Ali Osman Berk Şapcı

June 7, 2022

Contents

List of Figures

List of Tables

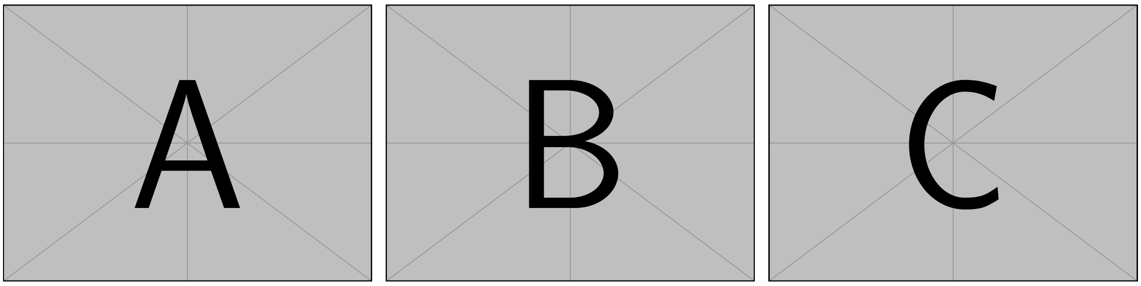
Abstract

A *Bastı* (English: *Basty*, Azerbaijani Turkish: *Basdı*, Anatolian Turkish: *Basdırık*) is an evil spirit in Turkic mythology which rides on people's chests while they sleep, bringing on nightmares. Bastı sits astride a sleeper's chest and becomes heavier until the crushing weight awakens the terrified and breathless dreamer. The victim awakes, unable to move under the Bastıs weight. It may also include lucid dreams.

Chapter 1

Introduction

1.1 Challenges



(a) (b) (c)

Figure 1.1: Orientations.

1.2 Contributions

Chapter 2

Biological Experiments and Data Collection

Chapter 3

Feature Extraction

For a single experiment data, i.e., single fruit fly recorded between ZT10 and ZT2 (zeitgeber time 10 and 2), feature extraction consists of four consecutive steps, where the input in one stage is the output of the previous one. The input of the first step is the raw output signal of the tracking and pose estimation model, in our case, the output of DeepLabCut, a toolbox for markerless pose estimation. The feature extraction steps are as follows:

1. Constructing pose values and preprocessing; dealing with occluded body parts, alignment of different orientations, filtering and imputation.
2. Computing spatio-temporal features, such as distances between body parts, velocity, angular velocity from body part positions.
3. Computing dynamic postural features by extending spatio-temporal features to multiple time-scales using wavelet transformation and sliding window statistics.
4. Computing normalized high-dimensional behavioral representations.

Matrices $\mathbf{X} \in \mathbb{R}^{T \times N}$ and $\mathbf{Y} \in \mathbb{R}^{T \times N}$ denotes a multivariate time series for x and y cartesian components of two-dimensional video recordings that are collected for N tracked body parts of the fly on T consecutive time stamps by a pose estimation model. This multivariate time series matrices $\mathbf{X}$ and $\mathbf{Y}$ are the raw input data that goes into the first step of the feature extraction. Note that the number of body parts, N, must be the same among all experiments conducted with different fruit flies, but the number of time stamps, T, might differ. Each column of the $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^\top$ and $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N]^\top$ can be written respectively as follows;

$$\begin{aligned}\mathbf{y}_i &= (y_{i,1}, y_{i,2}, \ldots, y_{i,t-1}, y_{i,t}, y_{i,t}, \ldots, y_{i,T}),\\ \mathbf{x}_i &= (x_{i,1}, x_{i,2}, \ldots, x_{i,t-1}, x_{i,t}, x_{i,t}, \ldots, x_{i,T}).\end{aligned}$$

Here i denotes the index of the body part, e.g., leg tip or proboscis.

In addition to $\mathbf{X}$ and $\mathbf{Y}$, a pose estimation model may report prediction scores for each tracked body part at each time step, which is the case for DeepLabCut as well. $L \in \mathbb{R}^{N \times T}$ denotes the time series of prediction scores, each column of the $L = [\mathbf{l_1}, \cdots, \mathbf{l_N}]^\top$ can be written as follows;

$$\mathbf{l}_i = (l_{i,1}, l_{i,2}, \ldots, l_{i,t-1}, l_{i,t}, l_{i,t}, \ldots, l_{i,T}).$$

The prediction scores tend to be very low when the body part is not visible. Thus, L provides valuable information about the occluded body parts. In the Section 3.1.1, how L is incorporated into construction of pose values is described in detail.

3.1 Preprocessing

The goal of in this step is to preprocess the signal, which involves filtering and imputation of certain video frames. But in addition to this, there are a couple of optional procedures that can be beneficial for our task of learning stereotypical behaviors. These additional procedures deal with occluded body parts of the fly, alignment of the fly orientations and defining new points of interest.

3.1.1 Dealing with Occluded Body Parts

As mentioned in the Section 1.1, the two-dimensionality of the video recordings introduces a number of important challenges, and one of them is the problem of occluded body parts. There are many types of occlusions, and some of them can be informally described as follows. One examples is short occlusions resulting from postural changes. Imputation of the time series $\mathbf{X}$ and $\mathbf{Y}$ for such short occlusions is relatively easy since the number of consecutive missing data points are not many. However, this is not the case for long occlusions, which usually occur when the fly is dormant for a long period of time. Especially for the body parts which have left and right counterparts, it is usual that the orientation of the fly results in one of the counterparts being occluded in long dormant epochs. We follow multiple approaches to handle different type occlusions; namely imputation and elimination of corresponding data-points. Before describing those approaches, we define a criterion for being occluded.

Oriented Pose Values for Body Parts with Left & Right Counterparts

If the fly is oriented perpendicular to the camera perspective, as in Figure 1.1c, then one of the left and right body parts is often occluded. In other orientations (e.g., Figure 1.1a and Figure 1.1b), both of them or none of them might be occluded as well. However, in the conducted experiments, flies usually choose to stay dormant perpendicular to the camera perspective in long dormant epochs, as mentioned in Chapter 2. In such cases, one can concede to use only one of the left and right counterparts to construct pose values. Therefore, this optional step is included in the behavioral mapping pipeline to reduce pose values of body parts with left and right counterparts to a single value.

We use prediction scores to determine which body part should be used to compute oriented pose values. Let i and j be a pair of body parts which are left and right counterparts of each other, e.g., left haltere and right haltere. Then, one can use the $\mathbf{l}_i$ and $\mathbf{l}_j$ to predict if one of them is occluded at a particular time step t. Let $\text{orient}(\mathbf{x}_i, \mathbf{x}_j)$ ($\text{orient}(\mathbf{y}_i, \mathbf{y}_j)$) be a new pose vector which will be computed based on $\mathbf{x}_i$ and $\mathbf{x}_j$ ($\mathbf{y}_i$ and $\mathbf{y}_j$), e.g., a vector of oriented haltere pose values, composed of left haltere and right haltere pose values. The following conditional procedures are proposed to compute oriented pose values from left and right pose values by deciding the orientation of the fly for a counterpart body pair. The procedures below can be used separately, or successively.

- If $l_{i,t} - l_{j,t} \geq \epsilon$, then, without loss of generality, $\text{orient}(\mathbf{x}_i, \mathbf{x}_j)_t = x_{i,t}$ and $\text{orient}(\mathbf{y}_i, \mathbf{y}_j)_t = y_{i,t}$ for a threshold ϵ, typically $\epsilon > 0.5$.
- If $|\{\, t' : \ l_{i,t'} > l_{j,t'},\ t' \in [t - w \,..\, t + w] \,\}| > w$, then, without loss of generality, $\text{orient}(\mathbf{x}_i, \mathbf{x}_j)_t = x_{i,t}$ and $\text{orient}(\mathbf{y}_i, \mathbf{y}_j)_t = y_{i,t}$, for a window of size $2w + 1$.
- If $l_{i,t} > l_{j,t}$ and if the nearest confident left orientation is closer than the nearest confident right orientation, i.e.,

  $$\arg\min_{t'} \{\, |t - t'| \ : \ l_{i,t'} - l_{j,t'} \geq \epsilon \,\} > \arg\min_{t'} \{\, |t - t'| \ : \ l_{j,t'} - l_{i,t'} \geq \epsilon \,\}\,,$$

  then, without loss of generality, $\text{orient}(\mathbf{x}_i, \mathbf{x}_j)_t = x_{i,t}$ and $\text{orient}(\mathbf{y}_i, \mathbf{y}_j)_t = y_{i,t}$.
- If simply $l_{i,t} > l_{j,t}$, then without loss of generality, $\text{orient}(\mathbf{x}_i, \mathbf{x}_j)_t = x_{i,t}$ and $\text{orient}(\mathbf{y}_i, \mathbf{y}_j)_t = y_{i,t}$.

Except directly comparing the prediction confidence scores as in the last procedure, some of the time points might be left with undecided orientations. If the number of such time points is acceptable, then directly comparing the prediction scores for those time points is convenient and handy.

After applying the above procedures for a left and right counterpart pair i and j, we can define oriented multivariate time series as;

$$\begin{aligned}
\mathbf{X}^o &= \left(\left[(\mathbf{x}_k)_{k \notin \bigcup_{\{\,i,j\,\} \in \mathcal{O}} \{\,i,j\,\}} \right]^\top \,\middle|\, \left[(\text{orient}(\mathbf{x}_i, \mathbf{x}_j))_{\{\,i,j\,\} \in \mathcal{O}} \right]^\top \right), \\
\mathbf{Y}^o &= \left(\left[(\mathbf{y}_k)_{k \notin \bigcup_{\{\,i,j\,\} \in \mathcal{O}} \{\,i,j\,\}} \right]^\top \,\middle|\, \left[(\text{orient}(\mathbf{y}_i, \mathbf{y}_j))_{\{\,i,j\,\} \in \mathcal{O}} \right]^\top \right),
\end{aligned}$$

where $\mathcal{O}$ is the set of index pairs of left and right counterparts and $\bigcup_{\{\,i,j\,\} \in \mathcal{O}} \{\,i,j\,\}$ is the union of all indexes of such body parts pairs. Applying described procedures for each left and right counterparts results in computing $\mathbf{X}^o$ and $\mathbf{Y}^o$. This oriented versions of original data matrices can be used instead of $\mathbf{X}$ and $\mathbf{Y}$ in the rest of the pipeline, if desired.

Detecting Occlusions Using Prediction Scores and Preternatural Tracking Predictions

3.1.2 Aligning Different Orientations

3.1.3 Filtering and Imputation

3.2 Computation of Spatio-temporal Features

After preprocessing of pose values, it is now feasible to go one step further towards learning stereotypical behaviors. Although tracking of relevant body parts and processing corresponding pose values is an essential step for quantifying behavior, a set of coordinate values is not sufficient to represent and capture complex spatio-temporal dynamics of animal behavior. There are thousands of unique postures, and behaviors are not even exhibited by some static set of postures. Instead, they are defined by expressive and meaningful spatio-temporal features such as distances, velocities, angles and angular velocities. Therefor, one need to compute such features from the coordinate values of body parts in two-dimensional space.

The second stage of the feature extraction is computation of spatio-temporal features from pose values. Two type of features are computed in this stage, as listed below.

1. **Snapshot features:** Spatio-temporal feature values computed at a snapshot of time, listed as follows:
    - distances,
    - angles,
    - cartesian pose values (i.e., per body part features).
2. **Gradient features:** Spatio-temporal feature values computed based how snapshot features change over time, listed as follows:
    - change of distances,
    - change of angles (i.e., angular velocities),
    - change of cartesian pose values (i.e., body part velocities).

The gradients of snapshot features are computed using second-order accurate central differences in the interior points. The resulting gradient features have the same shape, i.e., the number of features and the number of time-stamps, as the snapshot features.

3.2.1 Distances Between Body Parts

Given a body part pair (i, j), the distance between them at a time step t is calculated as a usual Euclidean distance, given below.

$$d_t^{i,j} = \sqrt{(x_{i,t} - x_{j,t})^2 + (y_{i,t} - y_{j,t})^2}. \tag{3.1}$$

The corresponding gradient feature, namely the change of distance between body part i and j, is computed using the second order gradient approximation,

$$\dot{d}_t^{i,j} = \begin{cases} \frac{\left|d_{t+1}^{i,j} - d_t^{i,j}\right|}{\Delta t} & \text{if t=0 or t=T,} \\ \frac{\left|d_{t+1}^{i,j} - d_{t-1}^{i,j}\right|}{2\Delta t} & \text{otherwise,} \end{cases} \tag{3.2}$$

where Δt is the sampling period, and it is equal to 1/FPS seconds.

3.2.2 Joint Angles Between Body Parts

Given a triplet of body parts (i, j, k), angle between i and k around j is computed using the 2-argument arctangent function as given below.

$$w_t^{i,j,k} = \text{atan2}\left(\det\begin{bmatrix} x_{i,t} - x_{j,t}, x_{k,t} - x_{j,t} \\ y_{i,t} - y_{j,t}, y_{k,t} - y_{j,t} \end{bmatrix}, \begin{bmatrix} x_{i,t} - x_{j,t} \\ y_{i,t} - y_{j,t} \end{bmatrix} \cdot \begin{bmatrix} x_{k,t} - x_{j,t} \\ y_{k,t} - y_{j,t} \end{bmatrix}\right) + \pi. \tag{3.3}$$

Then, similar to the change of distance features, angular velocities are approximated by,

$$\dot{w}_t^{i,j,k} = \begin{cases} \frac{\left|w_{t+1}^{i,j,k} - w_t^{i,j,k}\right|}{\Delta t} & \text{if t=0 or t=T,} \\ \frac{\left|w_{t+1}^{i,j,k} - w_{t-1}^{i,j,k}\right|}{2\Delta t} & \text{otherwise.} \end{cases} \tag{3.4}$$

3.2.3 Cartesian Pose Values of Body Parts

Cartesian pose values of a body part i is straightforwardly given by the x and y coordinate values as follows,

$$x_t^i = x_{i,t}, \tag{3.5}$$

$$y_t^i = y_{i,t}. \tag{3.6}$$

Note that for a single body part, two feature values are generated. Corresponding gradient features, namely the body part velocities along each cartesian component, are computed by

$$\dot{x}_t^i = \begin{cases} \frac{x_{t+1}^i - x_t^i}{\Delta t} & \text{if t=0 or t=T,} \\ \frac{x_{t+1}^i - x_{t-1}^i}{2\Delta t} & \text{otherwise,} \end{cases} \tag{3.7}$$

$$\dot{y}_t^i = \begin{cases} \frac{y_{t+1}^i - y_t^i}{\Delta t} & \text{if t=0 or t=T,} \\ \frac{y_{t+1}^i - y_{t-1}^i}{2\Delta t} & \text{otherwise.} \end{cases} \tag{3.8}$$

In order to compute overall two-dimensional velocity of a body part, one can always use the distance between origin and corresponding body part.

3.2.4 Constructing Spatio-temporal Feature Matrices

Let $\mathcal{C}, \mathcal{D}$, and $\mathcal{A}$ denote the sets of body parts, body part pairs, and body part triplets; respectively defining cartesian pose values, distances, and angles. Similarly, let $\mathcal{C}', \mathcal{D}'$, and $\mathcal{A}'$ denote sets which define sets of respective gradient features. Then snapshot feature matrix S constructed as follows;

$$\mathbf{S} = \left(x_t^i\right)_{1 \leq t \leq T,\, i \in \mathcal{C}} \mid \left(y_t^i\right)_{1 \leq t \leq T,\, i \in \mathcal{C}} \mid \left(d_t^{i,j}\right)_{1 \leq t \leq T,\, \{i,j\} \in \mathcal{D}} \mid \left(w_t^{i,j,k}\right)_{1 \leq t \leq T,\, \{i,j,k\} \in \mathcal{A}}. \tag{3.9}$$

Similarly, for gradient features, the feature matrix is constructed by concatenating change of distances, angular velocities and body part velocities; given by

$$\mathbf{G} = \left(\dot{x}_t^i\right)_{1 \leq t \leq T, i \in \mathcal{C}'} \mid \left(\dot{y}_t^i\right)_{1 \leq t \leq T, i \in \mathcal{C}'} \mid \left(\dot{d}_t^{i,j}\right)_{1 \leq t \leq T, \{i,j\} \in \mathcal{D}'} \mid \left(\dot{w}_t^{i,j,k}\right)_{1 \leq t \leq T, \{i,j,k\} \in \mathcal{A}'}. \tag{3.10}$$

The resulting two feature matrices are $\mathbf{S} \in \mathbb{R}^{T \times (2|\mathcal{C}|+|\mathcal{D}|+|\mathcal{A}|)}$ and $\mathbf{G} \in \mathbb{R}^{T \times (2|\mathcal{C}'|+|\mathcal{D}'|+|\mathcal{A}'|)}$. Let N_S denote the number of snapshot features, being equal to $2|\mathcal{C}| + |\mathcal{D}| + |\mathcal{A}|$ and let N_G denote the number of gradient features, which is equal to $2|\mathcal{C}'| + |\mathcal{D}'| + |\mathcal{A}'|$.

3.3 Computation of Dynamic Postural Features

Instantaneous values of spatio-temporal features do not provide a sufficient description of complex postural dynamics of behaviors. Understanding the output of a complex biological system, in our case behavior, can only be achieved by studying multiple time-scales together. Previous studies attempt to search behavioral motifs, e.g., repeated sub-sequences of actions with finite length, within the behavioral time series [CITE]. However, as Berman et al. [2014] states, this paradigm usually requires problems of temporal alignment and relative phasing between different scales. Alternatively, extending spatio-temporal features to capture postural dynamics at different time-scales eliminate requirements of temporal alignment and motif based analysis. In order to extend the spatio-temporal features to dynamic postural features, we applied wavelet transformation (similar to Berman et al. [2014]) and computed moving statistics at different time-scales (similar to Kabra et al. [2013]), respectively for snapshot feature set ($\mathbf{S}$) and gradient feature set ($\mathbf{G}$).

3.3.1 Moving Statistics of Gradient Features

Gradient features only reflect the instantaneous values of velocities with respect to the sampling rate. In order to capture how these values change within a given interval, the moving statistics, e.g., mean and standard deviation, of gradient features are computed with a sliding window approach. Let τ be the window size parameter, i.e., the timescale of interest, then the moving mean of the corresponding gradient feature $\mathbf{g}_i$ is given by the function μ_τ;

$$\mu_\tau(g_{i,t}) = {}^1\!/\!_{\min\{t+\tau,T\} - \max\{t-\tau,1\}+1} \sum_{t'=\max\{t-\tau,1\}}^{\min\{t+\tau,T\}} g_{i,t'}. \tag{3.11}$$

Similarly, the moving standard deviation of a gradient feature $g_{i,t}$ by is computed by σ_τ, as in the below equation.

$$\sigma_\tau(g_{i,t}) = \left({}^1\!/\!_{\min\{t+\tau,T\} - \max\{t-\tau,1\}+1} \sum_{t'=\max\{t-\tau,1\}}^{\min\{t+\tau,T\}} \left(\mu_\tau(g_{i,t}) - g_{i,t'}\right)^2 \right)^{1/2} \tag{3.12}$$

Moving statistics feature generation approach is used to learn animal behavior by Kabra et al. [2013], and Marshall et al. [2021] is also included such features into the analysis.

3.3.2 Wavelet Transformation of Snapshot Features

The wavelet domain is a useful representation of postural dynamics due to the following reasons given by Berman et al. [2014], and proposed spectrogram generation is used by others as well [Marshall et al., 2021, Todd et al., 2017].

- It describes dynamics over multiple time-scales simultaneously by possessing a multi-resolution time-frequency trade-off.
- It eliminates the requirement of precise temporal alignment for capturing periodic behaviors by taking amplitudes of the continuous wavelet transform of each snapshot feature at different scales.

Given a function $s(t)$, continuous wavelet transformation at a frequency $f > 0$ is expressed by the following integral;

$$W_{f,t'}\left[s\left(t\right)\right] = \frac{1}{\sqrt{a\left(f\right)}} \int_{-\infty}^{\infty} s\left(t\right)\, \varPsi^{*}\left(\frac{t-t'}{a(f)}\right) \mathrm{d}t\,, \tag{3.13}$$

where Ψ is the wavelet function and a is a function for converting frequencies to wavelet scale factor. The Morlet wavelet is suitable for describing postural dynamics which is closely related to human perception, both hearing and vision [Daugman, 1985], and is used in the pipeline. The corresponding wavelet function is given by

$$\Psi(t) = \exp\left\{\frac{t^2}{2}\right\}\cos(w_0 t), \tag{3.14}$$

where w_0 is a non-dimensional parameter. The frequency to scale conversion function a for Morlet wavelet is as follows;

$$a\left(f\right) = \frac{w_0 + \sqrt{2 + w_0^2}}{4\pi f},. \tag{3.15}$$

For the discrete sequence of snapshot feature $\mathbf{s}_i$ with sampling period Δt, $W_{f,t'}[s(t)]$ translates into;

$$W_f\left(\mathbf{s}_i, t'\right) = \frac{1}{\sqrt{a(f)}} \sum_{t=1}^{T} \Delta t s_{i,t}\, \varPsi^{*}\left(\frac{t-t'}{af}\right), \tag{3.16}$$

where $t', t \in \mathbb{Z}$ and $1 \leq t' \leq T$ [Torrence and Compo, 1998].

Normalization of Wavelet Power Spectrum

In order to ensure that wavelet transforms (Equation 3.16) at each frequency f are directly comparable to each other and to the other transformed time series, the transformation W_f has to be normalized at each frequency f to have unit energy. This normalization for Morlet wavelet at frequency f is given by

$$C(f) = \frac{\pi^{-\frac{1}{4}}}{\sqrt{2a(f)}} \exp\left\{{}^{1}\!/\!{}_{4}\left(w_0 - \sqrt{w_0^2+2}\right)^2\right\}. \tag{3.17}$$

So resulting normalized transformation which is also used to generate the spectrogram in Berman et al. [2014] is given by

$$W_f^0\left(\mathbf{s}_i, t'\right) = \frac{1}{C\left(f\right)} |W_f\left(\mathbf{s}_i, t'\right)|. \tag{3.18}$$

In addition to the above conventionally used normalization, we alternatively adopted the normalization proposed by Liu et al. [2007]. According to this alternative adjustment, the wavelet power spectrum should be equal to the transform coefficient squared divided by the scale it associates.

$$W_f^0\left(\mathbf{s}_i, t'\right) = \frac{W_f\left(\mathbf{s}_i, t'\right)^2}{a\left(f\right)} \tag{3.19}$$

We observed substantial improvements using this power spectrum (see Section 7).

Determining Spectrum Frequencies

We investigate two different approaches for computing a set of frequencies and, we include both of them in the behavior mapping pipeline. One set is dyadically spaced frequencies between $f_{\min}$ and $f_{\max}$ via;

$$f_i = f_{\max} 2^{-\frac{i-1}{N_f - 1} \log \frac{f_{\max}}{f_{\min}}}, \tag{3.20}$$

where $f_{\max} = {}^{FPS}\!/_2$ Hz is the Nyquist frequency.

The other alternative set of frequencies is linearly spaced between $f_{\min}$ and $f_{\max}$ by

$$f_i = f_{\min} + \frac{f_{\max} - f_{\min}}{N_f - 1} i, \tag{3.21}$$

for $i = 1, 2, \ldots, N_f$, and their corresponding wavelet scales are computed via $a\left(f_i\right)$.

3.3.3 Constructing Dynamic Postural Feature Tensors

Let $\mathcal{T}_S = \{\, {}^1\!/_{f_{\min}}, \cdots {}^1\!/_{f_{\max}} \,\}$ and $\mathcal{T}_G = \{\, \tau_{\min}, \cdots, \tau_{\max} \,\}$ denote the time-scale sets respectively for wavelet transforms of snapshot features and moving statistics of gradient features. Then corresponding feature tensors are given as follows:

$$\mathbf{W} = \left[W^0_{{}^1\!/_f}\left(\mathbf{s}_i, t\right)\right]_{1 \le i \le N_S,\, 1 \le t \le T,\, {}^1\!/_f \in \mathcal{T}_S}, \tag{3.22}$$

$$\mathbf{M}_\mu = \left[\mu_\tau\left(g_{i,t}\right)\right]_{1 \le i \le N_G,\, 1 \le t \le T,\, \tau \in \mathcal{T}_G}, \tag{3.23}$$

$$\mathbf{M}_\sigma = \left[\sigma_\tau\left(g_{i,t}\right)\right]_{1 \le i \le N_G,\, 1 \le t \le T,\, \tau \in \mathcal{T}_G}, \tag{3.24}$$

where $\mathbf{S} \in \mathbb{R}^{T \times N_S}$ and $\mathbf{G} \in \mathbb{R}^{T \times N_G}$.

The resulting extended feature tensors of dynamic postural representations are $\mathbf{W} \in \mathbb{R}^{T \times |\mathcal{T}_s| \times N_S}$, $\mathbf{M}_\mu \in \mathbb{R}^{T \times |\mathcal{T}_v| \times N_G}$ and $\mathbf{M}_\sigma \in \mathbb{R}^{T \times |\mathcal{T}_v| \times N_G}$.

3.4 Computation of Behavioral Representations

After applying wavelet transformation or computing moving statistics to extend extracted spatio-temporal features to dynamic postural features, a couple of additional operations are required to generate desired behavioral embeddings.

3.4.1 Flattening Dynamic Postural Feature Tensors

As constructed in Section 3.3, dynamic postural feature tensors are $\mathbf{W} \in \mathbb{R}^{T \times |\mathcal{T}_S| \times N_S}$, $\mathbf{M}_\mu \in \mathbb{R}^{T \times |\mathcal{T}_G| \times N_G}$, and $\mathbf{M}_\sigma \in \mathbb{R}^{T \times |\mathcal{T}_G| \times N_G}$. In order to apply manifold learning-based dimensionality reduction algorithms or traditional machine learning algorithms such as decision trees, one needs to flatten the last two dimensions of these feature tensors. As a result, feature matrices are $\mathbf{W}^* \in \mathbb{R}^{T \times (N_S |\mathcal{T}_S|)}$, $\mathbf{M}_\mu^* \in \mathbb{R}^{T \times (N_G |\mathcal{T}_G|)}$ and $\mathbf{M}_\sigma^* \in \mathbb{R}^{T \times (N_G |\mathcal{T}_G|)}$ are obtained.

3.4.2 L$_1$ Normalization of Frames

Wavelet power spectrum and gradient feature values are heavily dependent on the orientation of the flies and on the unique fly characteristics. In order to have a homogeneous feature space among flies and along the time, at each time step t, L$_1$ normalization is applied as follows;

$$\mathbf{W}' = \left(\frac{w_{t,i}^*}{\sum_{j=0}^{N_S |\mathcal{T}_s|} w_{t,j}^*} \right)_{1 \leq t \leq T,\, 1 \leq i \leq (N_S |\mathcal{T}_s|)} . \tag{3.25}$$

Similarly, L$_1$ normalized versions of $\mathbf{M}_\mu^*$ and $\mathbf{M}_\sigma^*$, namely $\mathbf{M}_\mu'$ and $\mathbf{M}'\sigma$ are obtained.

Here $\mathbf{W}'$, $\mathbf{M}_\mu'$ and $\mathbf{M}'\sigma$ are the final multivariate time series of normalized high dimensional behavioral representation of a fly experiment. Behavioral representation matrix $\mathbf{W}'$ will be used to compute behavioral embeddings.

3.5 Summary

Chapter 4

Experiment Outlining

Chapter 5

Behavior Mapping

5.1 Computing Behavioral Embeddings

McInnes et al. [2020] McInnes et al. [2017] Campello et al. [2013]

5.1.1 Supervised Disparate Embeddings

[NOTE: We compute Supervised-UMAP embeddings separately for each experiment. Computing supervised embeddings is only possible for annotated experiments. This is useful for exploring behavioral sub-categories. For instance, one annotation category, e.g. ”grooming”, can be consisted of two different clusters in the behavioral embedding space. We can further investigate how high-level behavioral annotations contain different but similar behaviors.]

5.1.2 Supervised Joint Embeddings

[NOTE: We compute Supervised-UMAP embeddings of all annotated experiments together. There is no specific benefit or use case (except visualization purposes) for computing this. The resulting embedding usually is not homogeneous in terms of fly experiments. Different flies do not mix well in the behavioral space.]

5.1.3 Unsupervised Disparate Embeddings

[NOTE: For each experiment, we compute a usual unsupervised embedding separately.]

5.1.4 Unsupervised Joint Embedding

[NOTE: We compute a single behavioral embedding using all experiments. The problem with this approach is that embedding space is too crowded and thus we can not find meaningful and homogeneous clusters right away. This embedding might be useful for visualization purposes.]

5.1.5 Semi-supervised Pair Embeddings

[NOTE: This is the novel and most useful embedding approach that we finally utilize to label unannotated experiments using annotated ones. We compute an embedding for each annotated and unannotated pair. For example, if there are N_A annotated experiments and N_U unannotated experiments we compute $N_A \cdot N_U$ embeddings in total. There are number of advantageous of this approach. Especially when, an behavioral repertoire of an annotated an unannotated are similar, resulting embedding is easy to interpret and use for clustering etc.]

5.2 Soft Clustering of Behavioral Embeddings

[NOTE: We always use soft clustering feature of HDBSCAN, since it is very beneficial to have a composite assignment for a data-point. For example, 0.7 grooming, 0.3 proboscis pumping may indicate that those two behaviors are simultaneously exhibited or a combination of both is exhibited etc. We can always take $\arg\max$ if a categorical label is needed.]

5.2.1 Disparate Clustering

[NOTE: If embedding is a disparate embedding, then we directly cluster each of them separately. If a joint embedding or pair embedding will be clustered, then experiments need to be extracted from the embedding space first and then they need to be clustered separately.]

5.2.2 Joint Clustering

[NOTE: This is only applicable to joint and pair embeddings. We cluster all experiments in the behavioral embedding together.]

5.2.3 Crosswise Clustering

[NOTE: This is again only applicable to joint and pair embeddings. For joint embeddings, we exclude a sub-group of experiments. For pair embeddings, we exclude one of the pair experiments. Then rest of the embedding is clustered and clusters are formed. Finally for each excluded embedding, soft cluster memberships vectors are computed based on the formed clusters.]

5.3 Computing Behavioral Correspondences

5.3.1 Mapping Clusters to Behavioral Categories

[NOTE: If a clustering contains annotated experiments, we map that clusters in that clustering to a behavioral composition as follows (*subject to change, there are couple of alternatives*)

$$m_\alpha = \frac{\text{number of frames with annotation } \alpha \text{ in the cluster}}{\text{total number of frames with annotation } \alpha}. \tag{5.1}$$

So for each cluster, we end up with a vector $\mathbf{m} = (m_\alpha)$, represent it behavioral composition.]

5.3.2 Computing Behavioral Scores

[NOTE: Behavioral score of unannotated experiment will be computed using clustering membership score and behavioral composition mapping of those clusters. For example, using semi-supervised pair embeddings and crosswise clustering; one can compute behavioral scores for a frame as follows;

$$y_\alpha = \sum_{c=0}^{C} m_\alpha^c \tag{5.2}$$

where C is the number of clusters, $\mathbf{m}^c$ is the behavioral composition of cluster c. As a result, for each frame we end up with a behavioral score vector $\mathbf{y} = (y_\alpha)$.]

Chapter 6

Analyzing Behavioral Repertoire of Asleep Fruit Fly

Chapter 7

Employing Proposed Pipeline for Collected Data

Chapter 8

Results

Chapter 9

`basty`: A Software Package for Automated Behavioral Analysis of Asleep Fruit Fly

Chapter 10

Conclusion

Bibliography

Gordon J. Berman, Daniel M. Choi, William Bialek, and Joshua W. Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99):20140672, October 2014. doi: 10.1098/rsif.2014.0672. URL `https://royalsocietypublishing.org/doi/full/10.1098/rsif.2014.0672`. Publisher: Royal Society.

Mayank Kabra, Alice A. Robie, Marta Rivera-Alba, Steven Branson, and Kristin Branson. JAABA: interactive machine learning for automatic annotation of animal behavior. *Nature Methods*, 10(1):64–67, January 2013. ISSN 1548-7105. doi: 10.1038/nmeth.2281. URL `https://www.nature.com/articles/nmeth.2281`. Number: 1 Publisher: Nature Publishing Group.

Jesse D. Marshall, Diego E. Aldarondo, Timothy W. Dunn, William L. Wang, Gordon J. Berman, and Bence P. Ölveczky. Continuous Whole-Body 3D Kinematic Recordings across the Rodent Behavioral Repertoire - SI. *Neuron*, 109(3):420–437.e8, February 2021. ISSN 08966273. doi: 10.1016/j.neuron.2020.11.016. URL `https://linkinghub.elsevier.com/retrieve/pii/S0896627320308941`.

Jeremy G. Todd, Jamey S. Kain, and Benjamin L. de Bivort. Systematic exploration of unsupervised methods for mapping behavior. *Physical Biology*, 14(1):015002, February 2017. ISSN 1478-3975. doi: 10.1088/1478-3975/14/1/015002. URL `https://doi.org/10.1088/1478-3975/14/1/015002`. Publisher: IOP Publishing.

John G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *JOSA A*, 2 (7):1160–1169, July 1985. ISSN 1520-8532. doi: 10.1364/JOSAA.2.001160. URL `https://opg.optica.org/josaa/abstract.cfm?uri=josaa-2-7-1160`. Publisher: Optica Publishing Group.

Christopher Torrence and Gilbert P. Compo. A Practical Guide to Wavelet Analysis. *Bulletin of the American Meteorological Society*, 79(1):61–78, January 1998. ISSN 0003-0007, 1520-0477. doi: 10.1175/1520-0477(1998)079⟨0061:APGTWA⟩2.0.CO;2. URL `http://journals.ametsoc.org/doi/10.1175/1520-0477(1998)079<0061:APGTWA>2.0.CO;2`.

Yonggang Liu, X. San Liang, and Robert H. Weisberg. Rectification of the Bias in the Wavelet Power Spectrum. *Journal of Atmospheric and Oceanic Technology*, 24(12):2093–2102, December 2007. ISSN 1520-0426, 0739-0572. doi: 10.1175/2007JTECHO511.1. URL `http://journals.ametsoc.org/doi/10.1175/2007JTECHO511.1`.

Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv:1802.03426 [cs, stat]*, September 2020. URL `http://arxiv.org/abs/1802.03426`. arXiv: 1802.03426.

Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205, March 2017. ISSN 2475-9066. doi: 10.21105/joss.00205. URL `https://joss.theoj.org/papers/10.21105/joss.00205`.

Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-Based Clustering Based on Hierarchical Density Estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 160–172, Berlin, Heidelberg, 2013. Springer. ISBN 978-3-642-37456-2. doi: 10.1007/978-3-642-37456-2_14.