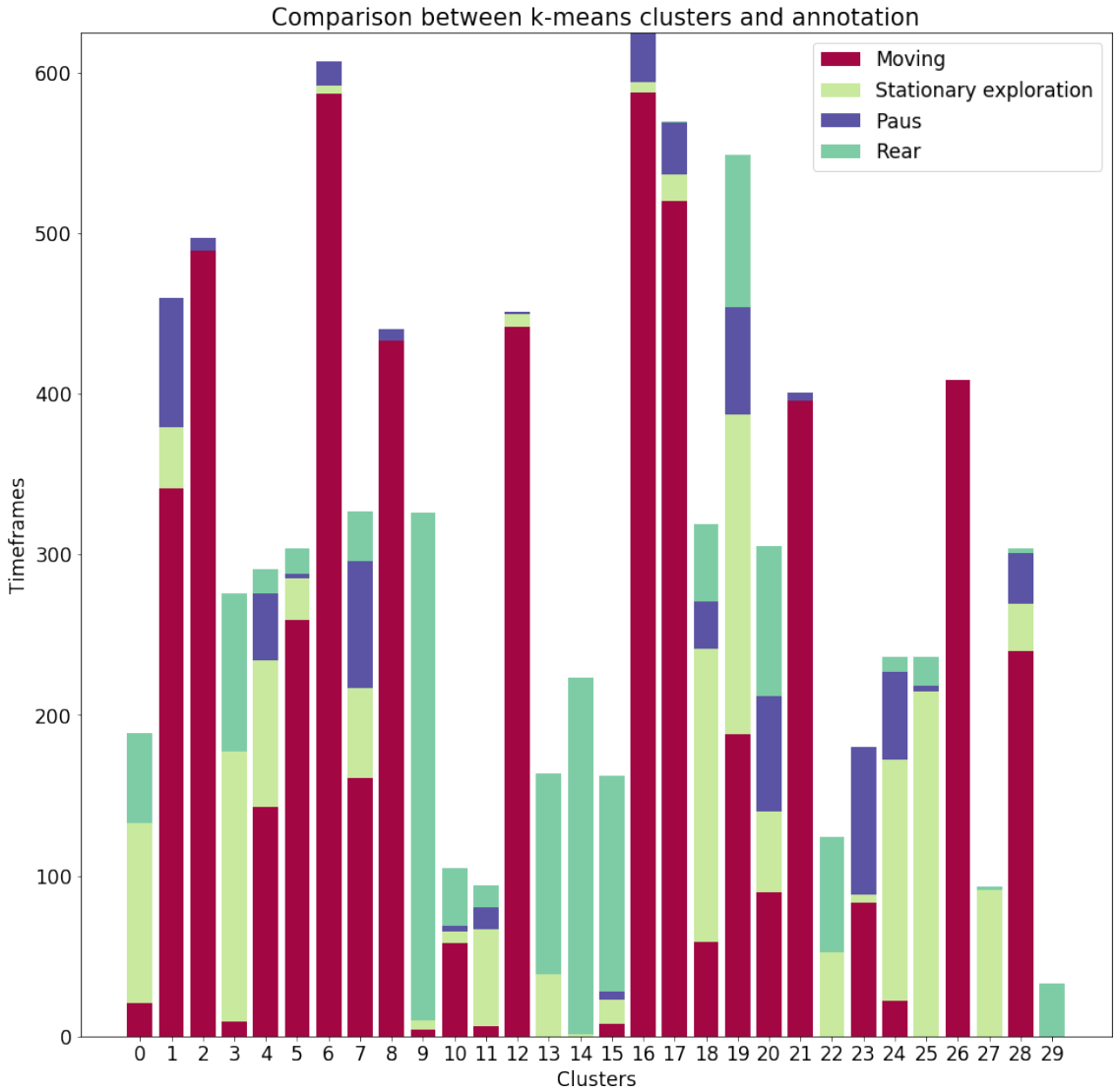


Figure 4.1: Comparison between clustering and annotated data in VAME results. The x axis represents the different clusters, the y axis represents the number of frames and the coloring represents the annotated behavior

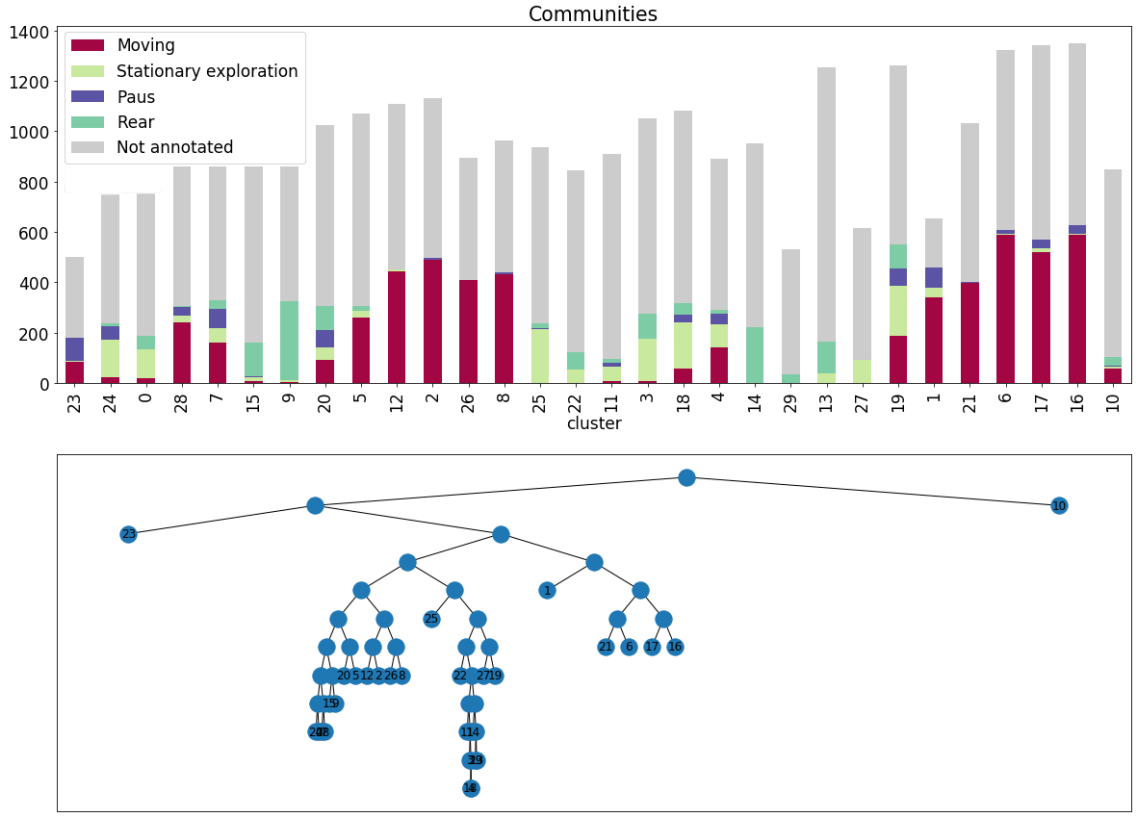


Figure 4.2: The hierarchial clustering of the VAME results. Note that the nodes in the tree are in the same order as the bars in the plot

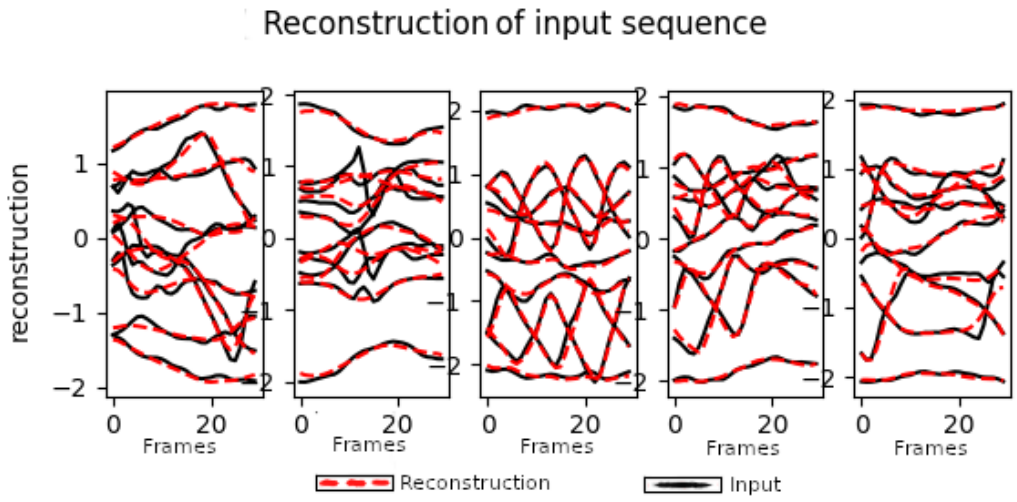


Figure 4.3: Random time window samples that compare the input with the VAE's reconstruction for the VAME data

With a few exceptions, most videos of clusters in the VAME data results showed a pattern that corresponded to a behavior that could be visualized. Moreover, in many clusters, the purity was high, indicating that the model performs well when identifying and separating behaviors. In particular, the clusters that represent the

behavior *moving* have a high average purity score as well as the behavior *rear*. Going beyond identifying the behavior *moving*, the model has clustered variations of *moving*, subbehaviors, into different clusters. For example, different clusters depending on which paw is taking the step.

The relatively low purity score for the behaviors *stationary behavior* and *paus* suggests that the model is not as successful when identifying these behaviors. However, these behaviors are arguably harder to define and annotate than behaviors such as *moving* and *rear* since their definitions are more loose. This will be discussed further in section 4.5.3.

The hierarchical tree presented in figure 4.2 shows how clusters can be grouped together based on how often the mouse transitions between different clusters. It can be seen that the clusters that identify the *moving* behavior can be grouped together into two larger groups, verifying that the model performs well when identifying the behavioral pattern *moving*. Furthermore, many of the clusters that do not contain *moving* are grouped together in the tree in figure 4.2. Most of these clusters contain the behaviors *stationary exploration* and *rear*. This grouping makes sense since the behavior *rear* is arguably a type of exploration and therefore would transition often to or from *stationary exploration*.

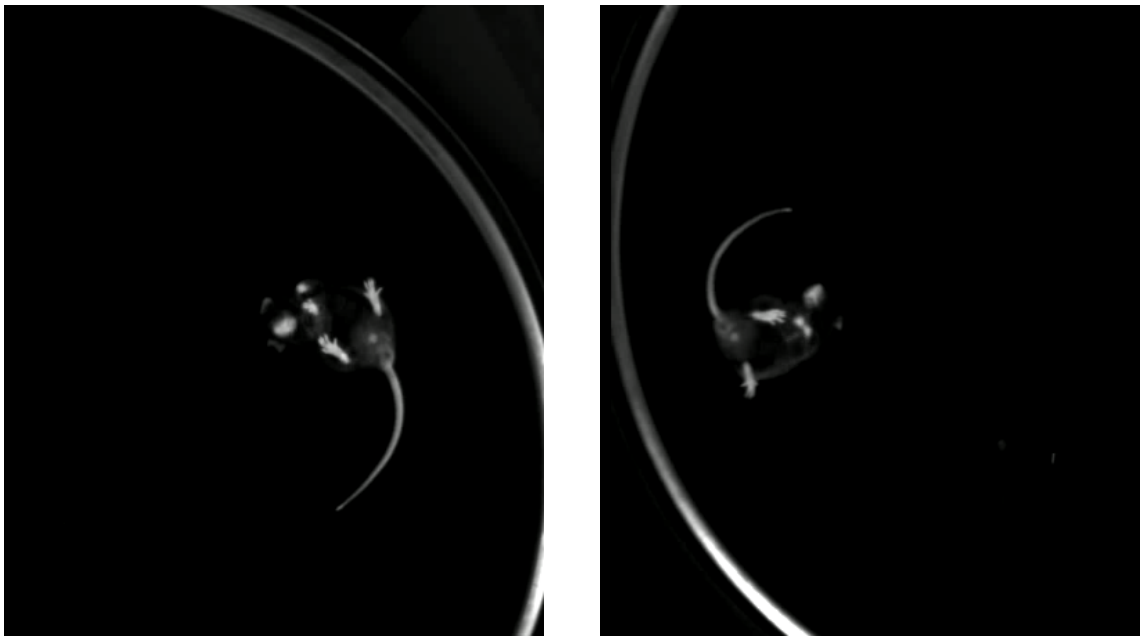


Figure 4.4: Two images that visualize the pattern in cluster 18.

While many clusters show a high purity, some clusters seem to contain all the different behavioral labels without an overwhelming majority of one behavior. Two examples of these clusters are cluster 18 and cluster 19. At first glance, this might seem like the model has failed to separate behaviors but that is not necessarily the case. When visualizing the timeframes in cluster 18, the video shows that the mouse is turning its head in one direction, images of this pattern are presented in figure

4.4. The model has therefore managed to find another subbehavior, however, this pattern is difficult to annotate since it can arise in combination with different behaviors that are used to annotate, for example *moving* and *stationary exploration*. Cluster 18 is an instance where the model seemingly falls short when only comparing with the annotated data but when looking at videos, it has actually found a behavior. Cluster 19 also contains a pattern that is difficult to annotate in a video but easy to identify when seeing the video snippets of the pattern. This highlights a shortcoming in the evaluation method which will be discussed further in section 4.5.3.

The plots presented in figure 4.3 show five random time windows of input and its reconstruction. It can be seen that the changes in the input are captured in the reconstruction and the VAE has therefore successfully reduced the data while keeping its information.

The results from the VAME data are more or less recreations of that of the previous project. However, the results offer insight when comparing with the results from the other datasets. Moreover, since it is known that the model performs well on this data, it offers optimal conditions for comparison with the results from the dilated RNN model. Table 4.4 compares the purity and NMI scores for Dilated RNN and the standard RNN with different sliding window sizes. These scores imply that the standard RNN model performs slightly better than the Dilated RNN model when comparing with annotated data. Putting a number on and comparing the subjective evaluation of looking at videos from different clusters is difficult and it is therefore difficult to see if a dilated model captures behavior better. Nonetheless, the dilated model successfully identifies behavioral patterns as well.

Finally, it is worth noting that the mouse in the VAME video exhibits the behavior *grooming* as defined in table 3.1. However, the mouse does not do this in the part of the video that was annotated. Additionally, the behavior was not separated into its own cluster. Therefore, the model's inability to cluster the behavior *grooming* is not indicated in the results shown in the figures.

	Bidir RNN	Dilated RNN
30 frame window	**0.764**	0.753
50 frame window	0.742	0.716

(a) Purity scores

	Bidir RNN	Dilated RNN
30 frame window	**0.256**	0.247
50 frame window	0.246	0.226

(b) NMI scores

Table 4.4: Purity and NMI scores for the VAME data with different parameter settings and encoders. The bold text indicates the best scores

Behavior	Purity (%)
Moving	83.4
Stationary exploration	66.4
Paus	51.1
Rear	77.7
Total	76.4

Table 4.5: Purity scores for different behaviors in the VAME dataset.

4.2 Irlab I

The video quality of the different videos in the Irlab I dataset varied and therefore mainly one video was used in the model. When using more videos, the model behaved poorly, but this will be discussed further in section 4.5.1. Moreover, all videos in Irlab I showed inactivity in approximately the last half hour of the videos and therefore, only the first half hour in the chosen video was used in order to achieve a more balanced dataset. As can be seen in table 4.1, the visibility rates of the left paws were significantly lower than the right paws and therefore, only the right paws were used in the dataset. Moreover, the middle- and endpoint of the tail were not included in the training dataset.

The results from the Irlab I dataset are presented in figures 4.5 and 4.6. The purity and NMI scores are presented in table 4.6 and the purity scores for different behaviors are presented in table 4.7. The parameter values for the Irlab I are the same as for Irlab II to allow for comparisons. These values were dictated by the Irlab II results since that data yielded better results. The parameter values for the results presented in the figures are latent dimensions of 35 and sliding window size 50.

At first glance, it seems that the model did not identify behavioral patterns successfully for the Irlab I data. In figure 4.5, it can be seen that many frames annotated with the behavior *still* have been placed in clusters 13, 11 and 2. However, other behaviors seem to not be as separated. Moreover, when watching the videos from each cluster, there is no distinguishable repeating pattern. A large part of this project was spent attempting to get the model to work for the Irlab I data without success. For example, the parameters of the model were changed and the body parts that made up the training data were also changed. In the final results, only the right paws were used since they showed a higher visibility rate than the left paws. Moreover, an exploration of the latent space was performed where other clustering algorithms were used in an attempt to find outliers and potentially kernels within clusters with a behavioral pattern. However, this did not yield better results. Figure 4.7 presents random time windows of input data and their reconstructions. It can be seen that the reconstruction does not capture the small changes in the input which suggests that the latent vector does not contain all the information of the input. This might seem like the reason for the model's inability to identify different behaviors, however it is not necessarily the case. Trainings were done with a smaller time window, namely 30 frames, and while they yielded better reconstructions, they did not yield

better scores. This suggests that there is perhaps not enough information in the pose data to separate patterns that correlate with observable behaviors.

While the clustering of the Irlab I data is seemingly quite poor, it does contain some indication of having identified patterns. This can be seen by looking at the hierarchical tree in figure 4.6 where, for example, cluster 13, 12 and 2 have been grouped together and these clusters all contain the behavior *still*. Moreover, the clusters that contain the behavior *moving* seem to be grouped close together. Furthermore, they are grouped separately from the *still* clusters 13, 12 and 2 that are arguably quite different from the behavior *moving*. Much like the behavior *moving*, some clusters containing the behavior *grooming*, clusters 3, 0, 22 and 9, have been grouped together. This means that under closer observation, the results from Irlab I are not as poor as they first might seem. However the behaviors *lean on wall* and *rear* are spread out.

	Bidir RNN	Dilated RNN
50 frame window	**0.655**	0.624
60 frame window	0.646	0.640

(a) Purity scores

	Bidir RNN	Dilated RNN
50 frame window	**0.262**	0.244
60 frame window	0.249	0.252

(b) NMI scores

Table 4.6: Purity and NMI scores for the Irlab I data with different parameter settings and encoders. The bold text indicates the best scores

Behavior	Purity (%)
Moving	43.6
Lean on wall	-
Still	70.1
Stationary exploration	63.6
Rear	100.0
Grooming	71.3
Total	65.5

Table 4.7: Purity scores for different behaviors in the datset Irlab I.

The purity scores for different behaviors are presented in table 4.7. The line for the *lean on wall* behavior is the result of that behavior not being the majority in any cluster. Moreover, the high score of 100% for the behavior *rear* is the result of cluster 7 only containing that label which is not an accurate evaluation of the clustering. This somewhat problematic aspect of the purity score will be discussed

further in section 4.5.3. Furthermore, the behavior *still* has a relatively high purity score which verifies the previous notion that the model has managed to identify *still*.

One issue with the Irlab I data was the video quality. As can be seen in the images in figures 4.14a and 4.14b, the video quality was quite poor and it also varied a lot between videos. The video quality probably resulted in lower precision in the pose estimations which likely affected the model's capabilities and the low quality also made it difficult when annotating behaviors which in turn affected the accuracy of the scores. Moreover, the difference in quality between videos resulted in poor results when using multiple videos to train the model. This will be discussed further in section 4.5.1. The video quality in combination with the angle of the camera resulted in the paws not being visible for the greater part of the video. During pre-procsseing, the data was aligned egocentrically which means that if the paws were not visible, it became difficult to capture when the rat was walking which is probably why the behavior *moving* is not separated much in the clustering. In attempts to bypass this problem, the speed of the rat was added to the training dataset, however, this did not improve the results. Furthermore, the interpolation method was changed so that if a paw was not visible for a long time, its coordinate values were set to a point below the rat's body in an attempt to imitate the paw's actual location. Nonetheless, this did not improve the results either. These attempts illustrate the difficulty of steering this model in order to achieve certain results, for example finding a specific behavior.

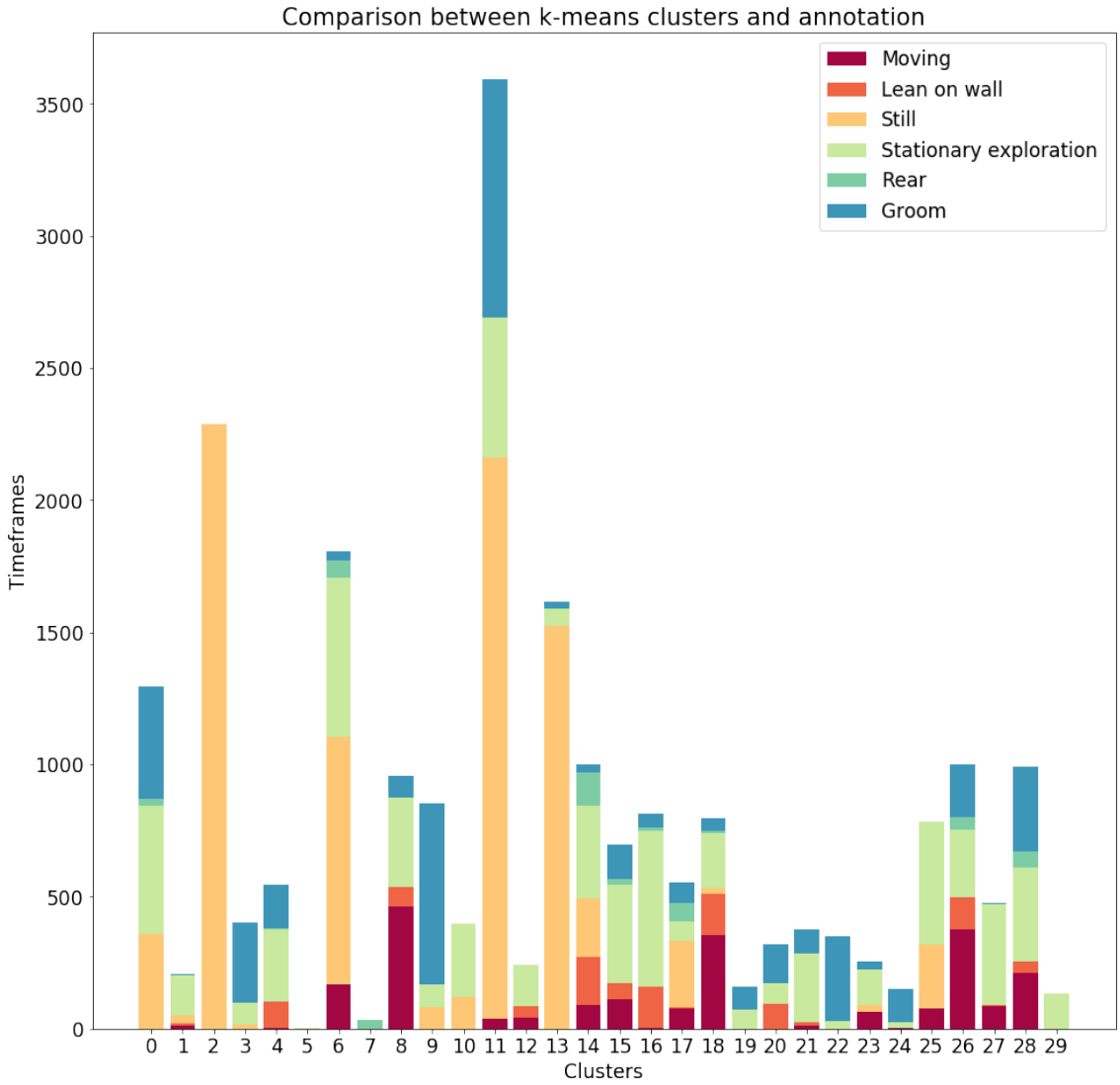


Figure 4.5: Comparison between clustering and annotated data in Irlab I results. The x axis represents the different clusters, the y axis represents the number of frames and the coloring represents the annotated behavior

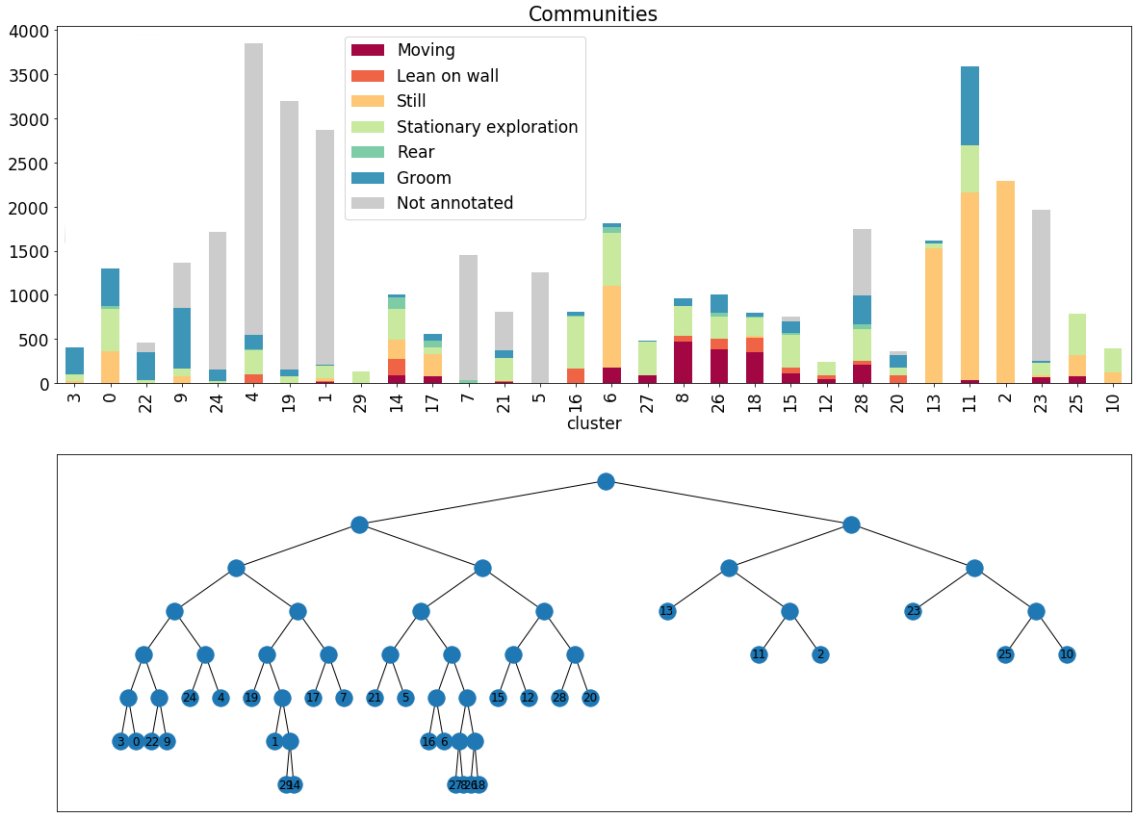


Figure 4.6: The hierarchial clustering of the Irlab I results. Note that the nodes in the tree are in the same order as the bars in the plot

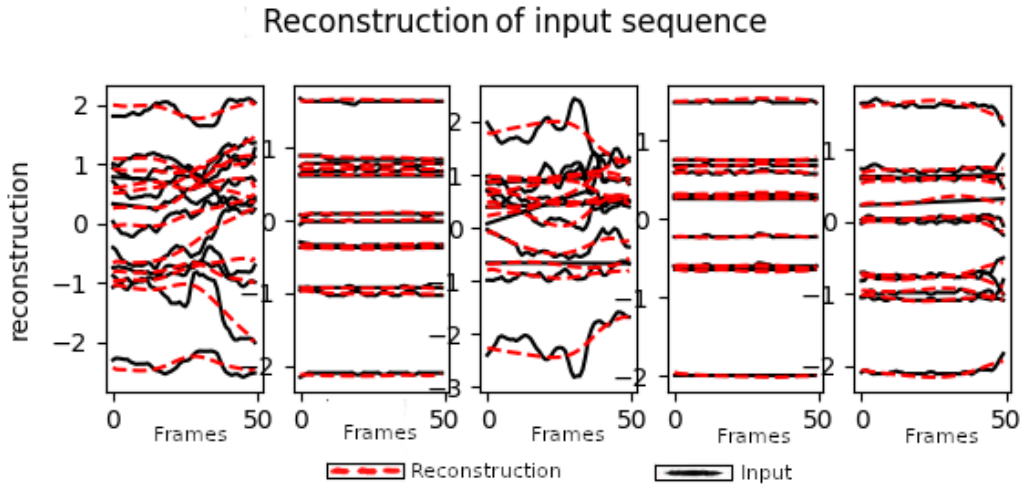


Figure 4.7: Random time window samples that compare the input with the VAE's reconstruction for the Irlab I data

4.3 Irlab II

Much like in the Irlab I video, the rat was inactive a large part of the Irlab II video and therefore only the first half hour was used. The video quality of the Irlab II dataset was significantly better than that of Irlab I which allowed for tracking of more bodyparts, namely the eyes. However, after running the model a couple of times on training data with different body part compositions, it was decided that the performance did not improve when the eyes were included. Moreover, out of the paws, only the paw with the highest visibility rate was included and the midpoint and endpoint of the tail were not included in the dataset.

While the quality of the Irlab II video was better, the visibility rate of the different bodyparts was not necessarily higher. The reason for this could simply be that the bodyparts are visible less time in the Irlab II video than Irlab I. However, the lower visibility rates could also mean that the pose estimation could be improved, perhaps by training the DeepLabCut model using more videos, which the documentation for DeepLabCut suggests. Moreover, the camera fails to capture two walls in the containment so that when the rat is leaning on the wall, it exits the frame. Therefore, another possible improvement in data acquisition would be to use a wider lens.

Even though the visibility rates are lower in Irlab II than Irlab I, the results are significantly better. The barplot comparing annotated data with the clustering is presented in figure 4.8, the hierarchical clustering is presented in figure 4.9 and the scores comparing different time windows and types of RNNs are presented in table 4.8. Moreover, the purity scores for different behaviors are presented in table 4.9. The results presented in the figures were yielded using latent dimensions 35 and sliding window size of 50 frames.

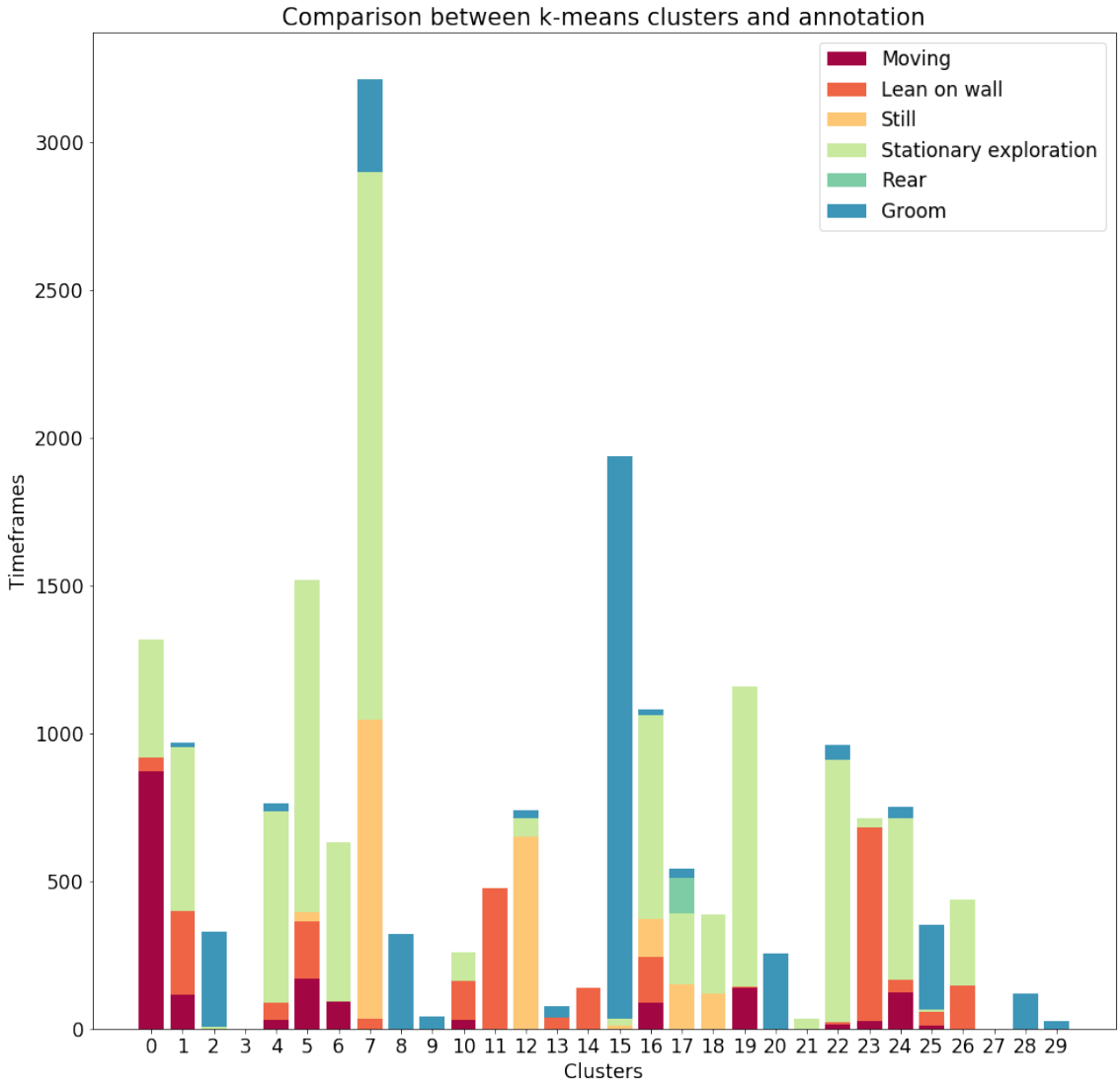


Figure 4.8: Comparison between clustering and annotated data in Irlab II results. The x axis represents the different clusters, the y axis represents the number of frames and the coloring represents the annotated behavior

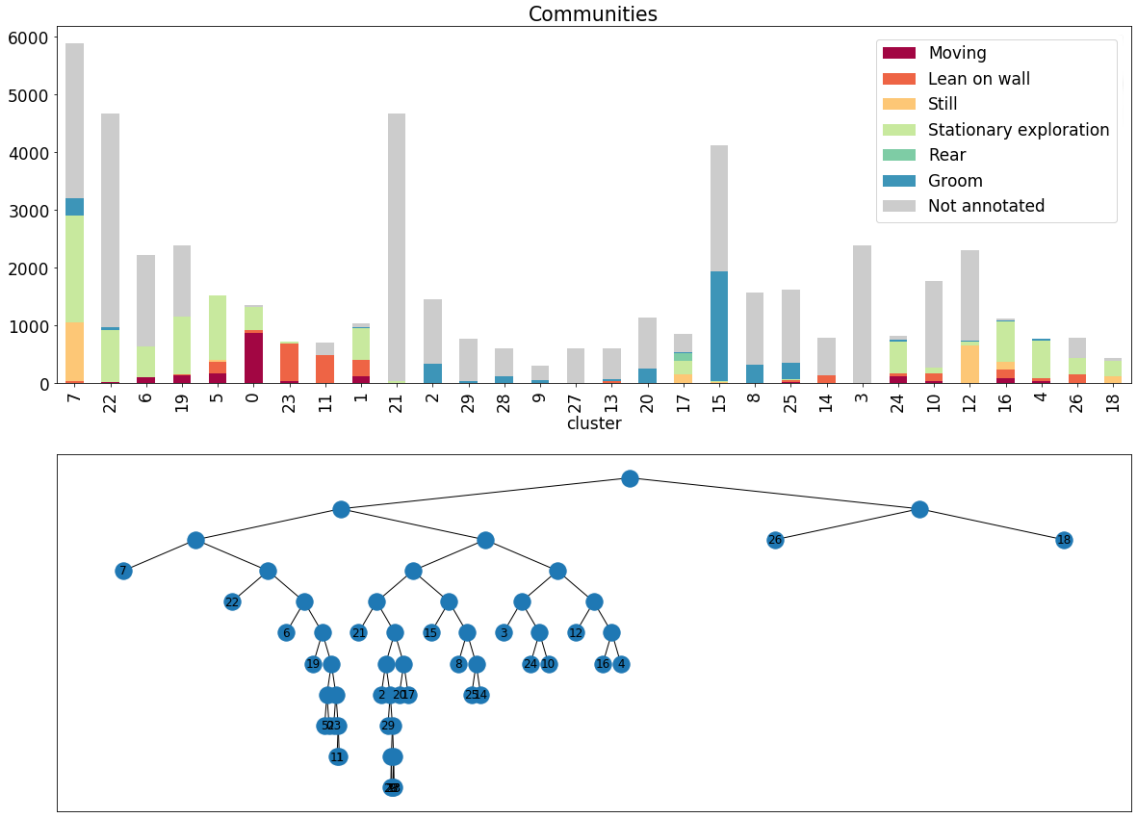


Figure 4.9: The hierarchial clustering of the Irlab II results. Note that the nodes in the tree are in the same order as the bars in the plot

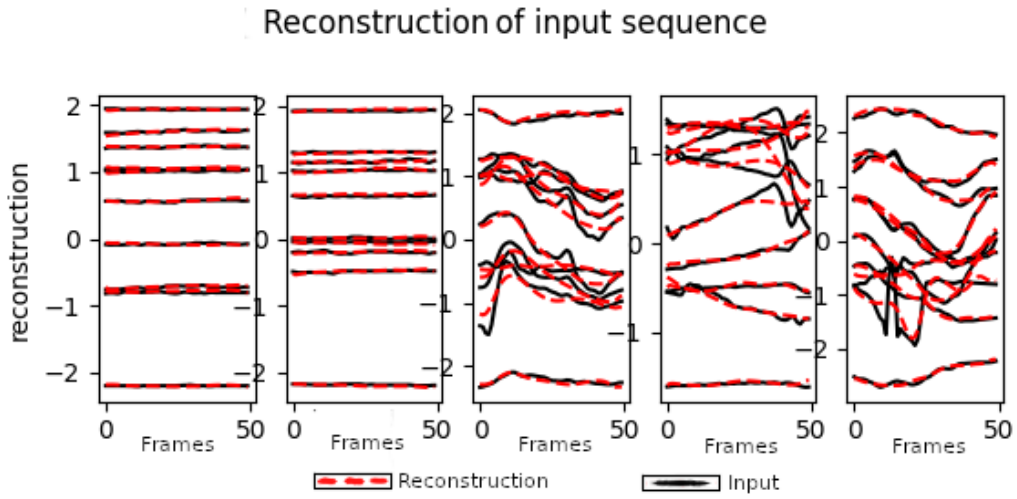


Figure 4.10: Random time window samples that compare the input with the VAE's reconstruction for the Irlab II data

The model appears to successfully identify behavioral patterns in the Irlab II data. In particular, the *grooming* behavior shows a high average purity where almost all the annotated *grooming* frames have been clustered together. The behavior with

the lowest average purity is *moving*. Much like with Irlab I, this makes sense considering the low visibility rates of the paws, however, the model identifies the behavior *moving* significantly better for Irlab II than for Irlab I. In the barplot in figure 4.8, it can be seen that the behaviors *stationary exploration* and *still* are often clustered together. There could be a couple of reasons for this; it could be because the two behaviors sometimes are arguably quite similar, however, it could also be the result of the model failing to capture the rat's movements and therefore interpreting the rat as still. This can be suspected by looking at the model's input and reconstruction in figure 4.10 where some changes in the input have not been captured in the reconstruction, suggesting that the latent vector is lacking information. Much like in Irlab I, the model's reconstruction accuracy improved when removing the body parts with lower visibility, for example not using any paws in the dataset, however this did not yield better pattern identification results, suggesting that at least the hind left paw provides necessary information.

The parameters of the model were tweaked and the bodyparts making up the training dataset were changed until the best solution, the one presented in the results, was reached. However, due to the number of trainings that took place, it was too time consuming and difficult to look at videos from each cluster from every training. Only the annotated data was used to find the optimal solution. The quality of the clustering was verified by looking at videos representing the frames that had not been annotated in each cluster to see if the unannotated frames in the cluster aligned with the annotation. This evaluation showed that some clusters contained a distinguishable behavior that matched the annotated portion of the cluster, for example the unannotated frames in cluster 8 and 25 showed grooming behavior. However, other clusters did not correlate with their annotated frames, for example cluster 14 is annotated as *lean on wall* but the unannotated frames show the behavior *grooming*. This evaluation corresponds well with the hierarchical representation in figure 4.9 where clusters 8, 25 and 14 have been grouped closely in the tree which makes sense since they show the same behavior (even though the annotation for cluster 14 implies a different behavior).

The hierarchical tree in figure 4.9 offers the opportunity to predict the behaviors in clusters that do not contain any annotated frames. For example, cluster 29 is expected to show grooming behavior since it is grouped with other *grooming* clusters. Looking at the videos representing the cluster 29 verifies this expectation. However, cluster 3, which does not contain annotated frames either, does not show an overwhelming majority of a specific behavioral pattern.

The Irlab II dataset was collected more than halfway into the project's duration when it started to become clear the model was unable to separate the behaviors in Irlab I. This means that there was not enough time to perform the same explorations done for Irlab I. Since these explorations proved to not bear fruit for Irlab I, it was assumed that this would also be the case for Irlab II. However, it would be insightful to see what the exploration of Irlab II's latent space would yield. For example, if it would be possible to identify and remove outliers by using a different clustering

algorithm.

	Bidir RNN	Dilated RNN
50 frame window	**0.814**	0.784
60 frame window	0.790	0.716

(a) Purity scores

	Bidir RNN	Dilated RNN
50 frame window	**0.368**	0.344
60 frame window	0.349	0.310

(b) NMI scores

Table 4.8: Purity and NMI scores for the Irlab II data with different parameter settings and encoders. The bold text indicates the best scores

Behavior	Purity (%)
Moving	65.1
Lean on wall	77.5
Still	88.1
Stationary exploration	73.0
Rear	-
Grooming	97.6
Total	81.4

Table 4.9: Purity scores for different behaviors in the datset Irlab II.

4.4 Lund data

The Lund data was, like the Irlab II dataset provided more than halfway into the duration of the project and it was not prioritized when optimizing results. However, since the Lund data was also filmed from below, it offers an opportunity to make comparisons with the VAME data to analyze the model's capabilities. Due to the comparison purpose of the Lund data, the same body parts were used in the training data, namely all that were tracked except for mid and end point of tail. Furthermore, the same parameter values were used in the model for the Lund data as the VAME data, namely latent dimensions of 30 and sliding window size of 30.

The dataset consisted of two videos but only one was used when implementing the model. The reason for this was the same as for Irlab I and will be discussed further in section 4.5.1. The video that was selected for the implementation of the model was the video in which the rat was more active and therefore showing more behaviors. Nonetheless, the rat was still a large part of the video and therefore only half was used.

Even though the videos were filmed from below, like the VAME data, the visibility

rates of the different bodyparts were lower than for VAME. This could be because the lighting contrasts are different in the Lund data than in the VAME data. In the VAME data, the paws are clearly discernible while in the Lund data it is at times more difficult to make out the paws. This can be visualized in images 3.1a and 3.1d. Moreover, in the Lund data, the rats have electrodes attached to their brain that are connected by tubes that are fed through a hole at the top of the containment. These tubes sway and move when the rat is moving and could therefore distract the tracker.

The hierarchical tree presented in figure 4.12 is unbalanced, making it difficult analyze.

	Bidir RNN	Dilated RNN
30 frame window	**0.621**	0.604
50 frame window	0.618	0.597

(a) Purity scores

	Bidir RNN	Dilated RNN
30 frame window	0.125	0.135
50 frame window	**0.163**	0.137

(b) NMI scores

Table 4.10: Purity and NMI scores for the Lund data with different parameter settings and encoders. The bold text indicates the best scores

Behavior	Purity (%)
Moving	67.4
Stationary exploration	52.2
Still	68.3
Groom	-
Total	62.1

Table 4.11: Purity scores for different behaviors in the Lund dataset.

It is worth noting that the model might have yielded better results if the parameters were tweaker more. However, since this dataset was added fairly late into the project, its purpose was to offer comparison possibilities against the VAME data. Therefore, the VAME data dictated the parameter choices for the Lund data.

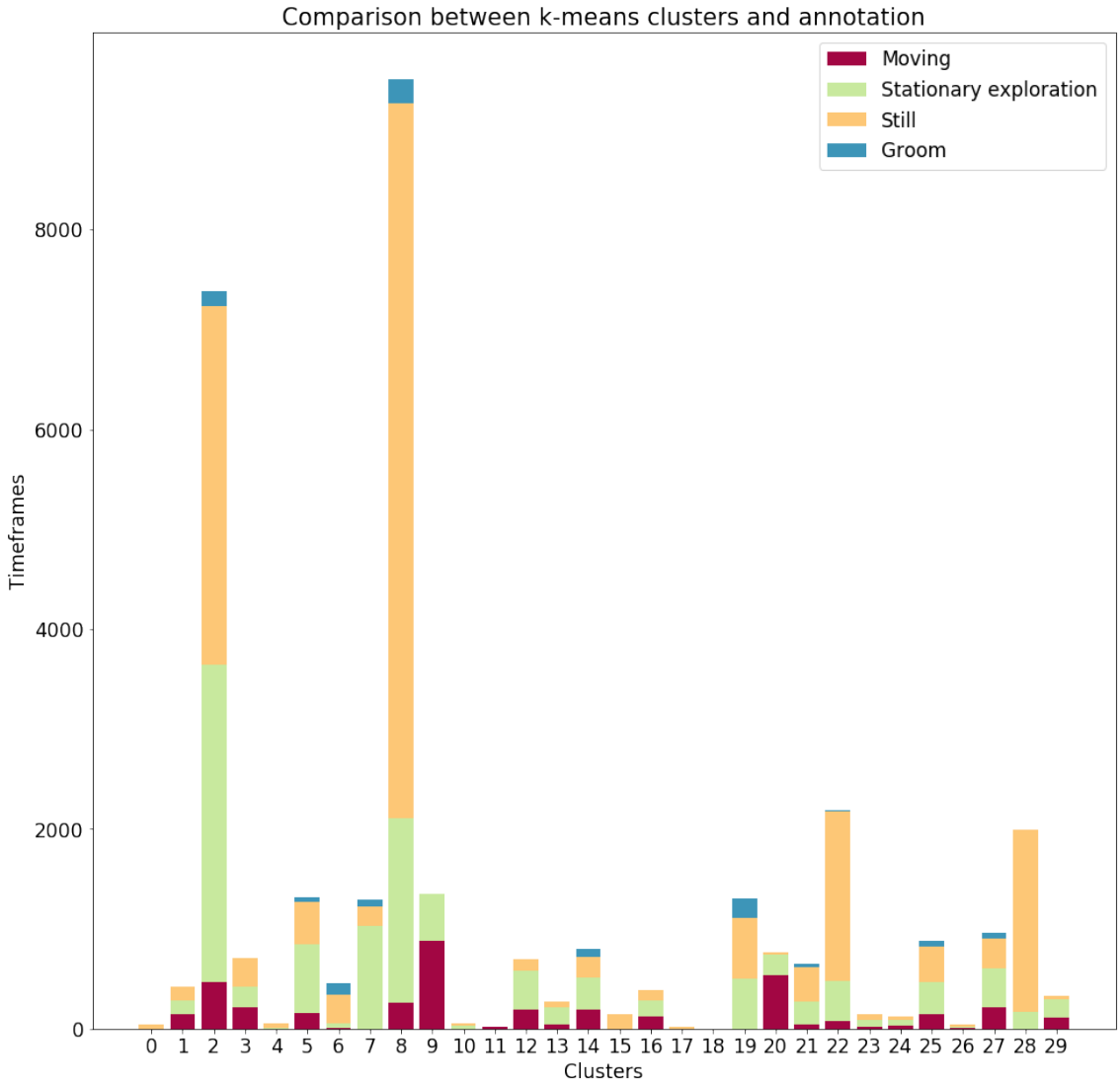


Figure 4.11: Comparison between clustering and annotated data in Lund results. The x axis represents the different clusters, the y axis represents the number of frames and the coloring represents the annotated behavior

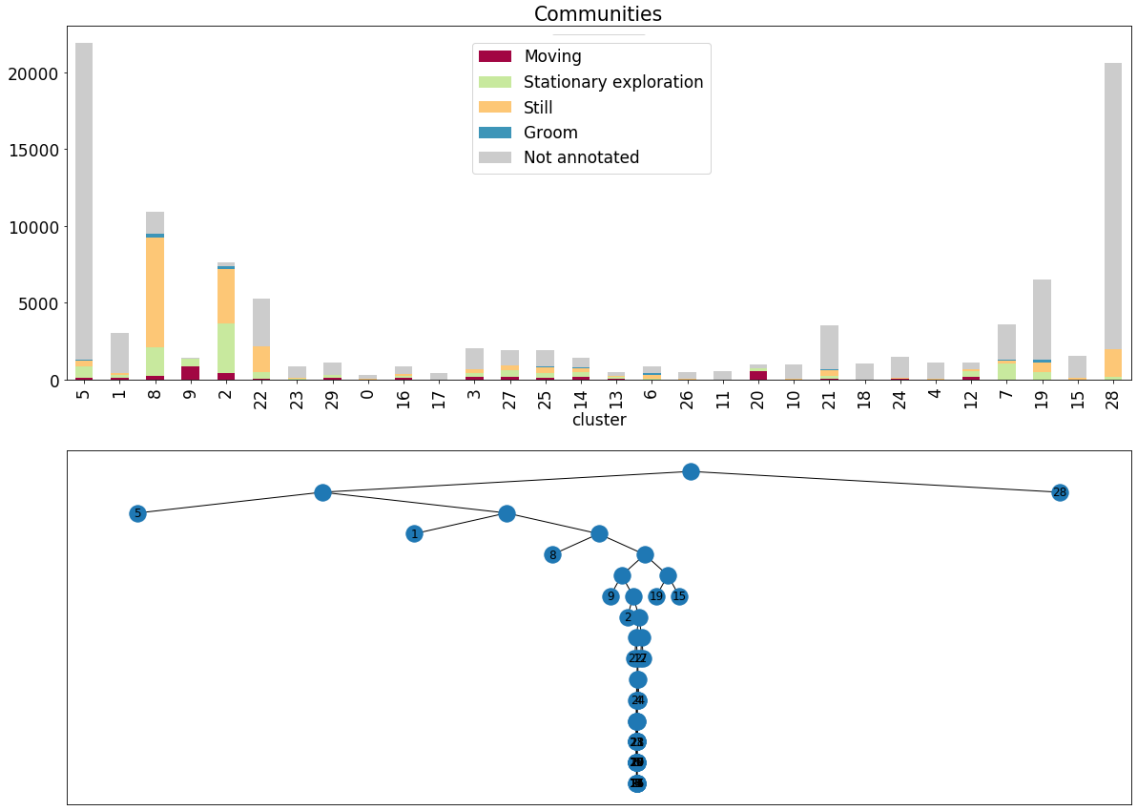


Figure 4.12: The hierarchial clustering of the Lund results. Note that the nodes in the tree are in the same order as the bars in the plot

4.5 Overall performance

Since the datasets differ in both the angle from which they were captured and video quality, it offers an opportunity to compare how the model performs on different types of data. The behavior *moving* had higher average purity in the results from the VAME dataset which was filmed from below than the Irlab datasets and this is reasonable since the paws have a much higher visibility rate. Similarily, the behavior *rear* also has higher purity in the VAME data than in the Irlab datasets, likely for the same reason. Grooming behavior, however, has a much higher purity score in the Irlab II results than in the other datasets. In neither the VAME data nor the Lund data is the grooming behavior separated into its own cluster. The behavior *stationary exploration* is pretty vague and has a lot of variations which might explain why it seems to be difficult to identify across the board.

For the majority of the duration of the project, the only datasets used were Irlab I and VAME. Due to the significant difference in performance between their results, it seemed that the camera angle was crucial in order to get this model to work properly. However, the addition of the Irlab II and Lund datasets put things into perspective. The results from the Irlab II data, which is filmed from above, have the highest scores which suggests that camera quality is just as important, if not more, than the angle from which the rodent is filmed. Moreover, the Lund data,

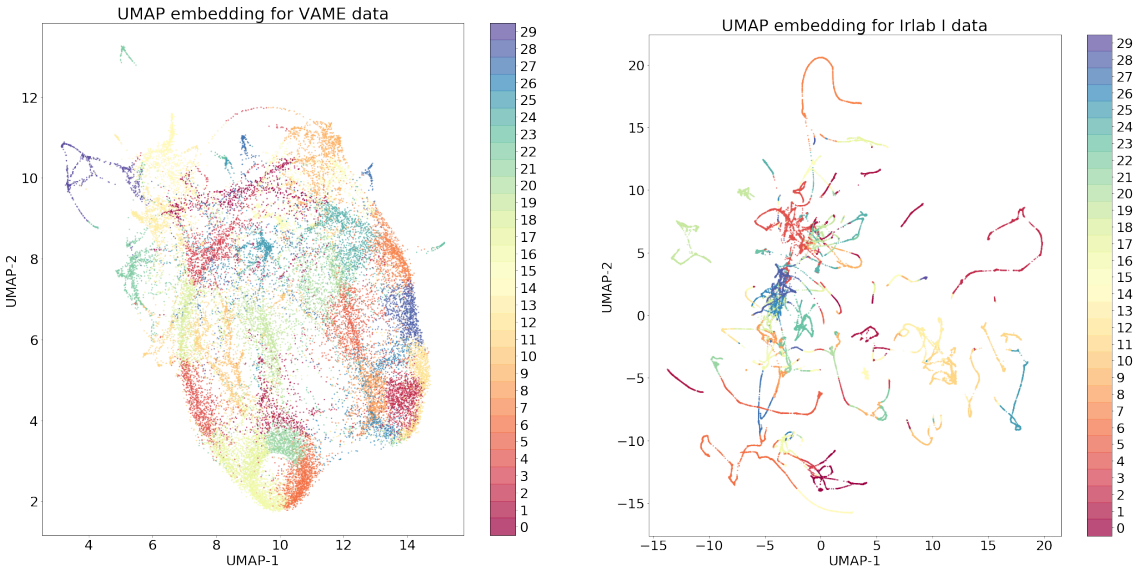
which was filmed from below did not achieve the same performance as the VAME data. The Irlab II results indicate that the model successfully identified the behavior *grooming*, a behavior that the model was unable to identify in the VAME data. Moreover, while the behavior *moving* is somewhat separated in the the results for Irlab II, the model identified the behavior, and variations of it, better in the VAME data. This suggests that the angle from which the data is collected affects the type of behaviors that the model identifies.

The results from the Lund data differ a great deal from the VAME data results in performance even though both videos are captured from below. This could be due to the pose data in the Lund dataset not being of the same quality as the pose data in the VAME dataset, however, the body part visibility rate in the Lund data is seemingly better than Irlab II but Irlab II performed significantly better. Since the Lund data and its results were not explored much, the reason for the difference in quality is unclear. One possible reason could be that the rat's general behavior in the Lund video differs from the other datasets. For the most part, the rat is still which could mean that the dataset is unbalanced and the model adapts to inactive behaviors. Moreover, the tube to which the electrodes are attached seems to affect the rat's movements by pulling the rat's head slightly when swaying which could affect the natural behavioral pattern. However, the result from the Lund dataset makes one thing clear; capturing the rodent from below does not guarantee good results from this model.

The plots showing the input data and reconstructed data presented in figures 4.3, 4.7 and 4.10 indicate a difference between the Irlab datasets and the VAME data. Namely, some of the plots of the random Irlab time window samples contain horizontal lines, indicating inactive behavior. The VAME samples, on the other hand, do not show the same type of plots. This reflects the rodents' behaviors in the videos because the mouse in the VAME video is never inactive whereas the rats in the Irlab datasets are. This could mean that the model adapted to the inactive sequences when training on the Irlab datasets, resulting in inferior reconstructions for the active behavior.

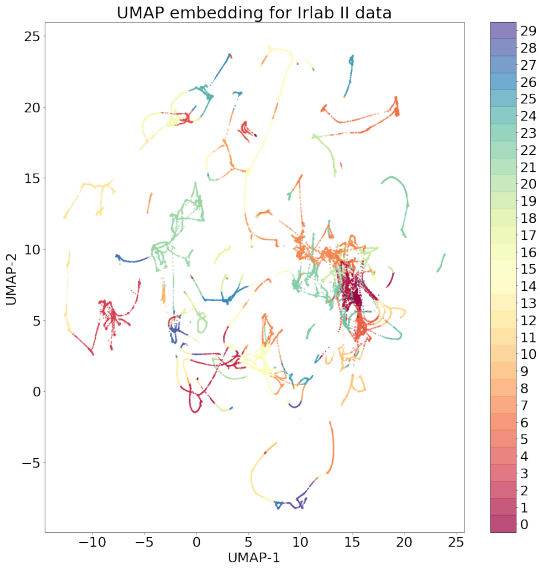
The UMAP embeddings for the different datasets are presented in figure 4.13. The purpose of the embedding was to get a visualization of the latent vector and the VAME data UMAP embedding stood out from the others. The embedding for the VAME data shows a cloud-like shape while the embeddings for the other datasets appear more straggly in structure and contain a lot of lines. The datapoints that are connected as lines are also connected in time, suggesting that datapoints that are close in time have been lumped together. While the UMAP embedding does not necessarily accurately represent the latent vector, its structure would offer an explanation to the unbalanced cluster sizes in the datasets with spidery embeddings. Moreover, when comparing videos of the clusters, the VAME clusters typically consisted of short snippets with a clear repeating pattern while the snippets were a lot longer in for example the Irlab II cluster videos, suggesting that timeframes' closeness in time affected the clustering. However, the difference in the UMAP structures

and shapes could simply be due to the UMAP parameter values.

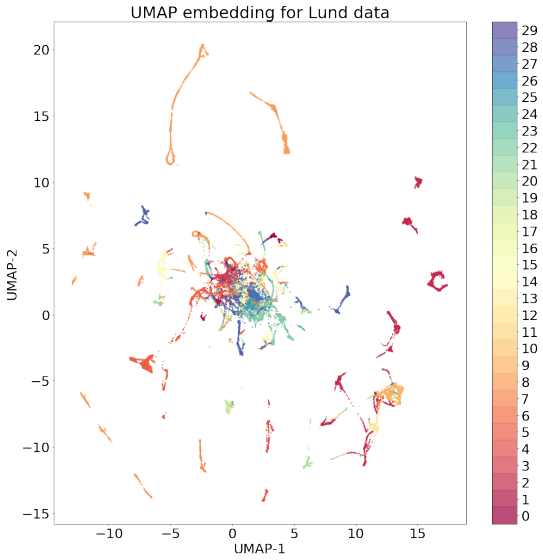


(a) VAME

(b) Irlab I



(c) Irlab II



(d) Lund

Figure 4.13: UMAP embeddings of the latent vectors from each dataset where the color indicates cluster

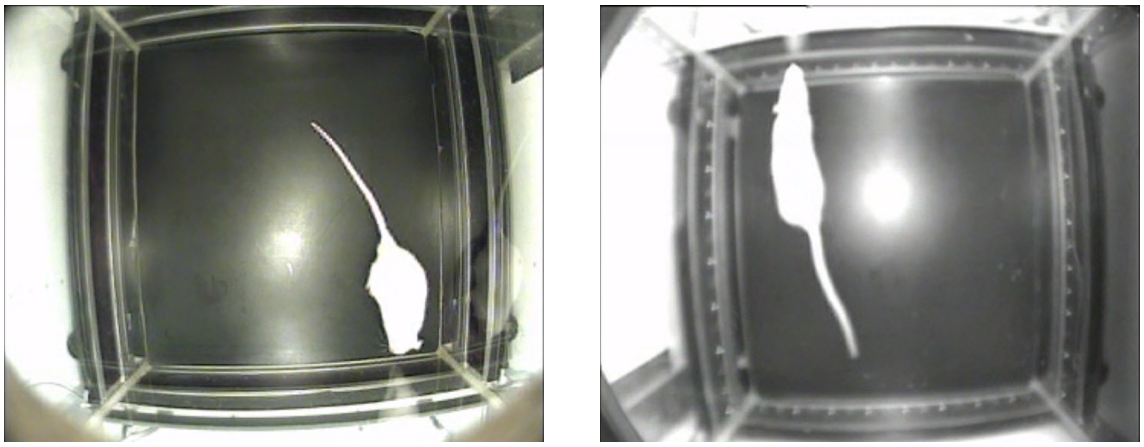
Only taking score into consideration, the Irlab II results have the best performance. However, as will be discussed further in section 4.5.3, the scores are not necessarily the only indicator of performance. When looking at the video snippets from the VAME clusters, almost all clusters contain a distinguishable pattern, however, the pattern might be difficult to annotate so its performance is not reflected accurately in its scores. Moreover, when observing the videos in the different clusters, the model seems to have found distinguishable variations of behaviors, subbehaviors,

to a greater extent in the VAME data. Furthermore, looking at the barplots that show the full cluster sizes, it can be seen that the cluster sizes for the VAME data are more even than in the Irlab II data. This results in some clusters in the Irlab II data having only a small portion of annotated data which could lead to a high purity score.

4.5.1 Multiple videos

The Irlab I and Lund datasets were the only datasets with more than one video. Though only one of the videos from the dataset was used when obtaining their presented results, the use of multiple videos was explored. These results presented an issue with the model; when clustering the latent space with more than one video, the videos could be clustered completely separately, depending on the videos used. The plots presented in figure 4.15 show UMAP embeddings of the latent space created from training on multiple videos where the colors indicate different videos in the latent vectors. Figure 4.15a presents an embedding created from 4 videos from Irlab I and in figure 4.15b is the embedding of the two videos in the Lund data. It is clear that different videos show varying degrees of separation. This means that the videos have been clustered separately and that the model has separated different videos instead of or as well as different behaviors. This could be due to difference in pose data quality, for example the images presented in figure 4.14 are taken from different videos in the Irlab I dataset and show a significant difference in quality which in turn likely affected pose quality. However, the two videos in the Lund dataset are separated yet the quality of its videos is seemingly the same. The reason for the separation in the Lund dataset could be due to difference in behavior in the two videos. In one video, the rat is more or less completely inactivate for the duration of the video while the other rat shows more activity. It could also be due to differences in rat sizes. Nonetheless, the embeddings in figure 4.15 show that using multiple videos in this video could lead to issues. However, it is worth noting that separation is not necessarily the case since two videos in figure 4.15a are overlapping. Moreover, the results from both the Irlab I data and the Lund was quite poor and using multiple videos from better performing datasets might yield different results.

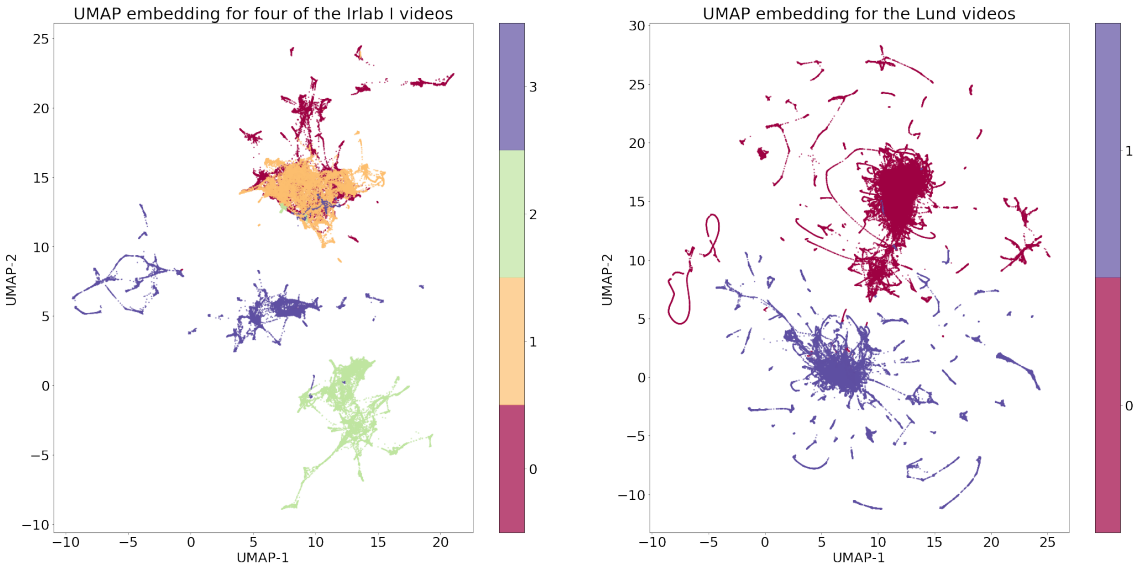
An additional potentially problematic aspect when using multiple videos is if it is decided to only use the body parts with the highest visibility rate in the dataset since these might differ for different rats, especially in videos filmed from above with a fish eye lens, like the Irlab datasets. The visibility rate of the paws is affected by the individual rat's behavior and its position in relation to the camera.



(a) Image from the Irlab I video used in the results

(b) Image from an Irlab I video that was not used

Figure 4.14: Images from two different videos in the Irlab I dataset.



(a) UMAP embedding of four Irlab I videos.

(b) UMAP embedding of the two Lund videos

Figure 4.15: UMAP embeddings where the color indicates the different videos in the latent space.

4.5.2 Dilated RNN

While the scores for the results obtained using the dilated RNN encoder were close to those of the bidirectional RNN encoder, they were nonetheless consistently lower. This might not necessarily mean the results were worse as the dilated RNN encoder might have encoded different behavioral patterns. However, the timeframes from clusters obtained through the dilated RNN encoder were observed and no new behavior was seen. Nonetheless, the architecture of the dilated RNN was not explored

much and considering how close the dilated scores were, a potential better solution using dilated RNN cannot be completely ruled out.

4.5.3 Evaluating performance

A challenge in this project was to evaluate performance of the behavioral pattern identification process. Ideally, it would be possible to observe distinguishable behaviors in each cluster from which to determine behaviors to annotate. To evaluate performance of the model, the data is annotated using the behaviors found in clusters to then estimate scores. However, issues arise if there are no distinguishable behaviors in the clusters, in which case it is difficult to know which behaviors to annotate in order to evaluate the performance and compare it with another result. This was solved by using pre-determined behaviors in this project but this comes with drawbacks. Due to the unsupervised approach, it is not known beforehand which behavioral patterns will be clustered and using pre-determined behaviors to estimate quality might result in inaccurately favoring suboptimal results. New behaviors that are not included in the pre-determined behaviors will go unnoticed if only this method of evaluation is used. Moreover, the annotated timeframes in a cluster might be quite few in comparison to the cluster size. For example, cluster 21 in Irlab data II presented in figure 4.9 has a purity score of 1 since all its annotated frames are have the label *stationary exploration* but only a small fraction of the total cluster has annotation and the score is therefore an unfair representation of the cluster. A subjective method of evaluation is therefore required for this model wherein the timeframes in the different clusters are observed.

An additional challenge when evaluating performance arose when dealing with the VAME data. Namely, the behavioral patterns in the clusters might be easy to identify when seeing them collected but actually annotating them in the video is trickier. Take the behavior *paus* for example; this behavior is easy to identify when seeing a collection of video snippets of a rodent pausing. However, annotating the behavior in a video is difficult, when does the behavior start and end and how does it differ from *stationary exploration*? This results in annotation of behaviors seen in clusters not actually aligning with the clusters. Once again, the scores comparing the annotation with the clustering might not offer an accurate evaluation.

Subjective evaluation obviously has many drawbacks. It is more or less impossible to fairly compare two sets of 30 clusters to determine which clustering is better. Moreover, it is time consuming to watch videos from each cluster for every results. Therefore, the final results presented in this report are not necessarily the best, they are simply the results with the highest scores and after having reached the results, subjective evaluation was used to offer more insight to the performance.

Finally, comparing performance between datasets is challenging because the animals exhibit different behaviors. The Lund rats do not rear or lean on the walls, likely due to their electrode helmets, the VAME mouse is never completely inactive, and the Irlab II rat almost never rears without wall support. It is impossible to know

how the model would interpret these behaviors in the respective videos, therefore making it difficult to properly compare performance between the datasets.

4.5.4 The unsupervised approach

The unsupervised approach to behavioral classification has advantages. There is, ideally, no need for annotated data and the unsupervised approach also allows for discovery of new behaviors that might not be expected. Nonetheless, it also comes with disadvantages, it might find behavioral patterns that are not desirable to identify or patterns that do not correspond to an observable behavior. Furthermore, it might end up clustering a behavior into subbehaviors, separating different patterns that correlate with the same behavior. Additionally, the "black box" nature of a deep learning method makes it difficult to steer the method to find pre-determined behaviors which was attempted with Irlab's datasets. When dealing with pre-determined behaviors it could be useful to use a bottom-up approach where each behavior is analyzed first to see how the pose data looks. This was done for Irlab data I in an attempt to improve the model's results. Namely, it was attempted to separate the behavior *still* in a pre-processing step and only use timeframes where the rat was actually doing something when training the model. While it was possible to discern the *still* behavior in a pre-processing step, it did not improve the classification accuracy of the rest of the data. Moreover, it created a dataset that was not continuous in time which is an undesirable input in the RNN. However, since it was possible to discern *still* behavior through a simple pre-processing step, it shows that when dealing with pre-determined behaviors it could be convenient to use a bottom-up approach. Nonetheless, when finding effects of substances on behavior for a pharmaceutical purpose it could be important to use an unsupervised approach in order to discover new or unexpected behaviors.

4.6 Potential further work

The model performs well on the right data and while Dilated RNNs did not improve the performance, there are other possible improvements. These were not explored or implemented in the duration of this project since they were outside the chosen scope.

4.6.1 Loss function

The loss function in this project consisted of the MSE loss for the two decoders, the KL divergence loss and a k-means objective. This loss function could potentially be improved. An additional loss component that could be used is a contrastive learning approach and triplet losses which aim to push dissimilar examples further from each other. Furthermore, when looking at the loss curves for the different trainings (A.1), the dominating loss component was the MSE loss while for example the k-means objective was small in comparison. Scaling the k-means objective so that it has greater effect on the total loss function could perhaps result in better clustering.

4.6.2 VAE framework

While the VAE framework has been shown to be successful in unsupervised behavioral classification, other VAE frameworks have been shown to outperform the VAE in other unsupervised learning tasks. For example, the Maximum Mean Discrepancy VAE (MMD-VAE)[21] which encourages a more informative latent space and decreases the risk of overfitting.

Since the paws show such a low visibility rate when the rat is filmed from above, using only pose data might remove too much information from the video. It would therefore be interesting to see how a VAE model that uses image data as input performs in comparison with one that uses pose data. A VAE model that uses image data as input is the Clockwork VAE (CW-VAE)[22]. The CW-VAE is used to make predictions of video sequences. While the goal of behavioral classification through a VAE is not necessarily to make predictions, the framework could probably be used for a dimensional reduction and clustering purpose.

5
Conclusion

An unsupervised approach to behavioral pattern identification during CNS drug research could offer valuable insights. It would allow for discovery of unexpected behaviors that might be the result of a CNS compound's effect on the brain. Furthermore, being able to perform said unsupervised approach without disturbing the animal during ongoing trials by only using a camera would allow for easy acquisition of data. This would result in obtaining more information from each trial during animal testing which in turn could lead to the research goal being reached faster. Moreover, it would lead to smaller need for animal testing. This kind of behavioral pattern identification would therefore be a valuable resource during research that makes use of animal testing, such as the CNS drug research at Irlab Therapeutics.

Variational embedding through the use of a VAE cosisting of Bidirectional RNNs or Dilated RNNs as encoders was implented on four different datasets in this project. Two of them were filmed from above and two were filmed from below. This allowed for analysis of the model's capabilities depending on data as well as on type of encoder. The performance of the identification process was evaluated by estimating scores that compared clustering and annotation. The results showed that the model can successfully identify behavioral patterns from above as well as below, clustering *grooming* behavior and *lean on wall* behavior in one video filmed from above and *moving* and *rear* behavior in one video filmed from below. The results further show that the angle from which the data is collected affects the type of behaviors that are identified.

All videos did not provide successful results. One of the videos filmed from above performed significantly worse than the other and the same applied for one of the two videos filmed from below. Given that both camera angles resulted in successful and unsuccessful results, it can be determined that the model can perform successfully from both camera angles. The differences in performance seem to be correlated with video quality which in turn affects the pose estimations that are used as input. The datasets that yielded the superior results are both of better video quality than their underperforming counterparts. Furthermore, it is suspected that an unbalanced dataset will cause less successful results, for example if the animal is still for the majority of the video. The dataset that yielded the worst results was more unbalanced in terms of behavior than all other videos.

The results demonstrated an aspect of the model that could lead to problems in the future. Namely, when attempting to train on multiple videos, the model is sen-

sitive to differences in the videos, for example slight differences in camera angle or camera quality. This means that in order to use multiple videos in the model, great care has to be taken when collecting the data so that the model does not differentiate between videos. If this model is to be used in the future, it would need to be able to take multiple videos as input in order to give results efficiently.

The aforementioned advantage of the unsupervised approach is that it will be able to identify behavioral patterns without the need for pre-determined annotated behaviors. The disadvantage on other side of this is of course that the model is difficult to steer if there is a desire to identify specific behaviors. The best performing dataset in this project according to scores, Irlab II, was evaluated using pre-determined behaviors. However, since the unsupervised approach is not meant to find pre-determined behaviors, this form of evaluation is not accurate for the unsupervised approach. A more accurate way to go would be to observe the patterns that the model has found and evaluate from there. This form of evaluation, however, is subjective and challenging to use when comparing different results. When only observing scores, the Irlab II data performed better. Using subjective evaluation, the VAME data might have yielded better results.

In conclusion, using a deep variational framework to reduce sequences of pose estimations with the goal of clustering behavioral patterns showed successful results for the datasets with better video quality. Estimating performance of this model is challenging and therefore determining the best dataset for the model is difficult. However, the successful results from Irlab data II indicate that variational embedding is a promising method with great potential for behavioral pattern identification at Irlab Therapeutics.

Bibliography

[1] *Parkinson's disease: Causes, symptoms, and treatments*, https://www.nia.nih.gov/health/parkinsons-disease, Accessed: 24.05.2022.

[2] *Irlab therapeutics: Parkinson's disease (pd)*, https://irlab.se/therapeutic-focus/parkinsons-disease, Accessed: 24.05.2022.

[3] K. Luxem, F. Fuhrmann, J. Kuersch, S. Remy, and P. Bauer, *Identifying behavioral structure from deep variational embeddings of animal motion*, May 2020. DOI: 10.1101/2020.05.14.095430.

[4] A. I. Hsu and E. A. Yttri, "An open source unsupervised algorithm for identification and fast prediction of behaviors," *bioRxiv*, 2021. DOI: 10.1101/770271. eprint: https://www.biorxiv.org/content/early/2021/03/19/770271.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2021/03/19/770271.

[5] A. Mathis, P. Mamidanna, K. M. Cury, and et al, "Deeplabcut: Markerless pose estimation of user-defined body parts with deep learning," *Nature Neuroscience*, vol. 21, pp. 1281–1289, 2018.

[6] A. B. Wiltschko, M. J. Johnson, G. Iurilli, R. E. Peterson, J. M. Katon, S. L. Pashkovski, V. E. Abraira, R. P. Adams, and S. R. Datta, "Mapping sub-second structure in mouse behavior," *Neuron*, vol. 88, no. 6, pp. 1121–1135, 2015, ISSN: 0896-6273. DOI: https://doi.org/10.1016/j.neuron.2015.11.031. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0896627315010375.

[7] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. Hasegawa-Johnson, and T. S. Huang, *Dilated recurrent neural networks*, 2017. arXiv: 1710.02224 [cs.AI].

[8] A. B. Gilbert, S. Kalouche, and P. S. Stanford, "Marker-less pose estimation," 2017.

[9] J. Brownlee. (). "Introduction to dimensionality reduction for machine learning," [Online]. Available: https://machinelearningmastery.com/dimensionality-reduction-for-machine-learning/. (accessed: 24.05.2022).

[10] A. Anwar. (). "Difference between autoencoder (ae) and variational autoencoder (vae)," [Online]. Available: https://towardsdatascience.com/difference-between-autoencoder-ae-and-variational-autoencoder-vae-ed7be1c038f2. (accessed: 24.05.2022).

[11] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, 2014. arXiv: 1312.6114 [stat.ML].

[12] N. Srivastava, E. Mansimov, and R. Salakhutdinov, *Unsupervised learning of video representations using lstms*, 2016. arXiv: 1502.04681 [cs.LG].

[13] Q. Ma, J. Zheng, S. Li, and G. W. Cottrell, "Learning representations for time series clustering," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/1359aa933b48b754a2f54adb688bfa77-Paper.pdf.

[14] H. Zha, X. He, C. Ding, M. Gu, and H. Simon, "Spectral relaxation for k-means clustering," in *Advances in Neural Information Processing Systems*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds., vol. 14, MIT Press, 2001. [Online]. Available: https://proceedings.neurips.cc/paper/2001/file/d5c186983b52c4551ee00f72316c6eaa-Paper.pdf.

[15] K. Fan and A. J. Hoffman, "Some metric inequalities in the space of matrices," 1955.

[16] *File:recurrent neural network unfold.svg*, https://commons.wikimedia.org/wiki/File:Recurrent_neural_network_unfold.svg, Accessed: 2022-06-03.

[17] *Bidirectional recurrent neural networks*, https://d2l.ai/chapter_recurrent-modern/bi-rnn.html, Accessed: 2022-06-03.

[18] *Vame github*, https://github.com/LINCellularNeuroscience/VAME, Accessed: 2022-06-03.

[19] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2014. DOI: 10.48550/ARXIV.1412.6980. [Online]. Available: https://arxiv.org/abs/1412.6980.

[20] V. Prasad, D. Das, and B. Bhowmick, *Variational clustering: Leveraging variational autoencoders for image clustering*, 2020. arXiv: 2005.04613 [cs.CV].

[21] "Infovae: Balancing learning and inference in variational autoencoders," vol. 33, pp. 5885–5892, Jul. 2019. DOI: 10.1609/aaai.v33i01.33015885. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/4538.

[22] V. Saxena, J. Ba, and D. Hafner, *Clockwork variational autoencoders*, 2021. DOI: 10.48550/ARXIV.2102.09532. [Online]. Available: https://arxiv.org/abs/2102.09532.

A

Appendix 1

A.1 Loss curves

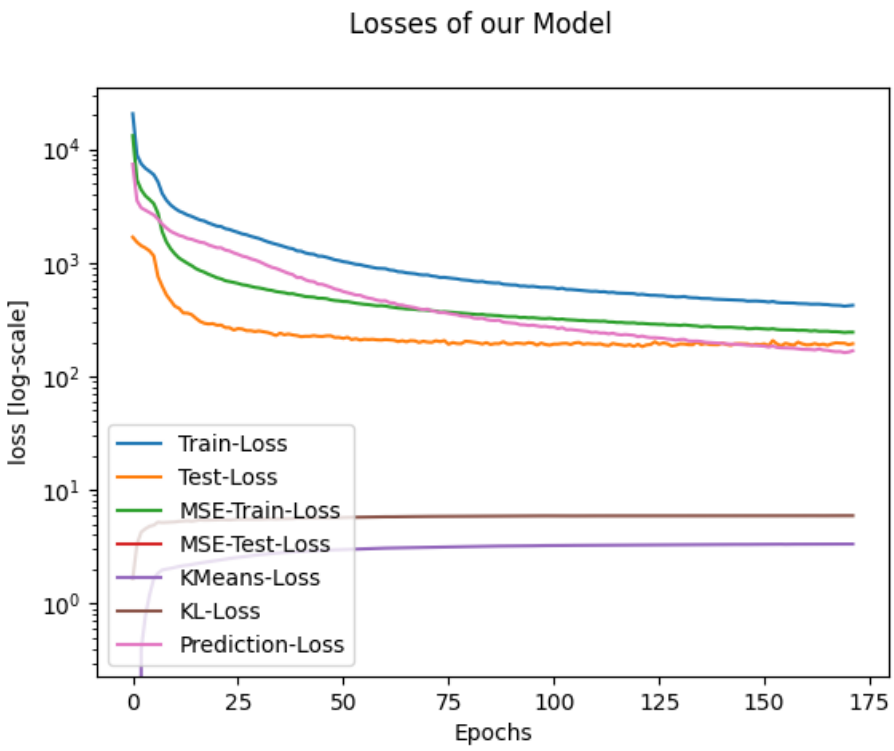


Figure A.1: Loss curves of VAME data training

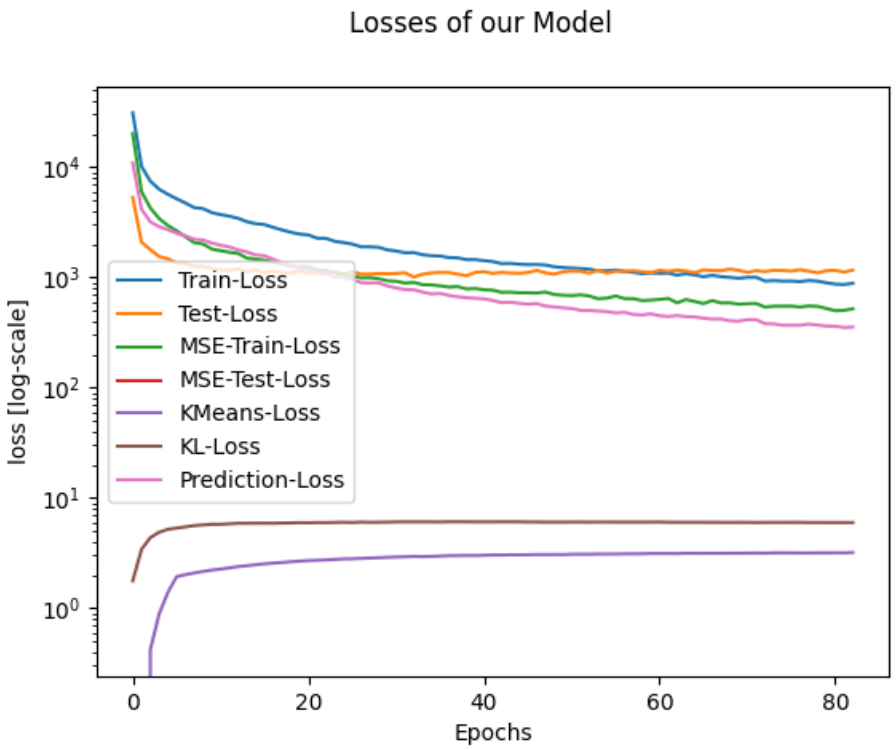


Figure A.2: Loss curves of Irlab I data training

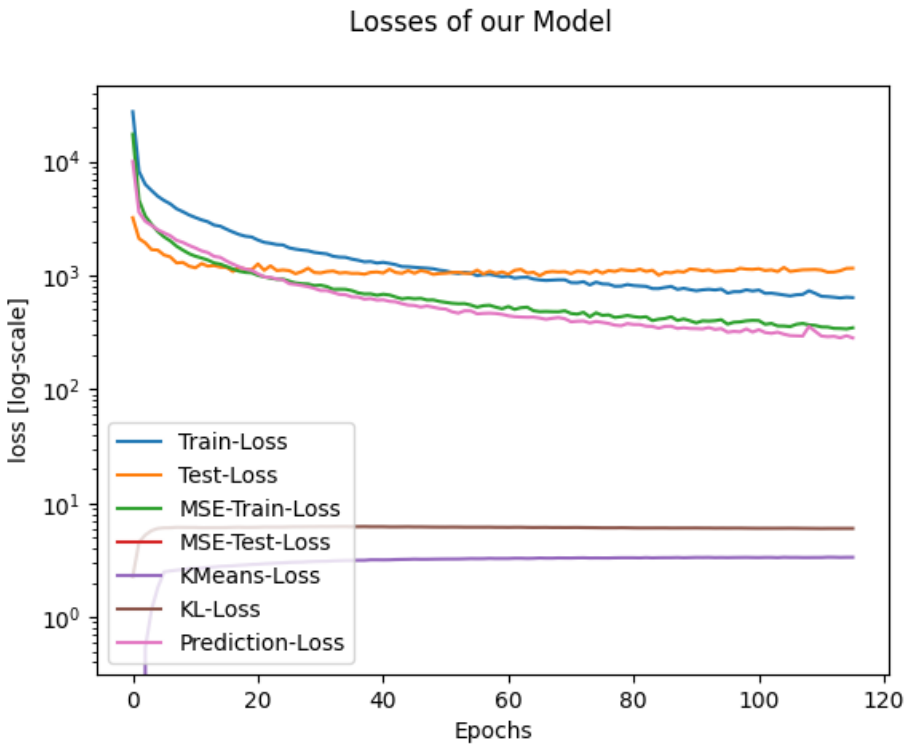


Figure A.3: Loss curves of Irlab II data training

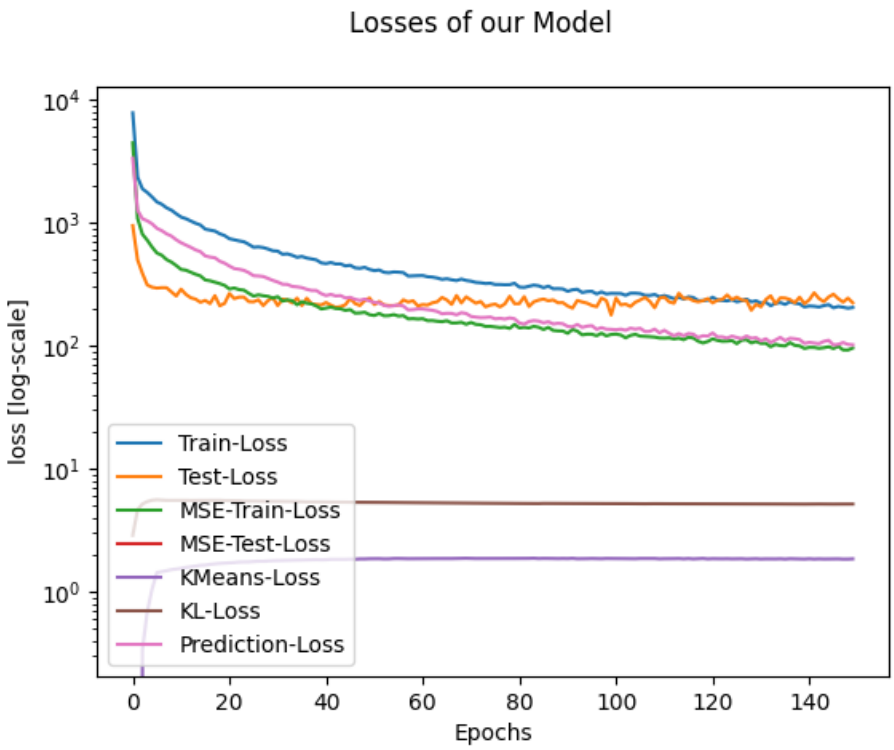


Figure A.4: Loss curves of Lund data training

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY