Extended Target Tracking Utilizing Machine-Learning Software–With Applications to Animal Classification

Magnus Malmström, Anton Kullberg, Isaac Skog, Daniel Axehill and Fredrik Gustafsson

Malmström, M., Kullberg, A., Skog, I., Axehill, D., Gustafsson, F., (2024), Extended Target Tracking Utilizing Machine-Learning Software–With Applications to Animal Classification, *IEEE Signal Processing Letters*, 31, 376-380. https://doi.org/10.1109/LSP.2024.3353165

Extended target tracking utilizing machine-learning software – with applications to animal classification

Magnus Malmström, Anton Kullberg, Isaac Skog *Senior Member, IEEE*, Daniel Axehill *Senior Member, IEEE* , Fredrik Gustafsson *Fellow, IEEE*

***Abstract*—This paper considers the problem of detecting and tracking objects in a sequence of images. The problem is formulated in a filtering framework, using the output of object-detection algorithms as measurements. An extension to the filtering formulation is proposed that incorporates class information from the previous frame to robustify the classification. Further, the properties of the object-detection algorithm are exploited to quantify the uncertainty of the bounding box detection in each frame. The complete filtering method is evaluated on camera trap images of the four large Swedish carnivores, bear, lynx, wolf, and wolverine. The experiments show that the class tracking formulation leads to a more robust classification.**

I. Introduction

This paper considers the problem of detecting and classifying objects in a sequence of images or a video and tracking them over time. In particular, it investigates how to incorporate information of the object's class to improve the robustness of the tracking algorithm. As object detection through neural networks (NNs) becomes widely used in safety-critical applications such as self-driving cars, it becomes of great importance that their predictions are robust and trustworthy. For example, if a pedestrian crosses the road, the car should detect the pedestrian in time to brake to avoid a collision. Further, it is also important to distinguish between different classes of objects, as this may influence the subsequent decision process.

Presented with a sequence of images, it is likely that the detected object belongs to the same class for the entire sequence. By classifying many images assumed to belong to the same class, the probability of correct classification has been shown to increase [1]. Hence, even though there is an error in a particular NN classification, it should be possible to correct the mistake by using information from classifications of previous images in the sequence. The problem can be split into two steps. Firstly, locate the object and classify it using an object-detection algorithm. Secondly, track the object over time, e.g., using a filter.

Manuscript submitted for review 1st of November 2023.

This work is supported by Sweden's innovation agency, Vinnova, through project iQDeep (project number 2018-02700). The authors would like to express their gratitude for the image data provided by Norwegian institute for nature research (NINA), and the Large Carnivore Center (Rovdjurscentret De 5 Stora), with financial support from WWF, for access to real-time camera trap images.

Magnus Malmström is with Linköping University and the Swedish Defence Research Agency (FOI) (e-mail: magnus.malmstrom@liu.se)

Anton Kullberg, Daniel Axehill, and Fredrik Gustafsson, are with Linköping University (e-mail: {firstname.lastname}@liu.se)

Isaac Skog, is with Uppsala University (e-mail: isaac.skog@angstrom.uu.se)

Lately, there have been numerous algorithms developed to solve the object-detection problem, e.g., Single Shot MultiBox detector (SSD) [2], you only look once (YOLO) [3] and its extension [4], region-based convolutional NNs (R-CNN) [5] and its extension [6–8], and CenterNet [9, 10]. These algorithms find and classify the object in the image, whereas none follow it over time.

There has been substantial work on developing algorithms that track bounding boxes in images, e.g., [11–23]. They usually consider one out of two problem formulations that use different solution strategies. The first problem formulation is called visual object tracking, where an object should be tracked over time given a reference frame or a template frame. This problem is commonly solved by introducing new NN architectures [11–19], e.g., a Siamese approach that measures the similarity to an template image. To measure the similarity, [17] use an attention mechanism while [18, 19] use an optimization based approach. The second problem is called multi-object tracking, where, given a video sequence, all objects of interest should be tracked over time. The standard strategy to solve this problem is to view the output of standard object detection algorithms as a measurement for a standard target tracking filter [20–22]. An advantage of the second problem formulation compared with the first one is that the filter formulation enables the fusion of detections from multiple object detection algorithms[24, 25]. Two examples of algorithms that aim to solve the multi-object tracking problem are ByteTrack [20] and SORT [21], which both rely on Kalman filters (KF) and simple motion models to track the objects. As a detection algorithm, ByteTrack uses YOLO-X [4] and SORT uses Faster-R-CNN [7], where here SSD is used. However, neither of the methods track the object's class, which is of interest in safety-critical applications where some classes might be of higher importance than others. Nor do they specify the uncertainty in the measurement of the bounding boxes to be tracked. Previous work has included class information in a tracking framework to make the association step more robust [26]. Thus, accurately tracking the class of the object(s) in the scene is of high interest.

With camera technology getting cheaper, more compact, and more durable, it enables the use of edge devices for camera surveillance systems over larger areas, e.g., for monitoring animals in national parks and animal sanctuaries. Carnivores, such as lynx and wolves, are keystone species in the European wilderness [27]. Hence, there have been attempts to reintroduce them by organizations such as Rewilding Europe. However, collaboration and acceptance from the general public

are important to reintroduce them successfully. Here, a camera monitoring system can warn the general public, count the number of individuals [28], and be used as a warning system for poaching [29]. Camera traps provide a sequence of images, often of bad quality and taken in poor lighting conditions. Here, one approach to increase the accuracy in the prediction of the object is to propagate the information over the entire image sequence.

The contribution of this work is threefold. Firstly, we formulate the tracking and detection of an object in a sequence of images as a filtering problem, where the measurements come from a standard object detection algorithm. The standard filtering problem formulation is extended, such that the uncertainty in the position of the bounding boxes is estimated. Secondly, we propose a method to systematically adjust how much information regarding the object's class from previous frames should be considered in the classification. Thirdly, the method is evaluated on a challenging task using camera trap images collected in Swedish forests for an animal conservation project.

II. Extended object tracking

Consider the problem of tracking an object and its class in a sequence of images given detections from a detection algorithm such as SSD. It will be assumed that every image only consists of one object to make the notation more concise. However, the method can easily be extended to cover several objects using an association process based on the intersection over union (IoU) between the bounding boxes.

A. States in the tracking algorithm

Denote $x \in \mathbb{R}^{n_x}$ as an image with n_x pixels, i.e., the input data to the detection algorithm. Further let $y_n \in \{1, \dots, M{+}1\}$ denotes the M class labels of the object in the image and the background class. Define the states

$$\chi^b = \begin{bmatrix} p_x & p_y & l & h \end{bmatrix}^\top, \text{ and} \tag{1a}$$

$$[\chi^c]_m = p(y = m|x), \quad m = 1, \dots, M+1, \tag{1b}$$

where $\chi^b \in \mathbb{R}^4$ represents the position and size of the bounding box of an object in the image and $\chi^c \in \mathbb{R}^{M+1}$ the confidence in the different classes in that box. Here $[\cdot]_m$ denotes the m'th element of a vector, i.e., in (1b), it is used to denote the probability that the object belongs to the m'th class. Further, (p_x, p_y) is the center, and l and h are the length and height of the bounding box, respectively.

At each time t, assume that a measurement $z_t^b = \chi_t^b + \boldsymbol{e}_t$ of the bounding box position and its size is available, with measurement noise $\boldsymbol{e}_t \sim \mathcal{N}(0, R_t^b)$. Here, R_t^b is the covariance of the estimated bounding boxes from the object detection algorithm. The position and size of the bounding box are assumed to follow a linear motion model with additive process noise $\boldsymbol{v}_t \sim \mathcal{N}(0, Q)$. The covariance of the process noise Q could be class dependent [30], e.g., different classes move at different speeds. Since the state-space model is linear, a KF can be used to solve the filtering problem. In the experiment, a constant position motion model is assumed.

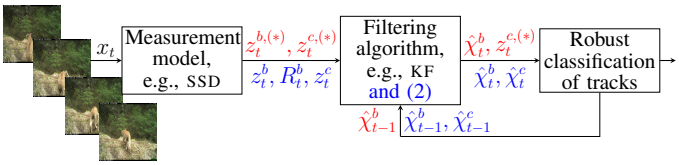

Fig. 1: Schematic illustration of the suggested filtering framework. In red are the quantities commonly used in multi-object tracking applications, and in blue are the quantities used in this paper.

B. Robust classification

Assume that the object's class in the image is categorically distributed and that the state χ_t^c stays the same between the images, i.e., $\chi_t^c = \chi_{t-1}^c$. An estimate of the probability vector for the categorical distribution is given by $\hat{\chi}_t^c$. A measurement of the probability for the object's class is given by z_t^c. Under the assumption that the estimate of the probability at time $t-1$ influences the estimated probability at time t, i.e., the same object is tracked over time, this influence should be included in the measurement update. Using a filtering formulation, this results in

$$\hat{\chi}_t^c = (1 - K_t^c)\hat{\chi}_{t-1}^c + K_t^c z_t^c \tag{2}$$

where $K_t^c \in [0, 1]$ weighs how much influence the measurement of the class probability at time t should have on the estimated PMF for the object's class in the image sequence. The formulation in (2) makes the tracking algorithm more robust against "incorrect" measurements, where $1 - K_t^c$ can be interpreted as a forgetting factor of the object's class. There are many different approaches to selecting K_t^c, e.g., formulating an optimization problem to weigh the influence between measurements and old states, using a forgetting factor or the median. In this paper, the value of K_t^c will be selected such that the estimated state $\hat{\chi}_t^c$ is an average of the previous measurements and the prior, i.e., $K_t^c = 1/(t+1)$. This choice is reasonable since if an object has been seen for a long time, it is unlikely its class would change, i.e., when $t \to \infty$ then $K_t^c \to 0$.

A schematic illustration of the filtering framework to solve the tracking problem can be seen in Fig. 1. Here, the filtering algorithm includes information of the object's class using (2). Note that other methods, e.g., ByteTrack and SORT, can also easily be extended to include the information of the object's class by extending the used filtering algorithm with (2).

This paper focuses on making the filter formulation more robust against incorrect classifications. In a target tracking framework, a *track* is often defined as the estimated history of a target. In such a framework, it is crucial to know when an object appears or disappears from the sensor's field of view to kill and give birth to new tracks. Here, the estimated probability mass function (PMF) $\hat{\chi}_t^c$ is used to determine whether to kill a track, e.g., if $\max \hat{\chi}_t^c$ is below a given threshold the track is killed. Similarly, a new track can be born if $\max \hat{\chi}_t^c$ is above some threshold.

III. Measurement model

This section will describe how to use standard object-detection algorithms to generate measurements for a tracking

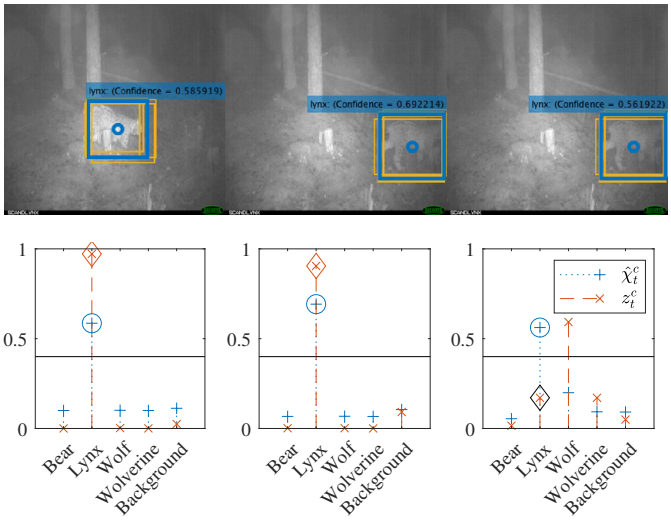

Fig. 2: Estimation of the PMF for the object class in a sequence of images. Top: The camera image sequence of the lynx with the estimated bounding box $\hat{\chi}_t^b$ in thick blue and the measurement from the detection algorithm in thin yellow, $z^{b,(i)}$. Bottom: In blue, the estimated PMF denoted $\hat{\chi}_t^c$ from (2), and in orange, the measurement of the PMF z_t^c from (8a). In black is a decision line on whether a track is lost. The dimond marker in the PMF plot denotes the class of the estimated track. It changes color when the track is lost.

TABLE I: Number of lost tracks at the last image in the sequence.

Detection using	Proposed, $\hat{\chi}_t^c$, (2)	Standard, z_t^c, (8a)
Number of lost tracks	2/20	20/20

algorithm.

A. Object detection

Consider the problem of learning a detector used to detect and classify objects in an image from the dataset

$$\mathcal{T} \triangleq \{x_n, y_n^j, b_n^j\}_{n=1}^N, \quad j = 1, \dots, J_n. \tag{3}$$

Here $y_n^j \in \{1, \dots, M\}$ is the class label of the object, and $b_n^j \in \mathbb{R}^4$ is the shape of the bounding box in which the object is located. The subindex j denotes the j'th object of the J_n objects in the image. From a statistical point of view, learning the detector can be formulated as a system identification problem where one simultaneously identifies a model $f^b(x;\theta)$ for the bounding boxes and a model $f^c(x;\theta)$ for the conditional PMF $p(y^j = m|x), \quad m = 1, \dots, M+1$, of a categorical distribution, where an extra class for the background is added. Here $\theta \in \mathbb{R}^{n_\theta}$ denotes the n_θ dimensional parameter vector of the parametrized model.

The models $f^b(x;\theta)$ and $f^c(x;\theta)$ are often based on a pre-trained convolutional NN (CNN) [2–7], here referred to as the backbone NN. The parameters in the model are the weights and biases of the CNN. The superscript c stands for classification and b for bounding box.

B. Single Shot MultiBox detector

This paper focuses on using SSD [2] as the detection algorithm. However, the proposed method is more general and could be applied to other detection algorithms that use anchor boxes, e.g., YOLO. Here, anchor boxes are predefined boxes bounding the object in the images. One of the key contributions of this paper is how to use the anchor boxes to compute the measurements for the tracking algorithm in such a way that the covariance of the measurements is included. This is something not commonly done in the literature. The knowledge of the covariance simplifies the tuning of the KF.

For SSD, the backbone NN is branched off at R different hidden layers, where each branch is responsible for detecting objects of different sizes. The classification and bounding box regression will be split into different branches. Each of those branches represents a predetermined grid. For grid r with γ_r grid points, α_r predetermined anchor boxes are specified. Then, anchor boxes are placed at every grid point. That is, for each image, the SSD detects $N_b = \prod_{r=1}^R \gamma_r \alpha_r$ bounding boxes with corresponding confidence per class, i.e., $f^c(x;\theta)^{(i)}$ and $f^b(x;\theta)^{(i)}$ where $i = 1 \dots N_b$.

The estimate of the model parameters is given by

$$\hat{\theta}_N = \arg\min_\theta \sum_{n=1}^N \frac{\big(L_c(\theta, x_n, y_n) + \alpha L_b(\theta, x_n, y_n, b_n)\big)}{N_m}. \tag{4}$$

Here the loss function is the weighted sum between a classification loss L_c and a location loss L_b, using the weighting parameter α [2]. Further, N_m is the number of matched boxes, i.e., boxes with a probability larger than some predetermined threshold. Define the so-called positive indicator variables $\xi_{ij}^{y^j} = \{0, 1\}$ and negative indicator variable $\xi_i^- = \{0, 1\}$. The positive indicator variable is equal to one if the predicted bounding box i matches the ground-truth bounding box j with the class label y^j, and the negative indicator variable is used to indicate that the predicted bounding box does not overlap with any of the ground-truth bounding boxes. The classification loss is based on the assumption that the classes (including the background class) in the boxes are categorically distributed. Hence, the classification loss is given as

$$\begin{aligned} L_c(\theta, x_n, y_n) = &- \sum_{i=1}^{N_b} \sum_{j=1}^{J_n} \xi_{ij}^{y_n^j} \log(f_{y_n^j}^c(x_n;\theta)^{(i)}) \\ &- \sum_{i=1}^{N_b} \xi_i^- \log(f_{M+1}^c(x_n;\theta)^{(i)}), \end{aligned} \tag{5a}$$

where both boxes containing objects and boxes not containing objects are represented. The localization loss was chosen as

$$L_b(\theta, x_n, y_n, b_n) = \sum_{i=1}^{N_b} \sum_{j=1}^{J_n} \xi_{ij}^{y_n^j} l_{L1}(f^b(x;\hat{\theta}_N)^{(i)} - b_n^j) \tag{5b}$$

where l_{L1} is the so-called smooth $L1$ loss defined as $l_{L1}(x) \triangleq \{||x||_2^2/(2\xi), \ ||x||_1 < \xi, \ ||x||_1 - 0.5\xi, \ \text{otherwise}\}$. Here $||.||_i$ is used to define the i'th norm of the vector.

In the prediction phase, non-maximum suppression (NMS) is typically used to remove overlapping boxes and boxes with too low confidence. Hence only keeping one box per object in the image. Define the most probable class as

$$\hat{y}^* = \arg\max_{m=1,\dots,M} f_m^c(x;\hat{\theta}_N)^{(*)}, \tag{6}$$

where $*$ indicates the index of the bounding box with the highest probability of including an object out of the N_b predicted boxes. If there are multiple boxes with high confidence for which there is no overlap, they are stored as separate objects.

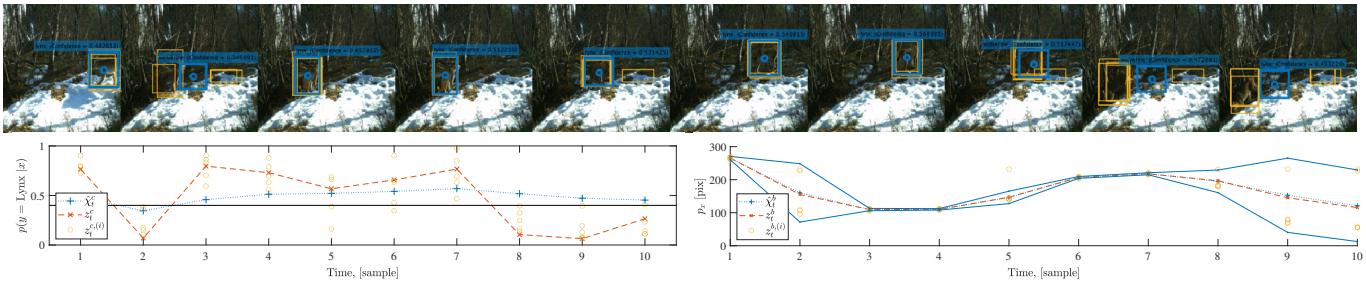

Fig. 3: On the top is the image sequence of a lynx, followed by the probability of a lynx in the image and the tracking of the x-position of the bounding box p_x over time. The estimated states are shown in blue dotted lines, the measurement to the filter in orange dashed lines, and the measurements from the individual anchor boxes in yellow circles. The blue solid lines shows the 3σ uncertainty in object location. The black line is the threshold used to determine if a track should be initiated or has been lost.

C. Measurement model

Instead of using NMS and only keeping the most likely detection of an object, the B most likely anchor boxes/proposals could be used. That is

$$z_t^{b,(i)} \triangleq f^b(x_t;\hat{\theta}_N)^{(i)},\ z_t^{b,(i)} \triangleq f^c(x_t;\hat{\theta}_N)^{(i)},\ i=1,\dots,B. \quad (7)$$

The proposals in (7) are used to create measurements z_t^b and z_t^c to the tracking algorithm. More precisely, a weighted mean of these proposals is used, i.e.,

$$z_t^b = \sum_{i=1}^{B} w_i z_t^{b,(i)}, \quad z_t^c = \sum_{i=1}^{B} w_i z_t^{c,(i)}. \quad (8a)$$

The weights are chosen proportional to the relative confidence that the proposed bounding boxes include an object of the most likely class, i.e.,

$$w_i = \frac{\tilde{w}_i}{\sum_{j=1}^{B}\tilde{w}_j} \text{ and } \tilde{w}_i = p(y=\hat{y}^*|z^{c,(i)}), \quad (8b)$$

where $\hat{y}^*$ denotes the most probable class of the object that is in the image, see (6). If there are multiple objects in the images, an association process using IoU can be used to create multiple measurements per image. However, to make the notation more concise, it will again be assumed that the images only include one object. Further, the covariance of the measurement can also be computed as

$$R_t^b = \sum_{i=1}^{B} w_i (z_t^{b,(i)} - z_t^b)(z_t^{b,(i)} - z_t^b)^\top, \quad (9)$$

which is used in the KF. Note that this is not commonly done in the literature.

IV. Experiments

This paper uses image sequences from camera traps in Swedish forests. The traps belong to a project to monitor the four Swedish top carnivores, i.e., bear, lynx, wolf, and wolverine. For the first experiment, a sequence of two correct measurements is followed by an incorrect one. The incorrect measurement is a copy of the previous measurement, but where z_t^c is artificially changed. Here, 20 such sequences are used to evaluate the proposed method. A track is considered lost if $\max \hat{\chi}_t^c < 0.4$ or if the most likely non-background class is changed.

The backbone NN used is a ResNet50 pre-trained on the ImageNet dataset [31], where the SSD is used as a detection algorithm. For the first image, χ_0^c is initialized uniformly distributed, and χ_0^b is initialized as the object's true position. The implementation of the SSD is done using the deep learning toolbox in MATLAB.

Fig. 2 shows one of the 20 sequences where the class measurement for the last frame has been artificially changed. As can be seen, using the information from the previous frames, the object probability stays above the threshold, and the track stays alive. This is even though the last measurement is incorrect. In the evaluation of all 20 sequences, the proposed method only lost track in 2 cases. This should be compared to the standard method, which lost the track in all 20 sequences. In Fig. 3, an experiment is shown where a lynx is tracked over ten frames. It can be seen that using the information from the previous frame results in a more robust prediction, i.e., even though the measurement from the SSD is incorrect $\hat{\chi}_t^c$ indicates the correct class. It is also shown how the position of the bounding box is correctly tracked using the specified measurement and associated measurment covariance.

V. Summary and Conclusions

A framework for joint object detection, classification, and tracking in sequences of images has been presented. Compared to previous works, the proposed framework differs in two key aspects. These are: (i) information about the object class is included in the tracking filter, and (ii) multiple anchor boxes are used to calculate the covariance associated with each object bounding box. The result is a more robust object classification and tracking. An important feature of the proposed method is that it can be used as a standalone add-on to any object detection algorithm that uses proposal anchor boxes without modifying the underlying detection algorithm. The evaluation of the proposed method on sequences of images of Swedish predators with manually induced miss-classifications shows that the method has significantly higher robustness than standard object classification and tracking methods.

References

[1] P. Braca *et al.*, "Statistical hypothesis testing based on machine learning: Large deviations analysis," *IEEE Open J. of Signal Process.*, vol. 3, pp. 464–495, 2022.

[2] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. of 14th European Conf. on Comput. Vision (ECCV)*. Amsterdam, The Netherlands: Springer, 2016, pp. 21–37, october 11–14.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. of IEEE Conf. on Comput. Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 779–788, jun 26 - July 1.

[4] Z. Ge *et al.*, "YOLOX: Exceeding YOLO series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. of IEEE Conf. on Comput. Vision and Pattern Recognition (CVPR)*, Sydney, Australia, 2014, pp. 580–587, 06–11 Aug.

[6] R. Girshick, "Fast R-CNN," in *Proc. of IEEE Int. Conf. on Comput. Vision (ICCV)*, Araucano Park, Las Condes, Chile, 2015, pp. 1440–1448, 11-18, Dec.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Adv. in Neural Inf. Process. Syst. (NIPS) 28*, vol. 28, Montréal, QC, Canada, 2015, 8-13 Dec.

[8] S. Chen, Z. Li, and Z. Tang, "Relation r-cnn: A graph based relation-aware network for object detection," *IEEE Signal Process. Lett.*, vol. 27, pp. 1680–1684, 2020.

[9] K. Duan *et al.*, "Centernet: Keypoint triplets for object detection," in *Proc. of IEEE Int. Conf. on Comput. Vision (ICCV)*, Seoul, South Korea, 2019, pp. 6569–6578, oct 27 – Nov 2.

[10] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[11] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ATOM: Accurate tracking by overlap maximization," in *Proc. of IEEE Conf. on Comput. Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 26–20 June 2019, pp. 4660–4669.

[12] L. Bertinetto *et al.*, "Fully-convolutional siamese networks for object tracking," in *Proc. of 14th European Conf. on Comput. Vision (ECCV) Workshops*. Amsterdam, The Netherlands: Springer, 2016, pp. 850–865, october 11–14.

[13] S. M. Marvasti-Zadeh, L. Cheng, H. Ghanei-Yakhdan, and S. Kasaei, "Deep learning for visual tracking: A comprehensive survey," *IEEE Trans. Intell. Transp. Syst.*, 2021.

[14] J. Zhang, Y. He, and S. Wang, "Learning adaptive sparse spatially-regularized correlation filters for visual tracking," *IEEE Signal Process. Lett.*, 2023.

[15] H. Wang *et al.*, "Robust visual tracking via semiadaptive weighted convolutional features," *IEEE Signal Process. Lett.*, vol. 25, no. 5, pp. 670–674, 2018.

[16] D. Yuan, X. Shu, Q. Liu, and Z. He, "Aligned spatial-temporal memory network for thermal infrared target tracking," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 70, no. 3, pp. 1224–1228, 2022.

[17] T. Xu, Z. Feng, X.-J. Wu, and J. Kittler, "Toward robust visual object tracking with independent target-agnostic detection and effective siamese cross-task interaction," *IEEE Trans. Image Process.*, vol. 32, pp. 1541–1554, 2023.

[18] D. Yuan *et al.*, "Active learning for deep visual tracking," *IEEE Trans. Neural Netw. Learn. Syst.*, 2023.

[19] G. Bhat, M. Danelljan, L. V. Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *Proc. of IEEE Int. Conf. on Comput. Vision (ICCV)*, Seol, South Korea, 27 Oct–2 Nov 2019, pp. 6182–6191.

[20] Y. Zhang *et al.*, "Bytetrack: Multi-object tracking by associating every detection box," in *Proc. of 17th European Conf. on Comput. Vision (ECCV)*. Tel Aviv, Israel: Springer, 2022, pp. 1–21, 23-27 Oct.

[21] A. Bewley *et al.*, "Simple online and realtime tracking," in *Proc. of IEEE Int. Conf. on Image Process. (ICIP)*. Phoenix, AZ, USA: IEEE, 2016, pp. 3464–3468, 25-28 Sep.

[22] Z. Wang *et al.*, "Towards real-time multi-object tracking," in *Proc. of 16th European Conf. on Comput. Vision (ECCV)*. Glasgow, UK/Online: Springer, 2020, pp. 107–122, 23-28 Aug.

[23] J. Han *et al.*, "Advanced deep-learning techniques for salient and category-specific object detection: a survey," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 84–100, 2018.

[24] D. Yuan *et al.*, "Robust thermal infrared tracking via an adaptively multi-feature fusion model," *Neural Comput. and App.*, vol. 35, no. 4, pp. 3423–3434, 2023.

[25] R. Fatima *et al.*, "Multiple passive-sensor distributed target tracking approach with machine learning feedback," *Expert Syst. with App.*, vol. 238, p. 122344, 2024.

[26] D. Gaglione *et al.*, "Classification-aided multitarget tracking using the sum-product algorithm," *IEEE Signal Process. Lett.*, vol. 27, pp. 1710–1714, 2020.

[27] S. Hoeks *et al.*, "Mechanistic insights into the role of large carnivores for ecosystem structure and functioning," *Ecography*, vol. 43, no. 12, pp. 1752–1763, 2020.

[28] O. R. Wearn and P. Glover-Kapfer, "Camera-trapping for conservation: a guide to best-practices," World Wildlife Fund (WWF), Technical Specification (TS), 10 2017. [Online]. Available: https://www.wwf.org.uk/sites/default/files/2019-04/CameraTraps-WWF-guidelines.pdf

[29] A. Tydén and S. Olsson, "Edge machine learning for animal detection, classification, and tracking," Master's thesis, Dept. Elect. Eng., Linköping University, Norrköping, Sweden, Jun. 2020.

[30] G. Soldi *et al.*, "Space-based global maritime surveillance. part ii: Artificial intelligence and data fusion techniques," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 36, no. 9, pp. 30–42, 2021.

[31] J. Deng *et al.*, "Imagenet: A large-scale hierarchical image database," in *Proc. of IEEE Conf. on Comput. Vision and Pattern Recognition (CVPR)*. Miami, FL, USA: Ieee, 2009, pp. 248–255, jun 20-25.