Scaling Up Social Behavior Studies: Real-Time, Large-Scale and Prolonged Social Behavior Analysis with PrecisionTrack

Vincent Coulombe[1,3,4], Sadegh Monfared[1,3,4], Khadijeh Aghel, Quentin Leboulleux[1], Modesto R. Peralta III[1], Benoit Gosselin[3,4], Benoit Labonté[1,2] ✉

1. CERVO Brain Research Centre, Université Laval, Québec, QC, Canada
2. Department of Psychiatry and Neuroscience, Faculty of Medicine, Université Laval, Québec, QC, Canada
3. Smart Biomedical Microsystems Laboratory, Quebec City, Canada
4. Department of electronics, Faculty of Science and Engineering, Université Laval, Québec, QC, Canada

* These authors contributed equally.

✉ Corresponding author: benoit.labonte@fmed.ulaval.ca

Abstract

Large-scale ethological behavioral studies can provide novel insights into the neuronal processes underlying complex and social behaviors, potentially opening new mental health research avenues. However, conducting such studies remains a tedious and time-consuming process, often requiring manual annotation of predefined ethological features. Although researchers are aware that current protocols constrain the potential scope of their experiments, they have limited alternatives. Despite recent advancements, automatically monitoring complex behaviors in socially interacting animals remains difficult. Even gold-standard tools struggle to reliably track more than a few interacting animals, forcing researchers to conduct short, small-scale social experiments in confined environments. Here, we present *PrecisionTrack*, designed to remain accurate and reliable, even when tracking large groups of interacting animals over prolonged periods. *PrecisionTrack* outperforms both *DeepLabCut* and *SLEAP* in crowded social settings, maintains its tracking performance over time, and supports real-time, online responsiveness. Researchers can therefore use it to gather ethological data on the fly during ongoing experiments. We also introduce the Multi-animal Action Recognition Transformer (*MART*), which can be seamlessly integrated with *PrecisionTrack* to extract behavioral insights beyond spatiotemporal localization, including individual actions and social interactions. The entire system is open source and accessible through high-level APIs. Users can follow step-by-step guides to configure, train, test, and deploy their own social behavior monitoring systems.

Introduction

Behavioral studies provide fundamental insights into the neuronal processes that underline complex constructs such as cognition, emotion, and social interactions. Behavioral analyses are essential not only for understanding the strategies that individuals from different species develop to cope with environmental challenges, but also for identifying how these strategies become maladaptive in models of psychiatric disorders. Traditionally, researchers have relied on manual annotation of predefined ethological features [1, 2, 3]. Although this approach has advantages, it becomes impractical in more complex contexts, particularly when analyzing social interactions or group dynamics. Manual scoring is time-consuming, subject to observer bias, and ill-suited to large-scale or long-term studies involving multiple individuals [4].

The development of computer vision–based artificial intelligence has revolutionized behavioral neuroscience. Tools such as *SLEAP* [5], *DeepLabCut* [6], and others [7, 8] use supervised machine learning to track multiple body parts with high spatial and temporal resolution [5, 6]. These pose-estimation frameworks provide coordinate data for specific anatomical landmarks, allowing researchers to monitor an animal's posture and movement throughout an experiment [5, 6]. Nevertheless, despite these advantages, several limitations restrict their broader implementation in complex behavioral studies.

First, they often fail to maintain persistent individual identities over time, particularly when tracking visually similar animals [9]. Although this issue is negligible in single-animal studies, it becomes a critical limitation in experiments aiming at tracking social behaviors within groups. Second, popular tools either rely on slow, two-stage detection algorithms [7] or are specifically designed for offline analysis [6], making them unsuitable for prolonged analysis regardless of their accuracy. Therefore, these limitations need to be addressed before considering applying AI-based approaches to large scale behavioral analyses. Another critical aspect of behavioral analysis is the automated recognition of specific actions. While traditionally achieved through manual coding, recent developments in supervised and

unsupervised learning have enabled scalable, data-driven approaches to action recognition [10, 11]. In supervised paradigms, algorithms are trained on annotated image datasets to recognize specific behaviors of interest. In unsupervised settings, algorithms can independently identify recurring patterns in pose and motion data, which can then be interpreted and validated experimentally. A growing number of algorithms now offer behavioral classification capabilities [10, 11], but most remain separate from gold-standard pose-estimation frameworks such as *DeepLabCut*, and *SLEAP*. This lack of integration creates latency bottlenecks, limiting their applicability in studies aiming to map complex behavioral repertoires, particularly those involving spontaneous social interactions in enriched or naturalistic environments over prolonged periods. Yet, achieving this level of behavioral granularity is critical for understanding the strategies employed by individual animals within social groups as they navigate environmental challenges.

In this study, we introduce *PrecisionTrack*, a fully automated, multi-animal tracking system designed for prolonged behavioral analysis of socially interacting animals. *PrecisionTrack* is shown to be faster, more accurate, and more robust than the existing multi-animal tracking gold standards, *DeppLabCut* and *SLEAP*, when tracking large groups of interacting animals. Furthermore, when combined with our marker-based data-driven re-identification module, *PrecisionTrack* consistently preserves identity assignments even in large social groups monitored over prolonged periods. *PrecisionTrack* also includes a novel action recognition algorithm predicting individual, spontaneous, and social actions at scale. By integrating robust multi-animal pose estimation, persistent ID tracking, and real-time action recognition in a single, scalable platform, *PrecisionTrack* reliably tracks and infers the actions of individuals within social groups with real-time online responsiveness, offering a powerful new solution for the automated analysis of complex behaviors in social contexts.

Results

PrecisionTrack is an open-source multi-animal pose-tracking and action recognition software (**Figure 1a**). It provides a rich ecosystem of high-level application programming interfaces (API)s, enabling users to build applications and extensions. Additionally, it offers configurable modules (**Figure 1b**) enabling the transformation of raw, unlabeled experimental recordings into fully trained, optimized, and deployable systems. A step-by-step configuration guide allows researchers to customize the system to support new species, skeletal landmarks, and behavioral repertoire. Advanced users can also refer to our comprehensive settings guide to gain fine-grained control over the system's neural network architecture, training schedules, hyperparameters, and many other options, including hardware acceleration backends, without having to write a single line of code. This added flexibility also enables fast implementation and testing of novel algorithmic and machine learning tools, ensuring the system's long-term relevance.

PrecisionTrack's provide various backend interfaces, allowing for a plug-and-play integration of diverse neural-network architectures and runtime compilations. We are currently supporting the three leading deep-learning runtimes, *PyTorch* [12], *ONNX* [13], and *TensorRT* [14] (**Figure 1c**). Additionally, our annotation format follows community standards, the Microsoft COCO specification for object detection, and the MOTChallenge format for multi-object tracking. This provides immediate compatibility with a plethora of other systems and benchmarks. It also allows users to prepare datasets using widely adopted open-sourced annotation suites such as *CVAT* [15], *LabelStudio* [16] and *COCOAnnotator* [17], then seamlessly import them into any *PrecisionTrack* workflow (**Figure 1d**).

PrecisionTrack is a dockerizable application, able to run on Windows, macOS, and Linux. Both CPU- and GPU- systems are supported, although training and inference are substantially faster on CUDA-enabled devices. Overall, *PrecisionTrack* offers a variety of flexible configuration options, standards-compliant I/O, modular architectures, and hardware-agnostic workflows, ensuring adaptability as computer-vision methods and computing infrastructure continue to evolve.

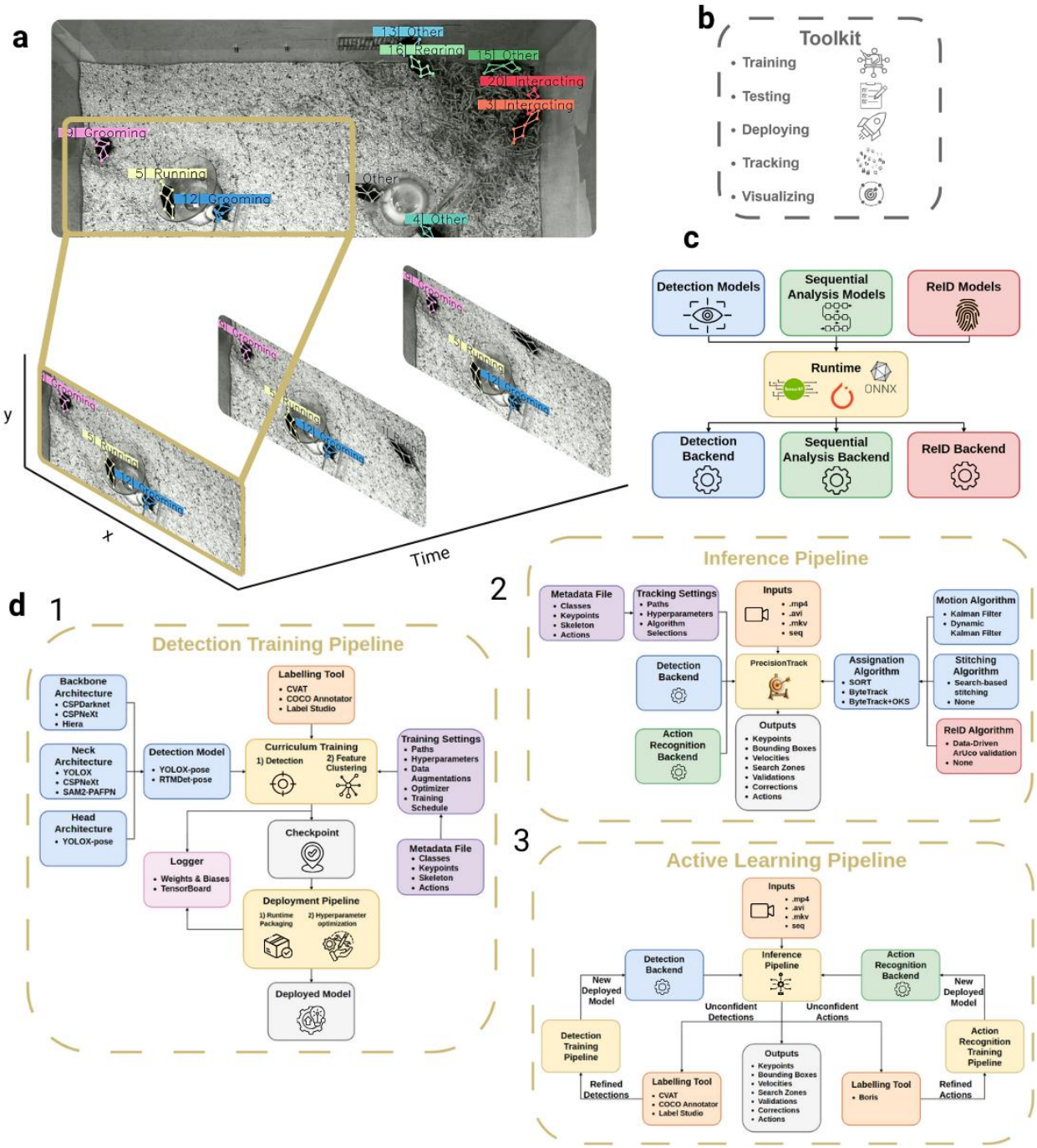

Figure 1: a, Schematic representation of *PrecisionTrack*'s capabilities, including subject detection, classification, and pose estimation, as well as multi-animal pose tracking, action recognition, and social behavior classification. **b**, Overview of the main tools available to *PrecisionTrack* users, enabling them to either run predefined, documented pipelines or create new ones. **c**, Illustration of the system's flexibility. Existing and future algorithms can be incorporated, provided they adhere to the documented interfaces. These models will then be compatible with available runtimes and backends. **d**, (1) Predefined training pipeline for detection, classification, and pose-estimation models. Users can follow our configuration guide to define custom settings (purple) used by the training tool (yellow). Advanced users can also modify the model architecture and hyperparameters directly via the configuration file (blue). Once the training is done, a checkpoint is saved to the specified path. The trained model can then be deployed using the deployment tool (yellow), resulting in a customized and ready-to-use *PrecisionTrack*.(2) Predefined inference pipeline. Users provide their customized settings (purple) as well as a recording (orange) to the tracking tool (yellow). After analysis, the algorithm outputs MOT-formatted subject keypoints, bounding boxes, velocities,

search zones, validations, corrections, and actions. (3) Active learning pipeline. Users leverage both the training and inference pipelines to automatically identify low-confidence predictions. These can be manually labeled to iteratively refine the *PrecisionTrack* system.

DATASETS AND BENCHMARKS

Currently available multi-mouse datasets fail to accurately reflect real-world, ethological conditions [18, 5, 6]. These datasets usually represent between 1-3 mice in overly simplified environments, repressing the animals' ability to exhibit their whole spontaneous and social behavioral repertoires [6]. We argue that these limitations prevent the field from progressing. Hence, we generated a novel and more challenging multi-mouse dataset representing real-world environmental conditions. We named this dataset MICE. MICE is composed of recordings of 20 mice socially interacting in a 1m x 1m open field area enriched with bedding, spinning wheels, water bottles, feeding trays, tubes, wooden blocks, and nesting material. Each mouse's pose, or skeleton, is defined by eight key points: snout, left ear, right ear, centroid, left hip, right hip, tail, and tag (**Figure 2a**). Overall, we have manually labelled over one million sequential locations and poses of socially interacting mice. To facilitate the development of reliable systems in challenging real-world scenarios, we also provide detailed testing benchmarks to help standardize future multi-mouse algorithm comparisons. This sequential dataset is freely available to the community and should serve for the development of more advanced multi-animal pose tracking and action recognition algorithms.

MULTI-SUBJECT DETECTION, CLASSIFICATION AND POSE ESTIMATION

To ensure *PrecisionTrack's* robustness and generalizability, we implemented a comprehensive evaluation pipeline (see Methods) combining metrics that jointly assess object detection and pose-estimation performances. We benchmarked *PrecisionTrack* using three comprehensive datasets, including MICE, Animal Pose (AP) [19] and Microsoft COCO [20].

To integrate pose-estimation and object detection efficiently, we first paired the anchor-free and inline pose-estimation method proposed by *YOLO-Pose* [21] with the performant *YOLOX* object-detection algorithm [22] (**Figure 2b**). This integration allows pose-estimation to

benefit from the latest object-detection advancements such as strongly augmented data (**Figure 2c**), as opposed to the traditional two-stage pipeline separating object-detection and pose-estimation tasks.

Next, we optimized *YOLOX*'s training hyperparameters and replaced its stochastic gradient descent (*SGD*) [23] optimizer with the *AdamW* [24]. This raised its macro-average performance on our evaluation metric (**Equation 5**) by 5.49%. Third, we replaced *YOLOX*'s original *CSPDarknet* backbone architecture with the more efficient *RTMDet*'s *CSPNeXt* [25] (**Figure 2d**), further improving its performance by 2.26% (**Figure 2e**) without compromising its inference speed. Indeed, across all three runtime environments tested, including Pytorch (4.17ms: 4.28ms), *ONNX* (5.59ms: 6.03ms), and *TensorRT* (3.02ms: 3.14ms), we observed no significant detection speed difference between the *CSPDarknet* and the *CSPNeXt* backbones (**Figure 2f**).

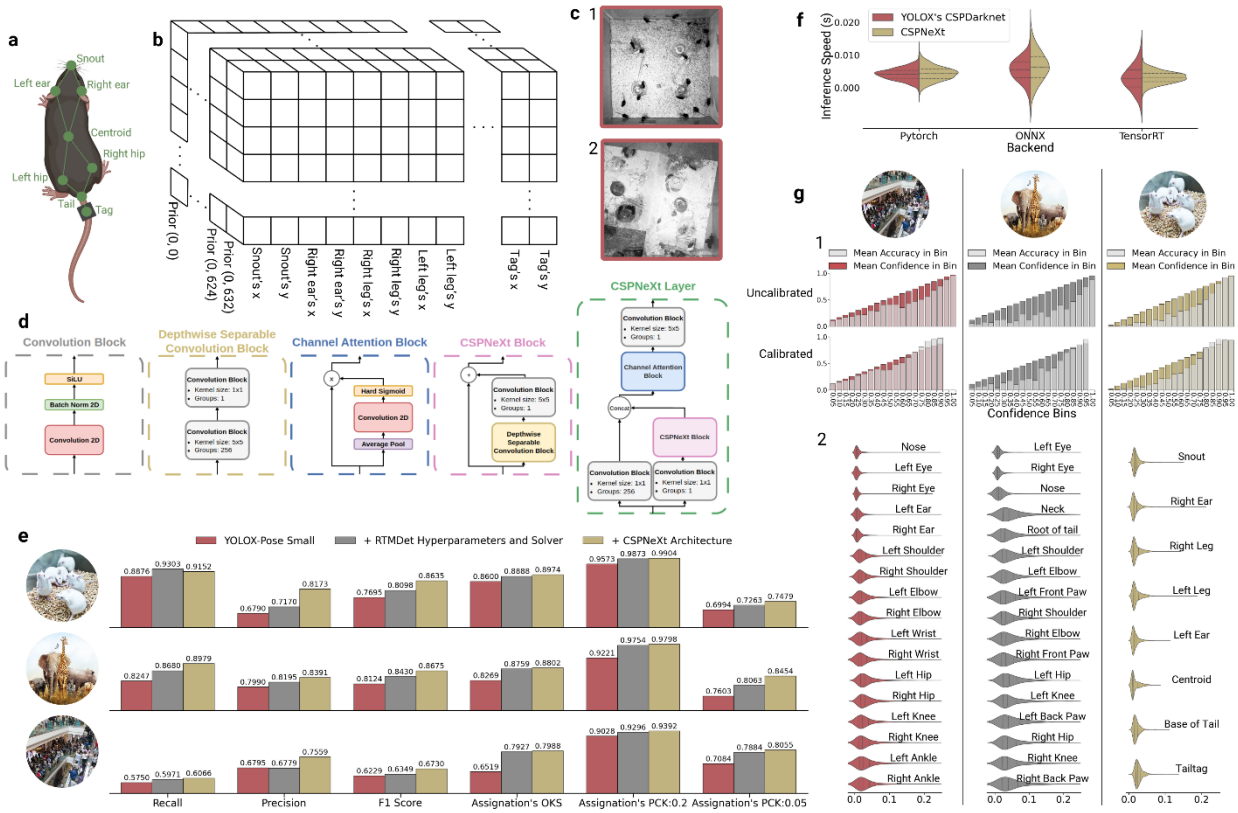

Figure 2: a, The MICE dataset defines a top-down view mouse skeleton with 8 keypoints: snout, left ear, right ear, centroid, left hip, right hip, tail, and tag. **b**, A breakdown of the decoupled keypoint head shows how the prior's features are processed in parallel, resulting in 8,400 pose predictions across the three levels of the feature pyramid. These predictions are then refined by multiplying them with the prior's stride and adding the prior's coordinates. **c**, (1) Raw training image from the MICE dataset. (2) Augmented training sample from the first training stage, where Mosaic and Mixup augmentations are applied alongside randomly selected geometric and color space transformations. **d**, Inference latency comparison. Measured inference delays of *YOLOX*-Pose (red) and the proposed model (dimmed yellow) across different backends (*PyTorch*, *ONNX*, *TensorRT*). All measurements were taken at FP16 precision with a batch size of 1 on an RTX 3090 GPU. It includes preprocessing, forward pass, and post-processing. **e**, Forward propagation through a *CSPNeXt* layer (green), consisting of *CSPNeXt* blocks (pink), which include a depth-wise convolution block (dimmed yellow), along with a Channel Attention (CA) block (blue) and standard convolution blocks (grey). **f**, Performance comparison between the baseline *YOLOX* model (red) and the proposed object detector (dimmed yellow) across three datasets: MICE (top), Animal-Pose (middle), and Microsoft COCO (bottom). Detection quality is evaluated using Recall, Precision, and F1 Score, while pose estimation accuracy is assessed with Object Keypoint Similarity (OKS) and Percentage of Correct Keypoints (PCK) at 5% and 20%. Only validation split results are reported. **g**, (1) Confidence distribution of the proposed model's detections on the validation split of the Microsoft COCO dataset (left), the Animal-Pose dataset (middle), and the MICE dataset (right). Detections are grouped into 20 bins, from the least confident (right) to the most confident (left), with the y-axis representing the average confidence within each bin. The pale gray overlay indicates the actual detection accuracy per bin, illustrating the model's reliability. A well-calibrated model maintains a close to one-to-one accuracy/confidence ratio across a wider upper confidence range. (2) Distribution of keypoint prediction errors for the proposed model across each validation split. Distances are normalized by the subject's size.

Entropy-based image classifiers are known to exhibit systematic overconfident prediction scores [26]. Our analysis revealed similar overconfidence in our detector's predictions across all three datasets (**Figure 2g**, Uncalibrated). This overconfidence is particularly problematic for downstream tasks, such as multi-animal pose-tracking, which rely on confidence scores as proxies for accuracy. To address this, we developed an object-detector neural network calibration scheme based on temperature scaling [26], a state-of-the-art post-processing algorithm to align predicted confidence scores with model accuracy. By optimizing the expected calibration error (ECE) metric [27], our calibrating process reduced the confidence/accuracy discrepancy by 16.3%, 23.1% and 21.8% on MICE, AP and COCO datasets, respectively (**Figure 2g**). Noticeably, this significantly reduced the accuracy drop previously noted in the critical 75–85 % confidence band (**Figure 2g**, Calibrated) and the pose-estimation errors on keypoints associated with fast-moving limbs (**Figure 2g**) across all datasets.

Overall, by building upon recent advancements in the Neural network Architecture Search (NAS), data augmentation and object-detection fields, we designed a multi-subject

classification, localization and pose-estimation algorithm shown both fast and accurate when tested on three separate benchmarks specifically designed to assess performances in crowded, highly occlusive scenarios.

FEATURE EXTRACTION

High dimensionality data often pose a significant challenge in pattern recognition due to their potentially high signal-to-noise ratio [28]. This is particularly true for high-resolution images [29]. Fortunately, established computer vision fields such as image classification have overcome this challenge on a plethora of benchmarks [30, 31] by leveraging deep neural networks to slowly reduce the feature space's spatial dimensions [30]. However, more specific architectures, such as those leveraging bigger receptive fields and feature pyramids (**Figure 3a**), were shown more performant for semantically dense tasks such as object-detection, pose-estimation, and instance segmentation [32]. Furthermore, our detector's *CSPNeXt* backbone architecture is not only part of the latter group but is also the current state-of-the-art in real-time instance segmentation [25]. This suggests that *PrecisionTrack's* extracted features could be semantically rich enough to serve as meaningful inputs for dense context-dependent downstream tasks such as action recognition and individual re-identification.

To verify this hypothesis, we evaluated the extracted feature maps' coherence, meaning its ability to cluster together features belonging to the same individuals, by measuring the latent feature map's silhouette score. The initial scores were relatively low (MICE: -10.26%, *AP*: 15.76%), indicating poorly clustered embeddings and a high variability in subject representations (**Figure 3b**). This means that our network's feature map, although semantically meaningful, lacks consistency, limiting its utility for downstream tasks. To address this issue, we added a novel Convolutional Neural Network (CNN) clustering head to our detector (**Figure 3a**). The head optimizes a semi-hard mined triplet loss function, as used in representation clustering methods (33). Most existing approaches jointly optimize representation clustering along with entity classification, detection, and pose-estimation during training [34, 35], with some reporting a degradation in detection and pose-estimation performance [35, 36]. We argue that this likely results from a misalignment between the

detection/pose-estimation and the feature extraction objectives. *PrecisionTrack* leverages a curriculum learning schedule instead. We first train the full network for entity classification, detection, and pose-estimation, as described in the previous section, then train the clustering head on frozen feature maps. This strategy increases the latent space's coherence (**Figure 3b**), achieving validation silhouette scores of 83,39% and 95.68% on the MICE and AP datasets, respectively, without affecting detection performances.

Globally, our method significantly enhances the extracted feature maps' coherence, which in turn improves the consistency of identity representations across time and space (**Figure 3c**). This enables downstream tasks such as action recognition to access both meaningful and coherent features without having to redundantly reprocess raw pixels. Thus, *PrecisionTrack* could be efficiently paired with a plethora of downstream modules, reducing their computational costs and improving their scalability.

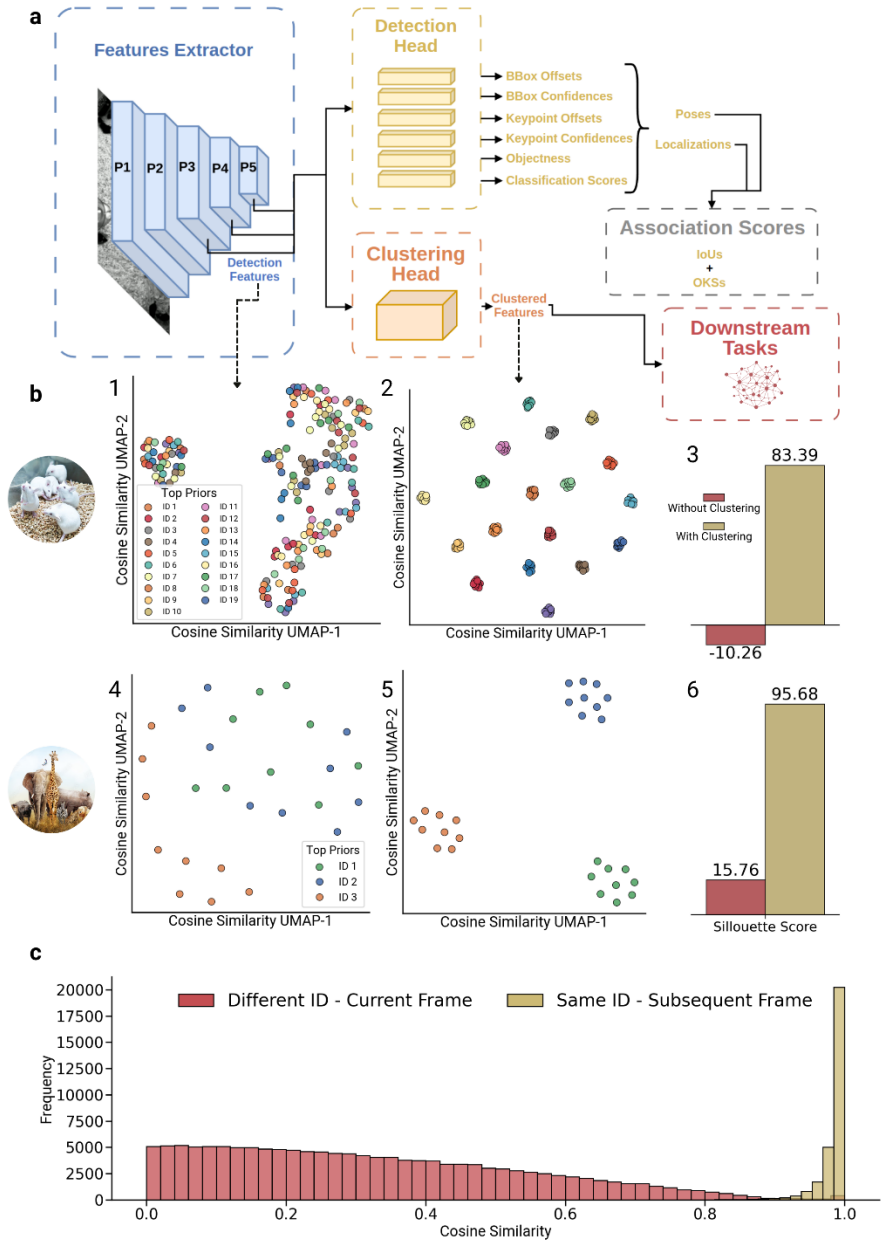

Figure 3: a, Schematic representation of *PrecisionTrack*'s architecture for subject detection, classification, pose estimation, and feature extraction. Raw RGB images are first processed by the feature extractor (blue), producing feature maps. The last three feature maps are selected for further processing. These selected maps are processed by various decoupled heads (yellow) to produce subject poses, bounding boxes, and classification scores. Simultaneously, the same feature maps are processed by the clustering head to extract entity-specific features (orange). **b**, Qualitative and quantitative evaluation of the coherence of the detection features and the extracted features. (1, 4) 2D visualization of each detected entity's top-k detection features from an image in the MICE dataset (top) and the AP dataset (bottom). (2, 5) 2D visualization of each detected entity's top-k extracted features from an image in the MICE dataset (top) and the AP dataset (bottom). (3, 6) Silhouette scores measuring cluster coherence for detection features (red) and extracted features (dimmed yellow) on the MICE (top) and AP (bottom) datasets. **c**, Extracted features remain coherent over time: features from subsequent timesteps belonging to the same entity remain similar (dimmed yellow), while features from different entities diverge (red).

MULTI-ANIMAL POSE TRACKING

To enable accurate tracking of multiple animals at once, we adopted the current dominant paradigm in multi-object tracking (MOT), namely Tracking-By-Detection (TBD) [37]. TBD algorithms are composed of two steps: (1) a detection step, as described in the previous section, and (2) the association step detailed here. Among recent TBD association algorithms including *SORT* [37], *DeepSORT* [38], and others [39], *ByteTrack* [40] is the current state-of-the-art. By incorporating high and low confidence detection in its association process, *ByteTrack* is particularly well adapted for crowded and occlusive environments [40]. However, although effective on standard *MOT* benchmarks involving pedestrians and cars moving unidirectionally at a steady pace [41], *ByteTrack* has not been optimized for multi-animal tracking, which features frequent path crossings, irregular trajectories, and extended occlusion periods.

We addressed these limitations by implementing a series of refinements to the original *ByteTrack* algorithm tailored for multi-animal tracking in complex environments. First, we augmented *ByteTrack*'s intersection over union (IoU)-based association cost with our variant of the object keypoint similarity (OKS) metric (see Method section). This modification allows the algorithm to consider not only the subjects' locations, but also their poses during association. Indeed, as shown in **Figure 4a**, relying solely on individuals' location fails to maintain reliable identification in highly occlusive scenarios. On our MICE benchmark, we found that implementing our OKS metric into *ByteTrack*'s association step improved its MOTA from 95.5% (IoU only) to 95.6% (IoU+OKS) and IDF1 from 88.4% (IoU only) to 90.7% (IoU+OKS).

ByteTrack employs the standard linear Kalman Filter with constant velocity model [42] to predict subject locations in successive frames. This algorithm assumes relatively consistent motion, which we argue is a poor assumption for tracking animals. This is because animals, especially in open fields, tend to exhibit inconsistent movement patterns composed of sudden shifts in both direction and speed. Thus, we next developed a more representative model, our Dynamic Kalman Filter (DKF), by incorporating 2D Newton's law of dynamics into the original Kalman Filter. As shown in **Figure 4c**, DKF enhanced trajectory prediction (red

corners) in scenarios showing rapid changes in directionality and velocity. Switching the original Kalman Filter with our DKF algorithm improved our motion prediction performance on both the MICE (KF: 83.1%; DKF: 92.4%) and the AP (KF:73.8%; DKF: 81.8%) benchmarks. It also improved the tracking IDF1 from 90.7% (IoU+OKS only) to 94.3% (IoU+OKS+DKF) on our MICE benchmark.

Finally, we used our challenging MICE benchmark to evaluate *PrecisionTrack*'s performance against gold-standard multi-animal pose tracking packages, *DeepLabCut* [6] and *SLEAP* [5]. As shown in **Figure 4d**, our results suggest that *PrecisionTrack* (beige) outperforms *DeepLabCut* (red) and *SLEAP* (grey) on both MOTA (*PrecisionTrack*: 95.6% *DLC*: 85.2%; *SLEAP*: 73.5%) and IDF1 (*PrecisionTrack*: 94.3% *DLC*: 84.9%; *SLEAP*: 72.2%). This suggests that *PrecisionTrack* performs well in crowded and naturalized environments where mice exhibit spontaneous and socially ethological behaviours. Furthermore, our results show that *PrecisionTrack* maintains fast processing latency even as the length of the processed recording increases. **Figure 4d** shows the processing latency for experiments of increasing duration. For example, processing a 40-minute experiment (pre-recorded at 25 FPS) from the MICE dataset required 12.5 minutes with *PrecisionTrack* compared to 29 minutes with *SLEAP* and 142 minutes with *DeepLabCut*. This suggests that *PrecisionTrack* is optimized to process long-duration experiments online and in a real-time manner, which is perfectly adapted for behavioral studies of social behaviors at scale.

We also evaluated *PrecisionTrack*'s generalization ability using the APTV2 tracking dataset [43], which comprises 2,749 video clips filtered and collected from 30 distinct animal species. *PrecisionTrack*'s pose-tracking performance did not significantly vary across species (**Figure 5a**).

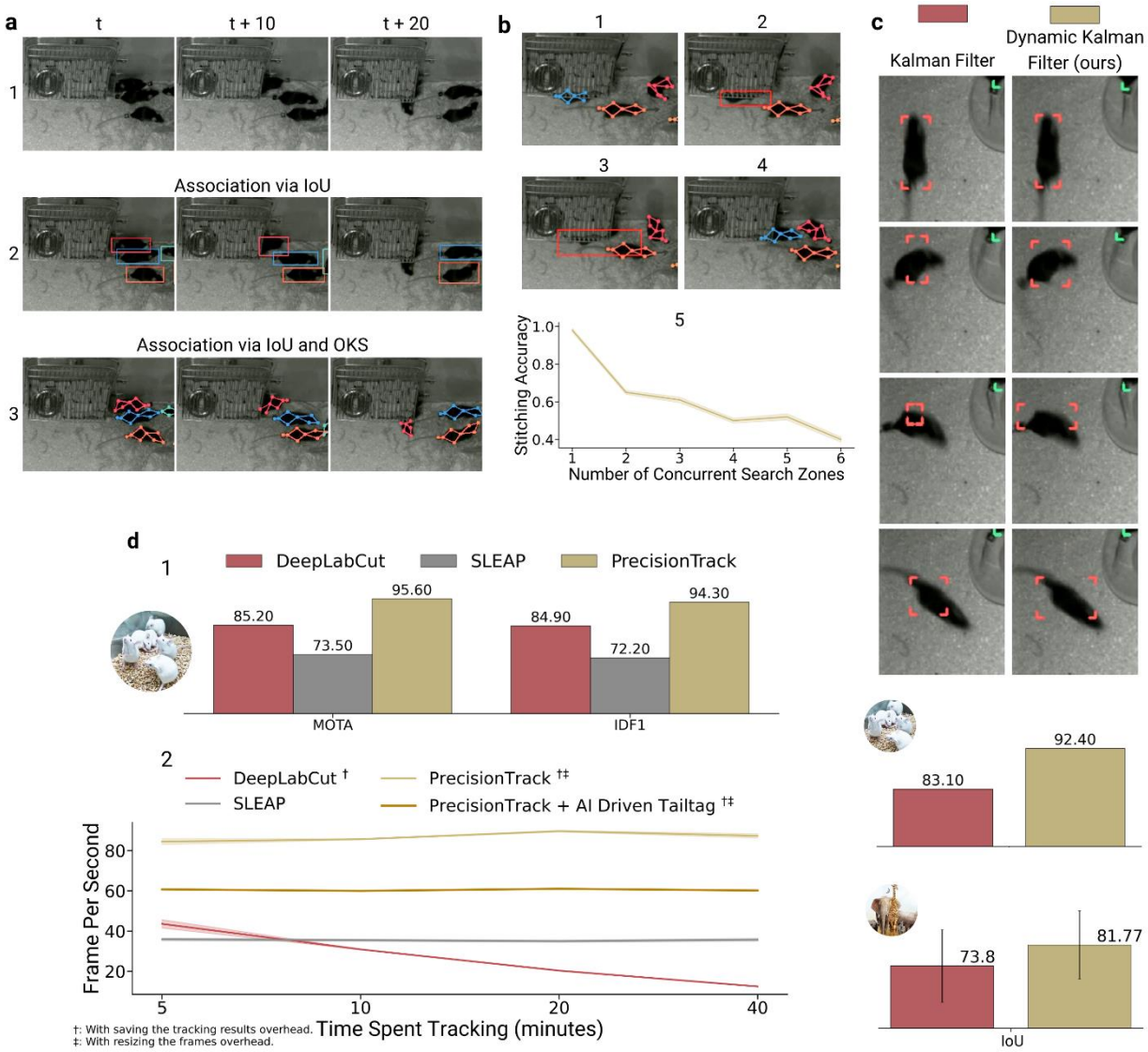

Figure 4: a, Real-world example illustrating the benefit of OKS-enhanced association. The red mouse is partially occluded by the feeder (1). An algorithm relying solely on IoU would fail to associate the newly visible head (2, t+20) with the red mouse, as there is no overlap with its last seen position (2, t+10). In contrast, an algorithm using both IoU and OKS succeeds, since the ears of the new detection (3, t+20) are close to the red mouse's last seen ears (3, t+10), enabling the association. **b**, Visualization of search zone evolution. Given the a lost subject's last seen dynamics and location (1), the search-based stitching algorithm recovers its trajectory (4) by incrementally expanding the assigned search zone (2, 3). (5) The algorithm's effectiveness rapidly degrades when multiple subjects are lost at the same location. **c**, Side-by-side comparison between the standard linear Kalman Filter with constant velocity and our Dynamic Kalman Filter. (1) Qualitative comparison when a tracked subject makes a sharp turn followed by rapid acceleration. Corners indicate predicted positions in the next frame, not actual tracking coordinates. (2) Quantitative comparison of predictive performance on the MICE dataset (top) and across all species in the AP dataset (bottom), showing IoU between predicted and actual next frame locations. **d**, (1) Multi-Object Tracking Accuracy (MOTA) and IDF1 scores for *PrecisionTrack*, *DeepLabCut*, and *SLEAP* on the 20-mice MICE benchmark. All models were trained on the MICE pose-estimation dataset. The selected DeepLabCut model is a bottom-up *dlcrnet_ms5* with *PAF*s enabled; the *SLEAP* model is a bottom-up *EfficientNetB0* with simple max tracks and IoU-based Hungarian matching. (2) Tracking speed (FPS) comparison of *PrecisionTrack*, *DeepLabCut*, and SLEAP. This evaluates temporal scaling. Online approaches (*PrecisionTrack* and SLEAP) maintain responsiveness as recording length increases, unlike the offline *DeepLabCut*. *PrecisionTrack* retains real-time responsiveness even when paired with our data-driven *Tailtag* re-identification module (dark yellow).

SUBJECT RE-IDENTIFICATION

Previous sections described how we designed *PrecisionTrack* to be both accurate and reliable even in crowded and enriched environments. However, even with these improvements, IDF1 tracking performances will inevitably decrease when tracked individuals get occluded for prolonged periods. This may happen when mice integrate their nest, dig into bedding, or enter housing enrichments. These periods of disappearance result in broken trajectories, limiting the system's ability to maintain its tracking accuracy over time.

To address this problem, we developed a novel trajectory stitching algorithm. This method defines search zones around each subject's last known position. The size of the zones increases proportionally to the subject's last seen 2D motion dynamics. **Figure 4b** illustrates how the search zone expands according to the likelihood of reappearance across successive frames. When tested on the MICE benchmark, our stitching algorithm effectively maintained subject identity but only when one subject reappeared within the search zone (**Figure 4b**). In contrast, in multi-subject reappearance scenarios, our stitching algorithm alone becomes insufficient (**Figure 4b**), with the re-identification accuracy falling to 40.8 % when six mice disappear in the same area.

In such cases, a more robust re-identification strategy is required. We argue that when subjects are visually indistinguishable, markerless vision-based re-identification methods are, by definition, unreliable. Therefore, visually marking near-identical animals remains the most reliable solution to re-identify them. To this end, we selected the *Tailtag* system [44], a lightweight visual marker designed for long-term use without behavioral interference. *Tailtag*s are visible under both daylight and infrared conditions and do not require specialized or expensive imaging hardware. We integrated this system into *PrecisionTrack* and optimized it in two ways. First, we restricted the refinement of the *Tailtag* identifications from full frame to areas around the tag's estimated positions (**Figure 5c**). This spatial restriction maximized identification speed without impacting precision (**Figure 5d**). Second, we improved the original *Tailtag*'s re-assignation algorithm by incorporating a data-driven tag reading

approach, accumulating evidence and reassigning IDs only when it surpasses a defined confidence threshold.

We evaluated re-identification performance using our MICE longitudinal benchmark, comprising over one million manually curated mouse localizations. Without the data-driven *Tailtag* re-ID module, *PrecisionTrack*, *SLEAP* and *DeepLabCut* show rapid decreases in their ability to maintain correct identification, as measured by IDF1 (**Figure 5b**). In contrast, when paired with our data-driven *Tailtag* system, all three were able to sustain their initial identification capabilities *(PrecisionTrack*: mIDF1 after 1 minute; 95.7%, mIDF1 after 10 minutes; 93.3%; *SLEAP*: mIDF1 after 1 minute; 69.7%, mIDF1 after 10 minutes; 66.6%; *DeepLabCut*: mIDF1 after 1 minute; 82.83%, mIDF1 after 10 minutes; 79.89%; **Figure 5b**). Furthermore, our analysis shows that our data-driven *Tailtag* system corrects most tracking errors within a few seconds, with more than 90% of errors corrected within 10 seconds and the remaining within 30 seconds (**Figure 5d**), with outliers often corresponding to cases where mice hid immediately after a tracking error.

Globally, our results show that implementing a two-stage re-identification process, combining our stitching algorithm with our data-driven *Tailtag* system, effectively overcomes one of the most important limitations of modern vision-based trackers, namely, their inability to accurately track near-identical subjects in crowded settings over prolonged periods.

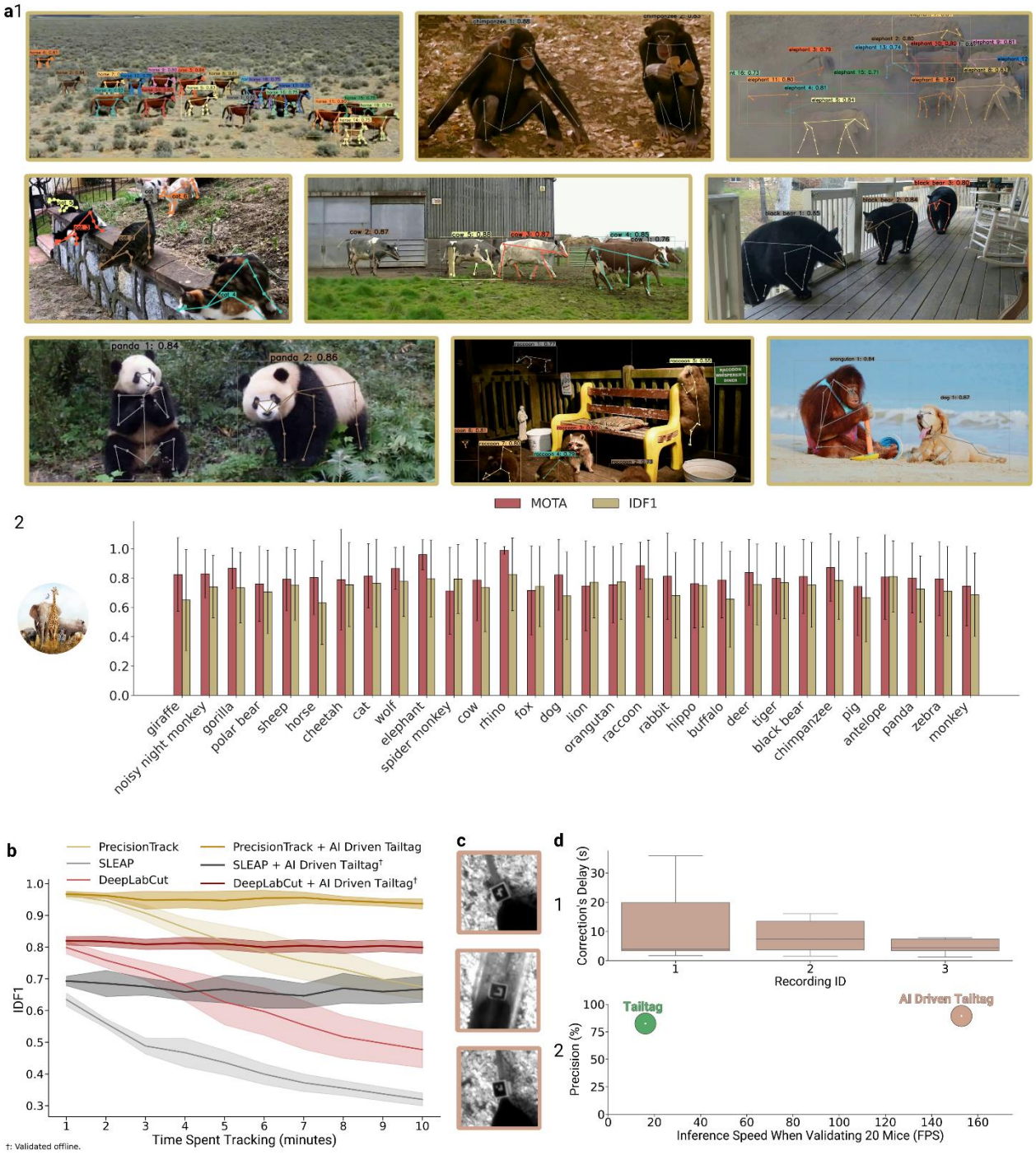

Figure 5: a, (1) Qualitative assessment of *PrecisionTrack*'s ability to detect, estimate poses, and classify species beyond mice. (2) Quantitative evaluation of *PrecisionTrack*'s Multi-Object Tracking Accuracy (MOTA) and IDF1 when tracking 30 distinct animal species from the APTv2 dataset. Tracking performance is not determined by species, but by viewpoint quality and occlusion level in the recordings. **b**, Evolution of IDF1 over 10-minute recordings for *PrecisionTrack* (dimmed yellow), *DeepLabCut* (red), and *SLEAP* (grey). We also evaluated each method when paired with the data-driven *Tailtag* system. All three methods maintain their performance levels when augmented with *Tailtag*. **c**, Visual representation of the improved marker detection context. The *Tailtag* system now searches for markers only in regions of interest, rather than across the full frame. **d**, (1) Distribution of correction delays across three 10-minute recordings. (2) Reading precision (y-axis) and speed (x-axis) comparison between the improved detection context (dimmed pink) and the original *Tailtag* system (green). By limiting marker searches to relevant regions, the system significantly reduces computational load and improves overall speed.

Action recognition and behavioural analysis

Being able to automatically extract insights beyond spatiotemporal localizations, such as the subject's actions and social behaviors, would benefit many research applications. It could enable new research directions and allow existing studies to scale efficiently, supporting larger cohorts and longer, more complex trials.

Current multi-animal tracking tools lack the capability to infer subjects' actions directly. As a result, researchers instead rely on various standalone tools [10 ,11] to perform post-hoc action recognition. This redundant externalization reduces the practicality of the whole process. It increases its computational cost, forces offline calculations, and introduces a manual step. As a result, automatically analyzing spontaneous and social behaviors in large groups over extended periods remains a tedious and partially manual process.

To address this limitation, we integrated a novel action recognition module into *PrecisionTrack*, called *MART* (Multi-animal Action Recognition Transformer). *MART*, when exploited in parallel via or *ParrallelTracker* API, does not affect *PrecisionTrack*'s real-time and online responsiveness, while enabling it to infer actions and social behaviors for each tracked subject.

MART is a transformer decoder (**Figure 6a**), structurally similar to GPT-2 [45], a state-of-the-art architecture for sequence classification tasks, particularly in the field of Natural Language Processing [46]. By casting the action recognition problem as the classification of sequences composed of each subject's timestep representations, we are able to leverage *MART*'s ability to selectively reflect over past and present inputs to predict ongoing behaviors. The length of the reflection sequence is known as the context window, meaning that increasing the context window allows *MART* to attend further in the past. Our experiments showed that expanding the context window beyond a single frame enriches the model's decision-making context and improves performance across all identified actions (**Figure 6b**).

For mice, we found that the optimal context window spans approximately 30 frames (~ 2 seconds) into the past, with a significant performance increase when the algorithm is allowed to attend to more than the current frame (Context window 1: 79.5%; 15: 84.1%, 30: 84.3%, 60: 83.5%). Interestingly, our results show that inference speed does not vary significantly across context-window sizes (**Figure 6b**), suggesting that *MART* is scalable on modern hardware.

Furthermore, we investigated whether enriching the sequence's timestep representations could improve performance. Our results show that using either the subject's poses (**Figure 6c**, red) or their extracted features (**Figure 6c**, grey) individually yields comparable performance. However, combining both leads to a clear improvement (**Figure 6c**, dimmed yellow). Also, considering the subject's velocities further enhanced performance for dynamic actions such as running and interacting (**Figure 6c**, green). Although *MART* was not explicitly trained for representation clustering, it effectively grouped both the supervised classes (**Figure 6d**) and several unlabeled "Other" behaviors (**Figure 6d**). Overall, leveraging an algorithm capable of reflecting over multiple timesteps, while providing rich representations at each timestep is effective at distinguishing between subjects' actions (**Figure 6e**).

When paired with *MART*, *PrecisionTrack* can gather results beyond spatiotemporal localizations, enabling more refined analyses. For example, we observed that, in our experiments, rearing occurred primarily near elevated water bottles and along the perimeter, social interactions clustered near the nests, running was concentrated on the wheels, and grooming mostly took place in isolated corners (**Figure 6c**). By pairing *MART*'s action labels with *PrecisionTrack*'s poses, we were able to refine social interactions and distinguish nose-to-nose from anogenital contacts. Using the resulting system, we constructed nose-to-nose and anogenital social networks (**Figure 6c**).

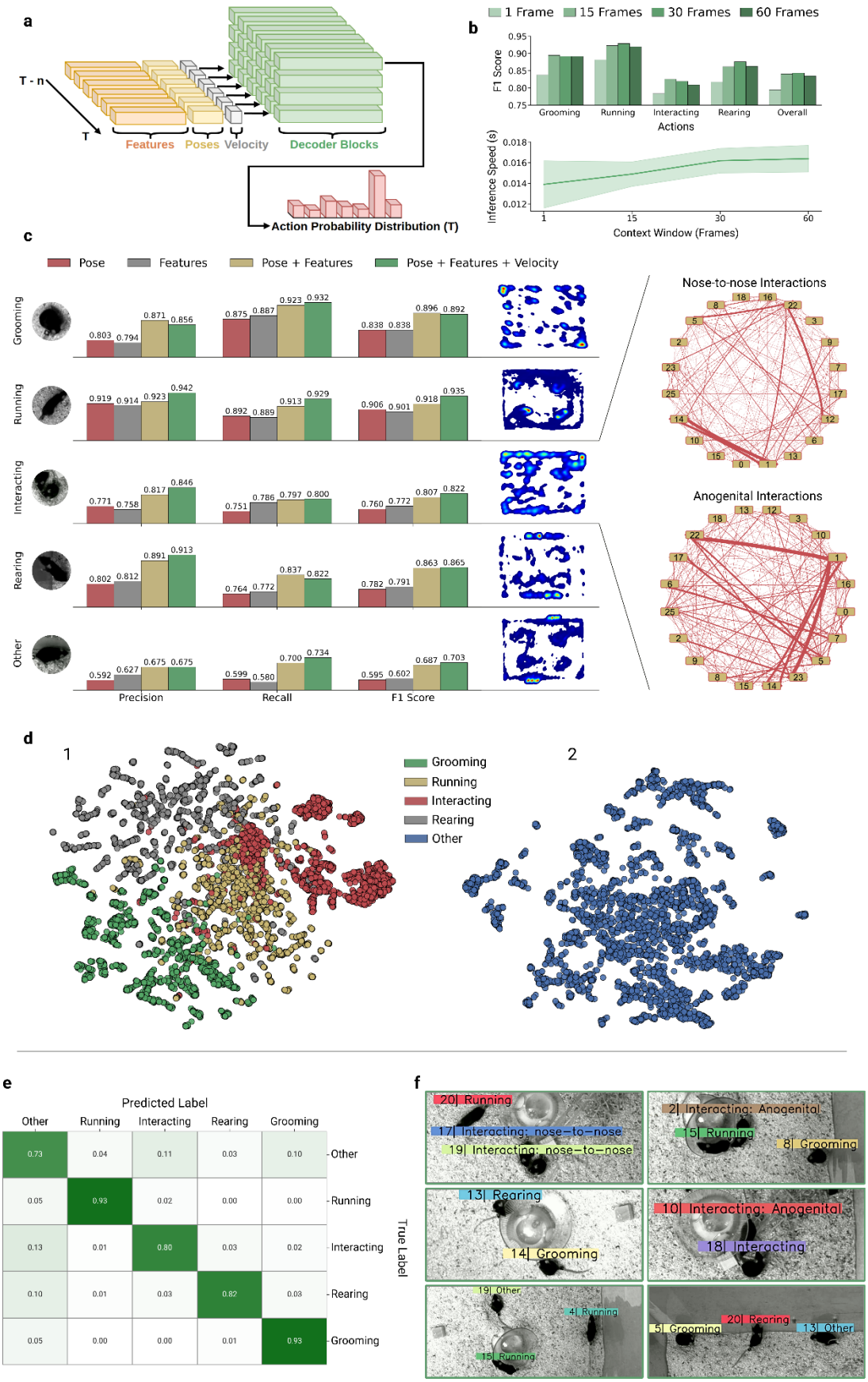
a
T - n
T
Features
Poses
Velocity
Decoder Blocks
Action Probability Distribution (T)
b
1 Frame
15 Frames
30 Frames
60 Frames
F1 Score
Grooming
Running
Interacting
Rearing
Overall
Actions
Inference Speed (s)
Context Window (Frames)
c
Pose
Features
Pose + Features
Pose + Features + Velocity
Grooming
Running
Interacting
Rearing
Other
Precision
Recall
F1 Score
Nose-to-nose Interactions
Anogenital Interactions
d
1
2
Grooming
Running
Interacting
Rearing
Other
e
Predicted Label
Other
Running
Interacting
Rearing
Grooming
True Label
f
20| Running
17| Interacting: nose-to-nose
19| Interacting: nose-to-nose
2| Interacting: Anogenital
15| Running
8| Grooming
13| Rearing
14| Grooming
10| Interacting: Anogenital
18| Interacting
19| Other
4| Running
15| Running
5| Grooming
20| Rearing
13| Other

Figure 6: a, Overall architecture of the Multi-animal Action Recognition Transformer (*MART*). The context window (containing n frames) spans the z-axis, the representations lie along the x-axis, and the four decoder blocks are shown on the y-axis. **b**, Context window ablation study. Top: performance on all labeled actions. Bottom: latency measurements for each tested context window. **c**, Representation ablation study for all labeled actions and the "Other" action, which serves as the negative class. Recall, Precision, and F1 Scores are shown for each representation. Action-specific heatmaps indicate spatial hotspots within the vivarium. Using *PrecisionTrack* poses, we refined some interactions to distinguish between nose-to-nose and anogenital interactions, enabling the construction of social networks that represent the subject's relationships for these two behaviors. **d**, The final decoder block of *MART* can cluster both labeled actions (1) and unlabeled actions (2). **e**, Confusion matrix showing agreement between predicted (x-axis) and manually labeled (y-axis) actions. **f**, Qualitative examples of inferred actions on video sequences not used during training.

Discussion

The ethological study of behavior, whether in insects, rodents, primates, or humans, has significantly advanced our understanding of behavioral repertoires and the diversity of social strategies that allow species to adapt to environmental challenges threatening their survival. With current multi-animal tracking algorithm achieving high spatiotemporal accuracy in semi-naturalistic environments, the field is on the verge of being able to effectively transition from manual annotation methods to efficient, tailored artificial intelligence approaches. However, three major limitations hinder this transition.

First, even gold-standard multi-animal tracking algorithms struggle to reliably re-identify physically similar individuals. This makes them unfit for tracking scenarios involving prolonged occlusion periods, such as subjects entering nests or forming tight crowds. This diminishes their practicality in real-world scenarios by limiting their application to short experiments involving a few individuals roaming in bare home cages [8, 6, 5]. To overcome this limitation, the field needs a reliable method to re-identify tracked individuals, no matter their species. While many markerless re-identification methods report interesting performances on subjects from easier-to-distinguish species [47], we argue that these levels of performance are not transferable to subjects from hard-to-distinguish or near-identical species. This is because visually re-identifying near identical subjects without markers is, by definition, nearly impossible. This means that the logical path forward for developing general and robust visual re-identification methods is to leverage marker-based methods.

Second, most multi-animal tracking methods prioritize accuracy over latency or scalability. Historically, this was not a major issue, as automatic multi-animal tracking algorithms are, to this day, only leveraged on short, controlled experiments that did not require real-time analysis. As a result, many popular approaches, such as *SIPEC*, which uses slow two-stage detectors [7], and *DeepLabCut*, which relies on offline tracklet stitching algorithms [6], optimize for accuracy at the cost of speed. Although perfectly adequate for the current state of the field, this design choice will become a limiting factor for researchers aiming at scaling to longer/more crowded experiments. Concretely**,** this means that weeklong trials could require months or even years to analyze.

Third, researchers seeking to gather information beyond basic spatiotemporal data in social experiments face limited options. This is because current multi-animal tracking tools cannot infer subject's actions directly. Consequently, researchers must externalize higher granularity analyses via various post-hoc tools [10, 11]. This decreases the overall practicality of the whole process by introducing manual steps and forcing offline analysis.

Here, we introduce *PrecisionTrack*, a novel multi-animal pose-tracking and action recognition tool designed for studying social dynamics in naturalized settings. *PrecisionTrack* achieves spatiotemporal accuracy comparable to both *SLEAP* and *DeepLabCut* (**Figure 4d**), while addressing the field's three major limitations. First, we address the subject re-identification problem via a marker-based approach. More specifically, we improved the *Tailtag* system [44], shown to be non-behavior altering and reliable day and night for at least a week. This new data-driven version is faster and more robust (**Figure 5d**). When paired with our data-driven *Tailtag* system, *DeepLabCut*, *SLEAP*, and *PrecisionTrack* all maintained their initial tracking accuracies (**Figure 5b**). This means that researchers can now prolong and complexify their experiments without experiencing unrecoverable tracking errors. Second, *PrecisionTrack* offers online, real-time responsiveness (**Figure 4b**), allowing researchers to access results as they are gathered, even during large-scale, complex trials. Third, we introduce the Multi-animal Action Recognition Transformer (*MART*). *MART* enables pose-tracking systems like *PrecisionTrack* to infer behavior automatically, extending analysis beyond spatiotemporal localization. It supports real-time and online action recognition, facilitating analyses of longitudinal interactions and group dynamics without manual intervention.

PrecisionTrack follows the tracking-by-detection paradigm, meaning that it relies on a static detection algorithm to localize subjects at each timestep. We argue that the quality of this localization is the primary bottleneck for tracking accuracy. Thus, we first focused on optimizing this step. Our strategy leveraged two key observations: 1) the object-detection field evolves rapidly, with frequent innovations, 2) the *YOLO-Pose* method extends object detectors directly instead of relying on the traditional top-down and bottom-up methods to perform pose-estimation [21]. Accordingly, we built our classification and detection algorithm by incorporating state-of-the-art object-detection techniques [22, 25] and a state-of-the-art backbone [25]. Furthermore, we adopted *YOLO-Pose*'s approach to solve the multi-subject pose estimation problem inline with object detection. This integration allows our pose-estimation step to directly benefit from the novel state-of-the-art techniques used by our detection algorithm. As a result, we improved *YOLOX*-pose's performance on three comprehensive benchmarks (**Figure 2f**) without increasing latency (**Figure 2d**).

Multi-subject tracking algorithm performances are typically measured by two main metrics. The MOTA (**Equation 21**) and the IDF1 (**Equation 22**). While both infer valuable insights upon the quality of the tracking algorithms, they do not measure exactly the same things. Indeed, the MOTA put more emphasis on the quality of the detections, while the IDF1 is instead directly measuring the ability of the algorithm to not switch trajectories. As such, the MOTA is more of a proxy for the quality of the detection step and the IDF1 is more representative of the accuracy of the assignation step. Since both *PrecisionTrack*'s assignment step and its

baseline use the same subject classification, localization, and pose-estimation model, they achieved similar MOTA scores (ByteTrack: 95.5%; *PrecisionTrack*: 95.6%) but different IDF1 scores (ByteTrack: 88.4%; *PrecisionTrack*: 94.3%). In contrast, *DeepLabCut* and *SLEAP* use different pose-estimation and assignment algorithms, *dlcrnet_ms5* and network flow optimization for *DeepLabCut*; *EfficientNetB0* and max-tracks *SORT* for *SLEAP*, resulting in different MOTA (*DeepLabCut*: 85.2%; SLEAP: 73.5%) and IDF1 (*DeepLabCut*: 84.9%; SLEAP: 72.2%) scores compared to *PrecisionTrack* (MOTA: 95.6%; IDF1: 94.3%).

To extract information beyond spatiotemporal localization, we propose two complementary methods. First, we extend *PrecisionTrack* with a feature extraction pipeline operating on the frozen feature map pyramid. This enables efficient retrieval of semantically rich embeddings directly from raw pixels (**Figure 3**). The intuition behind this method is the following: if the detection, classification and pose-estimation heads can accurately localize, estimate the pose and classify subjects directly from the feature map pyramid, then it must be encoding semantically rich spatial information about the subjects. Therefore, enabling downstream analyses to directly access these representations could lead to performance increases. This was later supported by our embedding richness ablation study (**Figure 6c**). Second, we introduce the Multi-animal Action Recognition Transformer (*MART*). *MART* enables multi-animal pose-tracking algorithms to perform automatic action recognition and social behavioral analysis, without degrading responsiveness. *MART* uses causal self-attention, similar to GPT-2 [45], but processes subject representations from past and present frames instead of tokenized sentences. This design allows *MART* to attend to the subject's current and past states within a defined context window when predicting ongoing behaviors. Our context-window ablation shows that being able to attend previous frames yields the largest relative performance gain across all actions (**Figure 6b**). Performance declines for windows spanning 60 frames (~2 s), likely due to noise from older actions. Additionally, *MART*'s final-layer representations cluster both labeled and unlabeled behaviors ("Other") (**Figure 6d**), suggesting that *MART* could operate in a self-supervised mode to discover behavioral patterns without manual labeling.

In this work, we presented practical solutions and a comprehensive method, *PrecisionTrack*, addressing the challenges that prevent the ethology field from moving beyond traditional manual annotation methods. While we do not claim to have fully solved these obstacles, our approach mitigates the three main current limitations and provides researchers with a method tailored to behavioral studies in real-world scenarios. Nonetheless, our approach still has limitations. First, the *Tailtag* system, while effective, requires daily inspection to ensure the reattachment of lost tags. Future work could build on its design to improve robustness. Second, both the pose-tracking and action recognition pipelines are fully supervised, requiring researchers to invest time and resources in data labeling. Although

transfer and active learning can reduce this burden, the development of self-supervised alternatives could significantly improve the overall practicality of the system. In conclusion, we believe that multi-animal tracking tools are on the verge of becoming reliable and practical enough to enable automatic, large-scale ethological behavioral studies, with *PrecisionTrack* representing a step in that direction.

Methods

MICE datasets and benchmark

Our pose-estimation dataset consists of over 1,500 images (1536x1536 resolution) featuring 20 mice. The images are from ethological experiments conducted in the laboratory of B. Labonté at CERVO in 2022. Each localization and skeleton (**Figure 2a**) were manually labelled internally. This dataset follows Microsoft's popular COCO formatting standard. This dataset was used to train *PrecisionTrack*, DeepLabCut and *SLEAP* before evaluating their respective performances on our benchmark. All animals' procedures followed the Canadian Guide for the Care and Use of Laboratory Animals and were approved by the Animal Protection Committee of Université Laval.

In addition to our pose-estimation dataset, we also compiled a sequential dataset comprising 35 minutes (30/5 train/val split) of recording at 28 Hz for a total 58,800 frames (2720x2720 resolution), which correspond to approximately a million localizations, poses, and actions of socially interacting mice in an enriched open field. The labelling of mice localizations follows the MOT standard, with each instance having a bounding boxe, keypoints and an action annotation. The bounding boxes are defined by ρ, while the keypoints are represented by 8 (x, y) coordinates corresponding to the snout, right ear, right leg, left ear, centroid, tail base, and tail tag, respectively (**Figure 2**a). A total of five distinct actions were labelled (Grooming, Running, Interacting, Rearing and Other), we wanted to also support Fighting, but we could not find enough fighting instances in our recordings to gather an interesting dataset. This dataset was used to train and evaluate our Multi-animal Action Recognition Transformer (*MART*; **Figure 6c**).

$$\rho = [x1, y1, w, h] \tag{1}$$

To complement these datasets, we provide a manually labelled evaluation benchmark, consisting of 20 interacting mice viewed from top-down. Here, we leveraged this benchmark to evaluate the tracking performances of *PrecisionTrack*, DeepLabCut and SLEAP on challenging, ethological settings (**Figure 4d**). These recordings were captured during additional ethological experiments conducted in B. Labonté's laboratory at CERVO, with a frame rate of 28Hz and a resolution of 1536x1536 pixels.

Detection step

PrecisionTrack follows the Tracking-By-Detection (TBD) paradigm, composed of two main autoregressive steps: 1) Detecting instances 2) Assigning current detections to memorized trajectories [38]. As such, *PrecisionTrack*'s first step, referred to as the detection step, focusses on localizing, classifying and estimating the pose of the subjects by processing raw RBG images. Besides some pre and post processing, our detection step is end-to-end, meaning it is solely composed of optimizable parameters, maximizing *PrecisionTrack*'s generability and transferability. These optimizable parameters formed our efficient multi-task CNN architecture inspired by [25, 21].

Neural network architecture

PrecisionTrack's neural network architecture integrates recent advances in real-time object detection to balance computational efficiency and accuracy. First, it leverages the *CSPNeXt* architecture (**Figure 2e**). Introduced by [25], *CSPNeXt* is designed to maximize receptive field without significantly increasing the network's floating-point operation (FLOP) count or its latency. As such, *CSPNeXt* relies on large convolution kernels (5x5) to expand the model's effective receptive field, capturing broader spatial context beneficial for dense prediction tasks like object detection and pose estimation at the cost of an increase computational demand relative to narrower convolution kernels (3x3). To counterbalance this increase, it also leverages depthwise separable convolutions (**Figure 2e**). However, since depthwise separable convolutions are composed of a sequence of two (depthwise and pointwise) convolution operations, blindly switching vanilla convolution layers for depthwise separable convolution layers in any given network will effectively double its depth, potentially impeding its parallelization potential. Therefore, the backbone architecture was adjusted by moderately increasing the filter count in each layer, thus widening the network and reducing the total number of layers from 9 to 6. This configuration yields an optimal computation-accuracy trade-off by decreasing latency without compromising detection performance [25]. Lastly, to efficiently focus on the relevant features extracted by the newly widened layers, a Channel Attention mechanism is added at the end of each of them (**Figure** 2e), this has been shown to lead to an increase in performance [25].

Multi-subjects detection and classification

PrecisionTrack's anchor-free detection mechanism offers multiple benefits over its anchor-based alternative. Most notably, it enables direct association of detections and features extracted by the backbone [35]. Thereby enabling efficient downstream processing relying of detection's features such as action recognition. Leveraging three distinct anchor-free heads, *PrecisionTrack* independently regresses, relative to each prior P, bounding boxes coordinates relative to each prior ($\sqrt{\mathrm{P}}$, $\sqrt{\mathrm{P}}$, 4), detection scores ($\sqrt{\mathrm{P}}$, $\sqrt{\mathrm{P}}$, 1) and classification scores ($\sqrt{\mathrm{P}}$, $\sqrt{\mathrm{P}}$, C), where C represents the number of possible classes. These priors correspond to unique feature maps, of dimensions (20 × 20), (40 × 40) and (80 × 80), coming directly from our three-stage feature pyramid (**Figure 3a**). By ensuring that the feature maps are symmetrical, their width (W) equals their height (H), we can express the priors as $\sqrt{\mathrm{P}_\mathrm{f}} = \mathrm{H}_\mathrm{f} = \mathrm{W}_\mathrm{f}$. Thus, each stage of the feature pyramid processing 640x640 images produces 400, 1600, and 6400 priors, with strides of 32, 16, and 8, respectively. Resulting in a total of 8400 priors, or potential detection, per image. To mitigate the overlapping of true positive coming from this high number of potential detections, *PrecisionTrack* leverages the fast and configurable Non-Max Suppression (NMS) algorithm to filter out bounding boxes overlapping by more than 65%.

Multi-subjects pose-estimation

Pose estimation algorithms typically follow either a top-down or bottom-up approach. Top-down (or two-stage) methods are generally more accurate, as they first detect all relevant subjects in the image and then perform single-individual pose estimation for each detection. However, the computational cost of these methods scales linearly with the number of processed individuals, leading to variable runtime. This makes top-down pose-estimator unsuitable for scenarios like ours, where the goal is to track large groups of subjects simultaneously and in real-time. In contrast, bottom-up approaches offer constant runtime, but they tend to be less accurate because of their two-stage process: 1) detect all keypoints in a single pass, 2) group them into individuals through various post-processing steps, such as Part Affinity Fields (*PAF*s) [48]. To overcome these limitations, we leverage recent advancements in both object detection [22, 25] and pose estimation [21], extending our existing multi-subject detector to support accurate pose estimation while maintaining a constant runtime and without adding additional post-processing. By adding two more anchor-free heads of shapes ($\sqrt{\mathrm{P}}$, $\sqrt{\mathrm{P}}$, $2 \cdot \mathrm{K}$) (**Figure 2b**) and ($\sqrt{\mathrm{P}}$, $\sqrt{\mathrm{P}}$, K) respectively, *PrecisionTrack* is able to regress keypoints directly from priors, inherently linking them to individuals. This approach thus directly resolves the keypoint-to-individual problem, eliminating the need for additional post-processing. As a result, we extend our object

detector to support pose estimation while only marginally increasing the network's computational cost at runtime.

Subject detection, classification and pose-estimation training

During the training process, predictions for each stage of the feature pyramid are matched with ground truth through the SimOTA [22] dynamic label assignment strategy. Each decouple head has its own loss function to optimize: 1) The classification head optimizes the Quality Focal (QFL) [49] Loss (**Equation 3**), 2) the bounding box regression head optimizes the General Intersection over Union (GIoU) [50] loss (**Equation 20**), 3) the keypoint regression head optimizes the Object Keypoint Similarity (OKS) [17] loss (**Equation 2**), 4) both the detection and keypoints confidence regression heads optimize the Binary Cross Entropy (BCE) loss. As such, the final loss function is composed of a weighted sum of each decoupled head's loss, as defined by **Equation 4**.

$$OKS_{loss} = \frac{1}{K} \cdot \sum_{i=1}^{K} \frac{\exp\left(-\frac{d_i^2}{2 \cdot s_i^2 \cdot k_i^2}\right) \cdot p_i}{\sum_{j=1}^{K} p_j} \tag{2}$$

Where:

- K : Total number of keypoints.
- $d_i = \|gt_i - pr_i\|_2$
- k_i: Weight of the i-th keypoint.
- p_i: 1 if the i-th ground truth is present, else 0.

$$QFL_{loss} = -|y - \sigma|^{\beta} \cdot \big((1 - y)log(1 - \sigma) + ylog(\sigma)\big) \tag{3}$$

Where:

- σ: The model's prediction.
- y: The ground truth.

- β: Down-weighting rate's hyperparameter (set to 2).

$$Total\ Loss = 2 \cdot QFL_{loss} + 5 \cdot GIoU_{loss} + 30 \cdot OKS_{loss} + 1 \cdot BCE_{loss} + 1 \cdot BCE_{losss} \quad (4)$$

Our two-stage training process closely follows [22]'s. Its first stage uses strong data augmentations, Mosaic and MixUp (**Figure 2c**), to teach the model that relevant subjects could be anywhere and at any scale. This step is crucial for minimizing common biases found in human-taken images, where objects of interest are often centered, non-occluded, and fully in the frame. It has also been shown to minimize the risk of learning these biases rather than the actual discriminative features necessary to accurately localize relevant subjects in real-world conditions [25]. The second training stage leverages weaker geometric augmentations, such as cropping and flipping, fine-tuning the model for the specific task at hand.

We adopt a hyperparameter-free cosine learning rate schedule, preceded by a brief warm-up period. Given our limited computational resources, a single RTX 3090, we utilized the linear learning rate scaling approach proposed by [51]. This allowed us to effectively replicate the performance benchmarks of *YOLOX* [22] on the COCO dataset [17] while using a fraction of the computational power.

Detection and pose-estimation evaluation

An optimal multi-animal pose tracking algorithm's detection step should prioritize three core qualities: 1) high detection coverage, meaning detecting as many subjects as possible in each frame, 2) accurate pose-estimation on each of the detected subjects, and 3) low latency to support real-time tracking. While the standard COCO metrics [17] technically encompass these qualities, they can be less intuitive to understand compared to other widely used metrics. Therefore, we report additional, flexible metrics with greater interpretability.

Specifically, we evaluated our model based on a weighted sum between the detection's F1 Score and the detected subject's Object Keypoint Similarity (OKS) metric (**Equation 13**). To closely reflect the inference scenario in our testing pipeline, our predicted detections are matched to labels using *PrecisionTrack*'s IoU-based cost function (**Equation 14**) and the Hungarian Algorithm [52].

For pose-estimation accuracy, we prioritize OKS and PCK over RMSE (**Figure 2f**), commonly used in other methods [6]. As both OKS and PCK are normalized by subject size, reducing the impact of regression errors on larger subjects.

$$\beta \cdot F1\ Score + \ (1 - \beta) \cdot OKS \tag{5}$$

Where:

- $\beta = 0.5$

Feature extraction

As explained in the Multi-subjects detection and classification section, the backbone outputs 8,400 feature vectors, one per prior. separate heads directly regress these vectors to predict each subject's class, pose, and location (**Figure 3a**). Therefore, these vectors necessarily encode semantics allowing downstream tasks to accurately locate, classify, and estimate poses. Consequently, these features could be interpreted as processed representations of each subject's appearance and offer rich inputs for other downstream tasks such as action recognition, enabling an efficient analysis pipeline.

An unintuitive finding has been that our one-stage, anchor-free and FPN-based object detector encoded uncoherent feature maps with respect to the inferred classes, localizations and poses. Specifically, features associated with highly overlapping bounding boxes, e.g., redundant localizations of the same entity, remained distant in the feature space (**Figure 3b**). This mismatch between detection and latent-space representation means that features linked to the same entity can drastically vary across timesteps, inevitably impacting the performances of sequential downstream tasks.

To solve this incoherence problem, we introduce a new convolution-based head trained to cluster relevant features by optimizing a triplet loss with easy and semi-hard mining. During inference, the resulting clustered features are normalized and aggregated via a weighted pooling mechanism taking into consideration each feature's associated confidence score. This process stabilizes each timestep's features, producing normalized and coherent feature maps (**Figure 3b**). Because the newly introduced head's clustering objective does not directly align with the network's other head's objectives, e.g., detection, classification and pose-estimation, we propose to replace the standard multi-task JDE approach popularized by [34, 35] by a curriculum-based strategy. By first training the network as previously described in the *Multi-subjects detection and classification and Multi-subjects pose-estimation* sections, and then training the clustering head separately, on frozen embeddings, *PrecisionTrack* enhances and stabilizes its feature extraction ability without impacting its accuracy.

$$Triplet\ loss = \sum_{i=1}^{t} max\ (\|f(a_i) - f(p_i)\|_2^2 - \|f(a_i) - f(n_i)\|_2^2 + \alpha, 0) \quad (6)$$

Where*:*

- a: The anchor, the reference point.
- p: The potitive, a point with the same identity as the anchor.
- n: The negative, a point with a different identity than the anchor.
- $\alpha = 0.2$

Association step

The association phase, which follows detection in the TBD pipeline, aims at pairing new detections to their respective live trajectories. Although conceptually simple, the task becomes combinatorially challenging whenever the detection count diverges from the number of active tracks. Following a recent trend in the Multi-Object Tracking (MOT) field [40], *PrecisionTrack*'s association step prioritizes a high recall approach. Therefore, while previous multi-animal pose tracking algorithms opted for systematically filtering out low-confidence detections before the association process [5, 6], *PrecisionTrack* salvages them in a second association step designed to perform low-confidence assignation. This method has been shown to yield more robust tracking performances in scenarios where the visible entities are partially occluded, such as within crowds or in complex environments [40].

The projection substep

Most current state-of-the-art TBD algorithms split the association step into two substeps: the projection substep and the matching substep. In the projection substep, motion estimation models are leveraged to project the tracked trajectories in the current frame, facilitating eventual matching [38]. Among these projection models, the Kalman Filter [42] remains the preferred choice due to its efficiency, reliability, and low computational cost. Especially in the multi-object tracking (MOT) community, whose benchmarks mainly include pedestrians and vehicles. The Kalman Filter's assumptions of linear motion and constant velocity align well with the typical movement patterns of these objects, such as pedestrians walking along sidewalks or vehicles following lanes.

However, these assumptions do not hold as well in multi-animal pose tracking where tracked subjects are often freely roaming in an open field while being recorded from a top-down view (**Figure 4c**). This could lead to sudden changes in the subject's direction and velocities (**Figure 4c**). As such, we argue that this significant shift in data dynamics imposes a corresponding shift in the leveraged motion projection techniques. To bridge this domain gap, we propose a simple, yet effective motion algorithm called the Dynamic Kalman Filter (DKF), which incorporates 2D Newton's laws of motion into the vanilla Kalman filter to more accurately project trajectory dynamics over time, even in cases of abrupt 2D changes in both direction and velocity (**Figure** 4c).

The DKF updates each trajectory's dynamics after each association step using the following:

$$v_{t+1} = \frac{x_{t+1} - x_t}{\Delta t} \tag{7}$$

Where:

- v: The trajectory velocity
- x: The trajectory position

For numerical stability, given the frequent directional changes observed in animals, an exponential moving average (EMA) is then applied to the velocity estimate:

$$v_{t+1} = \alpha v_t + (1 - \alpha) v_t \tag{8}$$

This smoothed velocity estimate is then used to infer the trajectory's accelerations using Newton's second law of motion:

$$a_{t+1} = \frac{v_{t+1} - v_t}{\Delta t} \tag{9}$$

Where:

- a: The trajectory acceleration

To further accommodate the complexities of animal tracking, we also adjusted the Kalman Filter's noise estimation. We set the initial state of the covariance matrix P to 1000 times the identity matrix, representing high uncertainty in the absence of prior knowledge about the trajectory's states. P is updated at every time step using **Equation 12**. The noise covariance Q is scaled by 0.1 and dynamically adjusted based on the time step Δt. This models the position noise growing quadratically and the velocity noise linearly in respect to eventual delays between measurements, allowing us to account for potential broken trajectories. The measurement H and measurement noise covariance R are both set to diagonal matrices with values of 1.0, assuming equal noise in both the x- and y-axes.

$$S = H \cdot P \cdot H^T + R \tag{10}$$
$$K = P \cdot H^T \cdot S^{-1} \tag{11}$$
$$P = (I - K \cdot H) \cdot P \tag{12}$$

The matching substep

The matching substep involves linking the projected trajectories to new detections. Typically, matching costs are defined by the Intersection over Union (IoU) (**Equation 18**) between the bounding boxes of detections. However, we argue that relying solely on bounding boxes overlooks critical information that could lead to tracking errors. For instance, two trajectories may overlap and exhibit similar IoU scores with newly acquired detections; however, the two detected subjects might not even be facing the same direction. As such, incorporating additional information, such as the pose of the subjects, can enhance the robustness of the association step. Thus, we propose to complement the standard IoU scores with our slightly modified version of the Object Keypoint Similarity (OKS) loss function [17]. Our implementation of OKS normalizes the Euclidean distances (**Equation 13**) by the trajectory's areas *A*. The exponential term in the OKS calculation is also multiplied by the weight of each detection *w*.

$$Association\ OKS_{p,t} = \frac{1}{K}\sum_{i=1}^{K} w_i \cdot \exp(-\frac{d_i}{\sqrt{A_i}}) \tag{13}$$

Where:

- $d_i = \|p_i - t_i\|_2$
- $A_i = h_i \cdot w_i$

By combining a weighted sum of the IoU and OKS metrics with a lambda of 0.5, we achieved a richer matching cost while retaining the advantages of the IoU metric, such as the ability to easily identify non-overlapping trajectory-detection pairs.

$$C = \lambda \cdot IoU + (1 - \lambda) \cdot Association\ OKS \tag{14}$$

Trajectory stitching

Trajectory stitching is an optional plugin for *PrecisionTrack* tracking pipeline. It allows the reconstruction of sparse trajectories for classes of subjects confined within the camera's field of view. For example, if a user wishes to track *n* marmosets (*Callithrix jacchus*) in their home-cage while also counting the number of new researchers entering the camera's field of view, they can configure the stitching plugin to only consider the *n* marmosets. This ensures that whenever a new marmoset is detected, e.g., emerging from a hideout,

PrecisionTrack will link it to a loss marmoset trajectory instead of considering it as a new trajectory, preventing an incorrect increment of the marmoset count.

While offline methods such as DeepLabCut, by processing whole recordings as once, can leverage clever techniques such as network flow minimization algorithm to solve this problem, online methods like *PrecisionTrack*, since it does not have access to global trajectories, must instead solely rely on trajectory's past information to eventually stitch them on the fly.

Working within this constraint, we developed a fast and flexible heuristic algorithm called Search-Based Stitching (SBS) that delivers strong practical performance. SBS defines a dynamically evolving search zone around lost trajectories' last known location. These search zones expand over time according to the lost trajectories last recorded dynamics (**Figure 4b**). To minimize the risk of excessively expanding the search zone, which could lead to a higher rate of false-positive matches, the rate of expansion decays at an exponential rate (**Equation 15**).

$$R_{t+1} = R_t \cdot e^{-\beta x} \tag{15}$$

Where:

- R: The search zone expansion
- β: Decay parameter, default to 0.1

Additionally, the search zones expand asymmetrically, guided by each side's cosine similarity with the lost trajectory's last recorded EMA 2D velocities (**Equation** 16).

$$cos\theta = \frac{sides \cdot velocity}{\|sides\|_2 \cdot \|velocity\|_2} \tag{16}$$

$$Cosines = \frac{cos\theta + 1}{2} \tag{17}$$

This biases the search towards the area the lost tracklet is most likely heading for (**Figure 4b**). Lastly, to account for the frequent use cases where multiple search zones are active and/or multiple new trajectories are detected at the same time, we cast the search zone-to-new trajectory linking as an assignment problem, which we solve using the Hungarian algorithm [52]. The cost matrix for the association is based on our implementation of a normalized

Generalized Intersection over Union (GIoU) (**Equation 19**) between each active search zone and each new relevant detection.

$$IoU(A,B) = \frac{Area(A \cap B)}{Area(A) + Area(B) - Area(A \cap B) + \epsilon} \tag{18}$$

Where:

- $A = [acx - \frac{aw}{2}, acy - \frac{ah}{2}, acx + \frac{aw}{2}, acy + \frac{ah}{2}]$
- $B = [bcx - \frac{bw}{2}, bcy - \frac{bh}{2}, bcx + \frac{bw}{2}, bcy + \frac{bh}{2}]$

$$GIoU(A,B) = IoU(A,B) - \frac{Area(enclosing\ box) - \cup\ (A,B)}{Area(enclosing\ box) +\ \epsilon} \tag{19}$$

Where:

- $enclosing\ box = \min(A_1, B_1), \max\ (A_2, B_2)$

$$normalized\ GIoU = \frac{GIoU(A,B) + 1}{2} \tag{20}$$

Tailtag re-identification

While the *Tailtag* system is shown to be effective and non-behavioural altering [44], it still has some weaknesses. Mainly; 1) the detection process, while robust, is not impervious to mistakes, and 2) searching for tags on the whole frame is computationally expensive. In fact, leveraging OpenCV's ArUco detection functionalities as is would bottleneck *PrecisionTrack*'s latency and harm its ability to run at real-time speed. Therefore, we opted to refine [44]'s approach in two ways. First, leveraging the fact that we are already tracking the tags' locations via our pose-estimation pipeline (**Figure 3a**), we were able to significantly reduce the marker search area from the whole frame to only regions around the subject's tags (**Figure 5c**). Additionally, we built a timeout priority queue, ensuring that the most uncertain trajectories have re-identification priority while also ensuring that the *Tailtag* re-

identification will not bottleneck the whole system's latency via a timeout mechanism. Second, the system perpetually accumulates evidence in the form of detections on each of the subject's tag areas. Tags are assigned to subjects only when confidence thresholds are met. This assignment could lead to what we define as a correction, meaning that a subject (given enough accumulated evidence) could get assigned to another one's tag. In this scenario, the *Tailtag* re-identification system declares a correction and the subject's identities are corrected accordingly. Paired with the *Tailtag* re-identification plugin, *PrecisionTrack* maintains reliable tracking performances over extended periods (**Figure 5b**), while remaining computationally efficient enough to remain real-time (**Figure 4d**).

Tracking evaluation

Tracking performances are assessed using the CLEAR metrics [53], which have been the standard for multi-object tracking (MOT) evaluations for over a decade. Specifically, we employ the Multiple Object Tracking Accuracy (MOTA) metric (**Equation 30**), which quantifies three main types of errors that can occur during tracking:

1. False Positives (*FP*): Instances where the algorithm incorrectly tracks irrelevant objects.
2. False Negatives (*FN*): Instances where the algorithm fails to track relevant objects.
3. Identity Switches (*IDSW*): Instances where the algorithm misidentifies two trajectories, resulting in swapped identities.

As such, MOTA focuses on assessing the overall quality of the tracking algorithm:

$$MOTA = 1 - \frac{\sum_{t=1}^{T}(FP_t + FN_t + IDSW_t)}{\sum_{t=1}^{T} O_t} \tag{21}$$

Where:

- FP_t: Total number of false positives at time t.
- FN_t: Total number of false negatives at time t.
- $IDSW_t$: Total number of identity switches at time t.
- O_t: Total number of visible objects at time t.
- T: Total number frames.

Our evaluation pipeline also incorporates the IDF1 score (**Equation 31**), which focuses on evaluating the algorithm's ability to maintain the correct identities of tracked objects over time:

$$IDF1 = \frac{2 \cdot IDPrecision \cdot IDRecall}{IDPrecision + IDRecall} \tag{22}$$

Action recognition

The Multi-animal Action Recognition Transformer (*MART*) is designed to accommodate the naturally dynamic tendency of actions. Modern transformer–based natural language processing methods achieves state-of-the-art performance by selectively reasoning over linear sequences [45, 46]. By repurposing transformer decoder blocks to process subjects' states instead of tokenized words, *MART* effectively captures the dynamic nature of actions. While traditional action recognition challenges focusses on sparsely predicting the actions of a single actor [54, 55], *MART* aims at simultaneously predicting the actions of large groups of individuals with frame-level temporal granularity. This focus shift introduces new challenges: 1) the inability to isolate specific actions leads to class imbalance, because some actions naturally occur more often than others, 2) the model must scale with increases in both context length and number of subjects, and 3) considering only subjects' poses may provide inadequate context, as it omits information about appearance and surroundings that may be required to accurately infer actions in social settings.

We address the first problem by labeling undesired and overrepresented actions as a single "Other" class. During training, we enforce class uniformity by randomly selecting an action and then sampling a sequence containing that action. This procedure mitigates the overconfidence typically observed when training on imbalanced data. Our best accuracy–latency trade-off was achieved with four decoder blocks, each featuring four attention heads and a block size of 30 (**Figure 6c**). Because no pretrained LLM is available at this scale, we trained the network from scratch. Notably, we obtained strong performances (**Figure 6c**) without the next-token prediction pretraining phase usually required for language models.

We address the second problem by fully parallelizing the subjects' dimension on modern hardware. Since Transformer blocks are optimized for GPU execution, we can concatenate each subject's input to build up a fifth tensor dimension. Consequently, *MART* scales efficiently across both temporal and subject dimensions while only marginally increasing

runtime (**Figure 6b**). The sole non-parallelizable component is the stack of decoder layers. Therefore, we limited it to four, achieving latency faster than real time on an RTX-3090.

The third problem was solved by leveraging our feature extraction pipeline, enabling an ablation study of alternative state-embedding representations. The most comprehensive embedding, composed of subject's poses, appearance features, and velocities, produced the best overall performance across classes (**Figure 6c**).

Computational Resource Usage

All experiments were conducted on a machine equipped with 32 GB of SSD storage, an Intel i7-10700k CPU, and an RTX-3090 GPU. The *PrecisionTrack* algorithm requires approximately 4 GB of RAM and 2.5 GB of VRAM for execution. Additionally, saving the tracking results incurs a disk space cost of about 7.3 MB per minute when monitoring 20 subjects.

Acknowledgements

BL holds a Sentinelle Nord Research Chair, is supported by the Canadian Institutes of Health Research (Grant No. PJT-451728 and PJT-451858), and the Natural Science and Engineering Research Council of Canada (Grant No. RGPIN-2019-06496) and receives Fonds de Recherche en Santé du Québec (FRQS) Junior-2 salary support. BG holds the Canada Research Chair in Smart Biomedical Microsystem and is supported by the Natural Science and Engineering Research Council of Canada (Grant No. RGPIN-2022-03984).

This work is made possible by the MMLab's open-source MMEngine [56], MMCV [57], MMdetection [58], MMdeploy [59] and MMPose [60] toolkits. We also thank Arturo Marroquin Rivera and Sam Mansouri for their comments on both the methods and the manuscript. As well as Maggy Shatov and David-Alexandre Roussel for refining the MICE dataset's annotations and providing ideas regarding the data-driven ArUco reading system respectively.

Author contributions

B.L. supervised the project and wrote the manuscript. BG reviewed the manuscript. VC developed the classification, subject detection and pose-estimation algorithm, developed *PrecisionTrack*, developed the data-driven *Tailtag* system, developed the *MART* algorithm, performed the analyses, made the figures and wrote the manuscript. MSM Labelled the MICE dataset and reviewed the manuscript. KA Labelled the MICE dataset and reviewed the manuscript. QL and MRP contributed to the animal experiments and video data generation. All authors contributed to the preparation of the manuscript.

Disclosures

The authors declare no competing financial interests.

[1] Ishii, K. K. et al. A labeled-line neural circuit for pheromone-mediated sexual behaviors in mice. Neuron 95, 123–137. https://doi.org/10.1016/j.neuron.2017.05.038 (2017).

[2] Seeholzer L. F. et al. Evolution of a central neural circuit underlies Drosophila mate Preferences. Nature 559, 564–569. https://doi.org/10.1038/s41586-018-0322-9 (2018).

[3] Yamashita, J. et al. Male-predominant galanin mediates androgen-dependent aggressive chases in medaka. Elife 9, e59470. https://doi.org/10.7554/eLife.59470 (2020).

[4] Button, K.S. et al. Power failure: why small sample size undermines the reliability of neuroscience Nat. Rev. Neurosci. 2013; 14:365-376

[5] Talmo D Pereira. et al. SLEAP: A deep learning system for multi-animal pose tracking. In: Nature Methods 19 (2022), pp. 486–495. doi: https://doi.org/10.1038/s41592-022-01426-1. url: https://www.nature. com/articles/s41592-022-01426-1.

[6] Jessy Lauer. et al. Multi-animal pose estimation, identification and tracking with DeepLabCut. In: Nature Methods 19 (2022), pp. 496–504. doi: https://doi.org/10.1038/s41592-022-01443-0. url: https://www.nature.com/articles/s41592-022-01443-0.

[7] Marks M. et. Al. Deep-learning based identification, tracking, pose estimation, and behavior classification of interacting primates and mice in complex environments. In: Nature Machine Intelligence 4 (2022), pp. 331-340. doi: https://doi.org/10.1038/s42256-022-00477-5. url : https://www.nature.com/articles/s42256-022-00477-5

[8] Romero-Ferrero, F. et al. idtracker.ai: tracking all individuals in small or large collectives of unmarked animals. Nat Methods 16, 179–182 (2019). https://doi.org/10.1038/s41592-018-0295-5

[9] James D. Crall et al. BEEtag: A Low-Cost, Image-Based Tracking System for the Study of Animal Behavior and Locomotion. In: PLOS ONE 10.9 (Sept. 2015), pp. 1–13. doi: 10.1371/journal.pone.0136487. url: https://doi.org/10.1371/journal.pone.0136487.

[10] Goodwin, N.L., Choong, J.J., Hwang, S. et al. Simple Behavioral Analysis (SimBA) as a platform for explainable machine learning in behavioral neuroscience. Nat Neurosci 27, 1411–1424 (2024). https://doi.org/10.1038/s41593-024-01649-9

[11] Weinreb, C., Pearl, J.E., Lin, S. et al. Keypoint-MoSeq: parsing behavior by linking point tracking to pose dynamics. Nat Methods 21, 1329–1339 (2024). https://doi.org/10.1038/s41592-024-02318-2

[12] Paske, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems 32, 8024-8035 (2019)

[13] Bai, Junjie. et al. ONNX: Open Neural Network Exchange. (2019). url: https://github.com/onnx/onnx

[14] NVIDIA. TensorRT: High-Performance Deep Learning Inference SDK. (2024). url: https://developer.nvidia.com/tensorrt

[15] CVAT. Computer Vision Annotation Tool (CVAT). (2023). doi: 10.5281/zenodo.7863887. url: https://doi.org/10.5281/zenodo.7863887

[16] Maxim Tkachenko. et al. Label Studio: Data labeling software. (2020). url: https://github.com/HumanSignal/label-studio

[17] Justin Brooks. COCO Annotator: Web-based image segmentation tool for object detection, localization and keypoints. (2019). url: https://github.com/jsbroks/coco-annotator

[18] Panadeiro. et al. A review of 28 free animal-tracking software applications: current features and limitations. Lab Anim 50, 246–254 (2021). https://doi.org/10.1038/s41684-021-00811-1

[19] Tsung-Yi Lin. et al. Microsoft COCO: Common Objects in Context. (2015). url: https://arxiv.org/abs/1405.0312

[20] Yu, Hang. et al. AP-10K: A Benchmark for Animal Pose Estimation in the Wild. (2021). url: GitHub - AlexTheBad/AP-10K: NeurIPS 2021 Datasets and Benchmarks Track

[21] Debapriya Maji. et al. YOLO-Pose: Enhancing YOLO for Multi Person Pose Estima- tion Using Object Keypoint Similarity Loss. 2022. arXiv: 2204.06806 [cs.CV]. URL: https://arxiv.org/abs/2204.06806.

[22] Zheng Ge et al. YOLOX: Exceeding YOLO Series in 2021. 2021. arXiv: 2107.08430 [cs.CV]. URL: https://arxiv.org/abs/2107.08430.

[23] Ruder, Sebastian. An overview of gradient descent optimization algorithms. (2016). url: [1609.04747] An overview of gradient descent optimization algorithms

[24] Ilya Loshchilov. et al. Decoupled Weight Decay Regularization. (2019). url: [1711.05101] Decoupled Weight Decay Regularization

[25] Chengqi Lyu. et al. RTMDet: An Empirical Study of Designing Real-Time Object Detectors. (2022). url: [2212.07784] RTMDet: An Empirical Study of Designing Real-Time Object Detectors

[26] Guo, C. et al. On calibration of modern neural networks. In International conference on machine learning, pp. 1321–1330. PMLR, 2017.

[27] Naeini, Mahdi. et al. Obtaining well calibrated probabili ties using bayesian binning. In AAAI, pp. 2901, 2015.

[28] Jiang,S. (2023). Dimensionality reduction techniques for high dimensional data: State of the art. Applied and Computational Engineering,13,37-45.

[29] Alonso, María & Malpica, José & Martinez-Agirre, Alex. (2011). Consequences of the Hughes phenomenon on some classification Techniques. American Society for Photogrammetry and Remote Sensing Annual Conference 2011.

[30] Krichen, Moez. Convolutional Neural Networks: A Survey. Computers, 12-151 (2023). url: https://www.mdpi.com/2073-431X/12/8/151

[31] Tasnim Shahriar. Comparative Analysis of Lightweight Deep Learning Models for Memory-Constrained Devices. (2025). url: https://arxiv.org/abs/2505.03303

[32] Sara Bouraya. et al. Deep Learning based Neck Models for Object Detection: A Review and a Benchmarking Study. The Science and Information Organization 12-11 (2021). url: http://dx.doi.org/10.14569/IJACSA.2021.0121119

[33] Alexander Hermans. et al. In Defense of the Triplet Loss for Person Re-Identification. (2017). url: https://arxiv.org/abs/1703.07737

[34] Zhongdao Wang. et al. Towards Real-Time Multi-Object Tracking. (2020). url: https://arxiv.org/abs/1909.12605

[35] Zhang. et al., FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object Tracking. IJCV (2021). url: [2004.01888] FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object Tracking

[36] Emirhan Bayar. et al. When to Extract ReID Features: A Selective Approach for Improved Multiple Object Tracking. (2024). url: https://arxiv.org/abs/2409.06617

[37] Bewley, Alex. et al. Simple online and realtime tracking. 2016 IEEE International Conference on Image Processing (ICIP), 3464-3468, (2016). doi: 10.1109/ICIP.2016.7533003

[38] Wojke, Nicolai. et al. Simple Online and Realtime Tracking with a Deep Association Metric. 2017 IEEE International Conference on Image Processing (ICIP), 3645-3649, (2017). doi: 10.1109/ICIP.2017.8296962

[39] Momir Adžemović. et al. Deep Learning-Based Multi-Object Tracking: A Comprehensive Survey from Foundations to State-of-the-Art. (2025). url: [2506.13457] Deep Learning-Based Multi-Object Tracking: A Comprehensive Survey from Foundations to State-of-the-Art

[40] Zhang, Yifu. et al. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. Proceedings of the European Conference on Computer Vision (ECCV), (2022). url: [2110.06864] ByteTrack: Multi-Object Tracking by Associating Every Detection Box

[41] Laura Leal-Taixé et al. MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking. 2015. arXiv: 1504.01942 [cs.CV]. URL: https://arxiv.org/abs/1504. 01942.

[42] R. E. Kalman. “A New Approach to Linear Filtering and Prediction Prob lems”. In: Journal of Basic Engineering 82.1 (Mar. 1960), pp. 35–45. issn: 0021-9223. doi: 10.1115/1.3662552. eprint: https://asmedigitalcollection. asme.org/fluidsengineering/article-pdf/82/1/35/5518977/35\ _1.pdf. url: https://doi.org/10.1115/1.3662552.

[43] Yuxiang Yang. et al. APTv2: Benchmarking Animal Pose Estimation and Tracking with a Large-scale Dataset and Beyond. (2023). url: [2312.15612] APTv2: Benchmarking Animal Pose Estimation and Tracking with a Large-scale Dataset and Beyond

[44] Coulombe, V. et al. The Tailtag: A multi-mouse tracking system to measure social dynamics in complex environments. Neuropsychopharmacol. 50, 1336–1345 (2025). https://doi.org/10.1038/s41386-025-02126-y

[45] Radford, A. et al. Language Models are Unsupervised Multitask Learners. OpenAI technical report. (2019) . url: Language Models are Unsupervised Multitask Learners

[46] Zhang, H. et al. Survey of transformers and towards ensemble learning using transformers for natural language processing. Journal of Big Data, 11(25), 1-29 (2024).

[47] Lukáš Adam. et al. WildlifeReID-10k: Wildlife re-identification dataset with 10k individual animals. (2025). url: https://arxiv.org/abs/2406.09211

[48] Zhe Cao. et al. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. 2017. arXiv: 1611.08050 [cs.CV]. url: https://arxiv. org/abs/1611.08050.

[49] Xiang Li. et al. Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection. 2020. arXiv: 2006.04388 [cs.CV]. url: https://arxiv.org/abs/2006.04388.

[50] Gong Cheng. et al. A survey on object detection in optical remote sensing images. In: ISPRS Journal of Photogrammetry and Re mote Sensing 117 (July 2016), pp. 11–28. issn: 0924-2716. doi: 10.1016/ j.isprsjprs.2016.03.014. url: http://dx.doi.org/10.1016/j. isprsjprs.2016.03.014.

[51] Priya Goyal. et al. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. In: arXiv preprint arXiv:1706.02677 (2017).

[52] H. W. Kuhn. The Hungarian method for the assignment problem. In: Naval Research Logistics Quarterly 2 (1955), pp. 83–97. doi: https:// doi.org/10.1002/nav.3800020109.

[53] Keni Bernardin. et al. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. In: EURASIP Jour nal on Image and Video Processing 2008.1 (2008), p. 246309. issn: 1687 5281. doi: 10.1155/2008/246309. url: https://doi.org/10.1155/ 2008/246309.

[54] Ng, Xun Long. et al. Animal Kingdom: A Large and Diverse Dataset for Animal Behavior Understanding. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), (2022).

[55] Will Kay. et al. The Kinetics Human Action Video Dataset. (2017). url: https://arxiv.org/abs/1705.06950.

[56] MMEngine Contributors. MMEngine: OpenMMLab Foundational Library for Training Deep Learning Models. (2022).

[57] MMCV Contributors. MMCV: OpenMMLab Computer Vision Foundation. (2018).

[58] Chen, Kai. et al. MMDetection: Open MMLab Detection Toolbox and Benchmark. (2019).

[59] MMPose Contributors. OpenMMLab Pose Estimation Toolbox and Benchmark. (2020).

[60] MMDeploy Contributors. OpenMMLab's Model Deployment Toolbox. (2021).