SLEAP into Behavioral Analysis: A Deep Learning Approach

Master's Project

Presented in Partial Fulfillment of the Requirements for the Degree
Master of Science in the Graduate School of The Ohio State
University

By

Madhumitha Sekamuri, B.Tech.

Graduate Program in Computer Science and Engineering

The Ohio State University

2024

Master's Examination Committee:

Dr. Rajiv Ramnath, Advisor

Dr. Laurence Coutellier, Co-Advisor

Abstract

This project leverages the SLEAP framework to automate the analysis of mouse social behaviors, addressing the limitations of manual tracking in behavioral neuroscience. By utilizing deep learning and transfer learning techniques, the study accurately identifies and quantifies sniffing behaviors, including ano-genital, nose-to-nose, and ear-based interactions. Temporal patterns, such as behavior duration, frequency, and latency, are analyzed to uncover fundamental principles of social dynamics and their links to neural circuitry.

The methodology employs advanced neural network architectures, including U-Net, integrated with SLEAP's customizable pipeline, to deliver high-precision behavioral predictions with minimal labeled data. Behavioral insights from these analyses not only improve understanding of animal social interactions but also offer a scalable framework for further research. Future directions include the integration of MoSeq for temporal pattern recognition, enabling automatic classification of stereotyped behaviors and enhancing the toolkit for comprehensive behavioral studies. This project represents a significant step towards efficient, scalable, and reproducible analysis in behavioral neuroscience.

Acknowledgments

I would like to express my deepest gratitude to Professor Coutellier for her invaluable guidance and for providing me the opportunity to contribute to research on the prefrontal cortex and its role in stress-induced behavioral regulation. Her mentorship and passion for neuroscience have profoundly influenced and shaped this work.

I also sincerely thank my advisor, Dr. Rajiv Ramnath, for his unwavering support and insightful guidance in applying machine learning techniques to analyze complex behavioral data. His expertise has been instrumental in bridging computational methods with biological research, greatly enriching this interdisciplinary endeavor.

Vita

2017	B.Tech., Information Technology, R.M.K Engineering College, India
2023	M.S., Computer Science and Engineering, The Ohio State University, U.S

Fields of Study

Major Field: Computer Science and Engineering

Table of Contents

List of Figures

Figure **Page**

List of Tables

Table **Page**

Chapter 1: Introduction

1.1 Motivation

Social behavior analysis in animals, particularly in mice, plays a critical role in understanding the fundamental principles of communication, neural dynamics, and behavioral interactions. Sniffing behaviors such as ano-genital, nose-to-nose, and ear-based interactions are key indicators of social dynamics, offering insights into variations influenced by factors like age, sex, and individual traits. Traditional methods for analyzing these behaviors rely on manual annotation, which is labor-intensive, time-consuming, and prone to errors. Recent advancements in machine learning provide the opportunity to revolutionize behavioral neuroscience by automating these analyses with unprecedented efficiency and precision. Leveraging deep-learning frameworks such as SLEAP enables researchers to quantify complex social behaviors with minimal labeled data, facilitating faster insights and more reproducible studies. This project is motivated by the need to overcome existing limitations and establish scalable solutions to analyze animal behavior comprehensively and accurately. [**?**]

1.2 Problem Statement

The manual tracking and annotation of mouse social behaviors are time-intensive and error-prone, limiting the scalability and reproducibility of behavioral studies. To address this, the project aims to:

- Develop automated methods to predict specific social behaviors such as ano-genital sniffing, nose-to-nose interactions, and ear-based sniffing across individual videos and grouped datasets.
- Quantify behavioral parameters, including the frequency, duration, and latency of events, to capture the temporal and spatial nuances of social interactions.
- Analyze behavior patterns during critical timeframes, such as the initial three minutes of interaction, to identify trends and variations across age, sex, and individual traits.
- Eliminate the need for labor-intensive manual annotation by leveraging SLEAP's deep-learning capabilities, enabling precise and efficient behavioral analysis.

By addressing these challenges, this project aims to advance the field of behavioral neuroscience, providing a robust framework for exploring social interactions and their underlying neural mechanisms.

Chapter 2: Approach

In this chapter, we will discuss the approach and design of the baseline solution used to perform behavioral analysis of mice.

2.1 Solution

This project presents a sophisticated solution to automate the analysis of mouse social behaviors by integrating advanced machine learning methodologies and state-of-the-art computational tools. Transfer learning with pre-trained models such as U-Net, ResNet152, and Hourglass is used to capture and analyze complex behavioral patterns with high precision. The solution leverages open-source frameworks such as SLEAP and DeepLabCut, renowned for their efficacy in pose estimation and behavioral tracking, to streamline data processing. SLEAP, a cutting-edge deep learning framework, is employed as the core platform to streamline pose estimation and behavioral tracking. The workflow is meticulously designed to automate the tracking, classification, and correction of behaviors, with U-Net serving as the backbone architecture to ensure accurate and reliable behavioral identification. This approach is scalable, reproducible, and efficient, enabling the analysis of large-scale datasets while minimizing manual intervention.

The high-level outline of the baseline solution is shown in figure 2.1. The U-Net architecture is shown in figure 2.2. The detailed design chart for the project is shown in figure 2.3.

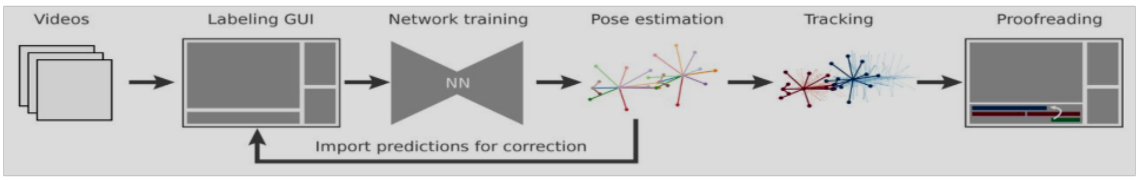

Figure 2.1: High-level outline of baseline solution

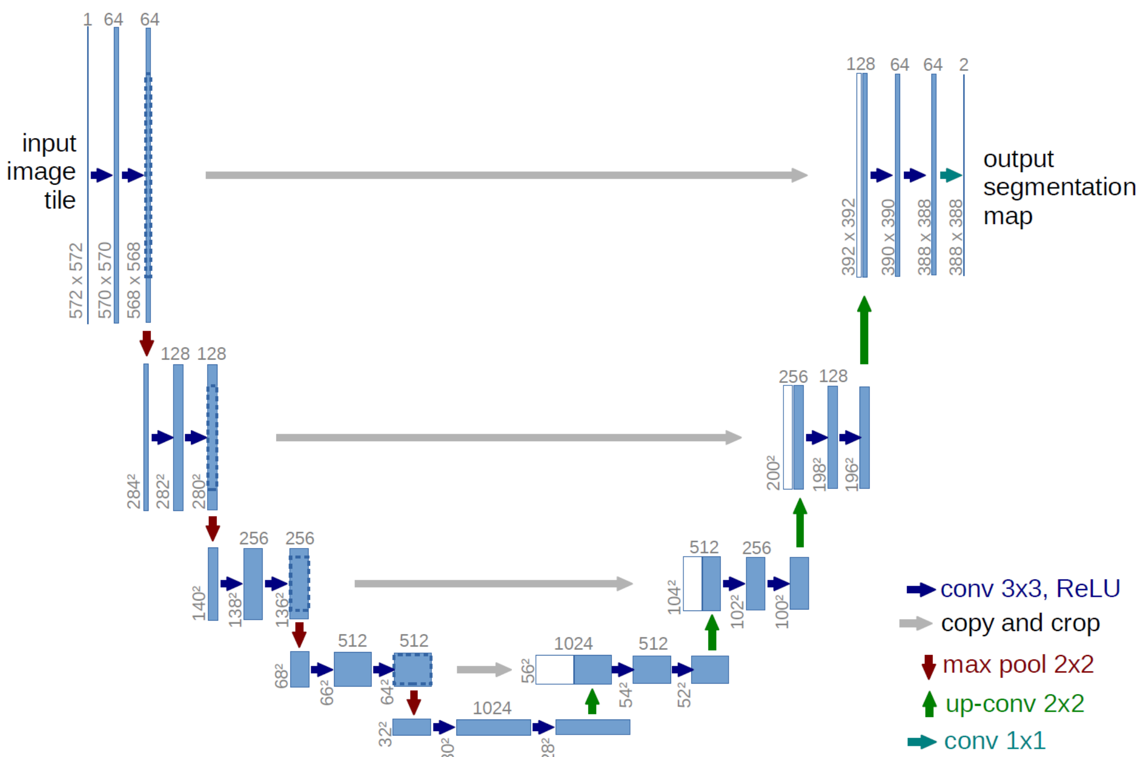

Figure 2.2: UNET Architecture

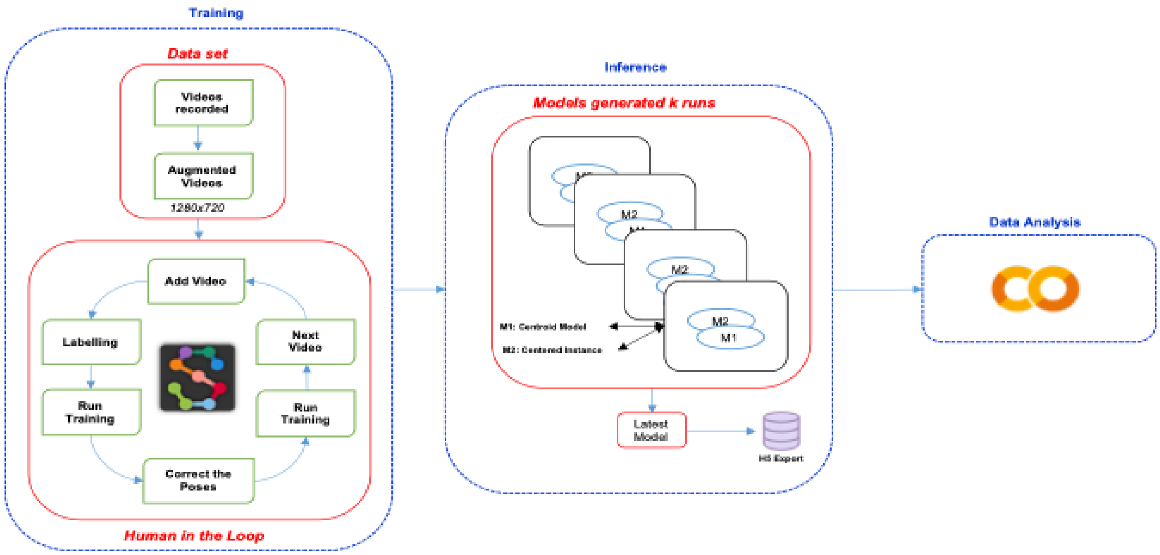

Figure 2.3: Design chart

2.1.1 Training

SLEAP's training process is designed to optimize pose estimation models using labeled datasets provided by the user. It incorporates transfer learning, leveraging pretrained models like U-Net or ResNet, to achieve superior accuracy even with limited training data. The workflow includes key steps such as data augmentation, which improves generalization, and iterative optimization, ensuring the model adapts effectively to the data. Training also includes validation on held-out datasets to monitor performance and prevent overfitting. This robust process results in models capable of accurately predicting animal poses across diverse scenarios.

Detailed explanation of the design chart will be done in Chapter 3

2.1.2 Inference

Inference in SLEAP involves using the trained model to predict animal poses in new, unseen videos. The process is highly efficient, leveraging the optimized neural network to detect keypoints with precision and speed. Inference outputs critical pose information such as joint locations, body orientation, and movement dynamics. It is designed for scalability, enabling the rapid processing of large datasets, and ensures accurate predictions in real-world scenarios. This capability makes SLEAP a powerful tool for automating pose estimation in behavioral studies.

Detailed explanation of the design chart will be done in Chapter 4

2.1.3 Analysis

Analysis using SLEAP data focuses on extracting meaningful insights from the pose estimation results generated during inference. The output data, including precise keypoint locations and behavioral patterns, is processed to calculate key metrics such as the frequency, duration, and latency of specific behaviors. Temporal and spatial dynamics of interactions are systematically analyzed to understand social behavior and its variations across individuals or groups. This process enables researchers to uncover critical patterns, link behaviors to neural mechanisms, and derive actionable insights for behavioral studies. The structured and scalable analysis pipeline ensures reproducibility and efficiency, making it an invaluable component of behavioral neuroscience research.

Detailed explanation of the design chart will be done in Chapter 5

Chapter 3: Training

3.1 SLEAP

SLEAP (Social LEAP Estimates Animal Poses) is a versatile and user-friendly toolkit designed for multi-animal pose estimation in behavioral neuroscience and related fields. It enables researchers to analyze animal behavior by detecting and tracking body parts across video frames with high accuracy. By providing an easy-to-use interface and support for advanced deep learning models, SLEAP facilitates robust pose estimation for complex datasets, including those involving multiple interacting animals.

SLEAP supports a wide range of functionalities, including project creation, labeling, and model training. Its modular design ensures that researchers can integrate custom workflows and adapt the software for specific experimental needs. The following works have contributed significantly to advancements in animal pose estimation and the development of SLEAP: [1–5].

3.1.1 Installation Procedure

To install and set up SLEAP, follow these steps:

i) **Install Prerequisites:**

- Ensure that you have Python 3.8 or later installed on your system. SLEAP requires a compatible Python environment.
- It is recommended to use a virtual environment manager such as `conda` or `mamba` to avoid dependency conflicts.

ii) **Install SLEAP Using Mamba:**

- Use the following command to create a new environment and install SLEAP:

```
mamba create -y -n sleap \
   -c conda-forge \
   -c nvidia \
   -c sleap \
   -c anaconda sleap=1.3.3
```

iii) **Activate the Environment:**

```
mamba activate sleap
```

iv) **Verify Installation:**

- Confirm that SLEAP is successfully installed by checking its version:

```
sleap --version
```

v) **Launch SLEAP GUI:**

- To open the SLEAP graphical user interface (GUI) for project creation and labeling, run the following command:

```
sleap-label
```

- The SLEAP labeling interface should pop up within a few moments.

3.1.2 Features of SLEAP

- **Project Creation:** Users can easily create new projects, define configurations, and import videos for processing.
- **Labeling:** SLEAP provides intuitive tools for annotating frames, including random frame generation and instance selection for efficient labeling.
- **Training:** SLEAP supports hyperparameter tuning, model selection, and tracking loss during training and validation phases.
- **Human-in-the-Loop Workflow:** The software integrates human feedback into the workflow, enhancing accuracy and adaptability.

3.2 Training Process

The training process in SLEAP involves several critical steps that ensure the model learns effectively and delivers accurate pose estimation results. The first step is **data preprocessing**, where videos are resized to a resolution of **1280x720** and aligned from a top-down perspective. This ensures consistency across the dataset and optimizes it for neural network input while preserving essential features for pose estimation. Next, a **project is created** in SLEAP to organize the workflow, serving as the foundation for data processing. A base project (`D1.slp`) is initialized, and videos are sequentially added, with each video linked to a specific training dataset. The third step involves **data labeling**, where frames from the videos are annotated to identify keypoints of interest, such as joints or body parts. This labeled data provides the essential foundation for training the model. Following this, **model and hyper-parameter selection** takes place, where the architecture is chosen and parameters such as learning rate, batch size, and number of epochs are fine-tuned to optimize the model's performance. Finally, the **model training** step iteratively updates the model weights using the labeled data and hyperparameters to minimize errors and improve prediction accuracy.

The overall workflow of the training process is illustrated in Figure 3.1.

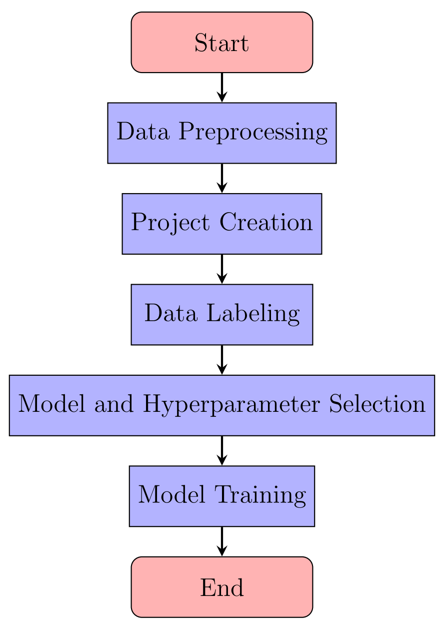

Figure 3.1: Workflow of the training process in SLEAP.

3.2.1 Data Preprocessing

Data preprocessing in SLEAP is a critical step to ensure the quality and consistency of input data, enabling the model to learn effectively from diverse datasets. One essential aspect of preprocessing is **video resizing**, which standardizes video dimensions to align with the input requirements of the neural network. Resizing ensures that videos, regardless of their original resolution, are efficiently processed without overloading computational resources, while maintaining essential details for accurate pose estimation.

In this project, we use a standardized video resolution of **1280x720** and resize the videos to ensure that each frame is correctly aligned as viewed from a top-down perspective. This alignment is crucial for consistent pose estimation, as it provides a uniform reference frame across all data.

3.2.2 Project Creation and Video Loading

The initial step in SLEAP involves creating a project and importing the videos to be analyzed. This process establishes the foundation for pose estimation by organizing the input data systematically. Figure 3.2 provides a detailed illustration of the steps required to create a new project and add videos.

In this study, a base project (`D1.slp`) is used as the starting point. Videos are then sequentially added to the project, allowing for efficient management and processing of data. For each video, the model is trained using samples extracted from that specific dataset, ensuring that the model learns effectively from the unique features and behaviors captured in each recording. This iterative approach not only enhances the training process but also enables the model to generalize well across different scenarios and datasets.

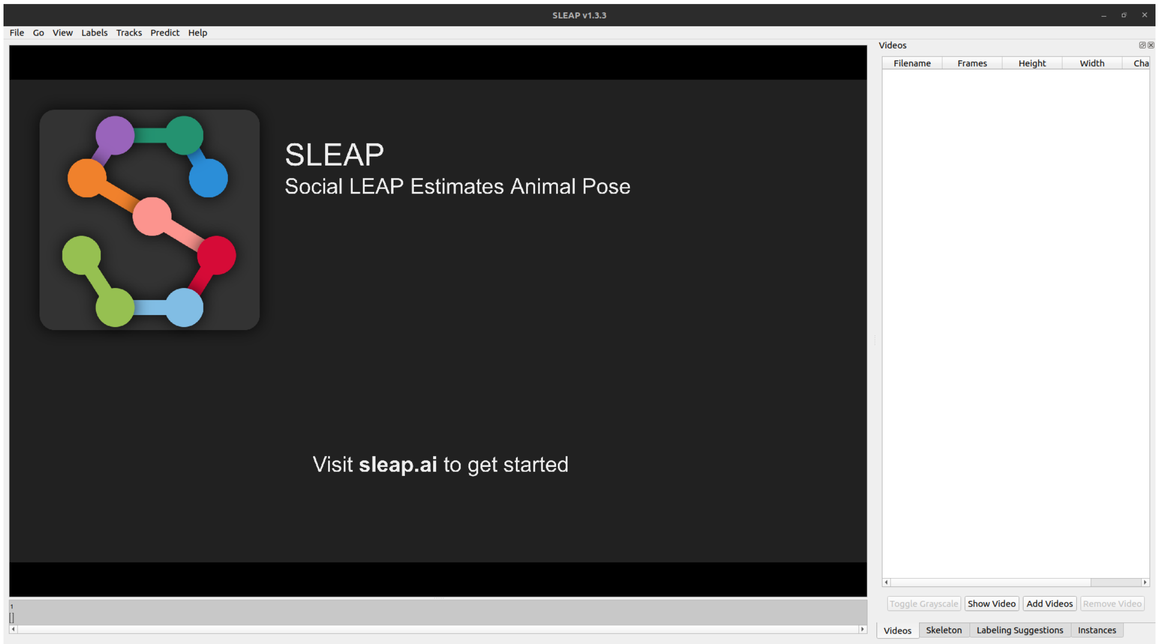

Figure 3.2: Adding a new video to the Project

3.2.3 Data Labeling

Labeling is a crucial step in the training process. Initially, 20 frames are labeled by annotating five key nodes for each instance (e.g., mice) in every frame. These nodes include the nose, neck, left ear, right ear, and tail. A skeleton is created with these nodes, which serves as a template for labeling instances across frames. This training data is used to localize and track instances in subsequent steps.

Random Frame Generation

Random frames are generated using the `Generate Suggestions` tab, which aids in selecting diverse frames for labeling (Figure 3.3).

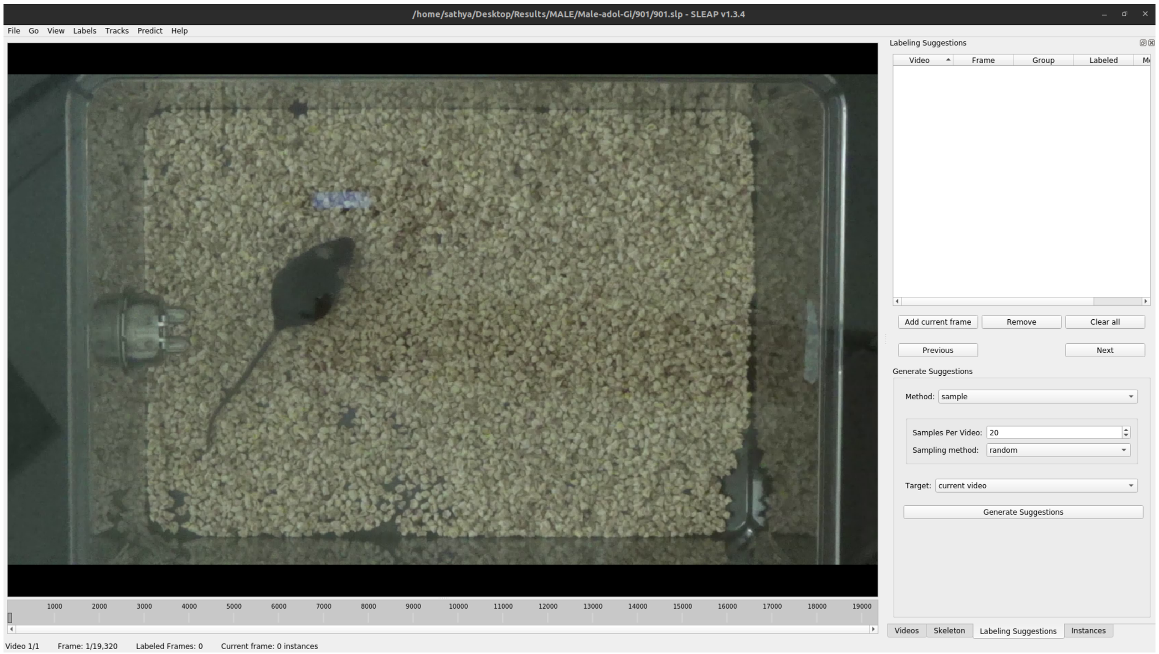

Figure 3.3: Random frame generation for labeling.

Instance Selection

Instances are labeled based on the number of mice present in each frame, providing essential training data for the model. The annotated frames serve as the foundation for accurate pose estimation and behavioral analysis, ensuring that the model can differentiate between individual subjects in complex scenarios (Figure 3.4).

3.2.4 Model Selection

For this study, we use UNet as the backbone for model training.

Figure 3.4: Labeled instance in a frame

3.2.5 Why U-Net?

U-Net is a robust convolutional neural network architecture designed for image segmentation, making it exceptionally suited for animal pose estimation tasks that require precise localization of intricate features. Its encoder-decoder architecture captures both global context and fine-grained spatial details, enabling accurate identification of keypoints such as joints and limbs.

Skip connections between the encoder and decoder layers preserve spatial consistency, preventing loss of critical details during downsampling. U-Net's modular design allows customization to balance computational efficiency and accuracy, adapting seamlessly to diverse datasets and hardware constraints. Its proven performance across varied datasets highlights its ability to generalize effectively, even in complex multi-animal scenarios.

The architecture also supports transfer learning with pretrained weights, reducing the need for extensive labeled data and streamlining the training process. U-Net integrates efficiently with frameworks like SLEAP, enabling multi-animal tracking, social interaction analysis, and scalability for large datasets.

3.2.6 Hyperparameter Selection

Several hyperparameters are adjusted to optimize the model's performance. The following settings are consistently applied across videos to ensure high accuracy. Figures 3.5, 3.6, and 3.7 detail the selected parameters.

Multi-Animal Top-Down Pipeline Parameters

The multi-animal top-down pipeline uses a combination of two models:

- **Centroid Model:** Locates and crops regions around each animal in the frame.
- **Centered Instance Model:** Generates a confidence map for predicting the node locations of each animal identified by the centroid model.

Key parameters for this pipeline are:

- **Max Instances:** 2
- **Sigma for Centroids:** 2.50
- **Anchor Part:** Neck
- **Sigma for Nodes:** 2.50

Prediction Settings

- **Predict On:** User-labeled frames (59 total frames)

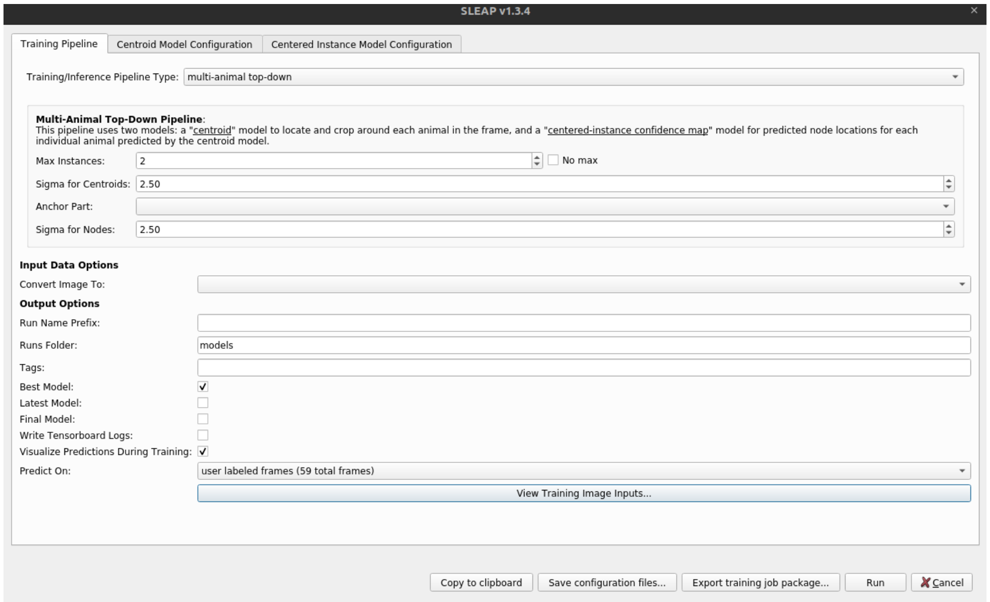

Figure 3.5: Hyperparameter selection interface.

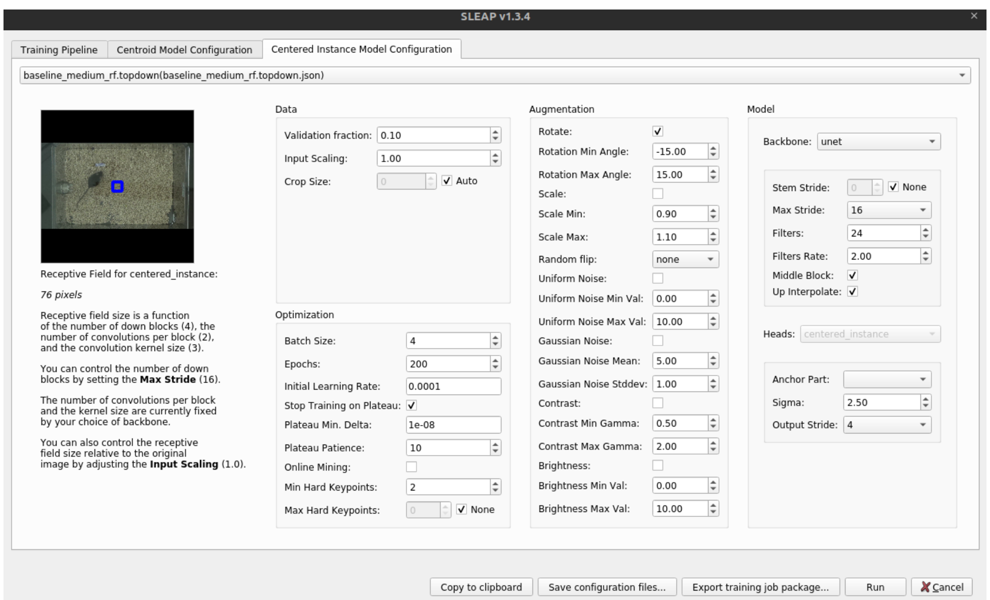

Figure 3.6: Centroid model hyperparameters.

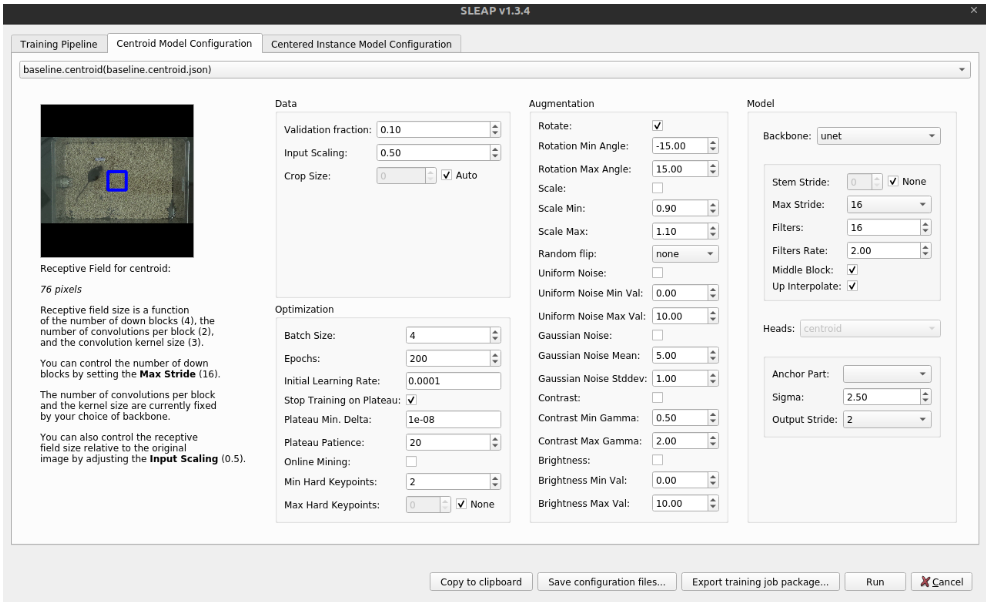

Figure 3.7: Centered instance model hyperparameters.

3.2.7 Training and Validation Loss

After preparing the labeled data and configuring the hyperparameters, the training and validation process is initiated. To ensure a robust evaluation of the model's performance, a **K-fold cross-validation approach** is employed. Specifically, the dataset is split into a **90:10 ratio**, where 90% of the data is used for training and 10% for validation in each fold. This approach helps mitigate overfitting and provides a more generalized assessment of the model's performance.

During training, the model iteratively adjusts its weights to minimize the error between the predicted and true keypoints. Validation is conducted simultaneously to monitor how well the model generalizes to unseen data. The **accuracy metric** is utilized to quantify the model's performance, evaluating how precisely the predicted keypoints align with the labeled ground truth.

Figures 3.8 and 3.9 illustrate the training and validation loss curves for the **Centroid Model** and the **Centered Instance Model**, respectively. These graphs provide critical insights into the learning dynamics of the models:

- **Training Loss:** Represents the error on the training dataset, indicating how well the model learns the patterns in the labeled data.
- **Validation Loss:** Reflects the model's ability to generalize to unseen data, providing an estimate of its real-world performance.

The convergence of the loss curves over time demonstrates the effectiveness of the training process.

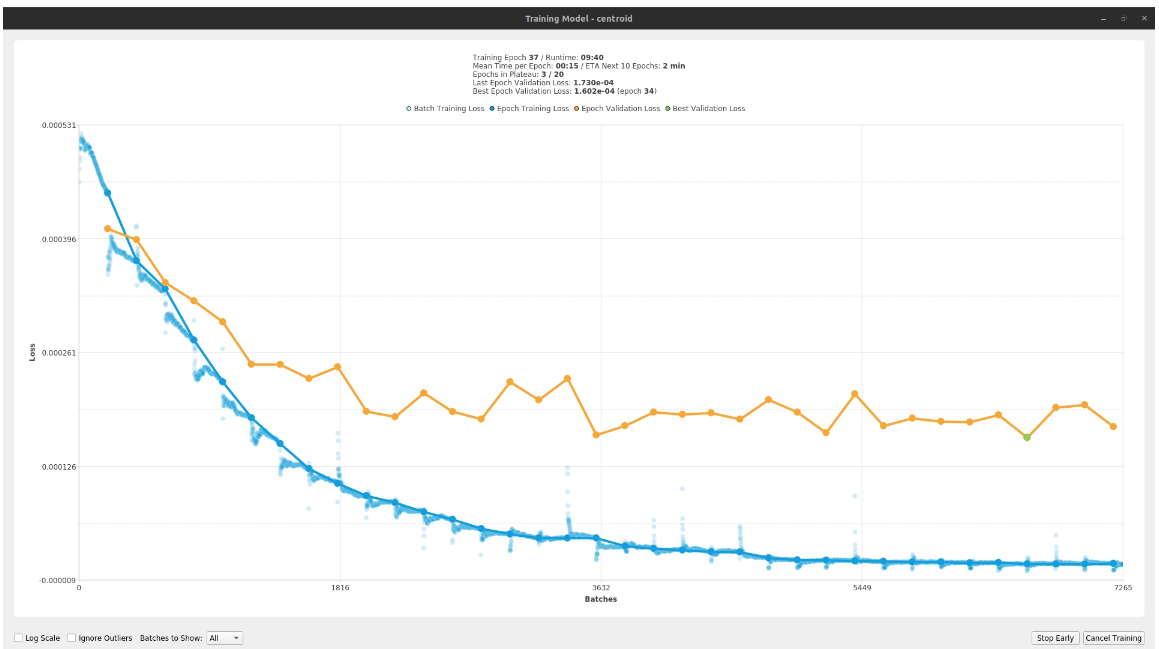

Figure 3.8: Training and validation loss for the Centroid model.

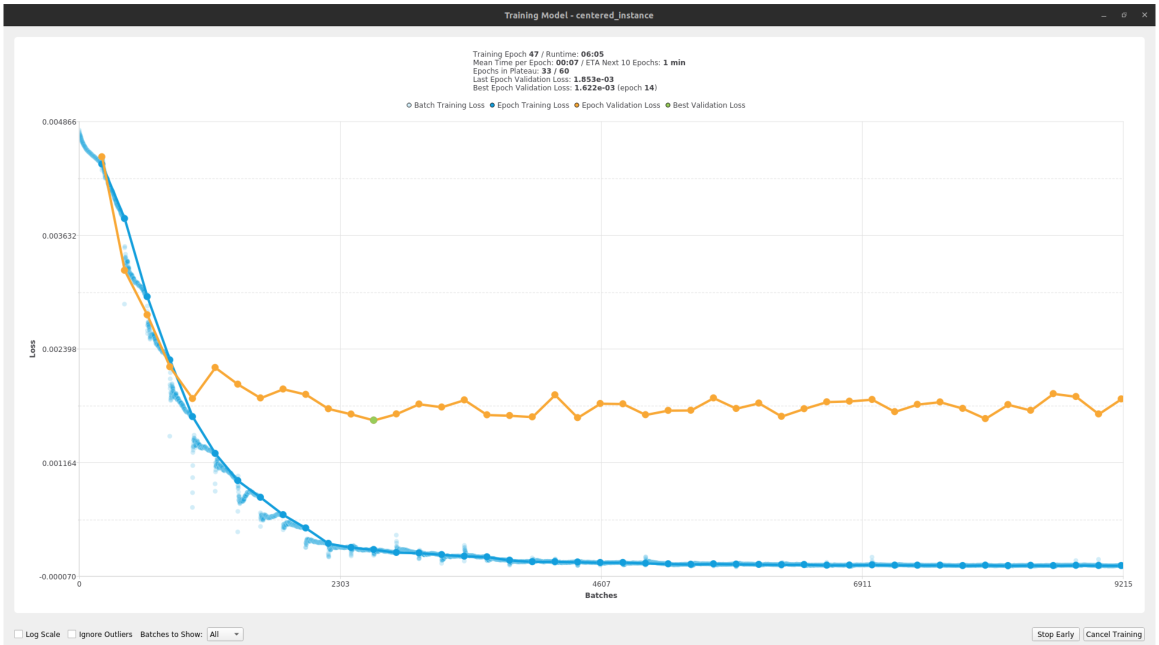

Figure 3.9: Training and validation loss for the Centered Instance model.

3.2.8 Human-in-the-Loop Training

The human-in-the-loop approach involves iteratively refining the model based on user feedback. After evaluating the model's predictions on labeled data, adjustments are made to improve mean prediction scores. Additional labeled data is incorporated, and the training process is repeated until the predictions achieve satisfactory accuracy.

This iterative process is extended to include new videos. For each new video, a few frames are labeled, and the model is retrained. This process ensures that the model generalizes effectively across diverse video datasets. The training loss typically reaches a plateau as the model becomes more robust across datasets.

Figures 3.10 and 3.11 illustrate the training loss progression after incorporating 642 labeled frames.

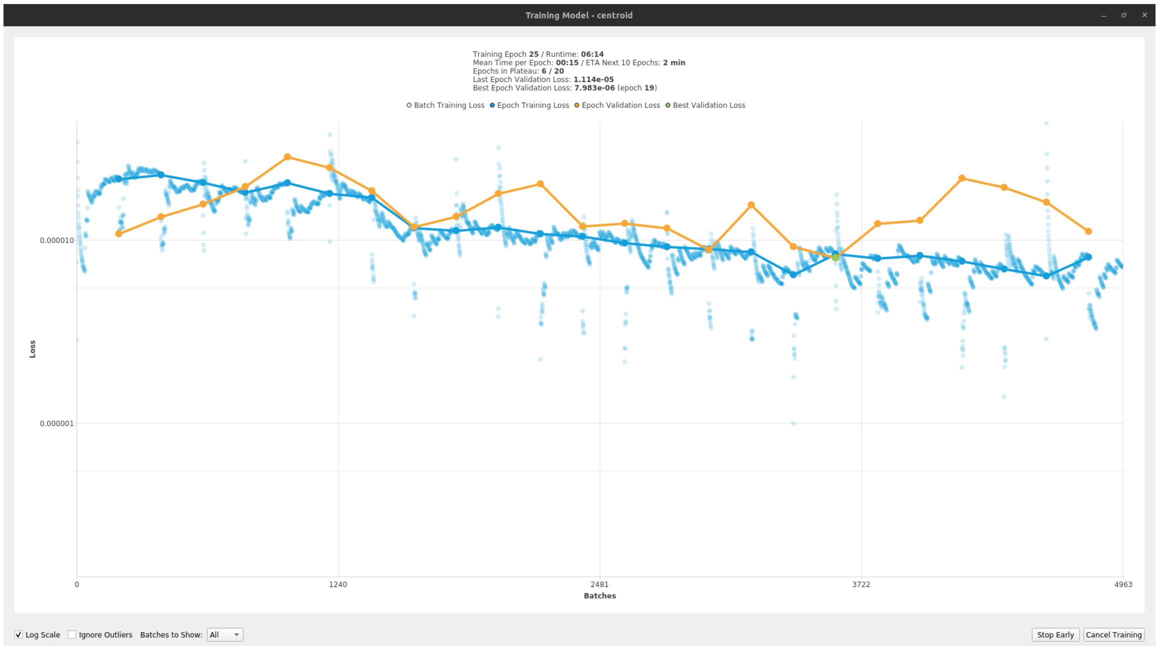

Figure 3.10: Training loss progression for the Centroid model with 642 labeled frames.

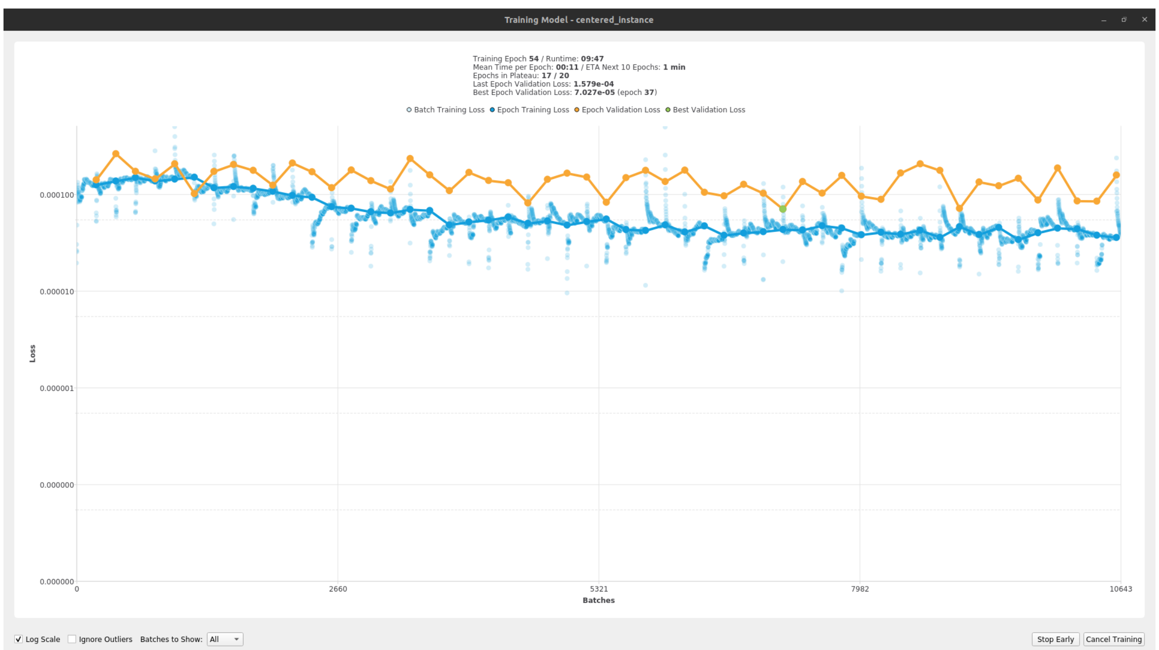

Figure 3.11: Training loss progression for the Centered Instance model with 642 labeled frames.

Chapter 4: Inference

4.1 Project Creation

In this chapter, we outline the process for running inference on videos using trained models and exporting the results. Each video will be handled individually to ensure precise analysis and structured outputs.

4.1.1 Project Setup

To begin, we create a new project for each video. This ensures that the data for each video is encapsulated within its own workspace. The workflow involves the following steps:

i) Load the video into the project.

ii) Apply the pre-trained model to run inference.

iii) Verify the predicted frames for accuracy.

If the overall prediction results are unsatisfactory, refer back to Chapter 3.2.8, *Human-in-the-Loop Training*, to retrain the model and improve its performance.

4.2 Run Inference

Once the project is set up, we proceed with running inference using the trained model. The process is depicted in Figure 4.1, which provides a visual representation of the inference workflow.

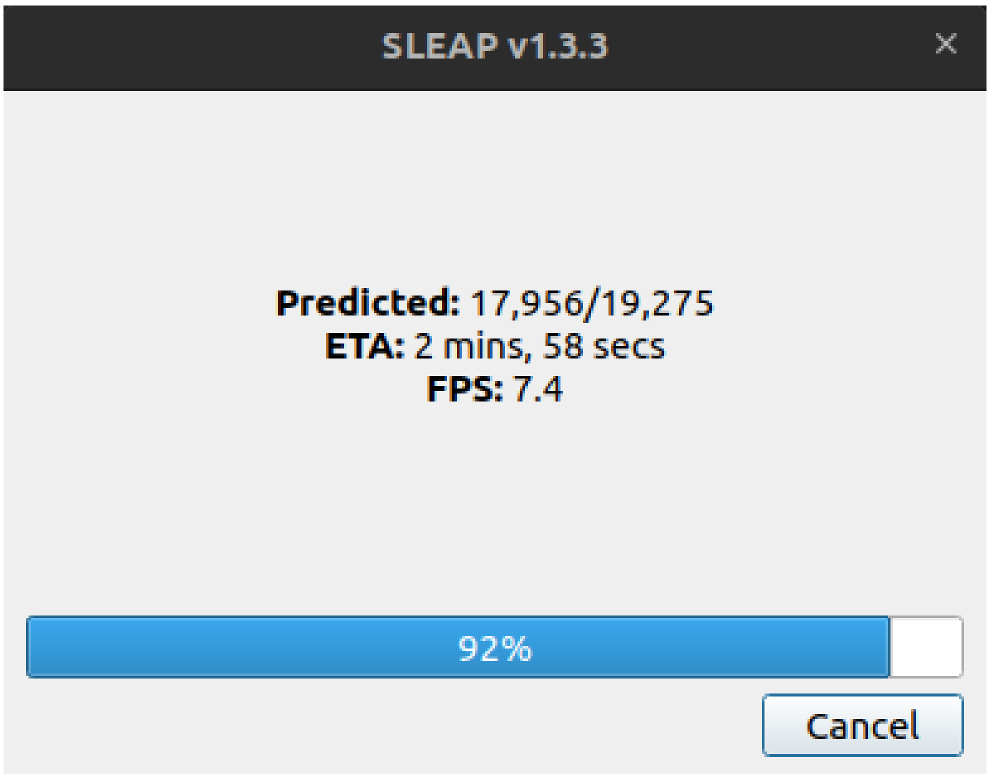

Figure 4.1: Illustration of the inference process.

During inference, the trained model processes each frame of the video to generate predictions. The results are reviewed to ensure that the model's output meets the desired accuracy standards.

4.3 Export H5 File for Each Video

Following the completion of inference, the results are exported as HDF5 (H5) files for each video. These H5 files are structured to include the following datasets:

- `edge_inds`
- `edge_names`
- `instance_scores`
- `labels_path`
- `node_names`
- `point_scores`
- `provenance`
- `track_names`
- `track_occupancy`
- `tracking_scores`
- `tracks`
- `video_ind`
- `video_path`

4.3.1 Future Use of H5 Files

The exported H5 files serve as the primary data source for subsequent analysis. The information encapsulated within these files will be utilized in the next chapter to conduct detailed evaluations and extract insights.

This chapter detailed the inference process, including project creation, running the trained model, and exporting results as H5 files. The structured datasets within these files form the foundation for further analysis, ensuring a streamlined workflow for handling video data.

Chapter 5: Data Analysis

5.1 Introduction

As part of the data analysis, various parameters, plots, and results are extracted to assist neuroscientists in analyzing the social behavior of mice. This chapter focuses on analyzing behavioral parameters from individual videos and groups of five videos. The specific behaviors analyzed include:

- Nose-to-tail sniffing (ano-genital sniffing)
- Nose-to-nose sniffing
- Nose-to-right-ear sniffing
- Nose-to-left-ear sniffing
- Nose-to-neck sniffing

The analysis includes the determination of the first occurrence of each behavior, quantification of the total occurrences, and examination of latency during the first three minutes of the video. The following results were derived from a video (853) of the Female Group C.

5.2 Behavioral Parameters and Statistics

5.2.1 Event Counts and Statistics

Summary of Sniffing Behaviors

Table 5.1 provides a summary of the sniffing behaviors observed during the study.

Table 5.1: Summary of Sniffing Behaviors

Event Type	Number of Events	Total Number of Events
Nose 0 to Tail 1	146	344
Nose 1 to Tail 0	198	
Nose 0 to Neck 1	392	810
Nose 1 to Neck 0	418	
Nose to Nose	406	406
Nose 0 to Left Ear 1	352	694
Nose 1 to Left Ear 0	342	
Nose 0 to Right Ear 1	270	574
Nose 1 to Right Ear 0	304	

5.2.2 First Occurrences of Behaviors

The first occurrence of each behavior is summarized in Table 5.2.

5.3 Latency Analysis

The latency for each behavior during the first three minutes of the video was analyzed to identify temporal patterns and dynamics in social interactions. The data extracted from the H5 file provides a comprehensive overview of these interactions. Table 5.3 summarizes the occurrences and total time for each behavior across the three-minute intervals.

Table 5.2: First Occurrences of Behaviors

Behavior	Frame	Time (s)
Nose 0 to Right Ear 1	663	22.10
Nose 0 to Left Ear 1	663	22.10
Nose-to-Nose	663	22.10
Nose 0 to Neck 1	663	22.10
Nose 1 to Neck 0	665	22.17
Nose 1 to Right Ear 0	679	22.63
Nose 1 to Left Ear 0	679	22.63
Nose 0 to Tail 1	690	23.00

5.4 Behavioral Graphs

The following graphs represent the occurrences of the analyzed behaviors:

5.4.1 Graphical Representation for video 853

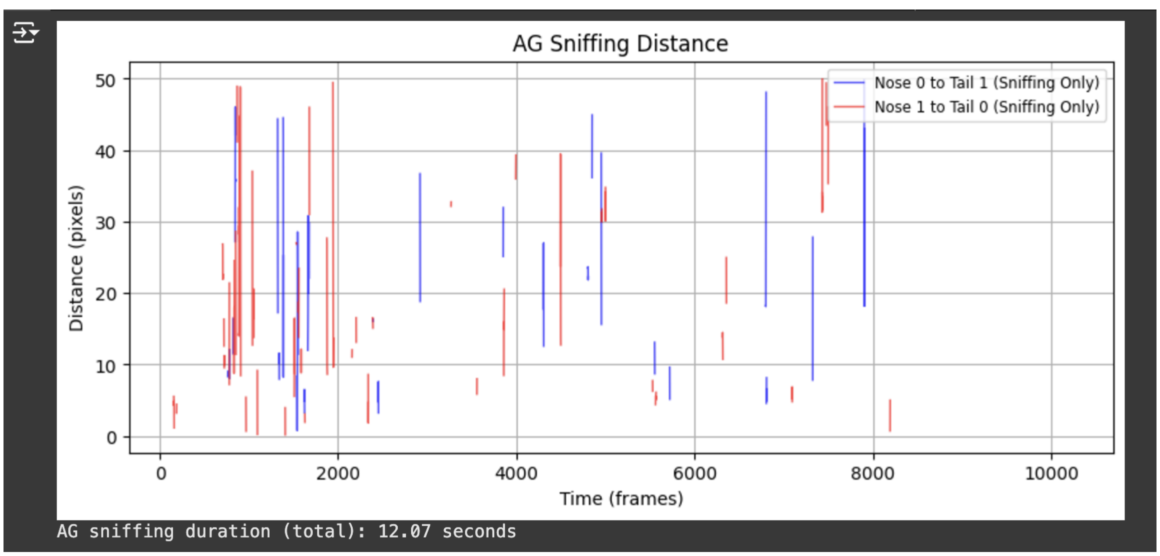

Figure 5.1: Graph representing Ano-Genital Sniffing events.

Table 5.3: Latency Behavior Statistics Over Three Minutes

Behavior	**Minute**	**Occurrences**	**Total Time (s)**
Nose 0 to Right Ear 1	1	60	2.00
	2	112	3.73
	3	73	2.43
Nose 1 to Right Ear 0	1	65	2.17
	2	105	3.50
	3	71	2.37
Nose 0 to Left Ear 1	1	67	2.23
	2	184	6.13
	3	98	3.27
Nose 1 to Left Ear 0	1	75	2.50
	2	177	5.90
	3	109	3.63
Nose to Nose	1	116	3.87
	2	229	7.63
	3	134	4.47
Nose 0 to Neck 1	1	82	2.73
	2	172	5.73
	3	98	3.27
Nose 1 to Neck 0	1	91	3.03
	2	181	6.03
	3	103	3.43
Nose 0 to Tail 1	1	64	2.13
	2	21	0.70
	3	23	0.77

5.4.2 Graphical Representation for Female Group C

The results from video 853 in Female Group C provide insights into the social behaviors of mice, which can be used by neuroscientists to further analyze patterns and interactions. By quantifying occurrences, examining latencies, and visualizing behaviors, these analyses contribute significantly to understanding the dynamics of social behavior in mice.

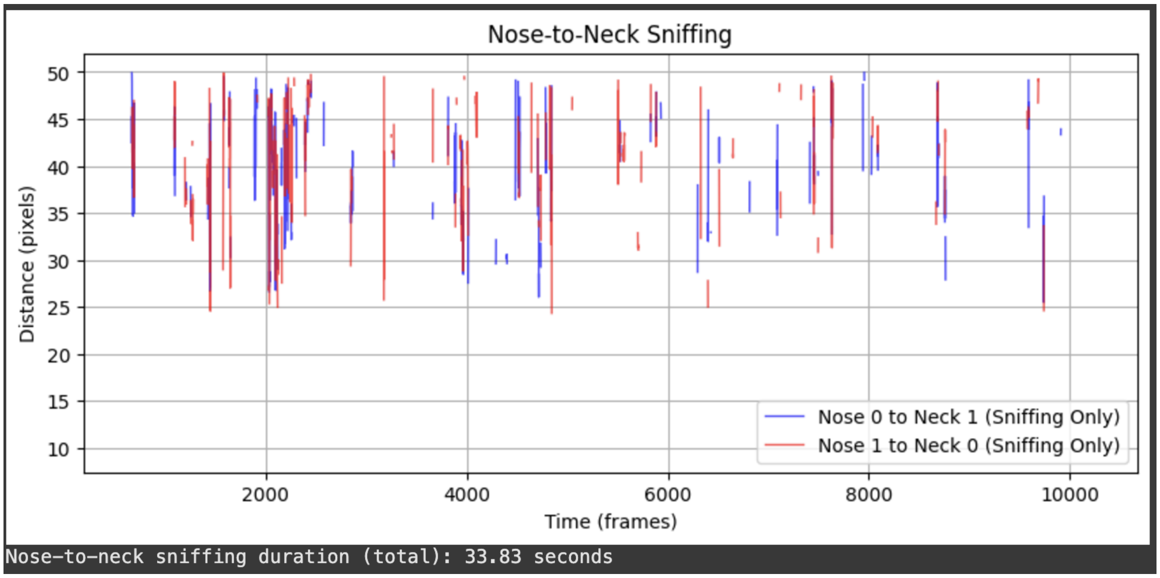

Figure 5.2: Graph representing Nose-to-neck Sniffing events.

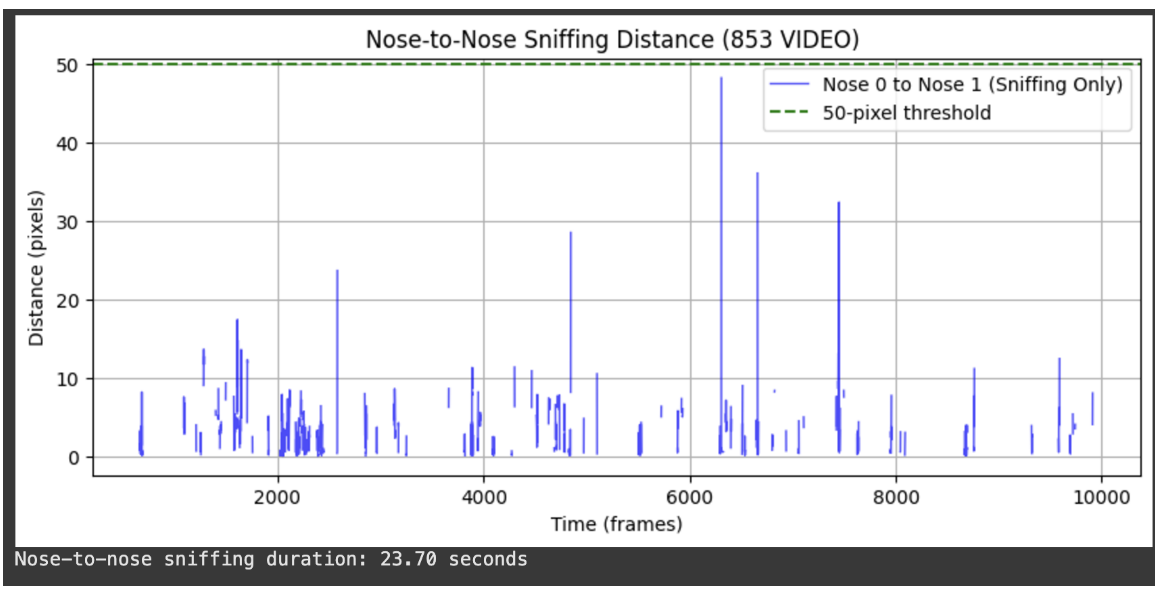

Figure 5.3: Graph representing Nose-to-Nose Sniffing events.

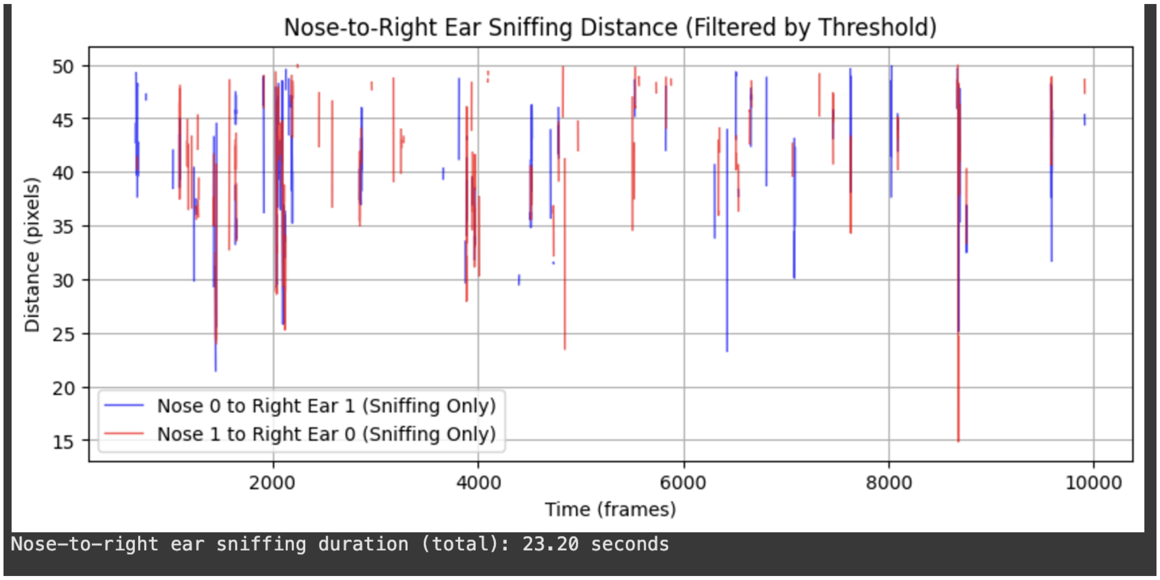

Figure 5.4: Graph representing Nose-to-Right Ear Sniffing events.

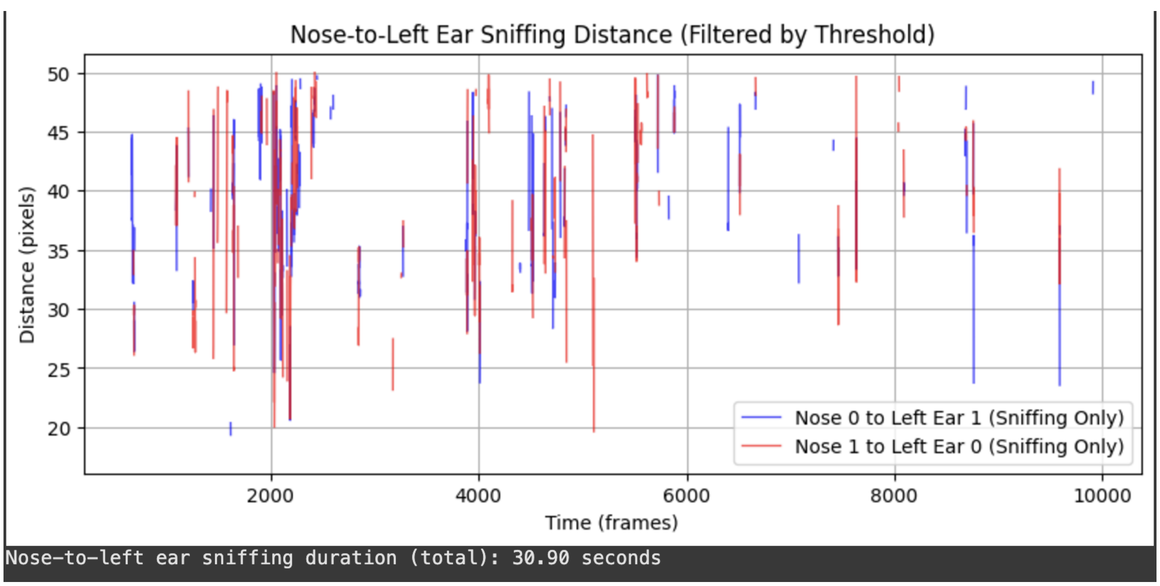

Figure 5.5: Graph representing Nose-to-Left Ear Sniffing events.

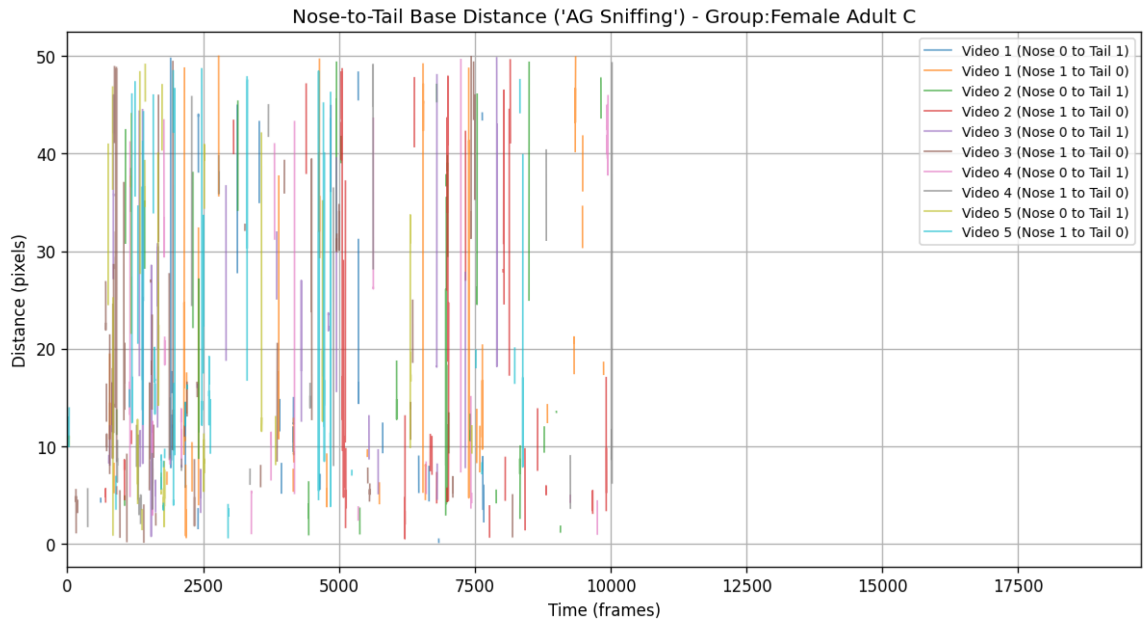

Figure 5.6: Graph representing Ano-Genital Sniffing events.

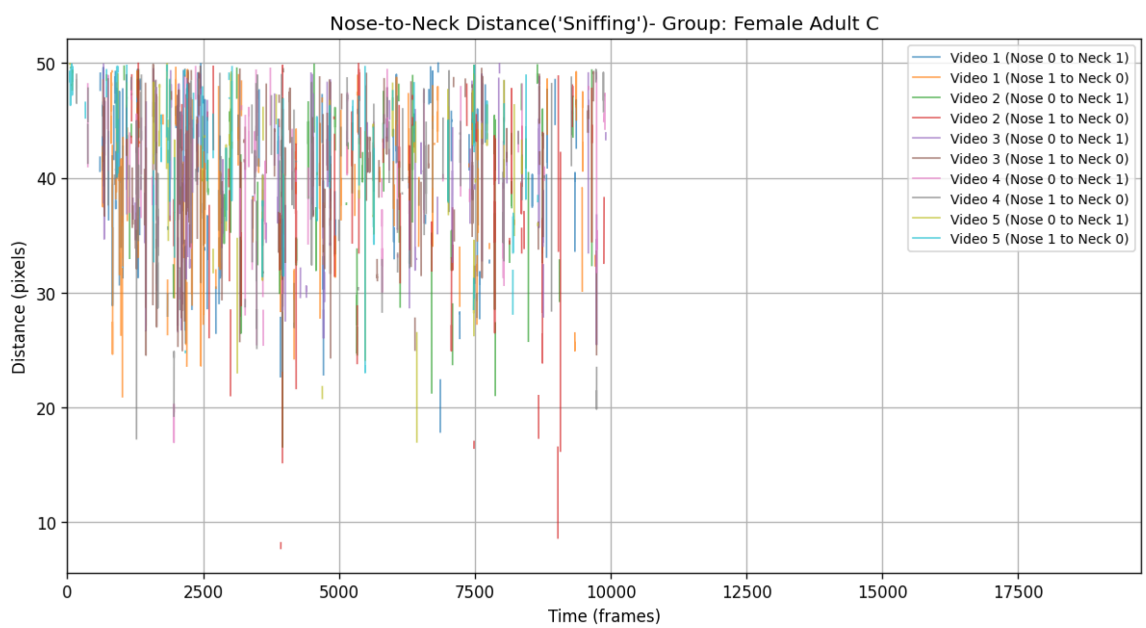

Figure 5.7: Graph representing Nose-to-neck Sniffing events.

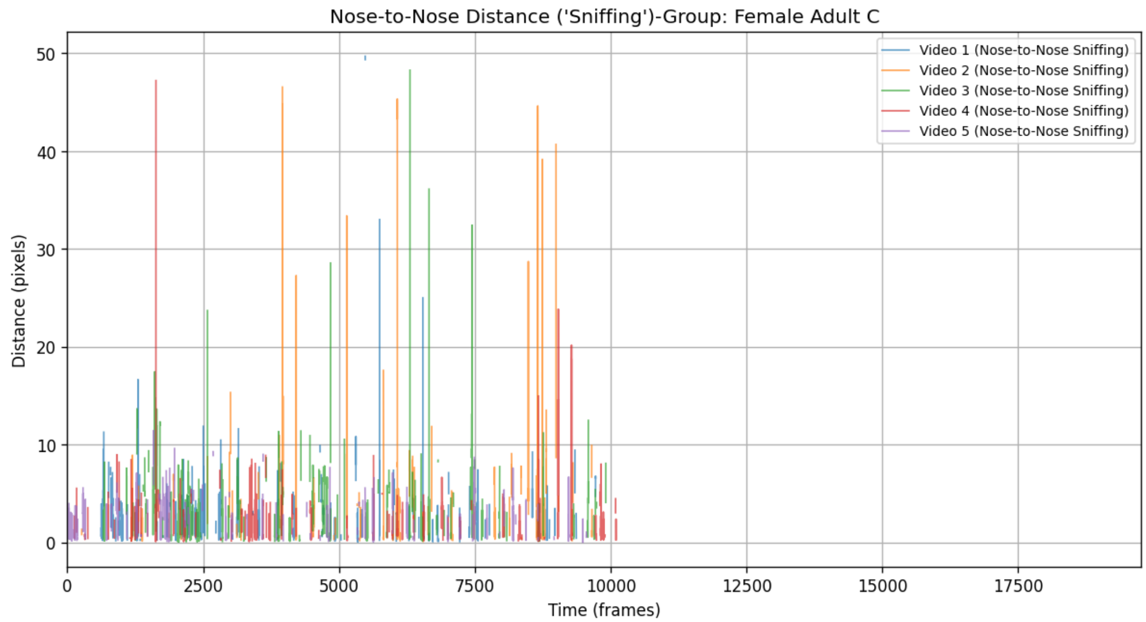

Figure 5.8: Graph representing Nose-to-Nose Sniffing events.

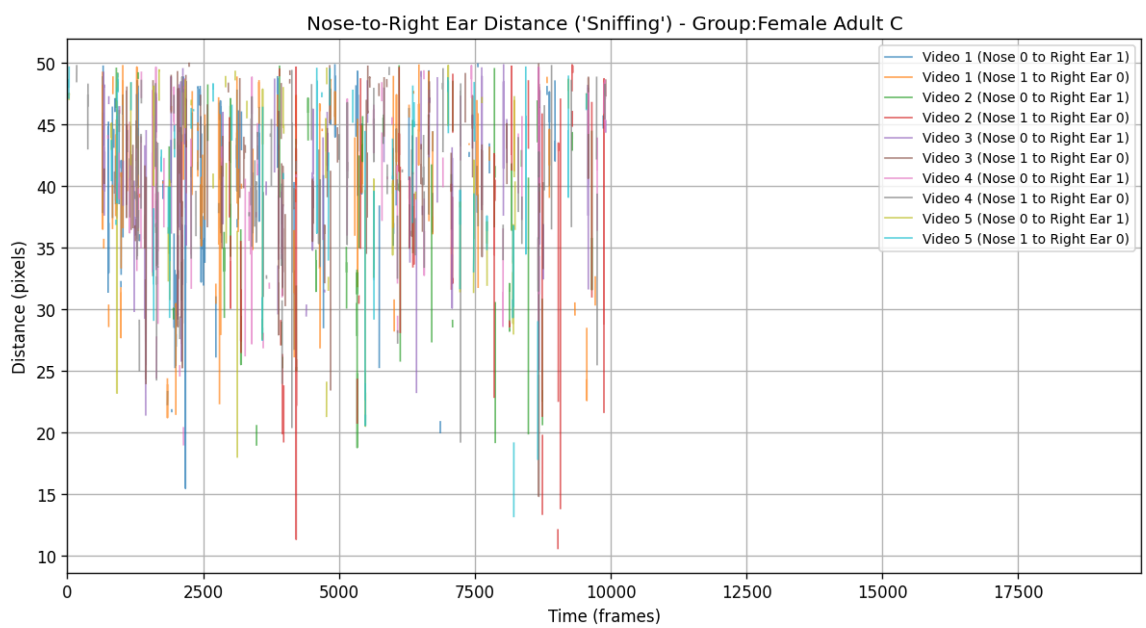

Figure 5.9: Graph representing Nose-to-Right Ear Sniffing events.

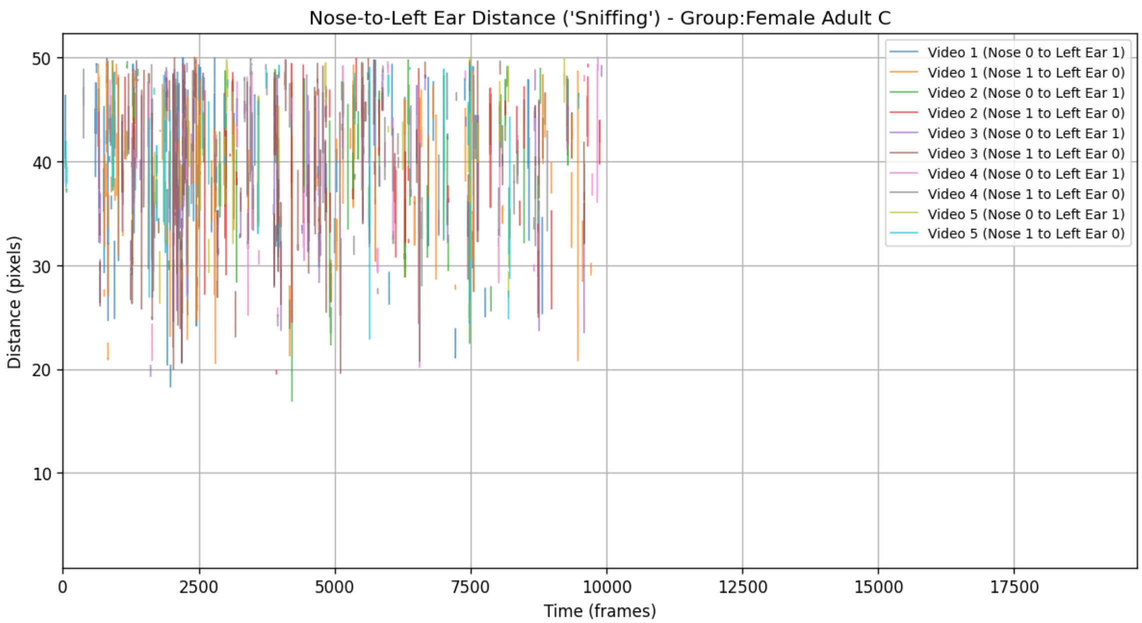

Figure 5.10: Graph representing Nose-to-Left Ear Sniffing events.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

This study demonstrates the successful application of SLEAP, a state-of-the-art deep learning framework, to automate the analysis of mouse social behaviors with precision and efficiency. By leveraging advanced neural architectures, such as U-Net, the project accurately quantified key behavioral metrics, including frequency, duration, and latency, providing detailed insights into the temporal and spatial dynamics of social interactions. The integration of automated pose estimation significantly reduces the manual effort required for behavioral studies while ensuring scalability and reproducibility across large datasets. This framework represents a pivotal advancement in behavioral neuroscience, bridging the gap between observed behaviors and their underlying neural mechanisms. The results of this study not only enhance our understanding of complex social dynamics in mice but also establish a robust and scalable foundation for future research in behavioral and computational neuroscience.

6.2 Future Work

Building upon the outcomes of this study, several promising directions can be explored to further enhance the framework's capabilities and applications. Integrating the methodology with tools like MoSeq could enable the identification and classification of stereotyped behaviors, facilitating advanced temporal and sequential behavioral analyses. Expanding the workflow to incorporate neural activity data would provide a comprehensive understanding of the correlation between specific behaviors and their underlying neural dynamics. Additionally, the framework could be adapted for real-time pose estimation, enabling immediate behavioral assessments and adaptive experimental interventions. Extending the application to cross-species analysis would uncover universal and species-specific behavioral patterns, broadening the scope of comparative neuroscience. Finally, enhancing cross-platform compatibility would allow seamless integration with other behavioral analysis tools, fostering interdisciplinary collaboration and innovation. These advancements would not only improve the scalability and precision of behavioral analysis but also deepen our understanding of the neural and environmental factors influencing social behavior.

Bibliography

[1] T. D. Pereira, D. E. Aldarondo, L. Willmore, M. Kislin, S. S. Wang, M. Murthy, and J. W. Shaevitz, "Fast animal pose estimation using deep neural networks," *Nature Methods*, vol. 16, no. 1, pp. 117–125, 2019. [Online]. Available: https://www.nature.com/articles/s41592-018-0234-5

[2] T. D. Pereira, N. Tabris, J. Li, S. Ravindranath, E. S. Papadoyannis, Z. Y. Wang, D. M. Turner, G. McKenzie-Smith, S. D. Kocher, A. L. Falkner, J. W. Shaevitz, and M. Murthy, "Sleap: Multi-animal pose tracking," *bioRxiv*, 2020. [Online]. Available: https://www.biorxiv.org/content/10.1101/2020.08.31.276246v1

[3] T. D. Pereira *et al.*, "Sleap: A deep learning system for multi-animal pose tracking," *Nature Methods*, vol. 19, pp. 486–495, 2022.

[4] T. D. Pereira, N. Tabris, A. Matsliah, D. M. Turner, J. Li, S. Ravindranath, E. S. Papadoyannis, E. Normand, D. S. Deutsch, Z. Y. Wang, G. C. McKenzie-Smith, C. C. Mitelut, M. D. Castro, J. D'Uva, M. Kislin, D. H. Sanes, S. D. Kocher, S. S-H, A. L. Falkner, J. W. Shaevitz, and M. Murthy, "Sleap: A deep learning system for multi-animal pose tracking," *Nature Methods*, vol. 19, no. 4, 2022.

[5] T. Pereira, N. Tabris, A. Matsliah, D. Turner, J. Li, S. Ravindranath, E. Papadoyannis, E. Normand, D. Deutsch, Z. Wang, G. McKenzie-Smith, C. Mitelut, M. Castro, J. D'Uva, M. Kislin, D. Sanes, S. Kocher, S. Wang, A. Falkner, J. Shaevitz, and M. Murthy, "Sleap: A deep learning system for multi-animal pose tracking," *Nature Methods*, vol. 19, no. 4, pp. 486–495, Apr. 2022, publisher Copyright: © 2022, The Author(s).