Puzzle: taking livestock tracking to the next level

Jehan-Antoine Vayssade, Mathieu Bonneau

scientific reports

OPEN

Puzzle: taking livestock tracking to the next level

Jehan-Antoine Vayssade & Mathieu Bonneau✉

Animal behavior is a critical aspect for a better understanding and management of animal health and welfare. The combination of cameras with artificial intelligence holds significant potential, particularly as it eliminates the need to handle animals and allows for the simultaneous measurement of various traits, including activity, space utilization, and inter-individual distance. The primary challenge in using these techniques is dealing with the individualization of data, known as the multiple object tracking problem in computer science. In this article, we propose an original solution called "Puzzle." Similar to solving a puzzle, where you start with the border pieces that are easy to position, our approach involves commencing with video sequences where tracking is straightforward. This initial phase aims to train a Convolutional Neural Network (CNN) capable of deriving the appearance clues of each animal. The CNN is then used on the entire video, together with distance-based metrics, in order to associate detections and animal id. We illustrated our method in the context of outdoor goat tracking, achieving a high percentage of good tracking, exceeding 90%. We discussed the impact of different criteria used for animal ID association, considering whether they are based solely on location, appearance, or a combination of both. Our findings indicate that, by adopting the puzzle paradigm and tailoring the appearance CNN to the specific video, relying solely on appearance can yield satisfactory results. Finally, we explored the influence of tracking efficacy on two behavioral studies, estimating space utilization and activity. The results demonstrated that the estimation error remained below 10%. The code is entirely open-source and extensively documented. Additionally, it is linked to a data-paper to facilitate the training of any automatic detection algorithm for goats, with the goal of fostering open access within the deep-learning livestock community.

Keywords Computer vision, Tracking, Monitoring, Monitoring, Goats

Animal behavior has a rich research history[1] and is receiving increased attention in recent years within livestock farming. Behavior often serves as an indicator of an animal's physiological state, enabling the detection or prediction of health and welfare issues[2]. Behavioral changes can be utilized for rapid and automated health problem detection, aiming to reduce reliance on chemical treatments and enhance their effectiveness, as seen in anthelmintic applications[3], with the hope of sustainably mitigating the concerning rise of antimicrobial resistance[4].

Behavior provides information on animal welfare, an escalating economic issue, with consumers expressing widespread concern about the housing conditions used in many animal production systems[5–7]. On the other hand, behavior is also a means of adaptation and could be considered as a management option. For example, behavior serves as a way to avoid diseases through fecal avoidance, limiting parasite infestation[8]. Despite its assumed high plasticity, behavior is a heritable trait[9], making it a potential inclusion in breeding programs [e.g.,[10] for maternal attachment in sheep].

In this context, there is a growing need to develop automatic and accurate methods to monitor livestock behavior. The availability of sensors and the development of artificial intelligence techniques offer new perspectives in this domain[11], for an example] or[12,13], for reviews]. Image analysis, in particular, emerges as a promising tool[14], capable of measuring various traits without the need for direct animal handling or multiple sensors. Although numerous promising studies rely on computer vision for monitoring behavior, the main challenge remains deriving individual information for group-housed animals. In the field of computer science, this challenge is referred to multi-object tracking problem (MOT)[15]. The most common approach is called tracking by detection[16], it initially involves detecting animals in every video frame and subsequently associating these detections across consecutive frames to assign a unique ID to each animal in the flock. Imperfect detections, due to scene elements or occlusion, pose a significant challenge.

The association of detections between frames is based on a score to quantify the consistency of the associations. Two common metrics[17]include distance-based metrics, assuming closer detections from consecutive frames belong to the same animal, and appearance-based metrics, where appearance clues are extracted using

UR143 ASSET, INRAE, 97170 Petit-Bourg, Guadeloupe, France. ✉email: mathieu.bonneau@inrae.fr

a convolutional neural network (CNN). These metrics can be used individually or in combination, employing an expert-weighted sum for example. Transforming the detections association problem into a minimization problem to find the optimal association.

The method proposed in this article, named Puzzle, builds on the seminal work of[18], and follows the same principle. Similar to solving a puzzle, our approach involves starting with the sequence of the video where tracking is straightforward (i.e., all animals are detected without occlusion). These sequences, called *tracklets* are then sorted, based on their duration. The longest tracklet, called *best-track*, is then used to automatically trained a classification CNN, with one class per animal. This CNN is then used on the rest of the video to extract appearance features from the detections. Appearance features are then combined with location data to propagate animal IDs throughout the rest of the video. The major advantage lies in the quality of the appearance CNN, directly trained with a sequence from the video, and thus customized for the analyzed video, eliminating the need for constructing a robust CNN based on a large and diverse training set, which is still lacking for livestock. A graphical summary of the method is available in Fig. 1. The code is open source and accessible online with comprehensive documentation.

Compared to the seminal version, three main improvements have been implemented:

Firstly, a deep-learning approach is now employed to calculate the association score. Instead of relying on a weighted sum of metrics, we propose using a Convolutional Neural Network (CNN) to optimally combine location and appearance metrics. The CNN used to combine the different metrics will be denoted ASN, for Association Score Network.

Secondly, we introduce several enhanced versions of the CNN designed to extract appearance features more effectively.

Thirdly, there is a novel approach to training this CNN, allowing the algorithm to be run in a semi-supervised manner, leading to a significant enhancement in results.

The method is discussed in the context of monitoring goats that graze outdoors, a particularly challenging scenario primarily due to the side-view camera, leading to frequent occlusions. Additionally, outdoor conditions introduce challenges such as the presence of shade and scene complexity (e.g., other animals in the background, objects, or humans).

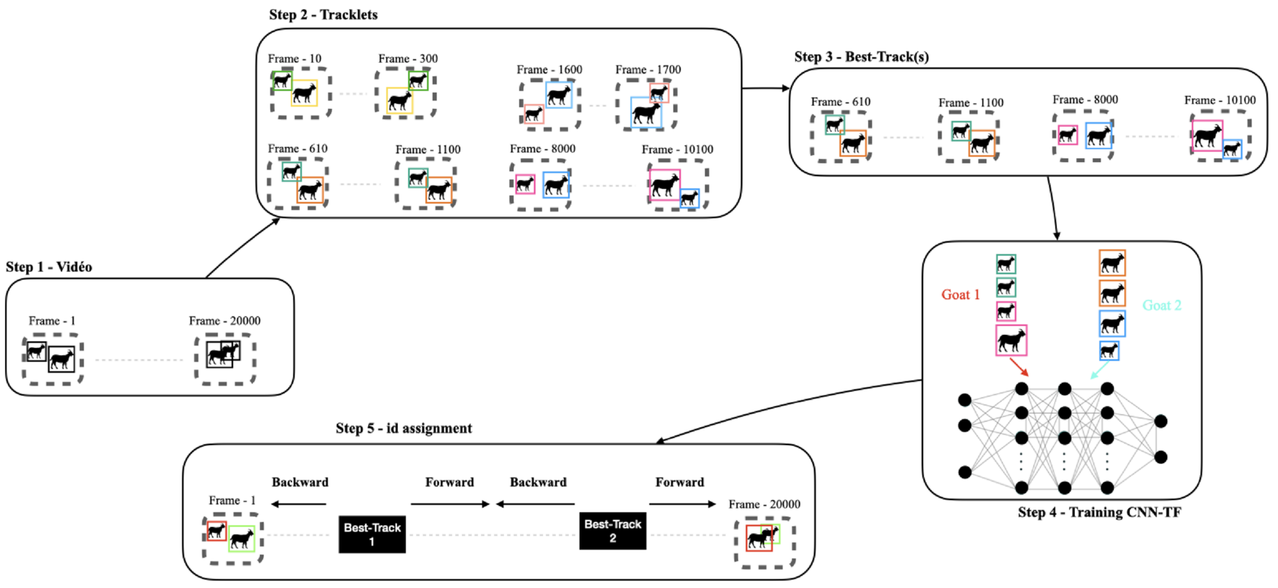

Figure 1. Graphical Abstract of the Puzzle Tracking Method: The initial phase involves implementing a detection method across the entire video. Following detections, each frame, represented by black-dotted rectangles, is subsequently linked to a set of detections, denoted as pixel coordinates framing the animals within rectangles-commonly referred to as bounding boxes or bboxes. The second phase entails identifying tracklets, which are sequences of frames wherein tracking is unambiguous, akin to the border pieces of a puzzle. This may encompass sequences where all animals are detected without overlap. The third phase consists in selecting one or several tracklets, that will be used to train the appearance CNN for this video. When no manual annotation is provided (i.e. fully unsupervised), only one best-track is used. Several best-tracks could be used when manual annotation is allowed (semi-supervised). In this example, two best-tracks are selected, and the id of each tracklet is provided manually. The fourth phase consists in training the appearance CNN, with one class per individual. The final phase consists in propagating the ids from the best-tracks, to all the detections of the video. This is executed through two types of passes: the backward pass, initiated at the beginning of the best-track, and the forward pass, originating from the end. Each pass involves iterative processes wherein known IDs from the animals in the preceding frame are propagated to the detections in the current frame. For video sequences between two best-tracks, the forward and backward passes end at the middle of the sequence.

Results

We tested several versions of the tracking method, as listed in Table 1, to explore the significance of: (i) the association score method, (ii) the CNN employed for deriving appearance clues, and (iii) the method used to train this CNN (either fully unsupervised or semi-supervised).

Comparison with other methods

The seminal method, Wizard[18], was compared with state-of-the art methods, named DeepSort[19], BoT-SORT[20] and ByteTrack[21]. Briefly, the methods were compared on 17 videos (4H23m52s) of goats grazing outdoor. The results indicated that Wizard achieved the lowest average number of IDs per animal, with 3.2, compared to 9.9 for DeepSort, 11 for BoT-SORT, and 10.8 for ByteTrack. Notably, all methods except Wizard were significantly affected by the frequency of occlusions in the videos.

As discussed later, Puzzle surpassed the latest version of Wizard. For the sake of clarity, we have not included a comparative analysis with other methods in this article. Additionally, Puzzle and state-of-the-art methods are not necessarily developed for the same objective and, as a consequence, are not easily comparable.

Comparison metric

State-of-the-art methods are designed to track animals in short video sequences. For example, they aim to estimate the speed of animals or the inter-individual distance. These pieces of information could then be utilized to recognize specific behaviors, such as panic, or interactions between individuals, such as fighting. In this configuration, it does not matter if one animal keeps the same ID throughout the entire video, as long as it remains consistent during the specific sequence. Thus, it is possible to detect specific behaviors, but not necessarily to automatically derive the identity of the animal. This might be seen as individual information but at the flock scale.

On the contrary, Puzzle is specifically designed to attribute a unique animal ID throughout the entire video, aiming to derive individual behavioral information. We thus proposed to use the Percentage of Good Tracking (PGT) to characterize the performances of Puzzle.

The PGT is computed for each animal, and is, for a given animal, the percentage of detections within the videos where the algorithm predicted the correct ID. It is the number of time the animal is detected in the

Name	Association score - method	Appearance CNN	Training
IOU	Location	∅	∅
pos-old	Location	∅	∅
Tex-FasterVit	Appearance	FasterVit	Unsupervised
Tex-Resnet12	Appearance	ResNet12	Unsupervised
Tex-FurryMixtureNet	Appearance	FurryMixtureNet	Unsupervised
Expert	Appearance and location—equation	FurryMixtureNet	Unsupervised
Wizard	Appearance and location—equation	Image Encoder	Unsupervised
NN-FatserVit	Appearance and location—Neural Network	FasterVit	Unsupervised
NN-Resnet12	Appearance and location—Neural Network	ResNet12	Unsupervised
NN-FurryMixtureNet	Appearance and location—Neural Network	FurryMixtureNet	Unsupervised
IOU-Multi	Location	∅	Semi-supervised
pos-old-Multi	Location	∅	Semi-supervised
Tex-FasterVit-Multi	Appearance	FasterVit	Semi-supervised
Tex-Resnet12-Multi	Appearance	ResNet12	Semi-supervised
Tex-FurryMixtureNet-Multi	Appearance	FurryMixtureNet	Semi-supervised
Expert-Multi	Appearance and location—equation	FurryMixtureNet	Semi-supervised
Wizard-Multi	Appearance and location—equation	Image Encoder	Semi-supervised
NN-FatserVit-Multi	Appearance and location—Neural Network	FatserVit	Semi-supervised
NN-Resnet12-Multi	Appearance and location—Neural Network	ResNet12	Semi-supervised
NN-FurryMixtureNet-Multi	Appearance and location—Neural Network	FurryMixtureNet	Semi-supervised

Table 1. Name and Definitions of Different Versions of the Tracking Methods. The naming convention for the various versions of the tracking method consists of three components. The first part of the name indicates the method employed for the association score, specifying whether it relies solely on location, texture (appearance), or a combination of both. If the combination involves both, it further specifies whether a manually defined equation or a neural network (NN) is used. The second part of the name denotes the convolutional neural network (CNN) utilized for extracting appearance information. The last part indicates the method used for training the appearance CNN, differentiating between fully unsupervised and semi-supervised approaches, where user-provided animal IDs for five tracklets to help the training. In cases where only location is considered for the association score, there is no appearance CNN to train. However, in the semi-supervised scenario, annotated tracklets are accounted for and not relabelled by the algorithm. All the tested methods follow the Puzzle principles. But methods using a neural network for the association score, and an appearance CNN represent the main contribution of this article. .

video, and where the tracking algorithm found back the correct id, divided by the number of times the animal is detected in the video. The PGT was specifically designed to quantify the ratio in which each animal is correctly tracked in the video, and there is no corresponding metric in the MOT or Clear MOT metrics. Discussion of these metrics was made on the previous article, together with a complete mathematical definition[18], section 4.1.

Importance of the association score
The Percentage of Good Tracking (PGT) is presented in Fig. 2.

Location methods solely rely on information related to the distance (in pixels) between detections for association. *Texture* methods exclusively depend on appearance, and several CNNs were assessed: FasterVit, a state-of-the-art architecture, ResNet12, a common architecture, and FurryMixtureNet, the architecture proposed in this article. *Equation* methods utilize an expert equation to combine location and appearance metrics, including Wizard. *Neural network* methods are based on a neural network to gather metrics, representing the original idea proposed in this article. The neural network used for the metrics association is denoted ASN, for Association Score Network. The *Multi* versions involve the semi-supervised version of Puzzle, where the user provides the IDs of the animals on four tracks suggested by the algorithm.

Methods relying solely on location lack robustness and yield the lowest PGT. It is interesting to note that there are not many differences between the unsupervised and semi-supervised versions of these methods. The improvement is solely due to the four manually annotated tracklets. In particular, this shows that the annotation of the tracklets alone cannot dramatically improve tracking efficacy.

When only appearance information is used, results are better and depend greatly on the CNN used to extract appearance features. One can observe that the FasterVit architecture provided interesting results, similar to the previous Wizard version. This is particularly interesting because these types of methods are the simplest, with no need to use an elaborate strategy to combine the association metrics. There is hope that, in the future, with improved architecture and datasets, robust networks could be used to significantly enhance the efficacy of tracking methods using only appearance clue.

The equation (expert weighted sum of metrics) methods generally lack robustness, although they could perform well in some videos. It remains challenging to find the optimal score association technique that will work in any situation.

Using a neural network to mix the association metrics is a novel and promising idea that provided very satisfying results on most of the videos, except when combined with the ResNet12 architecture, meaning that the output feature vector is erratic or temporally inconsistent. With the FasterVit architecture, animals are correctly

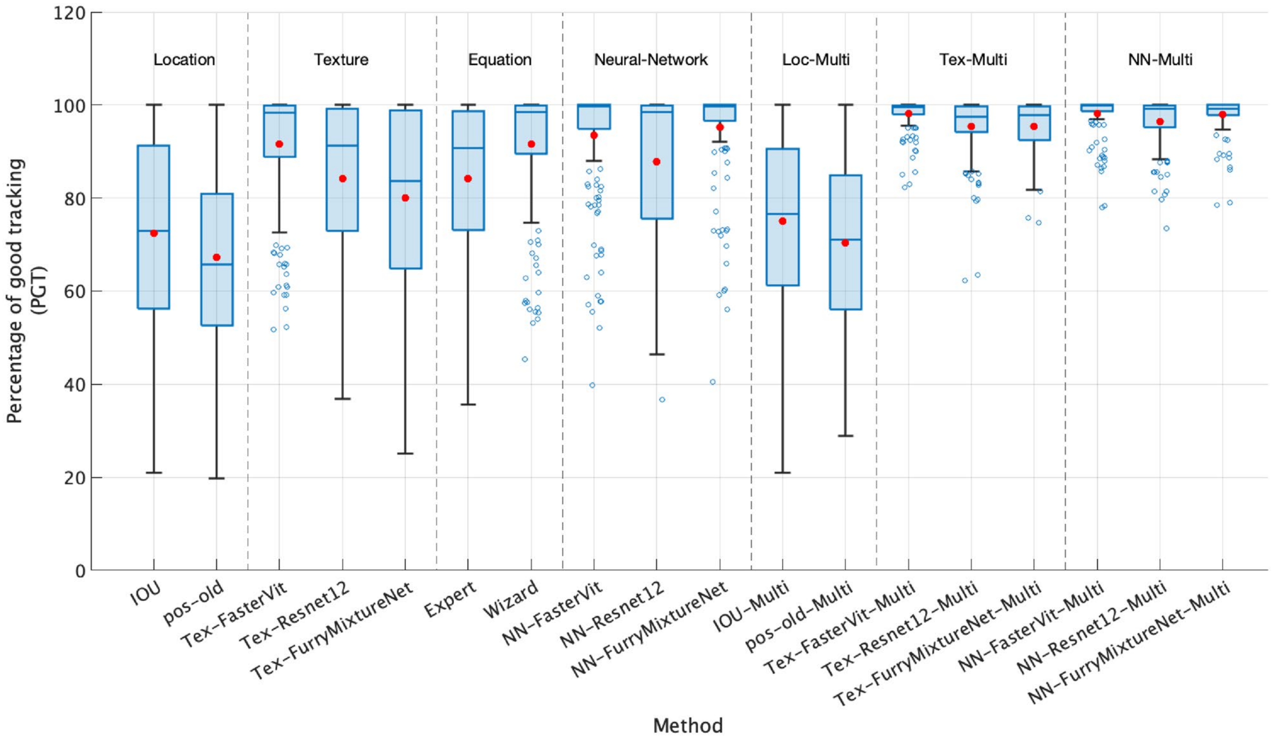

Figure 2. Distribution of the percentage of good tracking (PGT) of the different version of Puzzle, depending on the association score and the method used to train the appearance CNN. The PGT represents the percentage of detections where the method correctly identified the animal's ID. Methods were evaluated on 40 videos, and we displayed the distribution of the PGT of each individuals (156 in total) of the videos. The upper and lower bounds of each box denote the 0.25 and 0.75 quantiles, respectively. The inner line signifies the median, the red dots indicate the mean, and the black lines depict the whiskers, representing the non-outlier minimum and maximum values. Blue circles denote outliers.

identified on average in 93.4% of the frames ($q_{0.05} = 63\%, q_{0.25} = 95\%$), and 95.3% ($q_{0.05} = 72\%, q_{0.25} = 96.5\%$) using FurryMixtureNet. Although promising, this method relies on a set of fully annotated videos to train the metrics association network, which can be time-consuming and needs to be updated when using a different configuration (i.e., other species or camera viewpoint). Here, 13 videos were used to train this network.

Finally, the semi-supervised versions of the method provided the most satisfying results and, in many cases, allowed for an improvement in robustness. Most benefits are observed for the methods using appearance information for the association. This is not surprising, as the network is trained on more images from different parts of the videos, possibly with different points of view of the animals, increasing the diversity of the training set. In this case, again, a strategy based only on appearance information for the association provided very interesting results, suggesting that tracking could be used for behavioral studies. The average percentage of good tracking is 98.2% ($q_{0.05} = 92\%, q_{0.25} = 98\%$) when only appearance clues are extracted with FasterVit in the semi-supervised setup.

The semi-supervised setup also improved the results of the methods relying on a neural network to combine the association metrics, but in this case, the differences with the method based on texture only remain small. The average percentage of good tracking is 98.2% ($q_{0.05} = 88.6\%, q_{0.25} = 98.7\%$) when FasterVit is used, and 98.1% ($q_{0.05} = 89.5\%, q_{0.25} = 97.8\%$) with FurryMixtureNet.

Note that the FasterVit architecture is heavier than the other tested CNNs, which increases the computation time. Instead, the architecture of the FurryMixtureNet was intended to be small, fast and temporally consistent .For every method, preprocessing (i.e. detection and tracklets creation) took 0.05 seconds per frame. Then, tracking took 0.05 seconds per frame when using ResNet12 or FurryMixtureNet and 0.08 with FasterVit. The computation time was obtained on a video of 4 min and 33 s, using an RTX 3060 12GB GPU and an AMD Ryzen 5 5600X 6-core CPU.

Type of errors

A tracking error occurs when the algorithm assigns the wrong animal ID to a detection. The type of error may vary depending on the tracking method (i.e., the type of association score and supervision setup), as illustrated in Fig. 3.

When only location is considered as the association metric, most errors occur when an animal is re-detected. In some cases, due to shade or occlusion, some animals are temporarily lost by the detection method. Upon rediscovery, the animal might be at a different location than in the last frame where it was detected. Associating IDs solely based on location is challenging, representing a significant drawback of these methods. The inclusion of texture clearly improves performance in such cases.

When only texture is considered, most errors occur when the number of detections remains stable. In this scenario, errors are more likely due to variations in illumination, as the color of the animals may change for some time. Some errors also occur during postural or orientation changes. It is noteworthy that the length of the error sequence is significantly lower when appearance is used for association (Fig. 4), indicating that the network quickly reestablishes the correct identity. Conversely, accounting for texture increases the number of ID swaps between overlapping animals (dark green in Fig. 3).

When animals overlap, parts of each animal are present within the detections (bbox) used as input for the neural network, creating confusion. An error sequence refers to a video sequence in which the algorithm assigns the wrong animal ID to a series of detections.

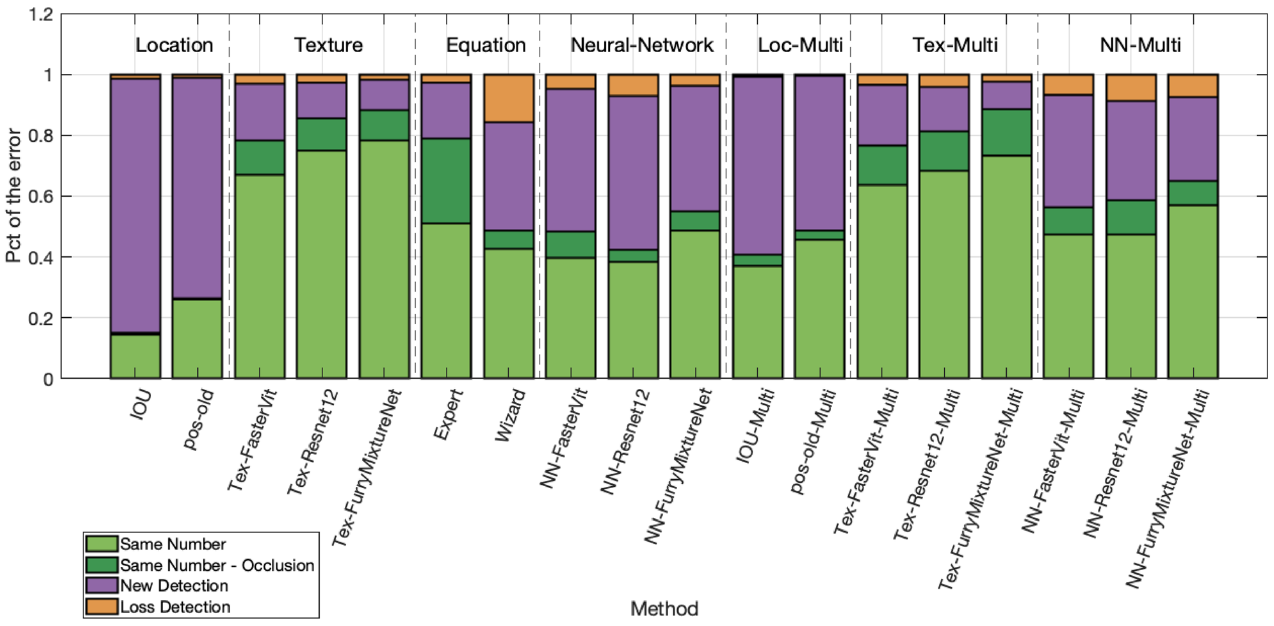

Figure 3. Percentage of errors attributed to detection issues. The proportion of errors occurring when a new animal was re-detected is shown in purple, in orange when an animal was lost, and in green when the same number of animals were detected.

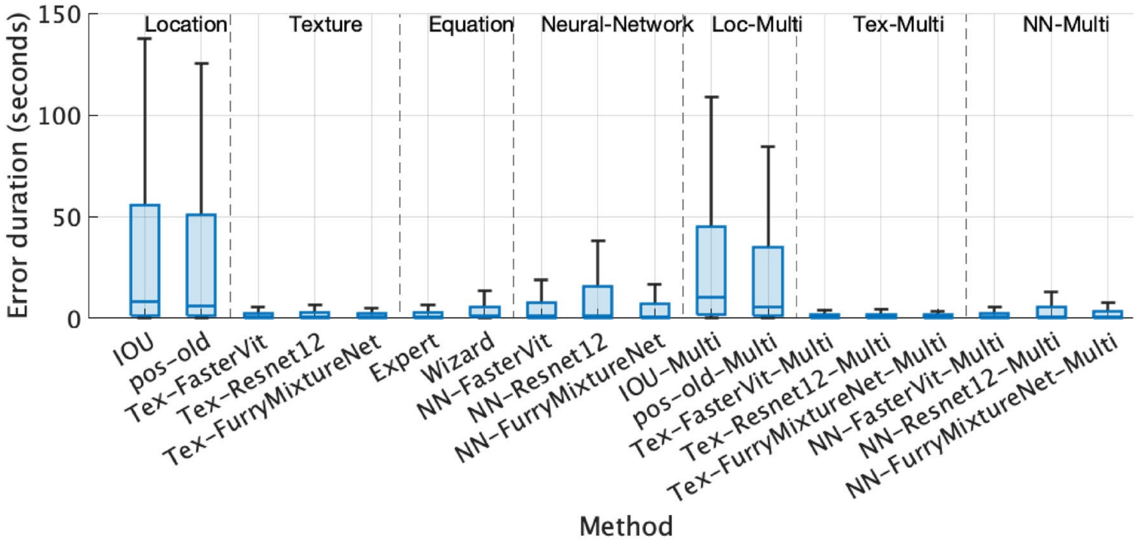

Figure 4. Duration of error sequences for various tracking methods.

Using both appearance and location information (i.e. for the Equation, NN and NN-Multi models) helps to balance both types of errors, as observed is the equation and neural network based methods for association.

Impact for monitoring behavior

Finally, as mentioned earlier, the challenge in using computer vision is to derive individual behavioral information. To better understand the impact of tracking errors, we illustrate how they translate into errors in terms of monitoring behavior. We considered two common behavioral traits: space use and activity time budget.

For space use, not all tracking errors have the same impact. For example, an ID swap between two animals located in the same area will not significantly affect the estimation of occupancy frequency. Similarly, for activity, an ID swap between two animals engaged in the same activity will not affect the estimation of the time budget.

However, it's important to note that the quality of the tracking method and the quality of the behavioral trait information are strongly correlated, as shown in Fig. 5. In other words, the better the tracking method, the more accurate are the estimations of behavioral traits.

All the semi-supervised methods, whether using a neural network for the association cost or relying on appearance only, allowed for an estimation error below 7%. The smallest error (i.e., 2.34%) is obtained by the method using FurryMixtureNet for texture and a neural network for the association cost. It is interesting to note that semi-supervised methods relying on FasterVit, either with a neural network for the association cost or using appearance only, exhibited similar estimation errors, below 5%.

Similar conclusions are drawn for the estimation of the activity time budget, with semi-supervised methods relying on FasterVit or on FurryMixtureNet and a neural network for the association cost showing errors below 10% when using texture information.

Discussion

Monitoring individual behavior is a significant challenge, particularly for animals raised outdoors and in groups. Despite its promise, computer vision is rarely employed for group-housed animals, mainly because identifying individual animals is a challenging task. This study introduces a robust method, demonstrated for monitoring goat behavior, a particularly complex problem due to outdoor conditions, side-view cameras, and the size of the observed animals.

Our approach highlights the effectiveness of utilizing a Convolutional Neural Network (CNN) to extract appearance information for animal ID estimation. The CNN is trained on a representative and diverse dataset. Unlike other species such as pigs or poultry, goats exhibit physical differences, often in terms of color, horn shape, or body size/shape, which facilitates the extraction of distinctive appearance information. The method was tested on videos from Psota et al. (2020), featuring housed pigs with very similar appearances. In this particular case, the method yielded unsatisfactory results. However, when tested on personal videos featuring marked pigs, the outcomes were comparable to those presented in this study. Regardless of the scenario, we demonstrated that relying solely on location for tracking was not a viable solution. The effectiveness primarily depends on the quality of detections. In instances where animals were lost and subsequently rediscovered, methods based solely on location struggled to accurately identify the correct ID. The optimal approach involves leveraging both location and appearance information, necessitating a method to effectively combine these two information. Our study showcased that a neural network presents a compelling option, significantly enhancing tracking efficacy. On the other hand, training the ASN requires the initial construction of a tracking dataset, which can be a challenging task. This challenge diminishes when the experimental setup is used in the long term, involving the same type of camera, angle, and field of view. A viable strategy is to initially use Puzzle in semi-supervised mode, utilizing FasterVit and texture exclusively, to analyze several videos. The results should then be corrected manually to

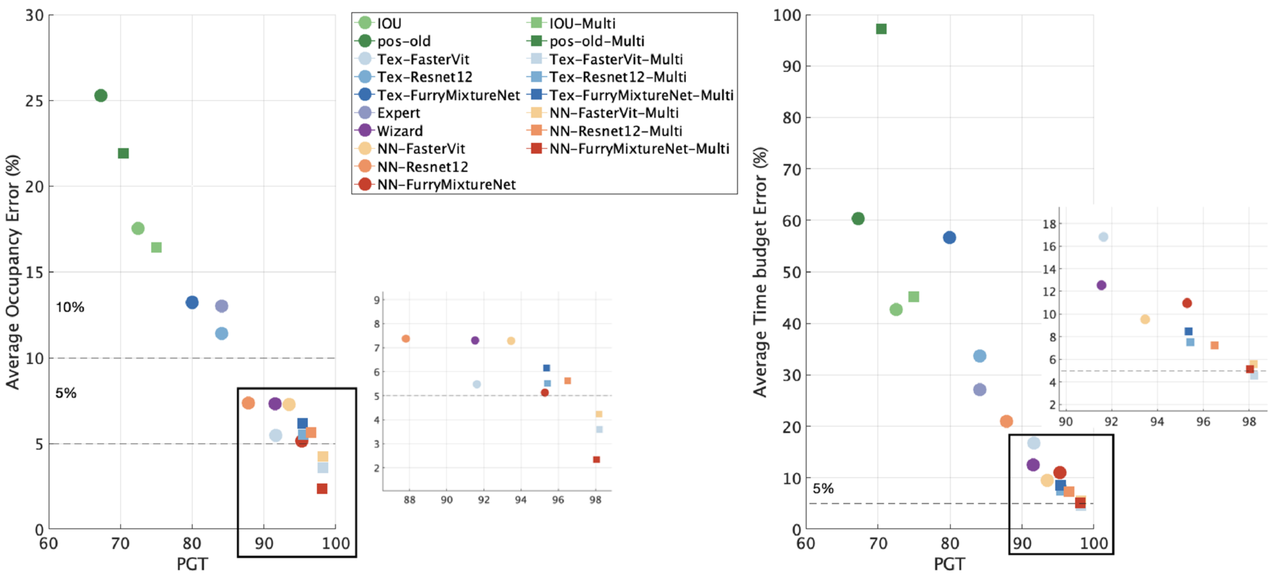

Figure 5. Correlation between behavioral traits estimation and tracking capacity. The x-axis represents the percentage of good tracking (PGT) for each method. For the occupancy error, the pasture was divided into a virtual grid of 10 pixels by 10 pixels quadrats. The occupancy frequency was computed using each tracking method and compared to the occupancy frequency computed with the ground truth annotated data. For a given quadrat, the occupancy frequency is the proportion of frames where the center of the animal was inside the quadrat. The average occupancy error is computed over all quadrats, animals, and videos for a given method. For the time budget, the time budget for grazing, lying, and other behaviors was computed for each animal and video for a given tracking method. It was then compared to the time budget computed with the ground truth annotated data and averaged over all animals, activities, and videos. In any case, the error is defined as $est = true\left(1 + \frac{error}{100}\right)$, where $est/true$ is either the estimated/true occupancy or time budget. A zoomed view is provided for the high PGT values.

create the dataset and train the ASN. Puzzle can then be deployed in fully unsupervised mode, incorporating FurryMixtureNet, for faster computation time within the remaining videos. In any case, the semi-supervised version of Puzzle, when combined with either the FasterVit or FurryMixtureNet architecture, yielded robust results. This robustness enables the method to be employed for initiating behavioral studies in a research context.

An interesting direction to improve our algorithm could involve reconsidering how the association problem is addressed. In the current version, we employed the common Hungarian algorithm, which has the main drawback that all between-frame detections should be associated, potentially including errors such as false positive detections.

To overcome this challenge, it is first crucial to depend on an efficient detection technique. However, exploring alternative methods beyond the Hungarian algorithm could also be worthwhile.

One possible improvement would be to integrate graph-based methods with min-cost flow algorithms[22]. Employing cutting-edge techniques, this approach could automatically eliminate inconsistent associations. Although Graph Neural Networks were tested for the same objective, they did not yield successful results.

Another appealing perspective involves training an optimal association strategy. This could be achieved using the framework of a Markov Decision Process, where an agent makes decisions based on the system's state and receives rewards contingent on their actions. In our case, the system would be represented by the association cost matrix, and the agent's decisions would be the different possible association, including the *no-association* action. Rewards would be computed based on the quality of the association. Any technique could then be applied to estimate an optimal association strategy, similar to the training of the FuzzyBudy network.

However, the use of image analysis for monitoring the behavior of group-housed animals still has important limitations. A high density of animals, which is not uncommon in typical housing conditions, generally results in significant occlusions between animals. In such situations, animals can remain undetected for long periods, losing any possibility of recording behavioral data, regardless of the tracking method employed. The presence of blind spots or shadow areas can also lead to some animals being undetected for extended periods. For example, Puzzle was used to monitor 14 goats inside a 30m^2 paddock. Due to practical constraints, we used a side-view camera, which caused at least one occlusion in all video images. In this case, Puzzle could not start the procedure, as it was not possible to compute any tracklets, and thus no best-track to train the A-CNN. The camera's location can potentially reduce the number of occlusions, such as when cameras are positioned on the ceiling, providing a top view of the flock. Unfortunately, this is not always feasible, considering the ceiling height and the camera's field of view. The field of view is another limitation, constraining the study area's size. It is challenging to imagine exceeding a few hundred square meters in most cases. There are several challenges that need to be addressed before combining image analysis and tracking algorithms to derive behavioral information in the most generic cases. However, for research purposes, where the size of the paddock and the density of animals can be adjusted, Puzzle is an interesting tool.

Our method is completely open-source and thoroughly documented, accompanied by a published open dataset used to train the goat detector[23]. The videos containing tracking annotations will be provided upon request. We are actively promoting the publication of open datasets and code within the livestock community, aiming to enhance and improve the use of deep-learning based methods. The software comes with a graphical interface allowing to check and correct the tracking results effortlessly. This can be helpful to ensure a minimal tracking quality, but also to grow the amount of annotated data that can be used to train any tracking method.

Method

General principle

The present algorithm, named Puzzle, adheres to the principles of the Wizard tracking algorithm[18] with some improvements. The general principle of the method is to leverage the detections from Yolo, recognizing that tracking might be straightforward in some *sequences* of the video. Indeed, it is common to observe sequences where all the animals are detected, with no occlusions or false negatives. In these sequences, tracking is easy, and a method based only on the bounding box (bbox) locations is sufficient. These sequences are called *tracklets*, i.e., successions of video frames where all the animals are detected, with little or no overlap between the bounding boxes.

On one tracklet, the same animal has a unique ID, but this ID might differ from one tracklet to another. In other words, the tracking problem is easy to solve on a tracklet scale but, of course, not at the video scale. The main idea of the method is to select one particular tracklet, called *best-track*, from which a CNN will be trained to extract texture/appearance information to identify each animal. Then, this appearance CNN, denoted as A-CNN, is used on the rest of the video to extract texture information from the detections.

Once the best-track is defined, and the A-CNN trained on it, the concept of tracklets can be forgotten. The inference on the entire video can start, which is divided into two other steps: the forward and backward passes. On the forward pass, tracking is conducted starting from the end of the best-track until the end of the video. Conversely, for the backward pass, tracking is performed from the beginning of the best-track until the start of the video, or to the end of the next best-track, in case of semi-supervised version. Each pass, like most tracking algorithms, is an ID assignment problem. It involves associating the detections from the previous (known) frame with the detections of the current frame (unknown).

Note that at the last (or first) frame of the best-track, the ID of all detections is known; this is the main principle of Puzzle. The two passes consist of propagating these IDs from the best-track to the entire video.

Assigning the ID to the detections of the current frame is based on a cost matrix C, where $C_{i,j}$ is the cost of assigning the animal ID j to the detection number i of the current frame, given the detection on the previous frame. The cost is generally a weighted sum of different metrics, such as texture or location distances.

The main improvements of Puzzle over Wizard are as follows. First, the assignment cost was redefined for the creation of tracklets to be more robust, especially for the false positive detections. Second, the assignment cost definition during the backward and forward pass was also improved by not considering the cost $C_{i,j}$ as a weighted sum of metrics, but as the output of a CNN, which takes different metrics as inputs. Third, the structure of the A-CNN was changed, adding an attention layer on the head to address illumination changes, while the classification layer was replaced by a Mixture of Expert. Fourth, there is an option to manually label several best-tracks, which takes limited time but can greatly improve the tracking results.

We will begin by describing the improvements of this new method. Then, we will describe the available data and the method used to detect animals. Finally, we will present the method used to compare the different tracking methods.

Puzzle main improvments

Assignment cost for the creation of tracklets

The main idea of Puzzle (and Wizard) is to first concentrate on simple sequences of the video, where tracking is easy. In this case, the assignment cost matrix C only relies on location information to associate the closest bounding boxes between frames.

For Puzzle, our intention was to refine the assignment cost due to the persistence of certain false positive detections, even after employing various elimination methods. Consequently, we introduced a new assignment cost based on two metrics: the distance between bounding boxes and their overlapping area.

$$\begin{aligned} overlapping_{i,j} &= 1 - IoA(B_i^t, B_j^{t+1}) = 1 - \frac{Area(B_i^t \cap B_j^{t+1})}{Area(B_j^t)}. \\ distance_{i,j} &= \sqrt{\sum (B(x,y)_i^t - B(x,y)_j^{t+1})^2}. \\ C_{i,j} &= (1 + overlapping_{i,j}) * distance_{i,j} * (1+\delta). \end{aligned} \tag{1}$$

Where B denotes a bbox or a detection, $B(x, y)$ denotes the centroid of the detection, and B_i^t denotes the detection numero i on frame t. t is used to denote the previous frame and $t+1$ the current frame. On the previous frame, an animal ID is associated to every bbox, and the animal ids have to be estimated for the current frame.

The overlapping metric is the intersection over area (IoA), used to quantify the spatial overlap between two consecutive bounding boxes. The $1-$ adjustment is applied to invert the outcome, as the assignment problem is a minimization problem. The distance metric is simply the Euclidean distance between the centroids of the two detections. The composite cost function 1 combines these two metrics and is computed for any pairs i and j. The bounding box association problem is then solved using the Hungarian algorithm with the defined cost matrix C.

Once associations are resolved, various thresholds are applied, mostly in the same way as Wizard, to remove False Positives and break the tracks into tracklets.

Definition of the A-CNN: FurryMixtureNet

The A-CNN is used to extract appearance information from the detections: what are the appearance clues allowing to distinguish each animal? The main proposition of both Wizard and Puzzle is to train the A-CNN directly on the images from the video being analyzed.

In Puzzle, we introduced a new version of the A-CNN called FurryMixtureNet. It is a classification network with *N* classes, one for each animal. It consists of two modules: (i) an image encoder module to extract features at different scales, and (ii) a classification module that combines these features for classification through a mixture of experts.

FurryMixtureNet: Image Encoder module

The initial component of the network is a conventional image feature extractor, resembling a lightweight VGG version. As suggested in the few-shot learning challenge and previous studies[24,25], we utilized a smaller structure than in Wizard, with an input size of (84, 84, 3).

Additionally, special convolutional blocks and skip connections were integrated to enhance the gradient path during training. Compared to Wizard, we also introduced the CBAM[26] and SqueezeExcitation[27] modules.

The Convolutional Block Attention Module (CBAM)[26] is a network module employed for enhanced feature representation and classification accuracy. CBAM consists of a two-stage attention mechanism designed to learn to focus on crucial image regions and channels.

In the first stage, the spatial attention module utilizes the global average pooling operation to generate a channel descriptor, which is then used to weigh the importance of each spatial location. In the second stage, the channel attention module employs the maximum pooling operation to generate a spatial descriptor, which is used to weigh the importance of each channel. The final output is obtained by combining the spatial and channel attention maps through element-wise multiplication.

By concentrating on significant regions and channels, CBAM contributes to improving the robustness of Convolutional Neural Networks (CNNs), particularly in the face of illumination changes. For exemple, CBAM module was integrated on yolov3[28] and in yolov5[29] for the same purpose.

The *DepthwiseConvBlock* consists of two stages of convolution, batch normalization, and a GELU activation function. This activation function is recognized for its faster convergence and greater resilience to noise compared to other activation functions like ReLU or ELU[30]. The initial convolutional layer in the first *DepthwiseConvBlock* includes a bias, while the subsequent layers do not. The first convolutional layer of each *DepthwiseConvBlock* employs a 3×3 kernel size with depth-wise operations, and the second layer uses a 1×1 kernel size with point-wise operations. This approach, introduced in[31], combines *DepthwiseConvBlock* with skip connections denoted by $\bigoplus$. An essential aspect of *DepthwiseConvBlock* is its computational speed, which is significantly faster than a regular convolutional block by minimizing the amount of multiplication and mathematical operations.

The incorporation of a SqueezeExcitation layer[27] was motivated by empirical observations. We found that this layer enhances our network's robustness by facilitating the extraction of a more diverse range of discriminative features, particularly concerning colors. The SqueezeExcitation mechanism focuses on modeling interdependencies between channels, explicitly capturing and leveraging the relationships among different channel responses. By integrating the SqueezeExcitation layer, we observed a reduction in confusion between goats of different colors. However, it should be noted that the introduction of multiple layers of SqueezeExcitation posed challenges during the learning process . Specifically, it affected the gradient path, causing divergent learning. Therefore, only one layer was used.

A detailed illustration of the image encoder's design can be found in Fig. 6.

FurryMixtureNet: Classification module

The classification process is illustrated in Fig. 7. The network takes the feature vector generated by the image encoder as input, feeding it into Mixture of Experts (MoE) modules[32], which mix and classify the features into their most appropriate categories. MoE is a neural network architecture type that divides the input data into multiple subspaces, using a separate "expert" network to make predictions for each subspace. The outputs of the expert networks are then combined to produce a final prediction. MoE is relevant for classification tasks because it enables the network to model complex, multi-modal distributions in the input data, which may not be accurately captured by a single expert network. In the context of individual classification in videos, MoE can help account for the large variability in appearance and goat posture.

The proposed architecture for the expert modules is provided in Fig. 8, and can be viewed as a small transformer block[33]. It is composed of several layers, including Dense, Batch Normalization, GELU, Multihead Attention, FeedForward, and a final Sigmoid layer for classification. Each expert specializes in certain features, and their contributions are dynamically weighed through the Gating Top-k. The Multihead Attention layer enables the model to learn how to concentrate on specific areas of the input that are pertinent to the classification task. Meanwhile, the FeedForward layer plays an important role in refining the feature representation obtained from the previous layers, helping the model captures high-level connections between features that are not linked to their spatial position in the image. As a result, the architecture can identify intricate patterns and interactions between features, learn to focus on important regions of the input, and detect high-level relationships between features beyond their spatial arrangement. Remember that the input features are derived from the image encoder, which includes CBAM and SqueezeExcitation layers, enabling diverse and representative features, in spatial and colorimetric space.

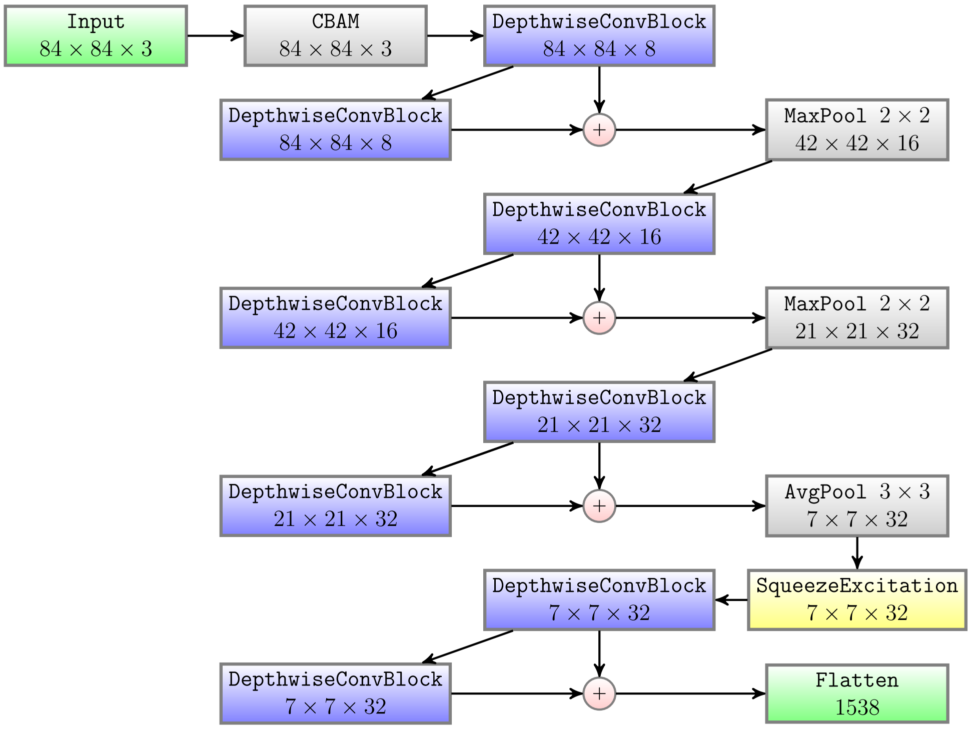

Figure 6. The image encoder architecture consists of a CBAM attention mechanism to focus on relevant parts of the image. Then, several depth-wise convolutional blocks are employed to efficiently extract features at the current scale. Additionally, skip connections, represented by the symbol $\bigoplus$, aid in preserving some features from previous scales. Similarly, the Squeeze Excitation layer allows capturing and focusing on distinct features, particularly color information. Finally, the extracted features are flattened and linearly combined to produce a feature vector of length 1538, which is then fed into the classification module.

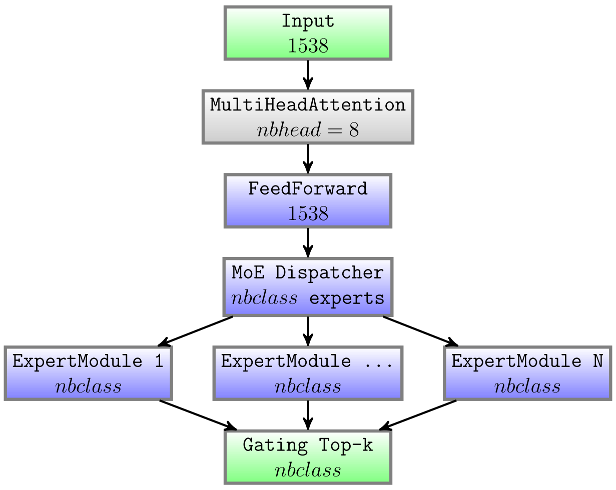

Figure 7. Classification module: the MoE approach.

A Sigmoid activation function was used instead of Softmax to achieve proportional representation: when goats are separated and do not overlap, the classification using the Sigmoid activation function is expected to be accurate, similar to a one-hot encoder. However, in cases where two or more goats are visible in the same image, the classification output is expected to provide a proportional representation of each goat based on their visibility. Achieving this functionality is challenging using the softmax activation function, as values tend to be maximized in a single class. Still, the sigmoid activation function enables the computation of proportions for each individual separately. To ensure this proportional representation, custom data augmentation techniques were designed.

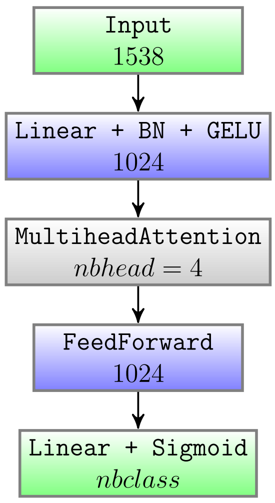

Figure 8. ExpertModule : small architecture based on "Attention is all you need".

Assignment cost for the forward and backward passes
The third aspect that differs from Wizard is the definition of the assignment cost matrix for assigning the ID between the previous and current detections over the forward and backward passes.

Let's denote $\left(\text{bbox}_i^{\text{past}}, \text{id} = i\right)_{i=1}^{N}$ as the set of past bbox with the associated animal ID. In other words, it is the most recently estimated bbox for each individual. Note that all bbox does not necessarily come from the same frame, as some individuals might be undetected on several consecutive frames. Then, for each new frame, called the current frame, we have a set of bbox $\left(\text{bbox}_i^{\text{current}}\right)_{i=1}^{n}$, with $n \leq N$, as again, some animals might not be detected. But also note that, in some rare cases, $n > N$ if false detections occur. The tracking problem consists of associating each current bbox to one of the past bbox. Once the bbox association is made, the past bbox set is updated, and so on.

In order to make the assignment, a cost matrix, C, is used, where $C_{i,j}, j \leq N$, is the cost of associating the ith bbox of the current set with the jth bbox of the past set, or equivalently, assigning animal ID j to the ith bbox of the current set. $C_{i,j}$ measures a distance or similarity between the two bbox. Once C is computed, making the association is a simple combinatorial optimization problem, called the assignment problem, easily solved in polynomial time ($O(n^3)$) using the Hungarian algorithm[34]. Mathematically, it consists in finding the optimal row numbers $(i_1^*, \ldots, i_N^*)$, with $i_{i_2}^* \neq i_{i_2}^*$, for which:

$$\sum_{i=1}^{N} C_{i_i^*,j} \leq \sum_{i=1}^{N} C_{i_i,j}, \qquad \forall 1 \leq i_i \leq N \text{ and } i_{i_2} \neq i_{i_2}$$

The major difficulty when designing a tracking algorithm is to propose a reliable cost definition that takes into account every aspect of the tracking problem, such as false negative/positive detections and occlusions.

A common approach is to minimize a global cost function that measures the dissimilarity between objects inside the bounding boxes of the previous and current frames. Typically, an empirical function is used to compute this cost, as proposed in Wizard[18], and in other popular MOT algorithms such as DeepSort[17], BOT-Sort[35], ByteTrack[36], or HybridSort[37].

Association metrics
One other approach to model assignment cost consists in using a learnable function that maps pairs of detections to a scalar cost $C_{i,j}$. This function can be learned using supervised learning[38], where pairs of detections are labeled with their true association cost. In this context, various approaches can be found, from GNN (Graph Neural Network)[39] to sequence classification[40]. In this work, we introduced a compact neural network that uses various association metrics as inputs and combines them to determine the accurate assignment cost. We proposed to compute the following metrics:

texture	The one-hot class Manhattan distance (based on the A-CNN output) between known previous and current vectors. By applying this distance function, we can quantitatively measure the dissimilarity between the past and current detections.
IoA	The Intersection over Area (IoA) is a similarity metric that quantifies the overlap between two bounding boxes. It is calculated by dividing the area of intersection between the two bbox by the total area of the union of the boxes. The IoA value ranges from 0 to 1, where a value of 0 indicates no overlap, and a value of 1 indicates complete overlap between the bounding boxes. By using the

	IoA as a similarity metric, we can assess the degree of spatial overlap between two bounding boxes and determine the level of similarity between objects or regions of interest.
IoU	The Intersection over Union (IoU) is a widely used similarity metric for assessing the overlap between two bounding boxes. It is calculated by dividing the area of intersection between the two bbox by the area of their union. The IoU value ranges from 0 to 1, where a value of 0 indicates no overlap and a value of 1 indicates complete overlap between the bbox. Here, the IoU was defined as the mean of IoU between the current detection B_j^{t+1} and each past detection in the last secondes of the same tracklet. The IoU quantitatively assesses the spatial correspondence between objects.
time	The absolute time distance quantifies the temporal proximity or similarity between two events or time points. If the time difference between two detections of the same object is extremely short, this implies minimal change in the object's location between the two frames. Conversely, a substantial time difference, for example, when the object remained undetected over several frames, might suggest significant alteration in the object's location across the frames, indicating a degree of dissimilarity. By combining the absolute time distance with other metrics, the assignment process can prioritize the temporal proximity of detections.
pos_old	The Euclidean distance between the centroids of the current and past detections is used as a dissimilarity metric. This metric quantifies the spatial disparity between the current detection and the most recent associated centroid. A smaller distance indicates a higher level of similarity, while a larger distance suggests greater dissimilarity. However, this metric cannot be robust in case of false association, even short.
pos_med	The Euclidean distance between the centroids of the current detection and the median of the last one-second detections is used as a dissimilarity metric. Again, it measures the spatial difference between the current and previous detections. A smaller distance indicates higher similarity, and a larger distance indicates greater dissimilarity. It should be more stable in case of short false association.
speed	The acceleration of the animal at each frame is computed for the last 3 seconds. To assess similarity, we compare the smallest (speed_a = quantile 10%) and the largest (speed_b = quantile 90%) acceleration values from this list with the actual acceleration of the animal, which is calculated as the distance traveled divided by the time interval. This approach serves as a similarity metric by evaluating the consistency of the animal's acceleration over time. A small difference between the smallest and largest acceleration and the current acceleration indicates a higher degree of similarity, suggesting that the animal is maintaining a relatively stable and consistent acceleration pattern. On the other hand, an acceleration difference between the two points (smallest and largest) indicates a deviation from the expected acceleration pattern, suggesting changes in the animal's movement dynamics. The quantile is used to remove possible outliers coming from short false association.
theta	Three spatio-temporal points are used: the current detection centroid, the last associated centroid, and the last 1-second median centroid. To measure the similarity between these points, we use the angle formed by these three locations. A smaller angle suggests that the objects are moving in a coherent and consistent manner, potentially towards the front. This indicates that the tracked objects maintain a relatively stable trajectory over time. On the other hand, a medium angle may indicate that the object is in motion, while a higher angle would be unlikely, suggesting abrupt changes in position that are not consistent with most movement patterns.

Balance between association metrics: FuzzyBody

To amplify the influence of the texture metric between individuals, the variable was duplicated thrice, resulting in a list of 11 metrics. This operation provides the neural network with more weights allocated to the texture distance, allowing the model to prioritize the correct tracks based on appearance, rather than relying on incorrect tracks due to other metrics based on spatial information. However, other metrics still play an important role in case of errors caused by the image classification network. Therefore, determining the weight for each distance metric is a delicate balance between the two aspects. To learn this delicate balance, we proposed the following neural network, named FuzzyBuddy, described in Fig. 9, to compute the value of the scalar $C_{i,j}$.

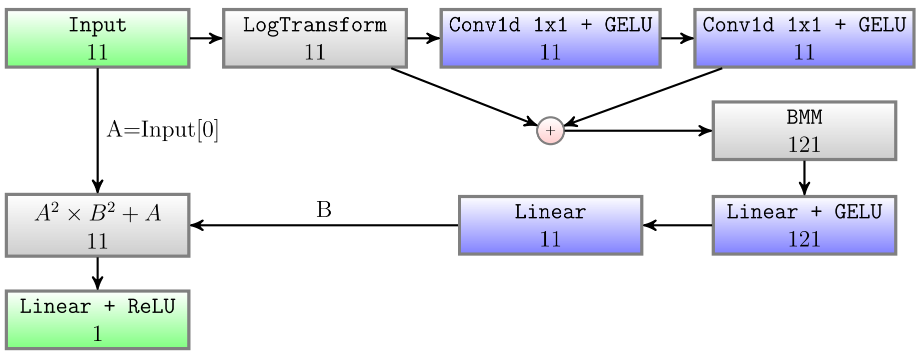

Figure 9. The proposed FuzzyBuddy network used to weight the distance metrics for the ID assignment problem.

Several studies[41,42] have reported that working with logarithmic data leads to more stable and robust neural networks, as well as faster convergence. In the FuzzyBuddy network, the first layer is thus a logarithmic transform, which can be expressed mathematically as $f(x) = \text{sign}(x) \cdot \log(\text{abs}(x) + 1)$. The use of the log transform was also motivated by the need to manage spatial distances, as high distance weights could disrupt the learning process and place too much emphasis on this metric compared to textural distances. The following two layers consist of convolutions and GELU activation. The choice of convolution over a linear layer was motivated by the group-based operation, which allows for working with specific properties separately rather than mixing them together. Additionally, the GELU activation was chosen for its convergence speed and the ability to create fuzzy thresholds. A skip connection is added to mix input data and learned "thresholds".

The BMM (Batch Matrix Multiplication) layer is a custom layer that utilizes batch matrix multiplication to generate novel combinations of properties that cannot be achieved through standard linear operations. Specifically, this layer computes the matrix product of the input variable x with its transpose x^T, resulting in a square symmetric matrix X. The BMM layer replaces the diagonal elements of the output matrix with the original input variable values, effectively removing the supposed unnecessary squared values. The BMM layer allows for the exploration of non-linear relationships and interactions between the input variables.

A compact dense layer was introduced to combine the features obtained from the previous flattened BMM layer. It learns polynomial coefficients since the previous operations can be considered as complex multivariate Bernstein expansions[43–45]. A GELU activation was used to introduce some additional non-linearity and enable a smooth/fuzzy transition between variables. A final dense layer was proposed to reduce the number of variables. Finally, to prioritize the texture metric (i.e., the appearance clue), the output was squared and multiplied by the texture metric ($A^2 \times B^2 + A$) before being fed into the final dense layer, which represents the optimal association cost. This ensures that the network gives greater importance to the texture metric while retaining some spatio-temporal information.

If the hard constraint of texture is removed, an interesting observation emerges: the learned network tends to become overly focused on spatial information, disregarding the valuable cues provided by texture. As a result, the network's performance may suffer, particularly in tasks that require accurate classification or identification based on both spatial and texture cues. For example, it becomes incapable of recovering the original track after a bad association, even if it's short. This approach allows the network to achieve a more balanced and holistic understanding of the input data, leading to improved performance and avoiding the pitfalls of overfitting on spatial information alone.

Training of the fuzzy buddy network

To be able to fit the model parameters, we used the ground truth of several videos, i.e., the detections and the associated true animal IDs. At each video frame, all the distance metrics defined earlier were computed between any pair of current versus previous detected objects. Then, we used the ground truth to characterize the value of the assignment (i.e., 0 if the assignment is correct, 1 otherwise).

The extraction of the distance metrics, together with the assignment value, was done on 15 videos. Note that we only focus on frames of the video where there exists at least one overlapping section between some detections (IoU > 0.05), while other detections might be non-overlapping. Indeed, on most of the videos, there is no overlapping between bbox, and if the network were trained on the entire video, these sequences would have a strong effect on the model parameters, resulting in a network that probably puts no weight on the texture metric.

Training was approached as a contrastive learning problem. During optimization, we clamped the CNN's output values to the range of [0,1], which can be seen as optimizing this decision boundary in probability p_i. Due to the imbalance in the ground truth data, with many soft cases and few hard cases (overlapping goats), we used the focal loss, a modified version of the standard cross-entropy loss function, as suggested in other studies[46,47], specifically for detection anomaly[42]. In our case, the binary cross-entropy was employed in the focal loss and defined as follows:

$$bce\_loss(p, y) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log p_i + (1 - y_i)\log(1 - p_i)\right]. \tag{2}$$

Where p is the predicted probability for each example, y is the corresponding true label (either 0 or 1), and N is the total number of examples in the batch. The focal loss introduces hyper-parameters α and γ, which balance the contribution of each class and down-weight the contribution of well-classified examples. The focal loss can be expressed as follows:

$$\begin{gathered} a = y\alpha + (1 - y)(1 - \alpha) \\ m = (1 - yp + (1 - y)(1 - p))^{\gamma} \\ focal\_loss(p, y) = \frac{1}{N}\sum_{i=1}^{N} a \cdot m \cdot bce\_loss(p_i, y_i) \end{gathered} \tag{3}$$

The focal loss is expressed as a combination of the modulating factor m, the class weight a, and the binary cross-entropy loss function. The binary cross-entropy loss function measures the differences between the predicted probabilities and the true labels, while the modulating factor reduces the weight given to well-classified examples. Finally, the focal loss is obtained by averaging the modified cross-entropy losses over the batch. We manually tuned $\gamma = 2.5$ and $\alpha = 0.25$.

Finally, we used the Adam optimizer with an initial learning rate of 0.0001 and beta parameters of 0.9 and 0.999. A ReduceLROnPlateau was also employed with default settings. The training process lasted for 20 epochs and was run on 15 videos. The remaining videos ($53 - 15 = 38$) were used for testing the tracking algorithm. Note that, as in Wizard, the common Hungarian algorithm is used to perform ID assignment once the cost matrix is built based on the neural network output ($C_{i,j} = NN(x) \mid |x| = 11$).

Puzzle semi-supervised learning option
We proposed a modified version of Wizard, where the A-CNN could be trained on several best-tracks. However, in this case, the ID of each animal in the best-track has to be recorded manually.

In the case of multiple best-tracks, the user was provided with a GUI interface to annotate five best-tracks. These best-tracks are strategically selected: one close to the beginning of the video, one close to the end, and finally, the three longest tracklets of the video. These annotated best-tracks are then used for the training of the A-CNN. Note that in this case, multiple forward and backward passes are used. A backward or forward pass always starts at the beginning or end of the best-track. Then, when a video sequence is located between two best-tracks, the forward and backward pass of each best-track ends in the middle of the video sequence (see Fig. 1).

Tracking efficacy
We proposed using the percentage of good tracking (PGT) to characterize the efficacy of the tracking algorithm. The PGT was computed for each animal in the video, representing the percentage of frames where the tracking algorithm provided the correct animal ID. It is also referred to as *precision* in some articles. It should be noted that commonly used measures, such as MOT or ClearMOT, were tested in the previous article. While these measures are commonly used, it was discussed in the previous article that they are not representative for our study.

Type and duration of errors
Common tracking errors occur when animals are not detected or when animals are close to each other, leading to bounding boxes (bbox) that contain parts of different animals. If an animal is lost and then re-detected, possibly in a different posture or location, it becomes challenging for the algorithm to make the correct association, as it relies on information from the previous detection. When animals are close to each other, confusion arises because their location information is similar, and their appearance information gets mixed.

Another possible error occurs when both a false positive and a false negative detection happen in the same image. In such cases, using the Hungarian Algorithm may lead to forcibly associating the false positive to an individual, resulting in a momentary error where all individuals may be swapped due to the cost minimization.

To assess the ability of different methods to handle specific cases, we employed the following methodology. For each method, video, and animal, video sequences with incorrectly predicted animal IDs were identified, and we recorded the frame number t at the beginning of these sequences, as well as na^t, the number of detected animals for this frame. We examined the detections from the frame just before the start of the error sequences, $t - 1$, and also recorded the number of detected animals na^{t-1}. We then compared the number of animals in these two frames.

If $na^t > na^{t-1}$, the error was attributed to a new detection, meaning an animal was lost and re-detected on frame t but was assigned the wrong animal ID. Conversely, if $na^t < na^{t-1}$, an animal was lost, which could also have caused an error. Finally, when $na^t = na^{t-1}$, the error was not due to a detection problem. In this case, we computed the overlapping ratio between the detections of the animals that received the wrong animal ID. We then calculated the proportion of cases where the bounding boxes (bbox) were overlapping (i.e., with a non-zero overlapping ratio), which we attributed to a tracking error due to occlusion.

We summarized the results for each method and computed the proportion of errors with the same number of detections, both with and without occlusions, and attributed to detection errors, with a lower or greater number of detections at the start of the error sequence. Additionally, we recorded the duration, in seconds, of the error sequences for each method during this analysis.

Impact for monitoring behavior
In order to assess the suitability of the tracking method for behavioral studies, we analyzed the impact of tracking errors on the estimation of two behavioral traits: space use and activity.

Space use
In behavioral studies, the utilization of space is often a crucial indicator, especially for estimating the time spent in specific areas such as resting places, water sources, or food dispensers. In our study, we focused on the estimation of occurrence frequency at the pasture scale, which can subsequently be used to derive other traits (e.g., frequency in a particular location).

To analyze space utilization, we divided the pasture in each video into a square grid of 10 pixels by 10 pixels, assigning a unique ID to each grid cell. For a given video, we calculated the number of detections of animals in each cell, resulting in an occurrence frequency map for each animal and video. These maps were generated using each tracking method and then compared to the ground truth labeled data. The error for each cell in the grid was computed as follows:

$$error = 100 \times abs\left(\frac{\tilde{f}_i}{f_i} - 1\right). \tag{4}$$

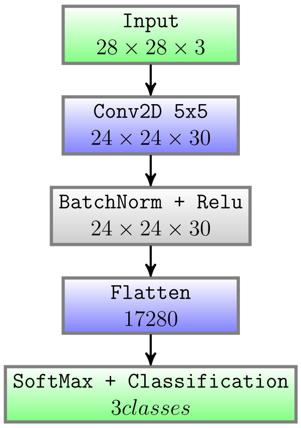

Figure 10. Activity prediction network.

Where $\tilde{f}_i$ and f_i are the estimated and true frequency on cell i. And *abs* is the absolute value function. The error is thus expressed as a percentage.

Activity
We categorized three types of activities: grazing, lying, and other. Grazing activity was determined based on the head position of the animal, specifically when the head was below the shoulder, often corresponding to situations where the animal's head touches the ground. An animal was considered lying when it was not on its feet, indicating a resting position on the ground. Any behaviors other than grazing or lying were classified as "other", including activities like resting while standing or moving.

To estimate the activities, we employed a neural network for each goat (see below), treating it as an image classification problem. The network took the image of the goat as input and predicted the activity class (grazing, lying, or other). The tracking method provided the images of each goat for every video frame, enabling us to predict their activities. We computed the activity time budget for each animal in each video, representing the proportion of frames where the animal was grazing, lying, or engaged in other behaviors. This activity time budget was then compared to the ground truth labeled data, and the estimation error was computed for each video, animal, and activity using the Eq. 4.

To predict activity from the image, we constructed a simple neural network, to speed up the prediction and training process. The input size of the image was set to 28 by 28 pixels. The network is only composed of one convolution layer, with 30 filters of size (5, 5) and a stride and dilation factor of (1, 1), with no padding. Followed by a Batch Normalization and a Relu layers, to finish with a fully connected layer, a softmax and a classification layer, using a cross-entropy loss function (see Fig. 10).

We used another study to quickly construct the training and test set. In 22 videos, the animals were equipped with accelerometers attached to one horn. Although these accelerometers were initially trained to predict more detailed behaviors, we repurposed their predictions to automatically generate images where goats were either grazing, lying, or engaged in other behaviors. For each video, we utilized the accelerometer predictions to create 100 images per animal-50 images indicating grazing and 50 images indicating non-grazing activities (i.e., either lying or other behaviors). Using the accelerometer data, we estimated the activity and automatically saved the images in the corresponding folders for the three activity types. All the images underwent manual inspection to correct potential misclassifications from the accelerometers, with images moved to the correct folders in case of errors. Images associated with challenging situations, such as those affected by occlusion, were deleted. In total, we collected 6530 images, with 1714 images representing grazing, 2022 lying, and 2794 other behaviors. We split the images into a training set (70%) and a test set (30%). On the test data, the recall rates were 96.1% for grazing, 89.8% for lying, and 86.4% for other behaviors. The precision rates were 83.3%, 92.2%, and 93.4% for grazing, lying, and other behaviors, respectively.

For both behavioral traits, the estimation errors were averaged per method. We were particularly interested in studying the variation of behavioral errors as a function of tracking efficacy.

Data availibility

The code are available on GitHub at the address https://gitlab.com/inra-urz/puzzle-livestock-tracking. The Yolo detector and the data used to train it are described in the datapaper Vayssade et al.[23].

Received: 4 March 2024; Accepted: 31 July 2024
Published online: 07 August 2024

References

1. Ord, T. J., Martins, E. P., Thakur, S., Mane, K. K. & Börner, K. Trends in animal behaviour research (1968–2002): Ethoinformatics and the mining of library databases. *Anim. Behav.* **69**(6), 1399–1413. https://doi.org/10.1016/j.anbehav.2004.08.020 (2005).

2. Weary, D. M., Huzzey, J. M. & Von Keyserlingk, M. A. G. Board-invited review: Using behavior to predict and identify ill health in animals. *J. Anim. Sci.* **87**(2), 770–777. https://doi.org/10.2527/jas.2008-1297 (2009).
3. Morgan-Davies, C. *et al.* Impacts of using a precision livestock system targeted approach in mountain sheep flocks. *Livest. Sci.* **208**, 67–76 (2018).
4. He, Y. *et al.* Antibiotic resistance genes from livestock waste: Occurrence, dissemination, and treatment. *NPJ Clean Water* **3**(1), 4 (2020).
5. Alonso, M. E., González-Montaña, J. R. & Lomillos, J. M. Consumers' concerns and perceptions of farm animal welfare. *Animals* **10**(3), 385 (2020).
6. Thibault, M., Pailler, S. & Freund, D. Why are they buying it?: United states consumers' intentions when purchasing meat, eggs, and dairy with welfare-related labels. *Food Ethics* **7**(2), 12 (2022).
7. Gorton, M. *et al.* Consumers' willingness to pay for an animal welfare food label. *Ecol. Econ.* **209**, 107852 (2023).
8. Hutchings, M. R., Knowler, K. J., McAnulty, R. & McEwan, J. C. Genetically resistant sheep avoid parasites to a greater extent than do susceptible sheep. *Proc. Biol. Sci.* **274**, 1839–1844. https://doi.org/10.1098/rspb.2007.0398 (2018).
9. Dochtermann, N. A., Schwab, T., Anderson Berdal, M., Dalos, J. & Royauté, R. The heritability of behavior: A meta-analysis. *J. Hered.* **110**(4), 403–410. https://doi.org/10.1093/jhered/esz023 (2019).
10. Hazard, D. *et al.* Genetic parameters estimates for ewes' behavioural reactivity towards their litter after lambing. *J. Anim. Breed. Genet.* **137**(4), 374–383. https://doi.org/10.1111/jbg.12474 (2020).
11. Girardie, O. *et al.* Analysis of image-based sow activity patterns reveals several associations with piglet survival and early growth. *Front. Vet. Sci.* **9**, 1051284 (2023).
12. Valletta, J. J., Torney, C., Kings, M., Thornton, A. & Madden, J. Applications of machine learning in animal behaviour studies. *Anim. Behav.* **124**, 203–220 (2017).
13. Kleanthous, N. *et al.* A survey of machine learning approaches in animal behaviour. *Neurocomputing* **491**, 442–463. https://doi.org/10.1016/j.neucom.2021.10.126 (2022).
14. Li, G. *et al.* Practices and applications of convolutional neural network-based computer vision systems in animal farming: A review. *Sensors* **21**(4), 1492 (2021).
15. Luo, W. *et al.* Multiple object tracking: A literature review. *Artif. Intell.* **293**, 103448. https://doi.org/10.1016/j.artint.2020.103448 (2021).
16. Zhang, Y., Wang, C., Wang, X., Zeng, W. & Liu, W. FairMOT: On the fairness of detection and re-identification in multiple object tracking. *Int. J. Comput. Vis.* **129**(11), 3069–3087 (2021).
17. Wojke, N., Bewley, A. & Paulus, D. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing* (ICIP) (2017).
18. Vayssade, J.-A., Godard, X. & Bonneau, M. Wizard: Unsupervised goats tracking algorithm. *Comput. Electron. Agric.* **209**, 107831. https://doi.org/10.1016/j.compag.2023.107831 (2023).
19. Wojke, N., Bewley, A. & Paulus, D. Simple online and realtime tracking with a deep association metric. arXiv:1703.07402 (2017).
20. Aharon, N., Orfaig, R. & Bobrovsky, B.-Z. BoT-SORT: Robust associations multi-pedestrian tracking. arXiv:2206.14651 (2022).
21. Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W. & Wang, X. ByteTrack: Multi-object tracking by associating every detection box. In *European Conference on Computer Vision*, 1–21 (Springer Nature, Cham, 2022).
22. Wang, C., Wang, Y., Wang, Y., Wu, C.-T., & Yu, G. muSSP: Efficient min-cost flow algorithm for multi-object tracking. In *Advances in Neural Information Processing Systems*, 423–432 (2019).
23. Vayssade, J.-A., Arquet, R., Troupe, W. & Bonneau, M. Cherrychèvre: A fine-grained dataset for goat detection in natural environments. *Sci. Data* **10**(1), 689 (2023).
24. Sun, Q., Liu, Y., Chua, T.-S. & Schiele, B. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 403–412 (2019).
25. Afham, M., Khan, S., Khan, M. H., Naseer, M. & Khan, F. S. Rich semantics improve few-shot learning. arXiv:2104.12709 (2021).
26. Woo, S., Park, J., Lee, J.-Y. & Kweon, I.S. CBAM: Convolutional block attention module. In *Proceedings of the European conference on computer vision* (ECCV) (2018).
27. Hu, J., Shen, L., Albanie, S., Sun, G. & Wu, E. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019).
28. Eclarinal, J. C. & Alampay, R. B. YOLO-BAM: Integrating CBAM to the YOLOv3 model for pedestrian detection in images. In *Proceedings of the 2023 5th World Symposium on Software Engineering*, 322–326 (2023).
29. He, L. & Wei, H. CBAM-YOLOv5: A promising network model for wear particle recognition. *Wirel. Commun. Mob. Comput.* **2023**(1), 2520933 (2023).
30. Hendrycks, D. & Gimpel, K. Gaussian error linear units (GELUS) (2016). arxiv:1606.08415.
31. Xu, M. & Zhang, X.-L. Depthwise separable convolutional ResNet with squeeze-and-excitation blocks for small-footprint keyword spotting. In *Interspeech 2020* (ISCA, 2020). https://doi.org/10.21437/interspeech.2020-1045.
32. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G. & Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv:1701.06538 (2017).
33. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems* (2017).
34. Bewley, A., Ge, Z., Ott, L., Ramos, F. & Upcroft, B. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, 3464–3468 (IEEE, 2016).
35. Aharon, N., Orfaig, R. & Bobrovsky, B.-Z. BoT-SORT: Robust associations multi-pedestrian tracking. arXiv:2206.14651 (2022).
36. Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W. & Wang, X. ByteTrack: Multi-object tracking by associating every detection box. In *European Conference on Computer Vision*, 1–21 (Springer Nature, Cham, 2022).
37. Yang, M., Han, G., Yan, B., Zhang, W., Qi, J., Lu, H. & Wang, D. Hybrid-sort: Weak cues matter for online multi-object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2023).
38. Jain, A., Chan, L., Brown, D. S. & Dragan, A. D. Optimal cost design for model predictive control. In *Learning for Dynamics and Control* (2021).
39. Li, S., Kong, Y. & Rezatofighi, H. Learning of global objective for network flow in multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022).
40. Chahine, M. *et al.* Robust flight navigation out of distribution with liquid neural networks. *Sci. Robot.* **8**(77), eadc8892. https://doi.org/10.1126/scirobotics.adc8892 (2023).
41. Miyashita, D., Lee, E.H. & Murmann, B. Convolutional neural networks using logarithmic data representation. arXiv:1603.01025 (2016).
42. Landauer, M., Onder, S., Skopik, F. & Wurzenberger, M. Deep learning for anomaly detection in log data: A survey. *Mach. Learn. Appl.* **12**, 100470 (2022).
43. Fellhauer, A. Approximation of smooth functions using Bernstein polynomials in multiple variables. arXiv:1609.01940 (2016).
44. Sun, H. *et al.* Solving partial differential equation based on Bernstein neural network and extreme learning machine algorithm. *Neural Process. Lett.* **50**, 1153–1172 (2019).
45. Vayssade, J.-A., Paoli, J.-N., Gée, C. & Jones, G. DeepIndices: Remote sensing indices based on approximation of functions through deep-learning, application to uncalibrated vegetation images. *Remote Sens.* **13**(12), 2261 (2021).

46. Vito, V. & Stefanus, L. Y. An asymmetric contrastive loss for handling imbalanced datasets. *Entropy* **24**(9), 1303. https://doi.org/10.3390/e24091303 (2022).
47. Chen, L., Song, J., Zhang, X. & Shang, M. MCFL: Multi-label contrastive focal loss for deep imbalanced pedestrian attribute recognition. *Neural Comput. Appl.* **34**(19), 16701–16715 (2022).

Acknowledgements
We would like to thank Rémy Arquet and all the members of PTEA staff.

Author contributions
J.-A.V.: Wrote the manuscript, Performed the experiment, Analyzed the results, Designed the algorithm. M.B.: Wrote the manuscript, Performed the experiment, Analyzed the results, Initiated and supervised the project.

Funding
The first author received financial support from the French Research National Research Institute for Agriculture, Food and Environment (INRAE-Animal Genetics division). This work was supported by the French National Research Agency, under the project HealtHavior ANR-22-CE32-0005-01. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to M.B.