

UNIVERSITY OF OKLAHOMA
GRADUATE COLLEGE

A COMPUTER VISION PIPELINE FOR AUTOMATED BIRD BEHAVIOR
RECOGNITION IN A FIXED-CAMERA AVIARY SETTING

A THESIS
SUBMITTED TO THE GRADUATE FACULTY
in partial fulfillment of the requirements for the
Degree of
MASTER OF SCIENCE

By
GLADIS MENACHERY SUNNY
Norman, Oklahoma
2026

A COMPUTER VISION PIPELINE FOR AUTOMATED BIRD BEHAVIOR
RECOGNITION IN A FIXED-CAMERA AVIARY SETTING

A THESIS APPROVED FOR THE
SCHOOL OF COMPUTER SCIENCE

BY THE COMMITTEE CONSISTING OF

Dr. Golnaz Habibi, Chair

Dr. Alexandra Bentz

Dr. Chao Lan

Abstract

Understanding the social behavior of house sparrows (*Passer domesticus*) is fundamental to studying how dominance hierarchies form, how nest site competition shapes breeding success, and how affiliative and aggressive interactions drive the structure of avian social networks. Characterizing these behaviors requires continuous observation of multiple individuals simultaneously in naturalistic settings, a task that is fundamentally beyond the capacity of manual annotation, which captures only a small fraction of behavioral events for a few individuals at a time and cannot scale to the population-level monitoring.

This thesis presents an automated bird behavior classification pipeline for detecting, tracking, and classifying the behaviors of house sparrows from fixed-camera aviary footage, enabling population-level behavioral analysis at a scale and temporal resolution that manual observation cannot achieve. The pipeline combines a fine-tuned YOLOv8 bird detector achieving a recall of 0.997 and mAP@0.5 of 0.979, a custom multi-object tracking system with novel ghost track logic for maintaining house sparrow aviary tracking continuity during nest box occlusion, and a temporal behavior recognition model built on a Vision Transformer backbone with a causal temporal transformer and a spatial feature vector encoding positional, motion, and social context information. The temporal model achieves an overall accuracy of 0.9958 and macro F1 of 0.9844 across four behavior classes - on_box, at_hole, in_box, and none. A specialized interaction detector extends the pipeline to side-by-side affiliative behavior classification, achieving mAP@0.5 of 0.984.

The system is developed and evaluated on footage of house sparrows housed in outdoor aviaries at the University of Oklahoma, using five synchronized GoPro cameras per enclosure and behavioral annotations provided by expert observers from the Department of Biology. The pipeline directly extends the 3D tracking infrastructure of prior work from our research group, which established where birds are in the aviary by addressing the complementary question of what behaviors those birds are performing.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Dr. Golnaz Habibi, for her guidance, support, and encouragement throughout this research. Their insights and feedback shaped every aspect of this work and I am deeply grateful for the opportunity to pursue this research under their supervision. I would like to thank my committee members, Dr. Alexandra Bentz, Dr. Chao Lan, for their valuable feedback and constructive suggestions during the course of this thesis.

I am grateful to the members of AIROU Lab at the University of Oklahoma for their support and collaboration. I would like to specifically thank Keon Moradi, Ethan Haque, and others whose prior work on 3D sparrow tracking provided the foundation on which this thesis builds. I would like to thank the Department of Biology at the University of Oklahoma, particularly Jasmeen Kaur, for providing the behavioral annotations that made this research possible. Their biological expertise was essential to the ground truth labels on which the entire pipeline depends. This research was conducted using resources provided by the Oklahoma Supercomputing Center for Education and Research (OSCER) at the University of Oklahoma. I gratefully acknowledge their computational support.

Finally I would like to thank my family and friends for their unwavering support and encouragement throughout my graduate studies. This work would not have been possible without them.

Table of Contents

List Of Tables	ix
List Of Figures	xi
1 Introduction	1
1.1 Thesis Organization	1
2 Related Work	3
2.1 Multi-view Bird Tracking in Aviary Settings	3
2.2 Animal Behavior Recognition	5
2.3 Multi-Object Tracking	8
2.4 Temporal Behavior Classification	8
2.5 Summary and Research Gap	9
2.6 Thesis Contributions	10
3 Dataset Collection and Annotation	11
3.1 Overview	11
3.2 Aviary Setup and Recording Infrastructure	11
3.3 Behavioral Annotations	12
3.3.1 Behavior Definitions	13
3.4 Video Clip Extraction	16
3.5 Frame Rate Conversion	18
3.6 CVAT Annotation	18
3.6.1 Annotation Schema	19
3.6.2 Annotation Workflow	19
3.6.3 Annotation Statistics	21
3.7 Dataset Challenges	22
3.8 Summary	23
4 System Architecture and Implementation	24
4.1 Overview of the Full Pipeline	24
4.2 Training Pipeline	25

4.2.1	Bird Detection	25
4.2.1.1	Model Selection and Architecture	25
4.2.2	Temporal Behavior Classification Model	27
4.2.2.1	Architecture Overview	27
4.2.2.2	ViT small/patch8 Backbone	28
4.2.2.3	Spatial Features	30
4.2.2.4	Fusion Layer	32
4.2.2.5	Causal Transformer	34
4.2.2.6	Social Context Layer	37
4.2.2.7	Classification Head	39
4.2.2.8	Loss Function and Optimization Design	39
4.3	Inference Pipeline	41
4.3.1	ByteTrack with Ghost Track Logic	41
4.3.2	Hole ROI Annotation	42
4.3.3	Weighted Box Fusion	43
4.3.4	Duplicate Track Suppression	43
4.3.5	Track Stitching	44
4.3.6	Behavior Inference	44
4.4	Visualization	45
4.5	Summary	46
5	Experiments and Results	47
5.1	Experimental Setup	47
5.2	Bird Detection	47
5.2.1	Training Procedure	47
5.2.2	Training Dynamics	49
5.2.3	Validation Results	50
5.2.4	Analysis of mAP@0.5 versus mAP@0.5:0.95 Gap	51
5.2.5	Social Interaction Detection	52
5.3	Temporal Behavior Classification	54
5.3.1	Training Configuration	54
5.3.2	Validation Results	55
5.3.3	Per-Class Analysis	56
5.3.4	Confusion Matrix Analysis	57
5.3.5	Effect of Focal Loss on Class Imbalance	58

5.3.6	Qualitative Results	58
5.4	Discussion	60
5.4.1	System Performance in Context	60
5.4.2	Key Design Decisions	61
5.4.3	Implications for Biology Research	61
5.4.4	Limitations and Future Work	62
6	Conclusion	64
6.1	Summary	64
6.2	Contributions	65
6.3	Broader Impact	66

List Of Tables

3.1	Annotation Statistics for Neutral Behavior Dataset	21
5.1	YOLOv8s bird detection validation results across two training stages. Both stages evaluated on the same held-out validation set of 461 images and 320 instances.	50
5.2	Interaction detector validation results for side-by-side behavior detection . .	52
5.3	Temporal model training configuration	54
5.4	Training frame distribution across behavior classes	55
5.5	Temporal behavior classification model validation results	56

List Of Figures

3.1	Aviary camera setup showing two selected views (GoPro3 and GoPro5), each capturing a different perspective of Enclosure 9.	12
3.2	Sample of the behavioral annotation file showing the column structure and sample rows of annotated behavioral events.	13
3.3	Examples of each behavior class as seen in the aviary camera footage. (a) on_box : bird perched on top of nest box. (b) at_hole : bird positioned at hole entrance. (c) in_box : bird peeking head out of hole.(d) side-by-side : two birds perched shoulder to shoulder on nest box.	16
3.4	CVAT annotation interface showing a bounding box annotation for a bird exhibiting on_box behavior. The Behavior attribute is set to on_box and the track panel on the left shows the active track for the annotated bird.	20
3.5	An annotated frame showing the three neutral behaviors (1) Bird1 : at_hole (2) Bird2 : on_box (3) Bird3 : on_box	22
4.1	Full pipeline diagram showing two phases. Training phase & Inference phase	24
4.2	Architecture diagram of the temporal behavior classification model. Input:T=16 cropped bird frames + 13-dimensional spatial feature vectors. Stages: ViT-small/patch8 backbone → fusion layer → causal transformer → social context layer → classification head.	27
4.3	The BoxROI annotation tool showing a reference frame from Enclosure 9 with circular ROI markers placed over each nest hole entrance. Each marker shows the hole center point and the circular proximity region used by the ghost track logic.	43
5.1	Stage 2 YOLOv8s training curves across 100 epochs. Top row shows training losses (box loss, classification loss, DFL loss) alongside validation precision and recall. Bottom row shows validation losses and detection metrics (mAP@0.5 and mAP@0.5:0.95). Training losses decrease smoothly throughout while validation mAP@0.5 stabilizes around 0.979, confirming model convergence on the expanded dataset.	49
5.2	Precision-Recall curve for the final Stage 2 YOLOv8s detector on the held-out validation set.	51

5.3	Precision-Recall curve for the interaction detector on the held-out validation set, showing high area under the curve consistent with mAP@0.5 of 0.984.	53
5.4	Confusion matrix for temporal behavior classification. Rows = true labels, Columns = predicted labels.	57
5.5	Sample annotated frames from the visualization layer with behavior predictions across three behavioral classes: on_box, at_hole, none	59
5.6	Sample annotated frames from the visualization layer with behavior predictions across 3 behavioral classes: in_box, on_box, none	59
5.7	Sample annotated frames from the visualization layer with side-by-side interaction detection. A combined interaction bounding box (magenta) is drawn around the interacting pair once the bout is confirmed over a minimum of 90 consecutive frames at 30 fps.	60

Chapter 1

Introduction

Animal behavior research has traditionally relied on direct observation and manual annotation to study the social dynamics of birds in controlled environments. While these methods provide valuable insights, they are inherently time-consuming, labor-intensive, and prone to observer bias. As studies scale to longer observation periods and larger populations, these limitations become increasingly significant. Recent advances in computer vision and deep learning have enabled automated detection and tracking of animals across diverse environments. However, most existing systems have been developed for large or visually prominent animals, such as livestock and primates, where detection is comparatively straightforward. In contrast, small birds such as sparrows present a significantly greater challenge: they are fast-moving, visually similar, and occupy only a few pixels in fixed-camera aviary recordings. These factors make both detection and behavior classification substantially more difficult.

Sparrows are widely studied in behavioral ecology due to their rich repertoire of social interactions, including competition, affiliation, and reproductive behaviors. In aviary settings, sparrows have been used to understand the dynamics of group living Kubitza et al. (2015) including neutral baseline behaviors such as perching on nest boxes, approaching nest hole entrances, and occupying nest boxes, as well as more complex interactions such as side-by-side perching, displacement, and copulation. Traditionally, identifying and quantifying these behaviors requires manual review of extensive video recordings, limiting both the scale and frequency of analysis. This thesis presents an automated pipeline for tracking and classifying sparrow behaviors across three enclosures in the outdoor aviary, each equipped with five synchronized GoPro cameras providing comprehensive multi-angle coverage.

1.1 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 reviews related work in animal detection, tracking, and behavior recognition. Chapter 3 describes the dataset collection and annotation methodology. Chapter 4 presents the system architecture and implementation

details. Chapter 5 reports experimental results and evaluation metrics. Chapter 6 concludes the thesis with a summary of findings and directions for future work.

Chapter 2

Related Work

The automated tracking and behavior recognition of animals using computer vision has emerged as a rapidly growing field, driven by the need to scale behavioral ecology research beyond what manual observation can achieve. This chapter reviews the relevant literature across four interconnected areas: multi-view bird tracking in aviary settings, animal behavior recognition using deep learning, multi-object tracking methods, and small object detection - each of which directly informs the methodology developed in this thesis.

2.1 Multi-view Bird Tracking in Aviary Settings

The most closely related body of work to this thesis involves multi-camera systems deployed in aviary environments to track groups of interacting birds. These systems share the fundamental challenge of maintaining consistent identities across multiple camera views for animals that are visually similar, move rapidly, and frequently occlude one another.

Xiao et al. (2023) introduced a comprehensive system for studying behavioral dynamics of a flock of songbirds in a 3D outdoor aviary. Their work studied 15 brown-headed cowbirds housed in a large covered arena using eight synchronized cameras, combining Mask R-CNN detection with background subtraction to reliably detect both stationary and flying birds. They employed a Reconstruction-then-Tracking (RT) approach, first reconstructing dense 3D point clouds from segmentation masks across camera views using epipolar geometry and triangulation, then linking 3D points across time using a Lagrangian Particle Tracking (LPT) algorithm. Their pipeline also incorporated a ResNet50-based re-identification network trained on leg band patterns to maintain individual identities over time. Critically, their system extracted behavioral ethograms from the tracked data and demonstrated through social network analysis that pair bond context significantly affects interaction dynamics between males and females. For example, pair-bonded females were more receptive to approaches from their bonded male than from other males. This work established that fine-grained social interaction data can be extracted automatically from aviary footage, and their WILD dataset provided a challenging benchmark for multi-view multi-bird tracking.

However, their system was designed for larger cowbirds with clearly visible colored leg bands that serve as reliable re-identification markers. In contrast, although colored leg bands are present in our setup, they are only observable in the security camera footage and are not visible in the GoPro recordings, thereby restricting their utility for consistent individual identification. Furthermore, while their pipeline extracts high-level interaction patterns such as approaches and vocalizations, it does not perform automated classification of specific behavioral states, such as nest box use, hole inspection, or social interactions like lunge, peck, displacement, and copulation - the precise behaviors that are biologically meaningful in our sparrow study.

The foundational prior work for this thesis is Moradi et al. (2024), a study from the same research group at the University of Oklahoma, which developed a multi-view 3D tracking pipeline specifically for visually similar house sparrows in the same outdoor aviary used in this thesis. Their system operates on five synchronized GoPro cameras recording 80 adult house sparrows across four separate enclosures, with each enclosure housing 20 mixed-sex birds. The pipeline follows an eight-step workflow: YOLOv5 detection fine-tuned on 600 manually annotated aviary frames, binary masking via Canny edge detection, SIFT keypoint extraction from masked bird regions, Brute-Force feature matching across camera pairs, landmark-based Voronoi diagram outlier rejection, feature clustering, 3D triangulation, and Kalman filter tracking. The core technical contribution is the context-aware outlier rejection algorithm, which constructs Voronoi diagrams from manually selected aviary landmarks and validates feature matches only when the matched keypoints in both camera views share the same nearest landmark. This approach eliminated approximately 80% of incorrect matches per frame while maintaining a 97% ratio of correct final matches, significantly improving 3D reconstruction accuracy. The tracking system achieved 77.1% bird tracking success over 10-second intervals, dropping to 26.7% over 60 seconds due to accumulated identity switches, with an average of 23.4 identity switches per minute. The authors concluded their paper by explicitly identifying behavior classification as future work, stating that "future work will involve testing our algorithm in different aviary datasets as well as combining it with animal behavior analyses." This thesis directly fulfills that objective. Where Moradi et al. (2024) answered the question of where birds are in 3D space at each moment, this thesis addresses what behaviors those birds are performing, building a behavior classification pipeline on top of the established tracking infrastructure using the same dataset and camera system.

Concurrent with this thesis, the Moradi et al. (2026) system is being developed within the same research group to further improve multi-view tracking of house sparrows in the same aviary. Sparrow3D introduces sex-semantic feature matching to enhance 3D reconstruction and identity preservation, leveraging a fine-tuned YOLOv5 detector paired with a ResNet-18 sex classifier to predict whether each detected bird is male or female. Male and female sparrows exhibit subtle sexual dimorphism—males have a distinctive black bib under the chin, while females have more uniform brown plumage - and this information is used as an additional semantic prior during cross-view feature matching, filtering out matches with inconsistent sex predictions. The system also explores an auto-calibration approach that uses bird detections as dynamic calibration targets, reducing reliance on manual checkerboard-based calibration and significantly lowering calibration time. Preliminary experiments suggest that incorporating semantic and sex-guided outlier rejection can substantially improve identity preservation compared to landmark-only methods. While Sparrow3D focuses on robust 3D tracking and identity consistency, this thesis addresses the complementary problem of classifying the behaviors performed by individual birds once reliable tracks are established. Together, these efforts aim to form a unified pipeline combining accurate 3D tracking with automated behavior recognition.

2.2 Animal Behavior Recognition

While multi-view tracking addresses the problem of maintaining individual identities, the automatic classification of what behaviors those individuals are performing requires a separate body of methodology. Animal behavior recognition from video has been approached through a spectrum of techniques ranging from rule-based systems to deep learning classifiers.

Ipek et al. (2023) developed a behavior analysis pipeline, building a computer vision system to detect agonistic interactions -chasing, fleeing, and provocative physical contact between commercially group-housed breeding rabbits. Their pipeline consisted of three main stages: group-level optical flow-based action segmentation to detect periods of movement activity, individual-level detection and re-identification using a fine-tuned Keypoint R-CNN network, and a hand-crafted rule-based action classifier that analyzed the velocity profiles and spatial distances between tracked animals to determine which animal was dominant and which was submissive in each interaction. The detected interactions were accumulated into time-dependent sociomatrices from which randomized Elo ratings were computed to derive

and visualize the evolution of dominance hierarchies over the 10-day group-housing period. Compared against human expert annotations, the system achieved 77% precision and 85% recall within a 5-minute tolerance interval, demonstrating that social hierarchy dynamics can be quantified automatically from video data. This work is relevant to this thesis as it models agonistic interactions using movement-based features derived from velocity and spatial relationships between animals, highlighting the importance of motion cues in capturing social behaviors. However, their approach relies heavily on rule-based classifiers derived from movement patterns rather than learned visual features, and their subjects are much larger than sparrows with significantly higher pixel coverage in the frame.

Building on the limitations of rule-based systems, recent work has shifted toward end-to-end deep learning frameworks that jointly model detection, tracking, and behavior recognition. Ma et al. (2024) introduced AlphaChimp, an end-to-end framework for simultaneous detection, tracking, and fine-grained behavioral recognition of chimpanzees in video. The model is evaluated on the ChimpACT dataset, a longitudinal collection of 163 video clips spanning four years and capturing over 20 group-living chimpanzees annotated with 23 behavior classes across locomotion, object interaction, social interaction, and other categories. AlphaChimp adopts a DETR-based architecture with a Video Swin Transformer backbone that extracts multi-scale spatiotemporal features over short temporal windows. These features are processed through a transformer encoder-decoder that jointly predicts bounding boxes and multi-label behavior classifications in a single forward pass. On the ChimpACT benchmark, AlphaChimp achieves significant improvements over prior methods, including approximately 10% higher tracking accuracy and 20% gains in behavior recognition, particularly for complex social behaviors such as playing and nursing, which were previously difficult to detect. The authors attribute these improvements to the model’s ability to incorporate contextual information from surrounding individuals, as confirmed through gradient-based visualizations highlighting attention over interacting animals. Despite these advances, AlphaChimp operates on large subjects in relatively controlled zoo environments, whereas our setting involves small birds occupying as few as 25 pixels in height and exhibiting complete disappearance when entering nest boxes - an occlusion scenario not encountered in the ChimpACT dataset.

Rodriguez-Juan et al. (2025) introduced the Visual WetlandBirds dataset, the first fine-grained video dataset specifically designed for bird behavior detection and species classification, comprising 178 videos of 13 waterbird species annotated with seven behavior classes

including alert, feeding, flying, preening, resting, swimming, and walking. Their dataset was created using a semi-automated annotation pipeline built on CVAT, where YOLOv7 was used to automatically generate initial bounding box proposals which were then corrected and annotated with behavior labels by professional ecologists. Baseline behavior detection experiments were conducted using four video classification architectures - Video MViT, Video S3D, Video SwinTransformer, and Video ResNet, with the best performing model, Video ResNet, achieving only 0.56 accuracy on the behavior classification task. The authors attributed this low performance to the limited quantity of training data available and highlighted the need for more advanced training strategies and architectures. This result is significant for contextualizing this thesis - even with state-of-the-art video classification architectures, behavior recognition for birds from video is a genuinely hard problem with considerable room for improvement. Critically, the WetlandBirds dataset focuses on large waterbirds such as gulls, herons, and moorhens that are substantially larger and more visually distinctive than house sparrows, and their behaviors are recorded in open wetland environments with minimal occlusion. In contrast, this thesis targets sparrows in fixed-camera footage, with behaviors that involve entering enclosed nest boxes and disappearing from view entirely, a fundamentally different and more challenging scenario that existing bird behavior datasets do not address.

Zhong et al. (2024) proposed AnimalMotionCLIP, which extended the CLIP visual language model for animal behavior recognition by interleaving dense optical flow fields computed by the RAFT algorithm with color video frames at the input level, embedding fine-grained motion information directly into the CLIP visual encoder. Their system also proposed multiple temporal sampling schemes - dense, semi-dense, and sparse - finding that sparse sampling across the full video duration best captured diverse behaviors. Experiments on the large-scale Animal Kingdom dataset demonstrated that combining RGB frames with optical flow improved mean average precision from 73.93% to 74.63% compared to RGB alone, outperforming all prior state-of-the-art approaches including MSQNet. This work demonstrates that incorporating motion information through optical flow provides a complementary signal to visual appearance, as evidenced by improved performance over RGB-only models for animal behavior recognition.

2.3 Multi-Object Tracking

The tracking component of our pipeline relies on ByteTrack Zhang et al. (2022), which introduces a simple but highly effective two-stage data association strategy for multi-object tracking. Rather than discarding low-confidence detections as most prior trackers did, ByteTrack associates every detection box - including low-confidence ones, with existing tracklets through two sequential matching steps: first matching high-confidence detections to tracked objects using IoU-based similarity and Kalman filter motion prediction, then attempting to match remaining unmatched tracklets with the low-confidence detections using a more lenient threshold. This two-stage approach significantly reduces fragmented trajectories and improves recovery of objects that are temporarily occluded or partially visible, achieving 80.3 MOTA and 77.3 IDF1 on MOT17 at 30 FPS. ByteTrack’s ability to associate low-confidence detections is particularly useful in scenarios involving occlusion or temporary disappearance of objects.

Tu et al. (2024) extended ByteTrack for livestock behavioral monitoring, developing Pig-ByteTrack, a YOLOX-based detection system combined with an improved ByteTrack tracker for group-housed pigs in commercial farm settings. Their system performed multi-object tracking and behavioral time computation for individual pigs, achieving HOTA of 72.9%, MOTA of 91.7%, and IDF1 of 89.0% over 1-minute video intervals. This work is the closest methodological parallel to our pipeline, a YOLO plus ByteTrack system applied to animal behavior analysis in a fixed-camera setting and demonstrates that this architecture is effective for continuous multi-animal tracking in farm environments. However, pigs are substantially larger animals than sparrows and occupy significantly more pixels per frame, making detection substantially easier. Additionally, pig behaviors such as lying, standing, and eating are visually distinct and do not involve the animal temporarily disappearing inside a structure as nest box entry does in our sparrow study.

2.4 Temporal Behavior Classification

Beyond detection and tracking, classifying the behavior of tracked animals from sequences of cropped image windows requires a temporal model capable of capturing both visual appearance and motion context across multiple frames. Vision Transformers Dosovitskiy et al. (2021) have become the dominant backbone for visual feature extraction in recent years, demonstrating superior performance to convolutional networks on image classification tasks

through their global self-attention mechanism.

For temporal modeling of behavior sequences, prior work has explored several architectures. The WetlandBirds paper (Rodriguez-Juan et al. (2025)) evaluated Video SwinTransformer and Video MViT, both ViT-based architectures extended to the temporal domain, finding that even these powerful models struggled with the limited training data available for bird behavior classification. The pig aggression classification work of Souza et al. (2023) evaluated ViViT, TimeSformer, and STAM against CNN-based approaches including ResNet3D and CnnLSTM for detecting aggressive behavior in pigs, finding that temporal modeling significantly improved classification accuracy compared to single-frame approaches. These results highlight the importance of temporal modeling for behavior recognition, particularly for distinguishing between behaviors that unfold over multiple frames, such as sustained presence near a location versus brief transient movement.

2.5 Summary and Research Gap

The reviewed literature reveals a consistent pattern: while significant progress has been made in multi-view tracking of birds in aviaries, and while animal behavior recognition has been demonstrated for larger animals in controlled settings, no prior work has combined automated multi-camera tracking with deep learning behavior classification specifically for small passerine birds in a naturalistic aviary environment. The WetlandBirds dataset (Rodriguez-Juan et al. (2025)) represents the closest attempt at bird behavior recognition from video but targets large waterbirds with visually distinct behaviors in open environments, achieving only 0.56 accuracy with state-of-the-art models on a comparatively simple classification problem. The specific behaviors relevant to sparrow social dynamics involve partial or complete occlusion of the bird from camera view, requiring tracking logic and temporal modeling strategies that go beyond what general-purpose behavior recognition systems provide. This thesis addresses this gap by building directly on the tracking infrastructure of Moradi et al. (2024) and Moradi et al. (2026) to add automated behavior tracking, fulfilling the future work direction they identified.

2.6 Thesis Contributions

The key contributions of this work are as follows:

- Fine-tuned object detection: A YOLOv8 model fine-tuned for small bird detection in complex outdoor aviary environments.
- Multi-bird identity tracking: Integration of ByteTrack for robust identity tracking across frames, addressing challenges such as track fragmentation during sudden flight and occlusion when birds enter nest boxes.
- Temporal behavior classification: A classification model combining a Vision Transformer (ViT) backbone with causal self-attention to enforce temporal ordering, enabling accurate recognition of behaviors that occur in close physical proximity, such as at-hole and in-box.
- Behavioral repertoire coverage: Automated recognition of baseline behaviors - perching on nest boxes, approaching nest hole entrances, and occupying nest boxes - while laying the foundation for extending the pipeline to more complex social behaviors such as lunge, peck, displacement, and copulation.
- Scalable multi-enclosure system: A pipeline designed to operate across multiple enclosures simultaneously, supporting large-scale behavioral analysis that would be infeasible through manual video review.

Chapter 3

Dataset Collection and Annotation

3.1 Overview

This chapter describes the data collection setup, behavioral annotation process, and video preprocessing pipeline used to create the training and validation datasets for the automated behavior classification system. The dataset consists of video recordings from an outdoor sparrow aviary, behavioral annotations provided by expert observers from the Department of Biology at the University of Oklahoma, and frame-level bounding box and behavior annotations created using the CVAT annotation tool.

3.2 Aviary Setup and Recording Infrastructure

All video data used in this thesis was collected from an outdoor aviary facility at the University of Oklahoma, housing adult house sparrows (*Passer domesticus*) in separate enclosures. This study focuses on three enclosures within the aviary: Enclosure 4, Enclosure 9, and Enclosure 10, each housing a mixed-sex group of 20 adult sparrows. Each enclosure is equipped with five synchronized GoPro11 cameras (1920×1080 pixels, 30 FPS) positioned at various angles to provide comprehensive coverage of the enclosure volume. The cameras were synchronized using GoPro software to ensure temporal alignment across views. Each recording session produced approximately 40-minute video clips per camera. Wooden nest boxes were mounted along the walls of each enclosure, with entry holes accessible to the birds, providing the physical context for the nest box behaviors targeted in this thesis - on_box, at_hole, and in_box. Food and water were distributed throughout the enclosure, and birds were given unique combinations of colored plastic leg bands for visual identification by the biology observers. Video recordings were collected across different dates for each enclosure. Due to the large volume of recorded footage and the availability of behavioral annotations, this thesis focuses on data from Enclosure 9 and Enclosure 4, utilizing only two camera views (GoPro3 and GoPro5). Enclosure 10 was excluded as it had fewer expert behavioral annotations available at the time of this work.



(a) Camera 3



(b) Camera 5

Figure 3.1: Aviary camera setup showing two selected views (GoPro3 and GoPro5), each capturing a different perspective of Enclosure 9.

3.3 Behavioral Annotations

Behavioral annotations were provided by observers from the Department of Biology using BORIS (Behavioral Observation Research Interactive Software). Observers reviewed the video recordings and manually logged each behavioral event they identified, and the resulting annotations were exported as structured Excel spreadsheets for analysis. Each row in the spreadsheet corresponds to a single observed behavioral event and contains the following columns:

- Date - the recording date
- Camera - the camera view used for observation
- Behavior - the specific behavior label (e.g., on_box, at_hole, in_box, displacement, side-by-side, lunge, peck etc.)
- Behavioral category - the broader category the behavior belongs to (neutral, affiliative, aggressive)
- Behavior Start - absolute clock time when the behavior began (HH:MM:SS.s format)
- Behavior End Time - absolute clock time when the behavior ended (HH:MM:SS.s format)
- Behavior Duration - duration of the behavior (HH:MM:SS.s format)
- Location - the spatial location within the enclosure where the behavior occurred

- Initiating Sex - sex of the bird initiating the behavior
- Band ID (initiating) - leg band color combination of the initiating bird
- Receiving Sex - sex of the receiving bird (for social interactions)
- Band ID (receiving) - leg band color combination of the receiving bird
- Notes - any additional observer notes

A	B	C	D	E	F	G	H	I	J	K	L	M
Date	Camera	Behavior	Behavioral category	Behavior Start (mins)	Behavior End Time (mins)	Behavior Duration (secs)	Location	Initiating Sex	Band ID	Receiving Sex	Band ID	Notes
4/14/2024	GoPro3	on box	neutral	9:04:51.13	9:05:07.28	0:00:16.15	box 23	M	AURR			
4/14/2024	GoPro3	on box	neutral	9:04:52.28	9:05:17.00	0:00:24.72	box 23	F	WUBA			
4/14/2024	GoPro3	side-by-side	affiliative	9:04:52.28	9:05:07.28	0:00:15.00	box 23	F	WUBA	M	AURR	
4/14/2024	GoPro3	at hole	neutral	9:09:35.19	9:09:43.19	0:00:08.00	box 23	F	WUBA			
4/14/2024	GoPro3	on box	neutral	9:09:44.03	9:10:01.10	0:00:17.07	box 23	F	WUBA			
4/14/2024	GoPro3	at hole	neutral	9:10:02.19	9:10:10.20	0:00:08.01	box 23	F	WUBA			
4/14/2024	GoPro3	at hole	neutral	9:11:36.28	9:11:39.01	0:00:02.73	box 23	F	WUBA			
4/14/2024	GoPro3	at hole	neutral	9:12:59.13	9:13:05.14	0:00:06.01	box 23	F	WUBA			
4/14/2024	GoPro3	on box	neutral	9:13:30.12	9:13:38.16	0:00:08.04	box 23	M	AURR			
4/14/2024	GoPro3	at hole	neutral	9:13:48.29	9:14:02.21	0:00:13.92	box 23	F	WUBA			
4/14/2024	GoPro3	on box	neutral	9:13:50.03	9:14:09.28	0:00:19.25	box 23	M	AURR			
4/14/2024	GoPro3	on box	neutral	9:14:03.01	9:14:23.02	0:00:20.01	box 23	F	WUBA			
4/14/2024	GoPro3	side-by-side	affiliative	9:14:03.01	9:14:09.28	0:00:06.27	box 23	F	WUBA	M	AURR	
4/14/2024	GoPro3	at hole	neutral	9:14:10.08	9:14:21.28	0:00:11.20	box 23	M	AURR			
4/14/2024	GoPro3	in box	neutral	9:14:21.28	9:14:41.20	0:00:19.92	box 23	M	AURR			
4/14/2024	GoPro3	at hole	neutral	9:14:23.11	9:14:29.19	0:00:06.08	box 23	F	WUBA			
4/14/2024	GoPro3	in box	neutral	9:14:29.19	9:20:21.22	0:05:52.03	box 23	F	WUBA			
4/14/2024	GoPro3	side-by-side	affiliative	9:14:29.19	9:14:41.20	0:00:12.01	box 23	F	WUBA	M	AURR	inside box together
4/14/2024	GoPro3	on box	neutral	9:14:42.01	9:15:32.17	0:00:50.16	box 23	M	AURR			
4/14/2024	GoPro3	on box	neutral	9:15:43.25	9:16:21.16	0:00:37.91	box 23	M	AURR			
4/14/2024	GoPro3	on box	neutral	9:17:08.19	9:18:23.28	0:01:15.09	box 23	M	AURR			
4/14/2024	GoPro3	on box	neutral	9:19:43.11	9:20:22.19	0:00:39.08	box 23	M	AURR			
4/14/2024	GoPro3	on box	neutral	9:20:22.00	9:20:45.00	0:00:23.00	box 23	F	WUBA			
4/14/2024	GoPro3	in box	neutral	9:20:23.04	9:22:40.01	0:02:16.97	box 23	M	AURR			
4/14/2024	GoPro3	at hole	neutral	9:22:03.24	9:22:17.20	0:00:13.96	box 23	F	WUBA			
4/14/2024	GoPro3	in box	neutral	9:22:48.02	9:23:36.17	0:00:48.15	box 23	F	WUBA			
4/14/2024	GoPro3	on box	neutral	9:22:55.21	9:23:44.25	0:00:49.04	box 23	M	AURR			

Figure 3.2: Sample of the behavioral annotation file showing the column structure and sample rows of annotated behavioral events.

3.3.1 Behavior Definitions

The behavioral categories used in this thesis were defined by observers in the Department of based on established ethological criteria for house sparrow social behavior. The following descriptions combine the biological definitions provided by the research group with the visual characteristics used during CVAT annotation to assign frame-level labels.

Neutral Behaviors

- on_box - A bird flies to and lands on the top surface of a wooden nest box, remaining perched on the box exterior. In the video frames this behavior is identified by the

bird's body being clearly positioned on the flat top surface of the nest box, not oriented toward the hole entrance. This is typically a sustained behavior lasting several seconds to minutes and represents the most frequently observed behavioral state in the dataset.

- `at_hole` - A bird is positioned at the entrance hole of the nest box, with its body oriented toward the hole opening. This behavior captures the moment of investigation or inspection of the nest box entrance, and may precede entry into the box. Visually, the bird is identified by its proximity to the hole, typically within one body length with its head directed toward or partially inside the hole entrance. The `at_hole` state is relatively brief, representing the transition between approaching the box and either entering it or departing.
- `in_box` - A bird has entered the nest box interior, either with or without nesting material. This behavior also includes frames where the bird is peeking its head out from inside the box - a partial emergence where only the head is visible at the hole entrance. In the video frames, `in_box` is identified in two ways: when the bird has completely disappeared inside the box and is no longer visible to any camera, or when only the bird's head is partially visible protruding from the hole.

Social Behaviors

- Displacement (aggressive) - One bird flies toward and lands next to or directly on top of another bird, taking its place on the perch, nest box, or feeding location. The displacing bird does not follow the displaced bird after taking its position. This behavior is classified as aggressive because it involves one bird forcing another to vacate a location. Visually, displacement is identified by a rapid approach flight followed by a landing that causes the receiving bird to depart, with the initiating bird occupying the vacated spot. Because displacement events are brief and fast, often completed within one to two seconds, these clips were retained at the original 30fps to preserve sufficient temporal resolution for classification.
- Side-by-side (affiliative) - Two birds are perched within less than one bird-width apart, essentially shoulder to shoulder - on a perch, near a food dish, or on or near a nest box, maintaining this proximity for at least three seconds. This behavior is classified as affiliative as it reflects tolerance and social proximity between individuals. Visually, side-by-side is identified by two birds simultaneously present in close spatial proximity

with no displacement or aggressive interaction occurring. Like displacement, side-by-side clips were retained at 30fps.

Transition Class

- None - A bird is visible in the frame but is not performing any of the four behaviors above. This includes birds in mid-flight between nest boxes, walking along the floor or perches, or pausing briefly in locations not associated with nest box activity. The none class was introduced to prevent the model from forcing an incorrect label onto transitional frames and to improve classification accuracy at behavioral boundaries. Frames labeled as none are particularly common in clips centered on social behaviors, where the surrounding context often includes birds moving freely before or after the interaction.

Other Behavioral Categories

The Excel annotations provided by the Department of Biology contain a broader range of behavioral categories beyond those described above, reflecting the full ethogram used by the biology observers during their recordings. These include additional aggressive behaviors such as chasing and lunging, additional affiliative behaviors such as preening and copulation. However, these behaviors were observed at extremely low frequencies in the recorded footage - individually representing only a small fraction of the already limited social behavior data which itself constitutes less than 1% of total annotations. As a result, insufficient training examples were available to train a reliable classifier for these behaviors. The current behavior classification model developed in this thesis therefore does not identify these additional behavioral categories.



(a) on_box



(b) at_hole



(c) in_box



(d) side-by-side

Figure 3.3: Examples of each behavior class as seen in the aviary camera footage. (a) on_box : bird perched on top of nest box. (b) at_hole : bird positioned at hole entrance. (c) in_box : bird peeking head out of hole. (d) side-by-side : two birds perched shoulder to shoulder on nest box.

3.4 Video Clip Extraction

The Excel behavioral annotations contain absolute clock timestamps, which required alignment with the video recording start times to determine the elapsed time within each video file. This elapsed time was computed by comparing the behavior timestamp with the video start time noted during the recording, using QuickTime Player on macOS to identify the corresponding position within the video file. An initial Python script was developed to automate clip extraction by computing elapsed times from the Excel timestamps and programmatically extracting video segments. However, due to the need for careful quality filtering, removing clips where the target bird was not visible, where the camera angle was obscured, or where the behavioral event was too brief or ambiguous, the final clip extraction was performed manually using LosslessCut, a lossless video trimming tool that preserves the original video quality without re-encoding.

For each neutral behavioral event in the Excel annotations, a 30-second clip was extracted centered on the behavior, taking 15 seconds before the behavior onset and 15 seconds after the behavior end. This windowing strategy ensures that the temporal context surrounding each behavior is captured, giving the model sufficient frames to observe the transition into and out of the behavior rather than just the behavior in isolation. For example, a clip centered on an on_box event captures the bird approaching the box, perching on it, and eventually departing, providing richer temporal context than cropping exactly to the behavior timestamps. For social behaviors - displacement and side-by-side, a different clipping strategy was used to account for their fundamentally different temporal characteristics. Unlike neutral behaviors which are sustained over several seconds to minutes, social interactions are typically brief and fast, often completing within one to three seconds. Extracting a full 30-second window centered on a displacement or side-by-side event would result in a clip dominated by unrelated neutral behavior context, diluting the social interaction signal that the model needs to learn. Instead, clips for social behaviors were extracted based on the exact behavior duration recorded in the Excel annotations, with a symmetric padding of 2 seconds before the behavior onset and 2 seconds after the behavior end. This tighter windowing strategy preserves the essential visual context of the interaction, capturing the approach, the interaction itself, and the immediate aftermath, while keeping the clip length proportional to the event duration. Since social behavior clips are retained at the original 30fps rather than being downsampled to 5fps, the total frame count per clip remains sufficient for the temporal classifier despite the shorter duration. For example, a displacement event lasting 1.5 seconds with ± 2 second padding produces a 5.5-second clip containing 165 frames at 30fps, providing dense temporal coverage of the interaction.

All clips were extracted exclusively from the GoPro 3 and GoPro 5 camera recordings across both Enclosure 9 and Enclosure 4. While each enclosure is equipped with five GoPro cameras, the behavioral annotations provided by the Department of were primarily recorded with reference to the GoPro 3 and GoPro 5 views, as these two camera angles provided the clearest and most consistent visibility of the nest boxes and the birds interacting with them. The remaining three camera angles: GoPro 1, GoPro 2, and GoPro 4, did not have sufficient behavioral annotations aligned to their footage to support clip extraction for model training. Approximately 300 clips were extracted in total across Enclosure 9 and Enclosure 4.

3.5 Frame Rate Conversion

The GoPro cameras record at 30 frames per second, producing approximately 900 frames per 30-second clip. For neutral behaviors - `on_box`, `at_hole`, and `in_box` - the original 30fps footage was downsampled to 5fps prior to annotation and model training. This downsampling was chosen for two reasons. First, at 30fps consecutive frames are nearly identical for stationary or slowly moving behaviors such as a bird perching on a nest box, introducing significant temporal redundancy that increases annotation effort and training time without adding behavioral information. Second, reducing to 5fps brings the frame rate in line with the temporal resolution at which meaningful behavioral state changes occur for neutral sparrow behaviors - a bird typically remains in a single behavioral state for several seconds, making 5fps sampling sufficient to capture full behavioral transitions. At 5fps, a 30-second clip produces 150 frames, which is a manageable annotation load while still providing sufficient temporal context for the sliding window temporal classifier.

Social behaviors - displacement and side-by-side interactions were retained at the original 30fps. Unlike neutral behaviors which are sustained over several seconds, social interactions such as displacement can be brief and rapid, lasting as little as one to two seconds. Downsampling these events to 5fps risks losing critical frames that capture the moment of physical displacement or the spatial proximity between two birds that defines the side-by-side behavior. Preserving 30fps for social behaviors ensures the model receives sufficient temporal resolution to distinguish these fast, transient events.

3.6 CVAT Annotation

All video clips were annotated using CVAT (Computer Vision Annotation Tool), an open-source annotation platform that supports video annotation with bounding boxes, tracking, and custom attribute labeling. CVAT was selected for this project because of its support for track-based annotations that propagate bounding boxes across frames using interpolation, its ability to attach custom behavioral attributes to each bounding box, and its support for multiple export formats suited to the two distinct training pipelines used in this thesis. Specifically, CVAT supports export in CVAT for Video 1.1 XML format, which preserves track IDs, per-frame behavior attributes, and occluded flags and is parsed directly by the

temporal behavior classification model training pipeline. Additionally, CVAT supports export in YOLO 1.1 format, which produces per-frame bounding box annotations in the normalized coordinate format required for fine-tuning the YOLOv8 bird detector. This dual export capability allowed the same set of annotations to serve both training objectives - behavior classification and object detection, without requiring any re-annotation or format conversion, significantly reducing the overall annotation effort.

3.6.1 Annotation Schema

A standardized annotation schema was defined in CVAT before annotation began. Each bounding box annotation has the following properties:

- Label: bird - a single object class label applied to all bird detections regardless of identity
- Attribute: Behavior - a categorical attribute attached to each bounding box specifying the behavior being performed. The valid values are: `on_box`, `at_hole`, `in_box`, and `none`
- Occluded flag - a per-frame flag set to `True` when the bird is inside the nest box and not visible to the camera, corresponding to fully occluded `in_box` frames

The behavior attribute was defined as a per-frame property rather than a track-level property, allowing the behavior label to change from frame to frame as a bird transitions between states. For example, `at_hole` when bird moves toward the hole entrance, and then to `in_box` as it disappears inside. The `none` behavior class was introduced to represent frames where a bird is visible but not performing any of the three primary nest box behaviors. For example, mid-flight between nest boxes, walking along the floor, or perching away from a nest box. This class is important for preventing the model from forcing a behavior label onto transitional frames where no clear behavior applies.

3.6.2 Annotation Workflow

The annotation workflow for each video clip proceeded as follows. First, the clip was loaded into CVAT and a new track was created for the target bird using the tracking mode. CVAT's built-in tracking interpolation was used to generate initial bounding box proposals across frames by placing a bounding box on the target bird in the first frame and allowing CVAT to propagate it forward. Each subsequent frame was then manually reviewed to verify that the automatically propagated bounding box remained correctly positioned on the bird. When

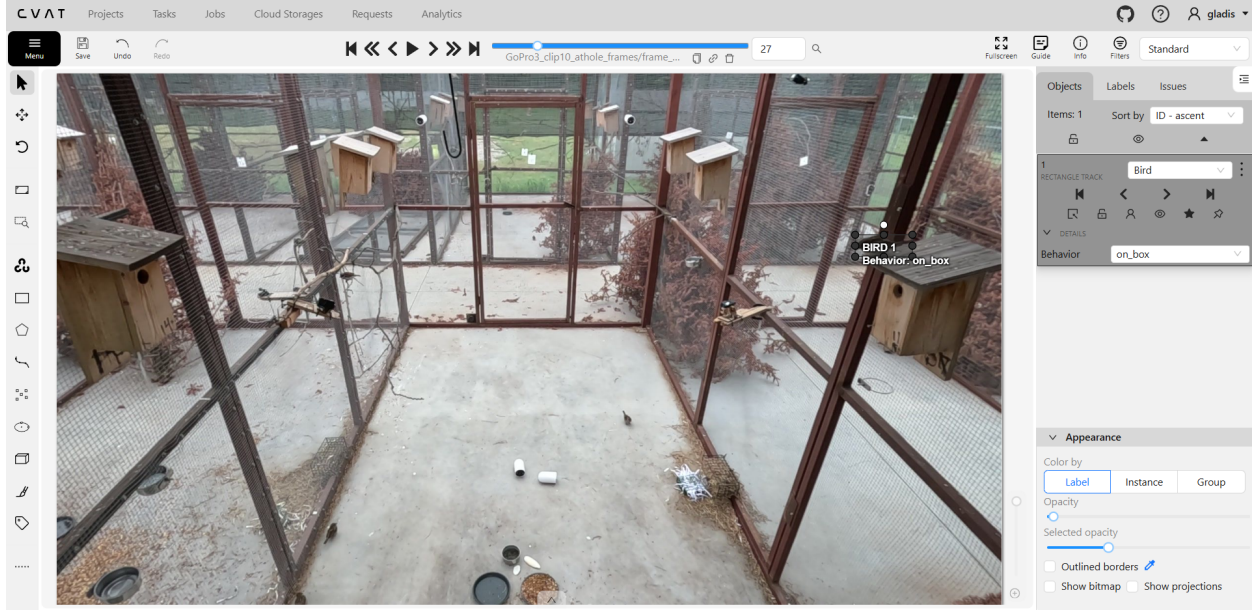


Figure 3.4: CVAT annotation interface showing a bounding box annotation for a bird exhibiting on_box behavior. The Behavior attribute is set to on_box and the track panel on the left shows the active track for the annotated bird.

the interpolated box drifted - typically during rapid flight or when the bird changed direction - the box was manually corrected at that frame and CVAT re-interpolated forward from the correction. This semi-automatic approach substantially reduced annotation time compared to purely manual frame-by-frame annotation while maintaining high spatial accuracy.

After completing bounding box placement across all frames, the behavior attribute was assigned frame by frame. For each frame, I selected the appropriate behavior value - on_box, at_hole, in_box, or none based on the bird's visible position and activity. Behavior transitions were annotated at the precise frame where the behavioral state changed, for example the frame where a bird first makes contact with the top surface of the nest box was labeled as on_box, and the frame where the bird's body crosses the threshold of the hole was labeled as in_box. For frames where the bird had entered the nest box and was no longer visible - the fully occluded in_box state, the occluded flag was set to True and the behavior attribute was set to in_box. The bounding box position was kept at the last known position near the hole entrance for these frames, providing a spatial reference for the ghost track logic in the ByteTrack pipeline. This occluded annotation ensures that the temporal model receives a signal that the bird is in an in_box state even when no visual information is available, through the is_occluded and bbox_confidence dimensions of the spatial feature vector. Social

behavior clips - displacement and side-by-side were annotated for both the initiating and receiving bird simultaneously, with each bird assigned its own track. This multi-track annotation enables the social context attention layer in the temporal model, which attends across all active bird tracks in the same frame to learn interaction-dependent behavior features.

3.6.3 Annotation Statistics

Both Enclosure 9 and Enclosure 4 clips were annotated and mixed together for training and validation. The dataset was split into training and validation sets at the clip level- entire clips were assigned to either train or validation rather than splitting individual frames to prevent data leakage between the two sets. Clips from both enclosures appear in both the training and validation splits to ensure the model is evaluated on footage from both environments.

Table 3.1: Annotation Statistics for Neutral Behavior Dataset

Category	Train	Val	Total
<i>Dataset Overview</i>			
Clips	20	7	27
Total Annotated Frames	9,840	2,128	11,968
<i>Behavior Class Distribution</i>			
on_box	7,338	1,796	9,134
at_hole	1,387	172	1,559
in_box	1,912	61	1,973
none	639	99	738
<i>Train/Val Split</i>			
Percentage	82.2%	17.8%	100%



Figure 3.5: An annotated frame showing the three neutral behaviors (1) Bird1 : at_hole (2) Bird2 : on_box (3) Bird3 : on_box

3.7 Dataset Challenges

Several challenges arose during data collection and annotation that are worth documenting as they directly influenced design decisions in the pipeline.

- **Class imbalance:** The neutral behavior classes - particularly on_box are substantially more frequent than at_hole and in_box. Social behaviors represent less than 1% of total annotations, with approximately 50 clips available for displacement and approximately 50 clips for side-by-side interactions across both enclosures. While these social behavior clips were annotated and included in the training pipeline, the extremely limited number of examples compared to the hundreds of neutral behavior clips, makes reliable classification of social behaviors a significant challenge. The scarcity of social behavior data reflects the natural rarity of these interactions in the aviary rather than a limitation of the annotation process; birds simply spend the vast majority of their time performing nest-related neutral behaviors punctuated by brief, infrequent social interactions. This extreme class imbalance across all behavior categories motivated the use of focal loss in the temporal classifier, which down-weights the contribution of

easy, well-represented classes such as `on_box` and focuses training gradient on harder, rarer examples such as `at_hole`, `in_box`, `displacement`, and `side-by-side`.

- Occlusion during `in_box` behavior. When a bird enters a nest box it becomes completely invisible to all five cameras. This produces a gap in the detection sequence that conventional trackers cannot bridge, since there is no detection to associate with the existing track. This challenge motivated the development of the ghost track logic in the ByteTrack pipeline, which maintains a virtual bounding box at the last known position near the hole entrance and propagates it forward using the CVAT occluded annotations as ground truth for when a bird should be considered in the `in_box` state.
- Behavior transition ambiguity. The transition between `on_box` and `at_hole` is visually gradual - a bird moves incrementally from the top of the box toward the hole, making it difficult to assign a precise frame-level label to the transition. The `none` class was introduced specifically to handle this mid-flight and transitional frames where forcing either `on_box` or `at_hole` would be incorrect.

3.8 Summary

This chapter described the complete data collection and annotation pipeline used to build the behavior classification dataset. Video recordings from Enclosure 9 and Enclosure 4 were processed using behavioral annotations from the Department of , with approximately 27 30-second clips extracted and annotated at the frame level in CVAT. Neutral behaviors were downsampled to 5fps to reduce temporal redundancy while social behaviors were retained at 30fps to preserve the temporal resolution needed for brief interaction events. The resulting dataset provides per-frame bounding boxes with behavior attributes and occluded flags, exported in CVAT for Video 1.1 XML format for direct ingestion by the temporal model training pipeline described in Chapter 4.

Chapter 4

System Architecture and Implementation

4.1 Overview of the Full Pipeline

The automated behavior analysis system developed in this thesis consists of two distinct phases: a training phase and an inference phase. During training, two models are trained independently : a YOLOv8 bird detector trained on frame-level bounding box annotations exported from CVAT in YOLO 1.1 format, and a temporal behavior classification model trained on per-track sliding window sequences parsed from CVAT XML annotations. During inference, the two models are applied sequentially in a four-stage pipeline: the YOLOv8 detector produces per-frame bounding boxes, ByteTrack links detections across frames into consistent tracks, and the temporal behavior model classifies each track's behavior across time. The final stage visualizes the predictions as color-coded annotated video.

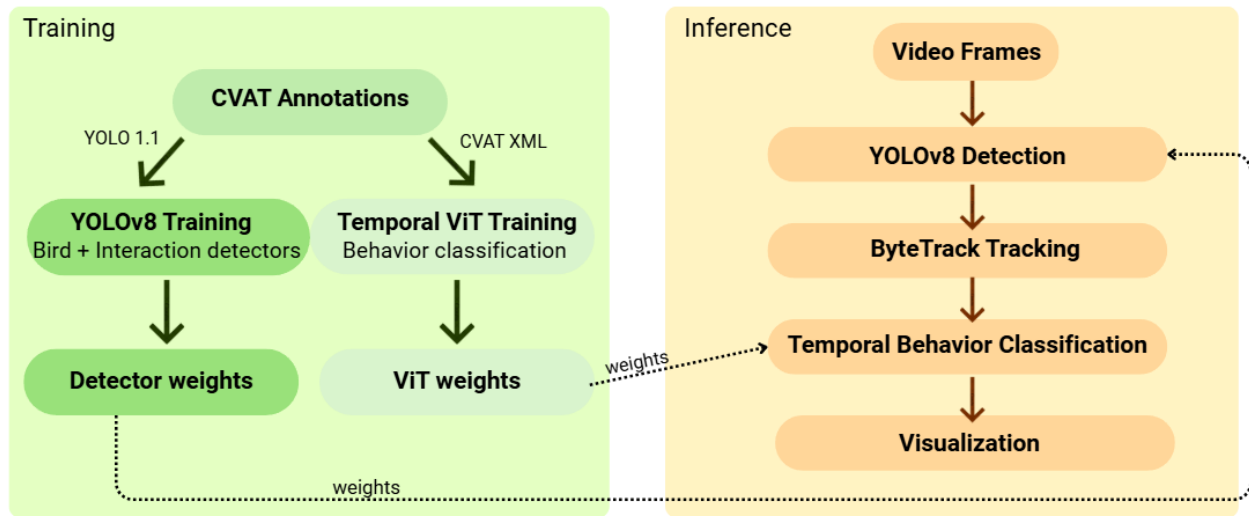


Figure 4.1: Full pipeline diagram showing two phases. Training phase & Inference phase

4.2 Training Pipeline

4.2.1 Bird Detection

4.2.1.1 Model Selection and Architecture

Bird detection is performed using YOLOv8s (small variant) from the Ultralytics framework, fine-tuned on annotated aviary frames. YOLOv8 Jocher et al. (2023) is a single-stage real-time object detector that directly predicts bounding box coordinates and class probabilities from image feature maps using a decoupled detection head and a CSPDarknet backbone with cross-stage partial connections. The model was initialized from pretrained YOLOv8s weights trained on COCO, providing a strong general visual feature representation that was then adapted to the specific domain of small sparrows in aviary conditions.

The selection of the detection architecture was an important design decision, and several candidate architectures were evaluated before settling on YOLOv8s. The parent work from our research group, Moradi et al. (2024), used YOLOv5 for bird detection in the same aviary setting, and the earlier work by Xiao et al. (2023) used Mask R-CNN combined with background subtraction for cowbird detection at UPenn. Mask R-CNN is built on the Faster R-CNN two-stage detection framework, which first generates region proposals using a Region Proposal Network and then classifies each proposed region in a second stage. While Mask R-CNN produces high-quality segmentation masks alongside detections, Moradi et al. (2024) directly compared Mask R-CNN against YOLOv5 on the same aviary footage used in this thesis and found that the base Mask R-CNN model detected only 823 bird instances compared to 1,448 for the base YOLOv5 model in the same one-minute video clip. This empirical evidence from our own dataset strongly argued against the R-CNN family of detectors for the aviary setting. Beyond detection counts, two-stage detectors such as Faster R-CNN are significantly slower than single-stage detectors at inference time, which is a practical concern when processing hundreds of frames per clip across multiple enclosures. The Region Proposal Network also requires careful anchor scale tuning to handle very small objects, sparrows occupying 25 to 120 pixels in height - adding considerable engineering overhead compared to the anchor-free YOLOv8 pipeline.

Moradi et al. (2024), used YOLOv5 for bird detection in the same aviary setting, requiring extensive iterative fine-tuning on 600 manually annotated frames to achieve reliable detection, with the base YOLOv5 model detecting only 16% of birds before fine-tuning.

This thesis upgrades the detection backbone to YOLOv8, which introduces architectural improvements over YOLOv5 including an anchor-free detection head, a redesigned neck with improved feature aggregation, and better training efficiency, all of which are beneficial for the small object detection challenges present in the aviary setting.

Within the YOLOv8 family, three model variants were considered: YOLOv8n (nano), YOLOv8s (small), and YOLOv8m (medium). YOLOv8n is the lightest and fastest variant but has the fewest parameters and shallowest feature representations, making it less capable of distinguishing small birds from visually similar background elements such as the hardware cloth mesh, wooden perches, and nest box surfaces present in the aviary footage. YOLOv8m offers the strongest feature representations among the three but introduces substantially higher computational cost per frame, which becomes prohibitive when processing hundreds of frames per clip across multiple enclosures. YOLOv8s was selected as the optimal balance between these two extremes. It provides sufficiently deep feature representations to reliably detect sparrows in challenging aviary conditions while maintaining inference speed suitable for the full pipeline. This decision is consistent with the practical constraints of the project, where detection accuracy on small, visually similar birds is the primary concern but computational efficiency is necessary to process the large volume of aviary footage.

A key design decision in detection was the input resolution. The model was trained at an image size of 832 pixels rather than the standard 640 pixels used in most YOLOv8 deployments. This higher resolution is critical for small bird detection because sparrows in the aviary footage occupy only 25 to 120 pixels in height. At standard 640-pixel input resolution, these small detections can fall below the effective receptive field of the feature pyramid network layers responsible for small object detection. Increasing the input resolution to 832 pixels preserves more spatial detail in the lower feature pyramid levels, significantly improving recall for small and partially occluded birds. This resolution increase is particularly important for the `at_hole` behavior, where only the bird’s head and upper body are visible at the nest hole entrance, often occupying as few as 15 to 20 pixels in height, making high-resolution feature maps essential for reliable detection.

4.2.2 Temporal Behavior Classification Model

4.2.2.1 Architecture Overview

The temporal behavior classification model is designed to classify the behavior of each tracked bird across a sliding window of frames, taking into account both visual appearance and motion context. The model follows a four-stage architecture: a Vision Transformer backbone extracts per-frame visual features from cropped bird images, a fusion layer combines visual features with hand-crafted spatial features, a causal temporal transformer models the sequence of fused features across time, and a social context attention layer allows each bird to attend to other birds active in the same frame. A classification head then produces per-frame behavior logits across four classes: `on_box` (0), `at_hole` (1), `in_box` (2), and `none` (3). While the Vision Transformer (ViT) backbone follows the architecture proposed by Dosovitskiy et al. (2021) and has been explored in the context of visual wetland bird recognition (Rodriguez-Juan et al. (2025)), the causal self-attention mechanism is drawn from established sequence modeling literature (Vaswani et al. (2017)), and the social context attention layer is inspired by prior work on chimpanzee behavior recognition (Ma et al. (2024)), the remaining components represent original contributions of this thesis. Specifically, the social context attention layer was adapted and modified from that setting to suit the dynamics of aviary sparrow interactions, and the fusion of spatial features with ViT-extracted visual representations, and the end-to-end integration and training of the full pipeline across three enclosures with five synchronized GoPro cameras are all contributions of this work.

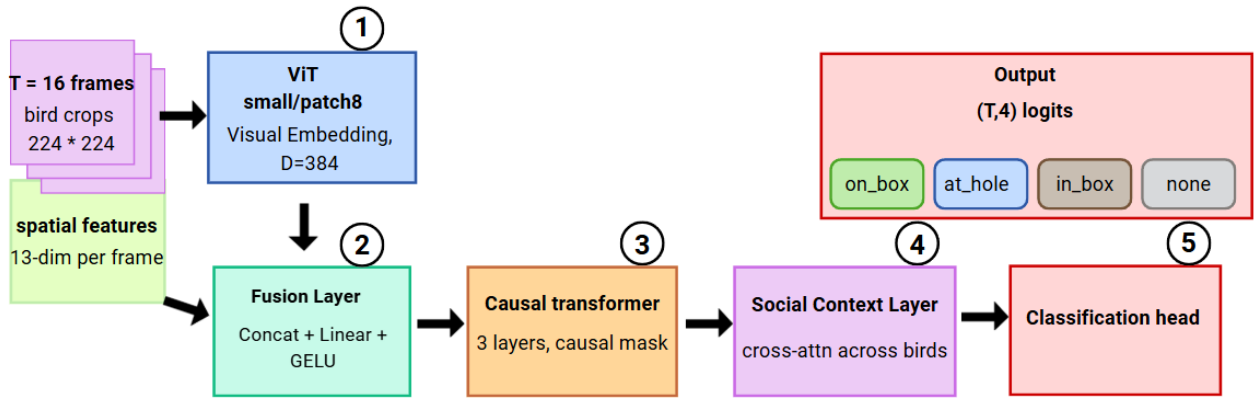


Figure 4.2: Architecture diagram of the temporal behavior classification model. Input: $T=16$ cropped bird frames + 13-dimensional spatial feature vectors. Stages: ViT-small/patch8 backbone $\rightarrow$ fusion layer $\rightarrow$ causal transformer $\rightarrow$ social context layer $\rightarrow$ classification head.

4.2.2.2 ViT small/patch8 Backbone

The visual backbone is responsible for extracting meaningful visual feature representations from cropped bird images that are then passed to the temporal classifier. The choice of backbone architecture was carefully considered, as the discriminating visual cues between behavioral states - body orientation relative to the hole entrance, head position, partial visibility of the hole edge, are subtle, spatially distributed, and require fine-grained feature extraction from small image regions.

Several backbone architectures were evaluated before settling on ViT-small/patch8. Convolutional neural network backbones -particularly ResNet He et al. (2016) variants - particularly ResNet-50 and ResNet-18, were the first candidates considered, given their widespread use in animal behavior recognition pipelines. The re-identification network in Xiao et al. (2023) used a ResNet-50 backbone for bird identity classification in the same aviary domain. ResNet architectures apply learned convolutional filters with fixed local receptive fields, building hierarchical feature representations from edges and textures in early layers to semantic object parts in deeper layers. While ResNets are computationally efficient and well-understood, their inductive bias toward local spatial patterns is a limitation for our setting, the discriminating visual features between behaviors such as `on_box` and `at_hole` are often subtle and spatially distributed across the entire cropped image, such as the bird’s body orientation relative to the hole entrance, the position of the head, and the partial visibility of the hole edge behind the bird. These spatially distributed cues are better captured by the global self-attention mechanism of the Vision Transformer, which can relate any patch in the image to any other patch regardless of spatial distance in a single attention operation.

ConvNeXt Liu et al. (2022) was also considered as a modern CNN architecture specifically redesigned to match ViT performance while retaining the convolutional structure. ConvNeXt incorporates several ViT-inspired design choices including larger kernel sizes and layer normalization, achieving competitive performance with ViT on ImageNet classification benchmarks. However, despite these improvements, ConvNeXt retains the fundamental limitation of convolutional architectures - locality of feature computation and does not provide the explicit global self-attention mechanism that makes ViT particularly well suited to relating spatially distant behavioral cues within a cropped bird image.

DeiT Touvron et al. (2021) (Data-efficient Image Transformer) was considered as a ViT variant specifically designed to train effectively on smaller datasets through knowledge distillation from a CNN teacher model during pretraining. Given the limited size of our annotated aviary dataset, DeiT’s data efficiency was an attractive property. However, DeiT was ultimately not selected because its pretraining used patch16 resolution — the same 16-pixel patch size as standard ViT-base and ViT-small models — which provides insufficient spatial granularity for fine-grained behavioral cue extraction from small bird crops. The patch8 resolution advantage of ViT-small/patch8 was judged to outweigh DeiT’s data efficiency benefits for this specific task.

The Vision Transformer Dosovitskiy et al. (2021) was ultimately selected because its self-attention mechanism is particularly well suited to the behavioral classification task. When a bird is at the hole entrance, the relevant visual cues, the hole opening, the bird’s head position, the surrounding box surface - are spatially separated within the cropped image. The ViT’s multi-head self-attention allows every patch to attend to every other patch, enabling the model to learn relationships between spatially distant regions that are jointly informative for behavior classification. This global receptive field from the very first transformer layer is a fundamental advantage over all convolutional architectures considered.

The specific choice of ViT-small/patch8_224 over ViT-small/patch16 or ViT-base/patch16 was motivated by the patch size. Standard ViT variants use a patch size of 16 pixels, dividing a 224×224 image into 196 patches. For cropped bird images where the bird occupies the majority of the 224×224 crop, a 16-pixel patch covers a substantial fraction of the bird’s body, potentially losing fine-grained detail about head orientation and body posture. The patch8 variant divides the same image into 784 patches of size 8×8 , providing four times the spatial resolution and preserving fine-grained detail about the bird’s position relative to the hole entrance. The ViT-small configuration produces a 384-dimensional embedding vector per image through global average pooling of the patch token outputs.

The pretrained ImageNet weights provide an important initialization advantage. ImageNet contains millions of natural images spanning a wide diversity of animals, textures, lighting conditions, and object poses. While ImageNet does not contain house sparrows in sufficient quantity to train a behavior classifier directly, the pretrained ViT has learned rich general visual representations - feather textures, beak shapes, body contours, and surface

materials such as wood and mesh - that transfer well to the aviary domain. These representations provide a strong initialization that significantly reduces the amount of aviary-specific training data needed to achieve reliable behavior classification. Without pretraining, training a ViT from scratch on our limited dataset would almost certainly result in severe overfitting.

To efficiently adapt the pretrained ViT to the aviary domain without overfitting, the backbone is partially frozen during training. All transformer blocks except the last two are frozen, with only the last two transformer blocks and the final layer normalization fine-tuned alongside all other model components. This strategy is grounded in the established understanding of transformer layer specialization - earlier layers capture low-level domain-agnostic features such as edges, textures, and color gradients that transfer well across visual domains, while later layers capture high-level semantic representations that benefit most from domain-specific fine-tuning. The last two blocks are fine-tuned to adapt high-level representations to the specific visual characteristics of house sparrows, their brown and gray plumage, the wooden nest box surfaces, and the hardware cloth background - across varying lighting conditions. The learning rate for the unfrozen backbone parameters is set ten times lower than the learning rate for other model components, a standard differential fine-tuning strategy to prevent catastrophic forgetting of the pretrained representations.

4.2.2.3 Spatial Features

In addition to visual features extracted by the ViT backbone, each frame is represented by a 13-dimensional hand-crafted spatial feature vector that encodes positional, motion, and social context information. This vector is constructed by the `build_spatial_feats` function and concatenated with the ViT embedding before the fusion layer. The 13 dimensions are organized as follows:

1. **cy_n** : normalized center x coordinate of the bounding box, computed as the horizontal midpoint of the bounding box divided by the image width W , giving a value in the range $[0,1]$ where 0 represents the left edge and 1 represents the right edge of the frame (dimension 0)
2. **w_n** : normalized bounding box width, computed as the box width divided by the image width (dimension 2)
3. **h_n** : normalized bounding box height, computed as the box height divided by the image height (dimension 3)

4. **area_n** : normalized bounding box area, computed as the box area divided by the total image area (dimension 4)
5. **is_occluded** : binary flag set to 1.0 when the bird is inside the nest box and not visible to the camera, corresponding to ghost track frames where ByteTrack has no active detection; set to 0.0 otherwise (dimension 5)
6. **bbox_conf** : detection confidence score from the YOLOv8 detector for the current frame; set to 0.0 for occluded ghost track frames where no YOLO detection exists (dimension 6)
7. **dx_n** : normalized frame-to-frame displacement in the horizontal direction, computed as the difference in center x between the current and previous frame divided by the image diagonal (dimension 7)
8. **dy_n** : normalized frame-to-frame displacement in the vertical direction, computed as the difference in center y between the current and previous frame divided by the image diagonal (dimension 8)
9. **speed_n** : normalized frame-to-frame Euclidean speed, computed as the Euclidean distance between the current and previous center position divided by the image diagonal (dimension 9)
10. **n_nearby_birds_n** : number of other birds whose normalized center position is within a proximity threshold of 15% of the image width, normalized by the maximum expected number of 8 birds (dimension 10)
11. **dist_nearest_n** : Euclidean distance to the nearest other bird in normalized image coordinates, divided by the image diagonal; set to 1.0 when no other birds are present in the frame (dimension 11)
12. **same_hole_zone_n** : binary flag set to 1.0 when the nearest other bird is within half the proximity threshold distance, indicating that two birds are in very close spatial competition at the same nest hole location; set to 0.0 otherwise (dimension 12)

The spatial feature vector serves multiple important functions that complement the visual features extracted by the ViT backbone. The positional dimensions `cx_n`, `cy_n`, `w_n`, `h_n`, and `area_n` provide explicit spatial grounding that the ViT crop alone cannot reliably encode - since the bird is always centered in the cropped image by design, the ViT has no direct access

to where in the frame the bird is located, which is informative for behavior classification because nest boxes are mounted at fixed positions along the enclosure walls. The `is_occluded` and `bbox_conf` dimensions together signal to the temporal model when a bird has entered a nest box and is no longer visually detectable - a state that cannot be inferred from visual features alone since the cropped image region shows only the empty hole exterior when the bird is fully inside. The `is_occluded` flag being set to 1.0 combined with `bbox_conf` dropping to 0.0 provides a strong and unambiguous signal to the temporal transformer that the current frame corresponds to the `in_box` behavioral state. The velocity dimensions `dx_n`, `dy_n`, and `speed_n` encode motion information across consecutive frames that helps distinguish between a stationary bird perching on the box top (`on_box`, near-zero speed) and a bird actively moving toward the hole entrance (transitioning to `at_hole`, non-zero directed velocity) and a bird in rapid flight (`none`, high speed). The first frame of each window has `dx_n`, `dy_n`, and `speed_n` set to zero as there is no previous frame to compute displacement from. The social context dimensions `n_nearby_birds_n`, `dist_nearest_n`, and `same_hole_zone_n` encode the presence and proximity of competing birds in the same frame, which are informative for classifying displacement and side-by-side social behaviors where the relative spatial positions of two birds are definitional to the behavior category. When no other birds are present in the frame, `dist_nearest_n` is set to its maximum value of 1.0 and `n_nearby_birds_n` and `same_hole_zone_n` are set to 0.0, providing a consistent default encoding for the single-bird case.

4.2.2.4 Fusion Layer

The visual embedding from the ViT backbone (384 dimensions) and the spatial feature vector (13 dimensions) are concatenated into a 397-dimensional vector, which is then projected back to the embedding dimension of 384 through a fusion layer. The fusion layer consists of a linear transformation followed by GELU activation and dropout with probability 0.1, implemented as:

$$\mathbf{z}_{\text{fused}} = \text{Dropout}(\text{GELU}(\mathbf{W}_{\text{fuse}} [\mathbf{v}_{\text{vit}} \parallel \mathbf{s}] + \mathbf{b}_{\text{fuse}})) \quad (4.1)$$

where $\mathbf{v}_{\text{vit}} \in \mathbb{R}^{384}$ is the ViT visual embedding, $\mathbf{s} \in \mathbb{R}^{13}$ is the spatial feature vector, $\parallel$ denotes concatenation, and $\mathbf{W}_{\text{fuse}} \in \mathbb{R}^{384 \times 397}$ is the learned projection matrix.

The design of this fusion layer reflects several deliberate architectural choices. The simplest approach to combining visual and spatial features would be direct addition, projecting

the spatial features to 384 dimensions and adding them element-wise to the ViT embedding. However, additive fusion assumes that visual and spatial features contribute equally and independently to each output dimension, which is an overly restrictive assumption for behavioral classification. The behavior-relevant information in the visual embedding and the spatial feature vector is heterogeneous - the ViT embedding encodes rich appearance information about feather texture, body posture, and the visual context of the hole entrance, while the spatial features encode positional, motion, and social context information that is entirely absent from the visual embedding. A learned linear projection over the concatenated vector allows the model to discover which combinations of visual and spatial dimensions are jointly informative for each behavior class, rather than constraining the fusion to be additive.

The choice of GELU (Gaussian Error Linear Unit) activation rather than ReLU is consistent with the rest of the ViT architecture, which also uses GELU throughout its feed-forward layers. GELU applies a smooth, probabilistic gating of the input values based on the Gaussian cumulative distribution function, producing a smoother gradient landscape compared to the hard zero-gating of ReLU. This smoothness is beneficial in a fine-tuning setting where the gradient signal from a small dataset can be noisy, as GELU’s soft gating prevents the dead neuron problem that can occur with ReLU when neurons become permanently inactive during training on limited data.

The dropout layer with probability 0.1 applied after the GELU activation serves as a regularization mechanism specific to the fusion layer. Since the fusion layer is the first learned component after the pretrained ViT backbone, it is the most exposed to the risk of overfitting to the specific visual-spatial co-occurrences present in the limited aviary training data. The dropout encourages the fusion layer to learn robust combinations of visual and spatial features that do not rely on any single dimension being present, improving generalization to unseen clips at inference time.

The output of the fusion layer, $\mathbf{z}_{\text{fused}} \in \mathbb{R}^{384}$, maintains the same embedding dimension as the ViT backbone output. This is a deliberate design choice that allows the subsequent temporal transformer to operate at a consistent dimensionality throughout the model. Maintaining a uniform embedding dimension of 384 across the ViT backbone, fusion layer, and temporal transformer avoids the need for additional projection layers between components and ensures that the positional encoding and attention weight dimensions remain consistent across the full model.

4.2.2.5 Causal Transformer

The fused embeddings across the $T=16$ frame window are processed by a three-layer causal Transformer Encoder to model the temporal dynamics of each bird’s behavioral sequence. Each encoder layer uses 6 attention heads, a feed-forward dimension of 1536 ($4 \times$ the embedding dimension of 384), and dropout of 0.1. The temporal transformer is the core reasoning component of the model. It is responsible for integrating information across the sequence of fused embeddings to produce a temporally-aware representation at each frame position that captures not just the current visual and spatial state of the bird but the entire history of its motion and appearance over the preceding frames in the window.

Why a Transformer over an LSTM or RNN?

Sequential behavioral data has historically been modeled using recurrent architectures such as Long Short-Term Memory networks (LSTMs) Hochreiter and Schmidhuber (1997) and Gated Recurrent Units (GRUs) Cho et al. (2014). These architectures process sequences step by step, maintaining a hidden state that is updated at each timestep and theoretically capable of capturing long-range temporal dependencies. However, recurrent architectures have two fundamental limitations for behavioral sequence modeling. First, they process sequences strictly sequentially, each timestep depends on the previous hidden state, making parallelization across the time dimension impossible during training and resulting in slower training convergence. Second, despite the gating mechanisms in LSTMs and GRUs, gradients still tend to diminish over long sequences, making it difficult to learn dependencies between frames that are many timesteps apart in the window. The Transformer architecture addresses both limitations through its self-attention mechanism, which computes relationships between all pairs of timesteps in a single operation regardless of their distance in the sequence, enabling full parallelization during training and direct gradient flow between any two timesteps without intermediate recurrent connections.

A simpler alternative to both recurrent and transformer architectures would be a temporal convolutional network (TCN), which applies 1D convolutions across the time dimension to capture local temporal patterns. While TCNs are efficient and effective for short-range temporal dependencies, their fixed convolutional kernel sizes limit the temporal context they can capture in a single layer. Stacking multiple layers to increase the temporal receptive field adds depth but also increases the risk of overfitting on a limited training dataset. The self-attention mechanism of the temporal transformer can directly attend to any frame in

the window from any other frame in a single layer, providing full temporal context without requiring deep stacking.

Architecture Details

The temporal transformer consists of three stacked `TransformerEncoderLayer` modules, each implementing the standard pre-norm transformer block with multi-head self-attention followed by a position-wise feed-forward network. The feed-forward dimension of 1536 is set to $4 \times$ the embedding dimension of 384, following the standard transformer scaling convention established in the original Transformer paper Vaswani et al. (2017) and adopted uniformly across ViT and BERT architectures. Three layers were selected as the depth of the temporal transformer based on the trade-off between model expressiveness and the risk of overfitting on the limited aviary training dataset, deeper stacks would provide more expressive temporal modeling but increase the number of trainable parameters beyond what the dataset size can reliably support.

The number of attention heads : 6, was chosen to divide evenly into the embedding dimension of 384, giving each head a key and value dimension of 64. This per-head dimension of 64 is consistent with the standard transformer design practice of maintaining a head dimension of 64 regardless of the total model dimension, which has been empirically shown to provide stable attention distributions across a wide range of tasks.

Causal Masking

The causal masking constraint is a critical design choice that distinguishes this temporal transformer from a standard bidirectional encoder such as BERT. The causal mask is implemented as an upper-triangular boolean matrix of shape (T,T):

$$M_{ij} = \begin{cases} \text{True} & \text{if } j > i \\ \text{False} & \text{if } j \leq i \end{cases} \quad (4.2)$$

where $M_{ij} = \text{True}$ indicates that position i is masked from attending to position j . This mask is passed to the PyTorch *TransformerEncoder* as the attention mask argument, causing the attention scores for all future positions to be set to $-\infty$ before softmax normalization, effectively preventing attention to future frames. As a result, the prediction at each time step t can only attend to positions $0, 1, \dots, t$ —the current and all preceding frames in the

window, but not to any future positions $t + 1, \dots, T - 1$.

The causal masking is essential for two reasons. First, it makes the temporal model suitable for online inference where future frames are not yet available - at inference time the model processes a sliding window of frames as they arrive, and the causal constraint ensures that predictions at earlier frames in the window are not contaminated by information from later frames that would not be available in a real-time deployment. Second, the causal constraint acts as a form of regularization during training by restricting the attention pattern to a lower-triangular structure, the number of active attention connections is halved compared to full bidirectional attention, reducing the effective capacity of the temporal transformer and mitigating overfitting on the small training dataset.

Temporal Reasoning for Behavior Classification

The temporal transformer enables the model to reason about behavioral sequences rather than classifying each frame independently, which is essential for the ambiguous behavioral transitions present in the aviary setting. To illustrate why temporal reasoning is necessary, consider the following challenging cases.

A single frame showing a bird positioned near a hole entrance is ambiguous in isolation, it could correspond to `on_box` (the bird is resting on top of the box near the hole), `at_hole` (the bird is actively investigating the hole entrance), or `none` (the bird is mid-flight passing near the nest box). With access to up to 15 preceding frames of motion and appearance history through the causal attention mechanism, the model can resolve this ambiguity by reasoning about the bird’s trajectory: if the bird has been stationary at the same position for several frames with near-zero velocity dimensions in the spatial feature vector, the temporal model learns to predict `on_box`; if the bird has been moving progressively toward the hole with directed velocity toward the hole center across preceding frames, the model predicts `at_hole`; if the bird has arrived with high speed in the immediately preceding frames indicating active flight, the model predicts `none`.

The `in_box` behavior presents an even stronger case for temporal modeling. When a bird is fully inside the nest box, ByteTrack produces a ghost track with the bounding box held at the last known position near the hole entrance. The ViT backbone crops this fixed ghost position and produces visual features of the hole entrance exterior with no bird visible, providing no

discriminative visual information for behavior classification. Without temporal context, the model would have no reliable basis for predicting `in_box` over any other class from this uninformative crop alone. However, with temporal context from the causal transformer, the model can learn that a sequence of `at_hole` frames followed by frames with `is_occluded=1.0` and `bbox_conf=0.0` in the spatial feature vector is a reliable indicator of the `in_box` state, even in the complete absence of visual information. The temporal transformer effectively learns to carry the behavioral state forward across the occluded frames, bridging the visual gap caused by nest box entry through the history encoded in the spatial feature dimensions. The transition from `in_box` back to visible - when a bird peeks its head out of the box or emerges fully, also requires temporal integration. In both cases the behavior is labeled as `in_box` in the annotation schema, so the model must learn to maintain the `in_box` prediction across both the fully occluded frames and the partially visible peeking frames that follow. The causal attention mechanism allows all transformer layers to directly attend to the preceding occluded frames and use this accumulated history to correctly sustain the `in_box` prediction rather than defaulting to none when the bird first becomes partially visible at the hole entrance.

4.2.2.6 Social Context Layer

After the temporal transformer produces per-bird temporal embeddings, a Social Context Layer is applied to enrich each bird’s representation with information about other birds active in the same frame. The layer is implemented as a multi-head self-attention module operating across the bird dimension - at each timestep t , the embeddings of all N active birds are treated as a sequence of length N , and each bird’s embedding queries the embeddings of all other birds at the same timestep as keys and values. This per-timestep attention operation is formally defined as:

$$\mathbf{z}_i^t = \text{LayerNorm} \left(\mathbf{z}_i^t + \text{Dropout} \left(\text{MHA} \left(\mathbf{z}^t, \mathbf{z}^t, \mathbf{z}^t \right)_i \right) \right) \quad (4.3)$$

where $\mathbf{z}^t \in \mathbb{R}^{N \times D}$ represents the embeddings of all N birds at time step t , MHA denotes multi-head attention with 4 heads applied over the bird dimension, and the subscript i indexes the output corresponding to the i -th bird. This operation is applied independently at each of the T time steps in the temporal window, processing one frame at a time rather than jointly across temporal and bird dimensions. Padding masks are used to prevent attention to empty bird slots when fewer than the maximum number of birds $N_{\max} = 8$ are present in

a frame, ensuring that padded positions do not contribute to the attention computation.

The motivation for the Social Context Layer is that social behaviors - displacement and side-by-side - are fundamentally relational behaviors defined by the interaction between two birds rather than the behavior of a single bird in isolation. A single bird’s temporal embedding, however rich, cannot encode whether it is performing a displacement behavior without information about the other bird being displaced. By attending across all birds present in the same frame, the layer allows each bird’s representation to be enriched with information about the positions, motion patterns, and behavioral states of its neighbors, enabling the model to learn that a bird whose embedding encodes rapid approach motion toward a nearby stationary bird is likely the initiating bird in a displacement event.

It is important to note, however, that the Social Context Layer is used during training but is bypassed during inference in the current implementation. During inference the model is called with *multi_bird_mode=False*, which skips the social attention computation entirely. This design decision was made to simplify the inference pipeline, grouping all active ByteTrack tracks per frame into a single social attention batch at inference time requires careful synchronization of track windows that adds implementation complexity. Instead, social context information at inference time is carried implicitly through spatial feature dimensions 10 through 12, *n_nearby_birds_n*, *dist_nearest_n*, and *same_hole_zone_n* which encode the proximity and count of neighboring birds at each frame and are computed from all active tracks simultaneously during the spatial feature construction step. These social spatial features are present at both training and inference time and provide the temporal transformer with a consistent social context signal regardless of whether the explicit cross-bird attention layer is active.

Enabling the Social Context Layer at inference time, by grouping all active ByteTrack tracks per frame into a multi-bird batch and running the full social attention computation, is a natural direction for future work that could improve classification accuracy for displacement and side-by-side behaviors, where the relational signal from explicit cross-bird attention is expected to be stronger than the implicit encoding through spatial feature dimensions alone.

4.2.2.7 Classification Head

The output of the social context layer passes through a LayerNorm followed by a linear projection to the number of classes, producing a $(T, 4)$ tensor of logits per bird per frame. The four output classes correspond to `on_box` (0), `at_hole` (1), `in_box` (2), and `none` (3). During inference, these logits are converted to predicted class labels via `argmax`, and a majority-vote smoothing pass with a window of 5 frames is applied to reduce frame-to-frame label flickering caused by borderline predictions at behavioral transition boundaries.

4.2.2.8 Loss Function and Optimization Design

Several design decisions were made regarding the training strategy to address the specific challenges of the aviary behavior classification problem.

Differential Learning Rates

The model uses a differential learning rate strategy that applies a lower learning rate to the unfrozen ViT backbone blocks than to the fusion layer, temporal transformer, social context layer, and classification head. This approach is standard practice when fine-tuning large pretrained vision models on small domain-specific datasets. The pretrained ViT backbone has already learned rich general visual representations from ImageNet that are valuable for the aviary classification task, applying the same learning rate to both the backbone and the task-specific components would risk overwriting these pretrained representations too rapidly with the noisy gradient signal from the limited aviary training data, a phenomenon known as catastrophic forgetting. By applying a lower learning rate to the backbone, the pretrained representations are updated gradually and conservatively while the task-specific components, which are initialized randomly and must learn entirely from the aviary data are updated more aggressively.

Focal Loss

Standard cross-entropy loss was not suitable for this task due to the severe class imbalance between `on_box` - the dominant class representing the majority of annotated frames, and the rarer classes `at_hole`, `in_box`, `displacement`, and `side-by-side`. With standard cross-entropy, the loss gradient is dominated by the abundant `on_box` examples, and the model tends to converge to a solution that performs well on `on_box` while largely ignoring the underrepresented classes. Focal loss addresses this imbalance by modulating the loss contribution of each

training example using a factor of $(1 - p_t)^\gamma$, where p_t is the model’s predicted probability for the correct class. Easy, well-classified examples, primarily the abundant *on_box* frames that the model learns quickly, receive a near-zero modulation factor and contribute minimally to the total loss. In contrast, hard misclassified examples from underrepresented classes receive a near-unity modulation factor and dominate the gradient update. The specific value of γ used during training is reported in Chapter 5.

Label Smoothing

Label smoothing is applied in combination with focal loss to prevent the model from becoming overconfident at behavioral boundaries where the ground truth label assignment is inherently ambiguous. The transition between *on_box* and *at_hole*, for example, is visually gradual - a bird moves incrementally from the box top toward the hole entrance, and the precise frame at which the behavior changes from one label to the other is a soft boundary rather than a hard discrete transition. Without label smoothing, the model is trained to assign probability 1.0 to the annotated label at every frame, including these ambiguous boundary frames, which can lead to overconfident incorrect predictions at behavioral transitions during inference. Label smoothing softens the target distribution by redistributing a small fraction of the probability mass from the correct class to all other classes, regularizing the model to maintain calibrated uncertainty at behavioral boundaries.

Data Augmentation

Data augmentation was applied to each cropped bird image during training to improve robustness to the natural variation in sparrow appearance across the dataset. The aviary footage was recorded across multiple dates and enclosures under natural lighting conditions, producing considerable variation in color temperature, shadow patterns, and contrast between recording sessions. Without augmentation, the model risks overfitting to the specific lighting and color conditions present in the training clips rather than learning lighting-invariant behavioral features. The augmentation pipeline includes color jitter, random horizontal flipping, and random grayscale conversion. The specific augmentation parameters are reported in Chapter 5.

Model Selection Metric

The best model checkpoint during training is selected based on macro F1 score across all behavior classes on the validation set, rather than overall classification accuracy. Overall

accuracy is a misleading metric in the presence of class imbalance - a model that perfectly classifies `on_box` while completely failing on `at_hole` and `in_box` would still achieve high overall accuracy due to the dominance of `on_box` in the dataset. Macro F1 computes the F1 score independently for each class and averages them with equal weight regardless of class frequency, ensuring that the saved checkpoint performs reliably across all behavior classes including the rarer ones.

4.3 Inference Pipeline

4.3.1 ByteTrack with Ghost Track Logic

We now look at the inference pipeline, as shown in Figure 4.1. During inference, bird tracking is performed using ByteTrackZhang et al. (2022), accessed through the Ultralytics `model.track()` interface. ByteTrack uses a two-stage data association strategy: in the first stage, high-confidence detections are matched to existing tracks using IoU-based similarity and Kalman filter motion prediction; in the second stage, unmatched tracks are re-attempted against low-confidence detections that would otherwise be discarded. This two-stage approach significantly reduces track fragmentation in the aviary setting where birds near hole entrances may produce low-confidence partial detections. The ByteTrack configuration used in this thesis sets `track_high_thresh=0.25`, `track_low_thresh=0.1`, `new_track_thresh=0.25`, `track_buffer=30`, `match_thresh=0.8`, and `fuse_score=True`.

In addition to standard ByteTrack, the pipeline implements custom ghost track logic to handle the `in_box` occlusion scenario - the most challenging tracking case in the aviary setting where a bird enters a nest box and produces zero detections for an extended period. The ghost track logic is implemented in the `TrackManager` class, which maintains a state object for each active track containing its position history, ghost frame counter, last known bounding box, and nearest hole ROI. When a tracked bird disappears from the detector output, the `on_missing()` method is called for that track ID at each subsequent frame. Ghost track continuation requires all of the following conditions to be satisfied simultaneously:

- Ghost frame limit: The track has been missing for fewer than `MAX_GHOST_FRAMES=30` consecutive frames. Tracks missing for more than 30 frames are dropped unconditionally.

- Proximity to hole: The last known position of the bird must be within `NEAR_HOLE_FACTOR=1.2` times the radius of a registered hole ROI. Tracks whose last position is not near any hole are dropped immediately, since a bird disappearing away from a hole is assumed to have flown out of frame rather than entering a box.
- Speed check: If the track has at least 3 position history entries, the average speed and most recent speed are computed. If the average speed is below 4.0 pixels per frame AND the most recent speed is below 4.5 pixels per frame, the track is dropped - this condition indicates a stationary bird that simply stopped being detected, likely due to a detection failure on an `on_box` bird, rather than a bird actively entering the box. Additionally, if the most recent speed exceeds 15 pixels per frame, the track is dropped - this indicates the bird flew away rapidly rather than entering the box.
- Trajectory direction check: The fraction of recent position-to-position steps that move toward the hole must meet or exceed `INWARD_THRESHOLD=0.5`, or the bird must be decelerating (slowing down toward the hole).

When all conditions are satisfied, the track is continued as a ghost track with the `occluded` flag set to `True` and confidence set to 0.0, and the bounding box is held at the last known position near the hole entrance. This ghost track row is written to the output CSV at every frame for which the bird is predicted to be inside the box, providing the temporal model with a continuous sequence of occluded frames to classify as `in_box`.

4.3.2 Hole ROI Annotation

The hole ROIs used for proximity checking are defined using a custom interactive annotation tool, developed as part of this pipeline. The tool loads a reference frame from the target video clip and allows the user to place circular ROI markers over each visible nest hole entrance by clicking on the hole center, with the scroll wheel used to adjust the ROI radius before each click. Each placed marker records the pixel coordinates of the hole center and the radius of the circular region, and the complete set of markers is saved to a JSON file.

Since the aviary uses fixed GoPro cameras mounted at static positions throughout the recording period, the pixel coordinates of each nest hole entrance remain constant across all frames of a given clip for a given camera view. A single ROI annotation session on one representative frame from a clip is therefore sufficient to define the hole positions for the

entire clip - the same json file is reused for all frames processed by the ByteTrack pipeline for that camera and enclosure combination. Different enclosures and different camera angles require separate ROI annotation sessions, since the nest boxes appear at different pixel positions in each camera view. In practice this means one file is created per camera angle per enclosure, and this file is loaded once at the start of the tracking pipeline and applied uniformly to every frame in the clip. The ROI radius parameter controls how close a bird's bounding box center must be to a hole entrance center to be considered near the hole for ghost track purposes. The default radius of 40 pixels was chosen based on the typical size of nest hole entrances in the aviary footage and the typical bounding box size of sparrows at the recording distance. The radius can be adjusted interactively using the scroll wheel in the annotation tool to account for variation in hole size and apparent size across different camera distances and angles.

Figure 4.3: The BoxROI annotation tool showing a reference frame from Enclosure 9 with circular ROI markers placed over each nest hole entrance. Each marker shows the hole center point and the circular proximity region used by the ghost track logic.

4.3.3 Weighted Box Fusion

Prior to ByteTrack association, a Weighted Box Fusion (WBF) step is applied to merge redundant detections produced by running YOLO at two confidence thresholds simultaneously - 0.25 and 0.60. Running at two thresholds produces two sets of detections per frame: a high-recall low-confidence set that captures faint or partially occluded birds, and a high-precision high-confidence set that captures clearly visible birds. WBF clusters overlapping detections from both sets using an IoU threshold of 0.30 and replaces each cluster with a single fused box whose coordinates are the confidence-weighted average of all boxes in the cluster. This fusion step reduces duplicate detections of the same bird from the two threshold passes while preserving detections of faint birds that would be missed by a single high-confidence threshold.

4.3.4 Duplicate Track Suppression

A second custom component of the tracking pipeline handles duplicate track creation, which occurs when ByteTrack assigns a new track ID to a bird that is already being tracked - typically during brief mid-frame detection failures followed by re-detection at a slightly different

position. The `on_detected()` method in `TrackManager` checks every newly assigned track ID against all existing active tracks. If the new detection is within 60 pixels of an existing track that has been active within the last 2 frames, the new track ID is suppressed and the detection is instead attributed to the existing track, updating its position and resetting its ghost frame counter.

An additional transition detection mechanism is implemented within duplicate suppression: when an existing track’s most recent speed was below 4.0 pixels per frame (stationary) and the new detection speed is above 8.0 pixels per frame (moving fast), the track’s position history is cleared. This reset prevents the temporal model from receiving a misleading motion history - for example, a long sequence of stationary `on_box` positions immediately followed by rapid flight frames, which would confuse the velocity-based spatial features and potentially cause misclassification of the behavior immediately after a behavioral transition.

4.3.5 Track Stitching

After `ByteTrack` and ghost track processing, a post-processing track stitching step is applied using a separate script to merge track fragments that were split by detection failures longer than the ghost track tolerance. The stitching algorithm examines all pairs of tracks and merges Track B into Track A when Track B starts after Track A ends, the gap between the end of Track A and the start of Track B is less than `max_gap_frames=10` frames (2 seconds at 5fps), and the spatial distance between the last position of Track A and the first position of Track B is less than `max_dist_px=200` pixels. When a merge is identified, Track B’s ID is reassigned to Track A’s ID in the output CSV, and chain merges ($B \rightarrow A$, and $C \rightarrow B$, becoming $C \rightarrow A$) are resolved before writing the final stitched output. Track stitching is applied after `ByteTrack` inference and before temporal behavior classification, ensuring the temporal model receives the longest possible continuous track sequences for each bird.

4.3.6 Behavior Inference

Behavior inference is performed by the `ByteTrackInferenceDataset`, which reads the stitched tracks CSV and constructs sliding windows of $T=16$ frames with `stride=1` for each track. For each window, the corresponding frame images are cropped to the bounding box region with a padding factor of 0.7 times the box dimensions to provide spatial context around the bird, resized to 224×224 pixels, and passed through the ImageNet normalization transform.

The 13-dimensional spatial feature vector is computed for each frame in the window using the `build_spatial_feats` function, incorporating the positions of all other active tracks at each frame as the social context input.

The model processes each window independently with `multi_bird_mode=False` during inference - the social context information is carried through the spatial feature dimensions rather than through explicit cross-bird batching, which would require grouping all active tracks per frame into a single batch. After all windows are processed, a per-frame prediction dictionary is accumulated for each track ID. The final label for each frame is determined by majority vote across all windows that include that frame, followed by a smoothing pass with a window of 5 frames that takes the majority-vote prediction across the surrounding frames. The smoothed predictions are written to a CSV file with columns `frame`, `track_id`, `behavior`, and `behavior_label`.

4.4 Visualization

The final stage of the inference pipeline renders the behavior predictions as annotated video frames using OpenCV. For each frame, bounding boxes are drawn for all tracked birds with colors indicating the predicted behavior: green for `on_box`, orange for `at_hole`, red for `in_box`, and gray for `none`. Birds predicted to be in the `in_box` state are drawn with a dashed bounding box rather than a solid one, visually distinguishing the occluded `in_box` state from visible detections. Each bounding box is labeled with the behavior name above the box and the track ID inside the box. A behavior legend is rendered in the top-right corner of each frame, a running frame counter is displayed in the top-left corner, and a bottom bar shows the cumulative count of frames attributed to each behavior class across all tracks. The annotated frames are saved as individual images and assembled into an MP4 video file using OpenCV's `VideoWriter`.

[Figure 4.3: Example annotated frame from the visualization pipeline showing multiple birds with color-coded bounding boxes. Green solid box - `on_box` bird perched on nest box top. Orange solid box - `at_hole` bird at hole entrance. Red dashed box - `in_box` bird inside nest box. Gray solid box - `none` bird in transition. Behavior legend and frame counter visible in corners.]

4.5 Summary

This chapter described the complete detection, tracking, and behavior classification pipeline developed in this thesis. The YOLOv8s detector fine-tuned at 832-pixel resolution provides reliable bird detections across the challenging aviary visual conditions. The custom ByteTrack pipeline with ghost track logic, duplicate suppression, and Weighted Box Fusion addresses the specific tracking challenges posed by in-box occlusion and brief detection failures. The temporal behavior model combining a partially-frozen ViT-small backbone, 13-dimensional spatial features, a three-layer causal transformer, and a social context cross-attention layer produces per-frame behavior classifications across four classes. Track stitching, majority-vote aggregation, and temporal smoothing post-processing further improve the stability and accuracy of the final behavior labels. Both models were trained on the OSCER A100 GPU cluster and the complete inference pipeline is fully automated from raw video frames to annotated behavior CSV and visualization video.

Chapter 5

Experiments and Results

5.1 Experimental Setup

All experiments were conducted on the OSCER (Oklahoma Supercomputing Center for Education and Research) GPU cluster at the University of Oklahoma. Both the YOLOv8s bird detector and the temporal behavior classification model were trained as separate jobs submitted to the GPU partition. The Python environment used Python 3.10.4 with GCCcore 11.3.0, PyTorch 2.6.0 with CUDA 11.8, and the Ultralytics 8.4.14 framework for YOLO training and inference. Each job was allocated one GPU, 8 CPU cores, and either 16GB or 32GB of system memory depending on the model being trained. GPU assignment on OSCER depends on node availability at submission time - training jobs in this work ran on NVIDIA RTX 6000 Ada (48GB VRAM) and NVIDIA A100 PCIe (80GB VRAM) nodes depending on cluster availability at the time of submission.

5.2 Bird Detection

5.2.1 Training Procedure

The YOLOv8s bird detector was trained in two stages using an iterative data expansion strategy. This approach was motivated by the practical challenge of building a detector that generalizes reliably across different enclosures, recording dates, and camera angles - each of which introduces variation in lighting conditions, background appearance, bird posture distributions, and apparent bird size, while managing the annotation effort required to cover this diversity.

Stage 1 : Initial Training

In the first stage, the model was initialized from the official YOLOv8s weights pretrained on COCO and fine-tuned on an initial annotated dataset consisting of clips from Enclosure 9 captured from the GoPro 3 camera angle. This single-enclosure single-camera dataset provided a controlled starting point for detector training, allowing the model to first learn

the core visual characteristics of house sparrows in the aviary setting, their size, plumage coloration, and appearance against the hardware cloth and nest box background before being exposed to greater visual diversity. Training ran for 60 epochs with a batch size of 32 and input resolution of 832 pixels on the OSCER GPU cluster.

Stage 2 : Expanded Dataset Training

In the second stage, the dataset was expanded along two independent axes of visual variation while the validation set was kept identical to Stage 1 to ensure direct comparability of results. Additional training clips were incorporated from Enclosure 9 GoPro 5 - introducing a different camera perspective of the same enclosure with variation in background appearance, apparent bird size, viewing angle, and shadow patterns, and from Enclosure 4 GoPro 3 - introducing a different enclosure with different background structure, and lighting conditions. Clips from different recording dates were also incorporated across both enclosures, adding temporal variation in natural lighting conditions. In total, Stage 2 training data covered three camera-enclosure combinations: Enclosure 9 GoPro 3, Enclosure 9 GoPro 5, and Enclosure 4 GoPro 3, compared to the single combination used in Stage 1. The model was initialized from the Stage 1 best checkpoint and trained for 100 epochs on the expanded dataset, completing in approximately 35 minutes on an NVIDIA RTX 6000 Ada GPU.

5.2.2 Training Dynamics

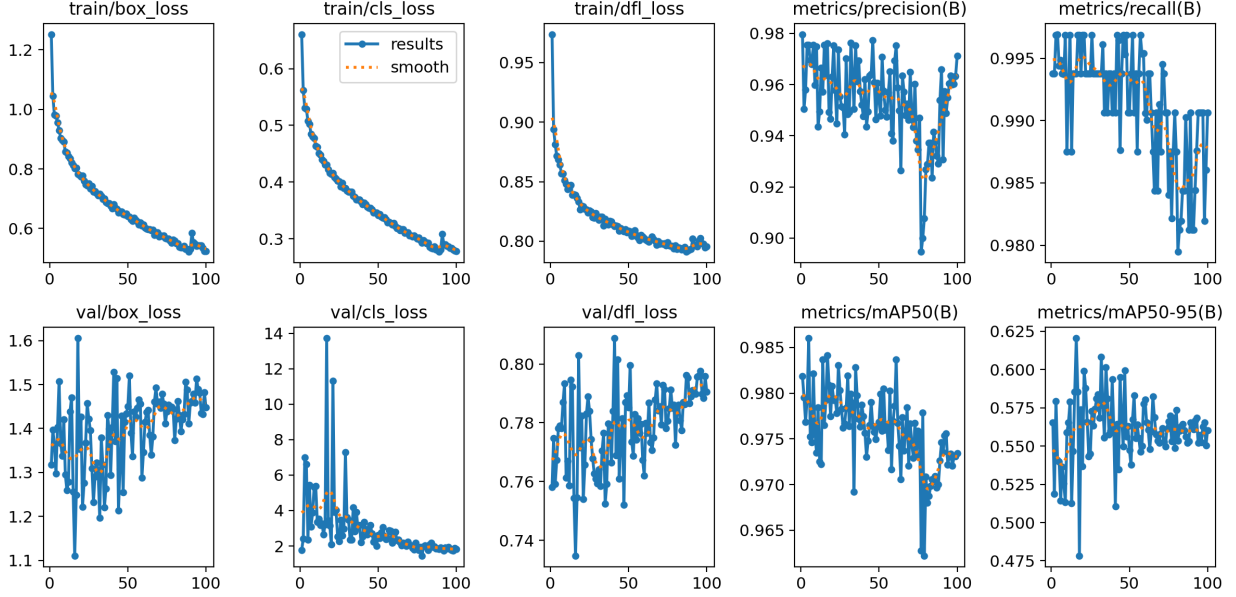


Figure 5.1: Stage 2 YOLOv8s training curves across 100 epochs. Top row shows training losses (box loss, classification loss, DFL loss) alongside validation precision and recall. Bottom row shows validation losses and detection metrics (mAP@0.5 and mAP@0.5:0.95). Training losses decrease smoothly throughout while validation mAP@0.5 stabilizes around 0.979, confirming model convergence on the expanded dataset.

The Stage 1 detector converged rapidly, achieving its best validation mAP@0.5 of 0.940 within the first few epochs of training. This rapid convergence reflects the strength of the COCO pretrained initialization - the backbone had already learned robust general bird detection features that required only a small number of aviary-specific gradient updates to adapt to the sparrow domain. After the initial convergence, the validation mAP@0.5 plateaued and showed no sustained improvement in the remaining epochs, indicating that the single-enclosure single-camera training data had been fully utilized and further training on the same data distribution would not improve generalization.

Stage 2 training, initialized from the Stage 1 best checkpoint on the expanded three-combination dataset, showed continued improvement in validation metrics - particularly in recall, reflecting the benefit of the expanded visual diversity. Training losses decreased steadily across all 100 epochs without signs of instability, and the best checkpoint was saved based on mAP@0.5 performance on the held-out validation set.

5.2.3 Validation Results

Both stages were evaluated on the same held-out validation set of 461 images containing 320 bird instances, ensuring direct comparability between Stage 1 and Stage 2 performance. Table 5.1 presents the validation metrics for both stages.

Training Stage	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Stage 1 (Encl9, GoPro3, 60 epochs)	0.987	0.893	0.940	0.603
Stage 2 (Encl(9+4), GoPro(3+5), 100 epochs)	0.964	0.997	0.979	0.620

Table 5.1: YOLOv8s bird detection validation results across two training stages. Both stages evaluated on the same held-out validation set of 461 images and 320 instances.

The results in Table 5.1 demonstrate clear and consistent improvements from Stage 1 to Stage 2 across all key metrics, with the most significant gain in recall - improving from 0.893 to 0.997. This near-perfect recall in Stage 2 indicates that the expanded dataset effectively addressed the generalization limitations of Stage 1, where the model trained on a single enclosure and camera angle missed approximately 11% of birds present in the validation frames. The recall improvement is directly attributable to the expanded training coverage by incorporating footage from Enclosure 4 and the GoPro 5 camera angle, the model was exposed to the full range of sparrow appearances, viewing angles, background conditions, and lighting variations that it encounters during inference across a larger dataset.

Precision decreased marginally from 0.987 to 0.964 - a drop of 2.3%, representing a small trade-off for the substantial recall gain. This slight precision reduction is expected when expanding training data to include more diverse visual conditions, as the model learns to detect birds in a wider range of appearances at the cost of occasionally producing detections in visually ambiguous regions. The mAP@0.5 improved from 0.940 to 0.979, reflecting the combined effect of improved recall and maintained localization accuracy across the full confidence threshold range. The mAP@0.5:0.95 improved from 0.603 to 0.620, indicating that the expanded dataset also produced modest improvements in bounding box localization precision under stricter IoU thresholds. The Stage 2 detector with recall of 0.997 and mAP@0.5 of 0.979 was selected as the final detection model for the behavior classification pipeline. The high recall is particularly important for the downstream tracking and behavior classification components - a missed detection breaks track continuity for that bird, potentially causing

the temporal model to misclassify frames with missing detections as the none behavioral state rather than the correct behavioral label.

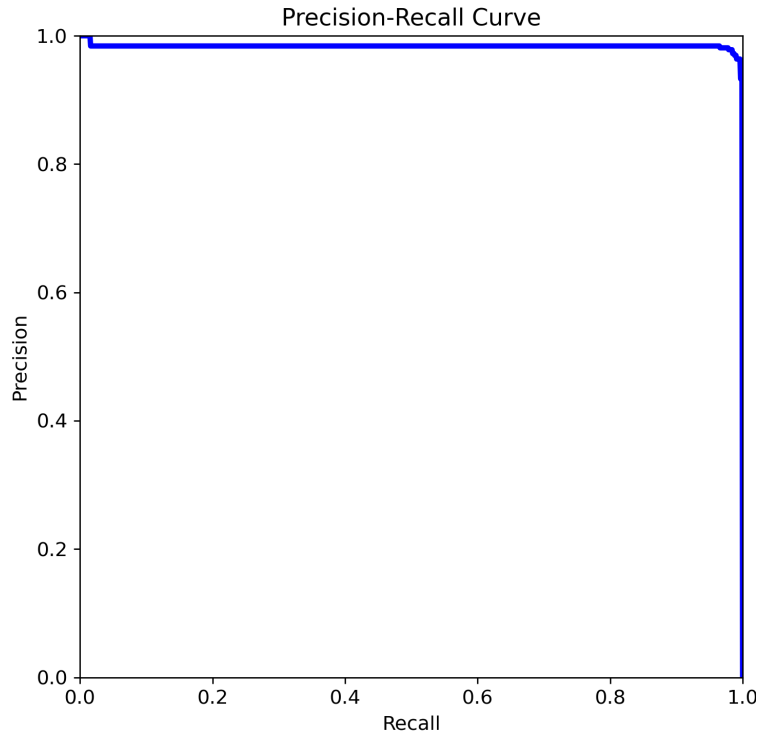


Figure 5.2: Precision-Recall curve for the final Stage 2 YOLOv8s detector on the held-out validation set.

5.2.4 Analysis of mAP@0.5 versus mAP@0.5:0.95 Gap

A notable observation in Table 5.1 is the gap between mAP@0.5 (0.979) and mAP@0.5:0.95 (0.620) in the final Stage 2 model. This gap is characteristic of small object detection. The mAP@0.5 metric counts a detection as correct if the predicted bounding box overlaps the ground truth by at least 50% IoU - a relatively lenient threshold at which the detector performs exceptionally well. The mAP@0.5:0.95 metric averages performance across ten increasingly strict IoU thresholds from 0.50 to 0.95 in steps of 0.05. At the strictest threshold of IoU=0.95, a detection is only correct if the predicted box overlaps the ground truth by 95% or more, allowing a margin of error of only a few pixels. For house sparrows occupying 25 to 40 pixels in height, a localization error of even 3 to 5 pixels constitutes a significant fraction of the bounding box dimensions, causing many otherwise correct detections to fail the stricter IoU thresholds.

This sensitivity to small localization errors is a well-documented property of small object detection. For the downstream behavior classification task in this thesis, precise sub-pixel bounding box localization is not required - the temporal model crops and resizes the bounding box region to 224×224 pixels regardless of small localization variations, and the behavioral cues visible in the crop are robust to localization errors of a few pixels. The mAP@0.5 of 0.979 is therefore the more meaningful metric for evaluating detector suitability for this pipeline.

5.2.5 Social Interaction Detection

In addition to the general bird detector, a specialized YOLOv8s model was trained specifically to detect side-by-side interaction events in the aviary footage. While the main bird detector identifies individual bird locations across all behavioral states, it does not explicitly detect the presence of a social interaction event as a whole. The interaction detector addresses this by treating each side-by-side event as a distinct object class - interaction, and detecting its spatial extent in frames extracted from social behavior clips. The interaction detector was trained exclusively on clips annotated for side-by-side behavior, using COCO pretrained weights as the starting checkpoint. Training ran for 80 epochs on the OSCER cluster on an NVIDIA A100 80GB GPU, completing in approximately 23 minutes. The model was evaluated on a held-out validation set of 228 images containing 58 interaction instances and 339 bird instances.

Class	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Interaction	0.965	0.957	0.984	0.829

Table 5.2: Interaction detector validation results for side-by-side behavior detection

The interaction detector achieves precision of 0.965 and recall of 0.957 on the held-out validation set, with mAP@0.5 of 0.984, indicating highly reliable detection of side-by-side events. The mAP@0.5:0.95 of 0.829 is substantially higher than the main bird detector’s 0.620, reflecting the fact that side-by-side interaction events produce larger and more spatially consistent bounding boxes than individual bird detections. A side-by-side interaction involves two birds in very close proximity within less than one bird-width apart — which together produce a combined bounding box substantially larger than a single bird detection, making precise localization under strict IoU thresholds more achievable than for the small

individual bird boxes.

In the inference pipeline, the ByteTrack tracks, produced by the bird detector is fed to both the temporal behavior model and the interaction detector in parallel. The temporal model processes each track's sliding window sequence to produce frame-level predictions for the four neutral and transitional behavior classes - `on_box`, `at_hole`, `in_box`, and `none`. The interaction detector processes the same tracks CSV independently and produces frame-level side-by-side labels for frames where a social interaction is detected. The final behavior output for each frame and track is a combination of both model outputs, frames labeled by the temporal model receive one of the four primary behavior labels, while frames identified by the interaction detector as containing a side-by-side event are labeled accordingly. This parallel dual-model inference architecture allows the rare side-by-side behavior to be handled by a specialized model trained specifically on interaction clips, rather than relying on the temporal model alone to classify a behavioral category that represents less than 1% of total training annotations.

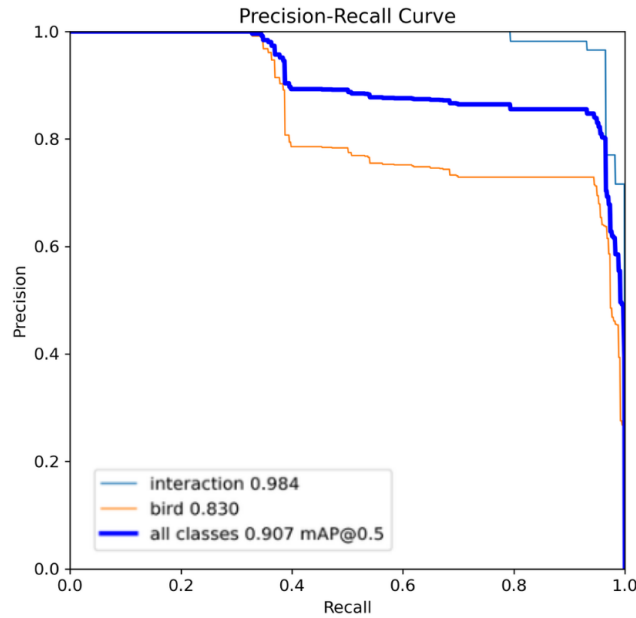


Figure 5.3: Precision-Recall curve for the interaction detector on the held-out validation set, showing high area under the curve consistent with mAP@0.5 of 0.984.

5.3 Temporal Behavior Classification

5.3.1 Training Configuration

The temporal behavior classification model was trained on the OSCER GPU cluster using the following configuration: The training dataset consisted of annotated clips from Enclosure

Hyperparameter	Value
Epochs	25
Batch size	8
Window size	16 frames
Stride (training)	10 frames
Optimizer	AdamW
Head learning rate	1e-4
Backbone learning rate	1e-5
Weight decay	1e-4
Scheduler	CosineAnnealingLR
Minimum learning rate	1e-6
Loss function	Focal Loss
Focal gamma (γ)	2.0
Label smoothing	0.1
Unfrozen ViT blocks	2
GPU	NVIDIA RTX 6000 Ada (48GB)
Allocated wall time	8 hours

Table 5.3: Temporal model training configuration

9 and Enclosure 4 across GoPro 3 and GoPro 5 camera angles, with clips split at the clip level into training and validation sets to prevent data leakage. The dataset exhibited severe class imbalance across the four behavior categories. The on_box class dominates the training data at 63.9% of frames, while the none class represents only 5.6% - an 11-fold imbalance between the most and least frequent classes. This imbalance was addressed through two complementary strategies. First, Focal Loss with $\gamma = 2.0$ was used instead of standard cross-entropy loss. Focal loss modulates the loss contribution of each training example by

$(1 - p_t)^\gamma$, where p_t is the model’s predicted probability for the correct class-down-weighting easy, well-classified examples from the dominant on_box class and focusing the gradient signal on harder examples from the underrepresented classes. Second, label smoothing of 0.1 was applied to prevent the model from becoming overconfident at behavioral boundaries where the ground truth label assignment is inherently ambiguous, particularly at the transition between on_box and at_hole, where the boundary is gradual rather than discrete.

Behavior	Training Frames	Proportion
on_box	7,338	63.9%
in_box	1,912	16.7%
at_hole	1,387	12.1%
none	639	5.6%
Total	11,276	100%

Table 5.4: Training frame distribution across behavior classes

The training stride of 10 frames with a window size of 16 produces an overlap of only 3 frames between consecutive training windows. This aggressive stride was chosen deliberately to maximize the diversity of training samples - with a smaller stride such as 2, consecutive windows would share 14 of 16 frames and be nearly identical, producing redundant samples that risk overfitting to the temporal patterns of individual clips. During inference, the stride is reduced to 1 to maximize prediction coverage, with each frame receiving predictions from up to 16 overlapping windows - the final label for each frame being determined by majority vote across all overlapping windows followed by a 5-frame smoothing pass. The best model checkpoint was selected based on macro F1 score across all four behavior classes on the validation set rather than overall accuracy. This metric choice ensures that the saved model performs reliably across all behavior classes, preventing the dominant on_box class from masking poor performance on the rarer classes.

5.3.2 Validation Results

The temporal behavior model was evaluated on the held-out validation set using per-class precision, recall, and F1 score alongside overall accuracy and macro F1.

The model achieves an overall accuracy of 0.9958 and a macro F1 of 0.9844 across all four behavior classes. These results demonstrate that the combined architecture, ViT-small/patch8 backbone, 13-dimensional spatial feature vector, three-layer causal temporal

Behavior	Precision	Recall	F1	Support
on_box	0.9983	0.9989	0.9986	1,796
at_hole	0.9773	1.0000	0.9885	172
in_box	1.0000	0.9836	0.9917	61
none	0.9789	0.9394	0.9588	99
Overall Accuracy	0.9958			
Macro F1	0.9844			

Table 5.5: Temporal behavior classification model validation results

transformer, and focal loss training, effectively handles both the visual ambiguity between behavioral states and the severe class imbalance present in the dataset.

5.3.3 Per-Class Analysis

All four behavior classes achieve F1 scores above 0.95, demonstrating consistent performance across the full behavioral taxonomy despite severe class imbalance. The on_box class achieves near-perfect performance with F1 of 0.9986, as the dominant class it provides the most training signal and produces the most visually consistent frames, with only 2 frames misclassified as none at behavioral boundaries where the bird has just landed and temporal context is limited.

The at_hole class achieves perfect recall of 1.0000 - the model never misses a genuine at_hole event, with precision of 0.9773 reflecting one in_box frame misclassified as at_hole. This specific confusion is behaviorally meaningful: when a bird peeks its head out of the nest box the visual appearance and spatial position closely resembles the at_hole state, making this the most natural confusion boundary in the behavioral taxonomy. The in_box class achieves perfect precision of 1.0000 with recall of 0.9836, with one in_box frame misclassified as at_hole. This is the mirror of the at_hole confusion above, the two states share the same spatial location near the hole entrance, and the transition between them is resolved through the is_occluded and bbox_conf spatial feature dimensions rather than visual features alone. The near-perfect performance on in_box despite being the smallest validation class (61 frames, 2.9% of support) directly demonstrates the effectiveness of the ghost track logic

and spatial feature encoding for handling nest box occlusion.

The none class achieves the lowest F1 of 0.9588, with 3 frames misclassified as on_box and 3 as at_hole. This is expected - none covers inherently ambiguous transitional frames including mid-flight and inter-behavioral pauses that are visually similar to the early stages of on_box and at_hole. Importantly, all errors across all classes occur only at adjacent behavioral boundaries - no misclassification occurs between behaviorally distant classes such as on_box and in_box, confirming that the model has learned semantically meaningful behavioral representations.

5.3.4 Confusion Matrix Analysis

The confusion matrix confirms the per-class analysis above and reveals the specific error patterns of the model.

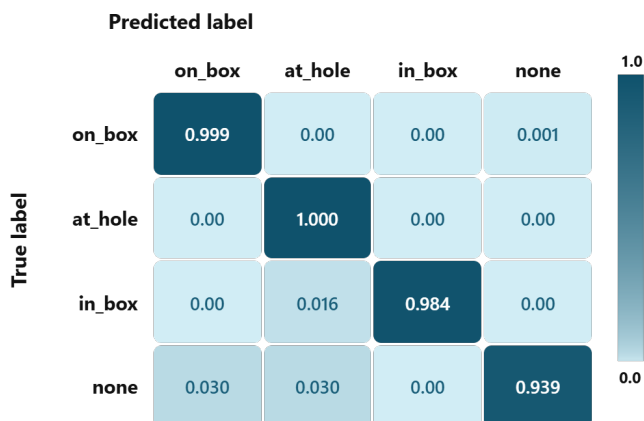


Figure 5.4: Confusion matrix for temporal behavior classification. Rows = true labels, Columns = predicted labels.

Some key observations are notable from the confusion matrix. Every misclassification involves adjacent behaviors that share spatial proximity or visual similarity, confirming that the model has learned semantically meaningful behavioral representations rather than superficial visual patterns. Second, the at_hole class is perfectly recalled with zero false negatives - every genuine at_hole frame is correctly identified. Third, the in_box class has zero false positives, the model never incorrectly assigns the in_box label to a bird that is not actually inside the box, which is the most critical constraint for the biology research since false positive in_box detections would incorrectly inflate nest box usage statistics.

5.3.5 Effect of Focal Loss on Class Imbalance

A key question in the experimental design was whether focal loss alone, without explicit class weights would be sufficient to address the severe 11-fold class imbalance between on_box and none. The results demonstrate that it was. Despite on_box constituting 63.9% of training frames and none constituting only 5.6%, the model achieves F1 scores of 0.9986 and 0.9588 respectively, a difference of only 3.98 % between the most and least frequent classes. For comparison, a naive classifier that always predicts on_box would achieve an overall accuracy of 84.4%, reflecting the dominance of on_box in the validation set - but would produce F1 scores of 0.0 for all three minority classes. The focal loss formulation with $\gamma = 2.0$ successfully redirected the gradient signal toward the harder minority class examples during training, producing a well-calibrated classifier across all four behavioral states without requiring explicit inverse-frequency class weights.

5.3.6 Qualitative Results

The visualization pipeline produces color-coded annotated frames that provide an intuitive representation of the predicted behavioral state for each tracked bird across all four behavioral classes, as illustrated in Figures 5.4 and 5.5. Green bounding boxes indicate the on_box class, blue boxes indicate at_hole, brown dashed boxes indicate in_box, and gray boxes indicate the none class. The dashed box style for the in_box class visually distinguishes ghost tracks from genuine YOLO detections, correctly indicating to a human reviewer that the bird is predicted to be inside the nest box at a position near the hole entrance where no active detection is present. Each bird is assigned a persistent unique track identifier to maintain individual identity across frames.

The side-by-side interaction detection pipeline produces a complementary visualization layer, shown in Figure 5.7. A magenta bounding box is drawn around the interacting bird pair once the interaction is confirmed, appearing only after the 90-frame temporal persistence rule is satisfied, corresponding to a minimum duration of three seconds at 30 fps. Candidate interactions that have not yet reached the confirmation threshold are rendered in pink, providing transparency into the state of the detection process at any given frame. The confirmed interaction region is labelled side-by-side to distinguish it from the individual bird tracks, which continue to display their neutral behavior predictions and track identifiers simultaneously. This combined visualization enables a human reviewer to verify both the

spatial extent of the interaction and the identity of the participating birds within a single frame.



Figure 5.5: Sample annotated frames from the visualization layer with behavior predictions across three behavioral classes: on_box, at_hole, none



Figure 5.6: Sample annotated frames from the visualization layer with behavior predictions across 3 behavioral classes: in_box, on_box, none



Figure 5.7: Sample annotated frames from the visualization layer with side-by-side interaction detection. A combined interaction bounding box (magenta) is drawn around the interacting pair once the bout is confirmed over a minimum of 90 consecutive frames at 30 fps.

5.4 Discussion

5.4.1 System Performance in Context

The complete pipeline, YOLOv8s detection, ByteTrack tracking with ghost track logic, and temporal behavior classification - achieves strong performance across all components, with the final temporal model reaching macro F1 of 0.9844 and the bird detector achieving recall of 0.997. The performance of this system should be interpreted in the context of the specific challenges of the aviary setting rather than as a direct comparison against existing bird behavior recognition systems, which target fundamentally different species, environments, and task formulations. The closest related work in bird behavior recognition Rodriguez-Juan et al. (2025) on the WetlandBirds dataset, reports a best accuracy of 0.56 using Video ResNet on 13 large waterbird species in open wetland environments with visually distinct behavior classes. While a direct numerical comparison is not meaningful given the differences in species scale, behavior taxonomy, dataset size, and task formulation, this result contextualizes the general difficulty of automated bird behavior classification and highlights that even state-of-the-art video classification backbones struggle with this problem domain. The present system addresses a more constrained but substantially harder sub-problem, classifying four spatially similar behavioral states for sparrow-scale birds in fixed-camera footage with a

challenging in-box occlusion scenario, and achieves 0.9958 overall accuracy through domain-specific architectural design rather than generic video classification approaches. The ghost track logic, spatial feature encoding, and causal temporal modeling each address specific challenges of the aviary setting that general-purpose video classification architectures do not account for.

5.4.2 Key Design Decisions

Three design decisions proved particularly impactful for the final system performance. First, the ghost track logic in ByteTrack, maintaining a virtual bounding box at the last known position near the hole entrance during nest box occlusion - was essential for achieving near-perfect in_box classification. Without it, the temporal model would receive no spatial signal during in_box frames and would have no basis for predicting box occupancy. Second, the 13-dimensional spatial feature vector, particularly the `is_occluded` and `bbox_conf` dimensions provided an unambiguous in_box signal that the temporal transformer could learn to associate with nest box occupancy independently of visual features. Third, focal loss with $\gamma = 2.0$ proved sufficient to handle the severe 11-fold class imbalance between on_box and none without requiring explicit class weights, simplifying the training procedure while achieving well-calibrated performance across all classes.

5.4.3 Implications for Biology Research

The results have direct implications for the sparrow social behavior study that motivated this thesis. The zero false positive rate for in_box classification ensures that nest box occupancy statistics derived from the automated pipeline are reliable - a false positive in_box detection would incorrectly inflate box usage counts and potentially distort analyses of nest site competition and breeding behavior. The perfect recall for at_hole ensures that every nest hole investigation event is captured, supporting complete ethogram generation without manual review. The high recall of the bird detector means that the vast majority of birds interacting with nest boxes are reliably tracked throughout the recording.

However, it is important to note that the current pipeline provides behavior-centric tracking rather than full population-level tracking. The system tracks birds that are performing or transitioning between the defined behavioral states - on_box, at_hole, in_box, and none - in the vicinity of nest boxes. Birds elsewhere in the enclosure that are not interacting with

nest boxes - for example birds feeding at food bowls, perching on non-box structures, or flying freely between distant locations - are not consistently tracked by the pipeline. This means the system is well suited for studying nest box-related behaviors and interactions but does not yet provide a complete behavioral record for all 20 birds per enclosure simultaneously, which would require full-enclosure tracking coverage beyond what the current behavior-focused annotation and training data support.

5.4.4 Limitations and Future Work

Several limitations of the current system were identified during development and evaluation.

- **Partial population coverage** - As described above, the current pipeline tracks birds primarily in the context of nest box interactions. Extending behavior classification to the full enclosure, covering all birds across all locations and behaviors simultaneously would require broader annotation coverage and potentially a multi-stage detection approach that first identifies all birds in the frame regardless of their proximity to nest boxes, then classifies their behaviors across the full enclosure space.
- **Track identity discontinuity.** The current tracking pipeline does not include a re-identification component. When a bird flies out of the camera frame and re-enters, either from a different direction or after a gap, ByteTrack assigns a new track ID to the re-appearing bird rather than recovering its original identity. This means the same physical bird may accumulate multiple track IDs across a recording session, leading to fragmented behavioral histories that cannot be automatically attributed to a single individual. The track stitching post-processor addresses short gaps within a clip, but cross-clip and long-gap identity recovery requires a dedicated re-identification module. Xiao et al. (2023) and Moradi et al. (2024) both identified re-identification as a key challenge in aviary bird tracking. This limitation is actively being addressed in parallel work within our research group, the 3D multi-view tracking system developed in Moradi et al. (2024) is currently being refined to handle track identity recovery across re-appearances, which when integrated with the behavior classification pipeline developed in this thesis will provide a complete solution for persistent individual-level behavioral analysis across the full recording session.
- **Single camera angle per clip.** The current inference pipeline processes one camera angle at a time. A bird that moves out of the GoPro 3 field of view may simultaneously

be visible in GoPro 5, but the single-camera pipeline cannot exploit this multi-view information to maintain track continuity. Integrating the multi-view 3D tracking approach from Moradi et al. (2024) with the behavior classification pipeline developed in this thesis would address both the re-identification and track continuity limitations simultaneously, providing a fully integrated multi-view behavior analysis system for the aviary dataset.

- **Limited enclosure coverage.** The pipeline currently covers Enclosure 9 and Enclosure 4 only - extending to Enclosure 10 requires additional annotation but is straightforward given the iterative training framework established here. The same two-stage detector training strategy used in this thesis can be applied directly to Enclosure 10 footage once behavioral annotations from the Department of Biology become available for that enclosure.
- **Social behavior coverage.** The current pipeline detects and classifies side-by-side affiliative behavior through the specialized interaction detector, and this work is actively being extended to cover a broader range of social behaviors. Aggressive behaviors, including lunging, chasing, biting, and pecking, and additional affiliative behaviors such as copulation are present in the aviary recordings and are of direct biological interest to the Department of Biology. However, extending the pipeline to these behaviors is currently limited by the availability of annotated training data. These social interactions are rare and brief by nature, occurring infrequently enough that the existing recordings do not provide sufficient annotated examples to train reliable detectors and classifiers for each behavior category. Targeted recording sessions specifically designed to capture higher frequencies of these rare interactions - combined with continued annotation effort from the biology observers are needed to build the training datasets required for extending the pipeline to the full sparrow social behavior ethogram.

Chapter 6

Conclusion

6.1 Summary

This thesis presented an automated computer vision pipeline for detecting, tracking, and classifying the behaviors of house sparrows (*Passer domesticus*) in a fixed-camera outdoor aviary setting. The work was motivated by the limitations of manual behavioral observation, which is labor-intensive, captures only a small subset of behavioral events for a few individuals at a time, and cannot scale to the population-level continuous monitoring required for studying complex sparrow social dynamics. The pipeline developed in this thesis addresses these limitations by providing a fully automated system capable of processing aviary video footage and producing per-frame, per-bird behavior labels for all birds interacting with nest boxes across an entire recording session.

The system was built in three main stages. First, a YOLOv8s bird detector was fine-tuned through an iterative two-stage training process on annotated aviary footage from Enclosure 9 and Enclosure 4, achieving a final recall of 0.997 and mAP@0.5 of 0.979 on the held-out validation set. The two-stage training strategy, beginning with a single enclosure and camera angle before expanding to multiple enclosures and camera angles - proved effective for building a detector that generalizes across the visual diversity of the aviary environment. A specialized interaction detector was additionally trained to identify side-by-side affiliative behavior events, achieving mAP@0.5 of 0.984 on held-out interaction clips. Second, a custom ByteTrack tracking pipeline was developed with several aviary-specific extensions - ghost track logic for maintaining track continuity during nest box occlusion, duplicate track suppression to prevent fragmented identities during brief detection failures, nest hole ROI configuration for proximity-based occlusion detection, and a track stitching post-processor for merging fragmented tracks across short gaps. These extensions directly address the unique tracking challenges of the aviary setting that standard ByteTrack cannot handle out of the box.

Third, a temporal behavior classification model was developed combining a partially-frozen ViT-small/patch8 visual backbone, a 13-dimensional spatial feature vector encoding positional, motion, and social context information, a three-layer causal temporal transformer for modeling behavioral sequences, and a social context attention layer for cross-bird interaction modeling. Trained with focal loss to address severe class imbalance, the model achieves overall accuracy of 0.9958 and macro F1 of 0.9844 across four behavior classes - on_box, at_hole, in_box, and none with F1 scores above 0.98 for the three biologically critical nest box behaviors.

6.2 Contributions

This thesis makes the following contributions to the automated analysis of animal behavior in naturalistic aviary settings. The primary contribution is the first automated behavior classification pipeline specifically designed for small passerine birds in a fixed-camera aviary, directly extending the 3D tracking infrastructure of Moradi et al. (2024) from localization to behavioral understanding. Where prior work established where birds are, this thesis establishes what they are doing.

A novel ghost track mechanism is introduced to handle the in-box occlusion challenge, a problem unique to nest box monitoring where birds disappear entirely from camera view, enabling reliable classification of nest box occupancy without visual information. This mechanism, combined with a 13-dimensional spatial feature vector that explicitly encodes occlusion state, motion, and social context, represents a domain-specific solution to a tracking and classification challenge that generic computer vision systems do not address. The thesis additionally contributes a frame-level annotated dataset of sparrow nest box behaviors across two enclosures and two camera angles, exported in formats compatible with both object detection and temporal classification training pipelines, providing a reusable resource for future aviary behavior research. Finally, a specialized interaction detector for side-by-side affiliative behavior extends the pipeline to social behavior classification, demonstrating that rare behaviors representing less than 1% of training data can be reliably detected through targeted model specialization.

6.3 Broader Impact

The automated behavior analysis pipeline developed in this thesis has direct implications for the sparrow social behavior research program at the University of Oklahoma. By replacing manual behavioral observation with an automated system capable of processing full 40-minute recording sessions across multiple enclosures, the pipeline enables behavioral analyses at a scale and temporal resolution that were previously infeasible. Behaviors that a human observer might miss during a brief or distant interaction - a fleeting displacement event, a momentary side-by-side proximity, are captured frame by frame across all birds simultaneously. The reliable detection of nest box occupancy with zero false positives ensures that nest box usage statistics derived from the automated pipeline can be directly incorporated into biological analyses of breeding behavior, social hierarchy, and nest site competition without requiring manual validation of every predicted event.

More broadly, the methodological contributions of this thesis - the ghost track mechanism for occlusion handling, the spatial feature vector design, and the iterative two-stage detector training strategy are applicable to other fixed-camera animal behavior monitoring scenarios beyond the sparrow aviary. Any setting where animals interact with fixed structures and periodically disappear from view - nest monitoring, burrow entry detection, den occupancy tracking, could benefit from the ghost track and spatial feature approaches developed here. The open-source codebase developed for this thesis provides a reusable foundation for future behavior classification work on the aviary dataset and similar fixed-camera animal monitoring systems.

Reference List

- Cho, K., B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, 2014: Learning phrase representations using RNN encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, <https://doi.org/10.3115/v1/D14-1179>.
- Dosovitskiy, A., and Coauthors, 2021: An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*.
- He, K., X. Zhang, S. Ren, and J. Sun, 2016: Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Hochreiter, S., and J. Schmidhuber, 1997: Long short-term memory. *Neural Computation*, **9** (8), 1735–1780.
- Ipek, N., L. G. W. Van Damme, F. A. M. Tuytens, and J. Verwaeren, 2023: Quantifying agonistic interactions between group-housed animals to derive social hierarchies using computer vision: a case study with commercially group-housed rabbits. *Scientific Reports*, **13**, 14138, <https://doi.org/10.1038/s41598-023-41104-6>.
- Jocher, G., A. Chaurasia, and J. Qiu, 2023: Ultralytics YOLOv8. URL <https://github.com/ultralytics/ultralytics>, accessed: September 7, 2025.
- Kubitza, R. J., J. Suhonen, and T. Vuorisalo, 2015: Effects of experimental perturbation of group structure on hierarchy formation and behaviour in House Sparrows. *Ornis Fennica*, **92**, 157–171.
- Liu, Z., H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, 2022: A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Ma, X., and Coauthors, 2024: Alphachimp: Tracking and behavior recognition of chimpanzees. *arXiv preprint arXiv:2410.17136*.
- Moradi, K., E. Haque, J. Kaur, A. B. Bentz, E. S. Bridge, and G. Habibi, 2024: Context-aware outlier rejection for robust multi-view 3d tracking of similar small birds in an outdoor aviary. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, IEEE.
- Moradi, K., E. Haque, J. Kaur, A. B. Bentz, E. S. Bridge, and G. Habibi, 2026: Sparrow3d: Sex-semantic feature matching for 3d reconstruction and tracking of visually similar house sparrows in an outdoor aviary. Work in progress, University of Oklahoma.

- Rodriguez-Juan, J., D. Ortiz-Perez, M. Benavent-Lledo, D. Mulero-Perez, P. Ruiz-Ponce, A. Orihuela-Torres, J. Garcia-Rodriguez, and E. Sebastian-Gonzalez, 2025: Visual wetlandbirds dataset: Bird species identification and behavior recognition in videos. *arXiv preprint arXiv:2501.08931*.
- Souza, J. S., E. Bedin, G. T. H. Higa, N. Loebens, and H. Pistori, 2023: Pig aggression classification using cnn, transformers and recurrent networks. *arXiv preprint arXiv:2403.08528*.
- Touvron, H., M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, 2021: Training data-efficient image transformers & distillation through attention. *Proceedings of the 38th International Conference on Machine Learning (ICML)*, PMLR, Proceedings of Machine Learning Research, Vol. 139, 10 347–10 357, URL <https://proceedings.mlr.press/v139/touvron21a.html>.
- Tu, S., H. Ou, L. Mao, J. Du, Y. Cao, and W. Chen, 2024: Behavior tracking and analyses of group-housed pigs based on improved bytetrack. *Animals*, **14** (22), 3299, <https://doi.org/10.3390/ani14223299>.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, 2017: Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*.
- Xiao, S., Y. Wang, A. Perkes, B. Pfrommer, M. Schmidt, K. Daniilidis, and M. Badger, 2023: Multi-view tracking, re-id, and social network analysis of a flock of visually similar birds in an outdoor aviary. *International Journal of Computer Vision*, **131**, 1532–1549.
- Zhang, Y., P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang, 2022: Byte-track: Multi-object tracking by associating every detection box. *European Conference on Computer Vision (ECCV)*.
- Zhong, E., C. R. del Blanco, D. Berjon, F. Jaureguizar, and N. Garcia, 2024: Animal-motionclip: Embedding motion in clip for animal behavior analysis. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA.