International Journal of Computer Vision (2023) 131:2267–2297
https://doi.org/10.1007/s11263-023-01781-2

Towards Automated Ethogramming: Cognitively-Inspired Event Segmentation for Streaming Wildlife Video Monitoring

Ramy Mounir[1] · **Ahmed Shahabaz**[1] · **Roman Gula**[2] · **Jörn Theuerkauf**[2] · **Sudeep Sarkar**[1]

Received: 29 April 2022 / Accepted: 12 March 2023 / Published online: 28 April 2023

Abstract

Advances in visual perceptual tasks have been mainly driven by the amount, and types, of annotations of large-scale datasets. Researchers have focused on fully-supervised settings to train models using offline epoch-based schemes. Despite the evident advancements, limitations and cost of manually annotated datasets have hindered further development for event perceptual tasks, such as detection and localization of objects and events in videos. The problem is more apparent in zoological applications due to the scarcity of annotations and length of videos-most videos are at most ten minutes long. Inspired by cognitive theories, we present a self-supervised perceptual prediction framework to tackle the problem of temporal event segmentation by building a stable representation of event-related objects. The approach is simple but effective. We rely on LSTM predictions of high-level features computed by a standard deep learning backbone. For spatial segmentation, the stable representation of the object is used by an attention mechanism to filter the input features before the prediction step. The self-learned attention maps effectively localize the object as a side effect of perceptual prediction. We demonstrate our approach on long videos from continuous wildlife video monitoring, spanning multiple days at 25 FPS. We aim to facilitate automated ethogramming by detecting and localizing events without the need for labels. Our approach is trained in an online manner on streaming input and requires only a single pass through the video, with no separate training set. Given the lack of long and realistic (includes real-world challenges) datasets, we introduce a new wildlife video dataset–nest monitoring of the Kagu (a flightless bird from New Caledonia)–to benchmark our approach. Our dataset features a video from 10 days (over 23 million frames) of continuous monitoring of the Kagu in its natural habitat. We annotate every frame with bounding boxes and event labels. Additionally, each frame is annotated with time-of-day and illumination conditions. We will make the dataset, which is the first of its kind, and the code available to the research community. We find that the approach significantly outperforms other self-supervised, traditional (e.g., Optical Flow, Background Subtraction) and NN-based (e.g., PA-DPC, DINO, iBOT), baselines and performs on par with supervised boundary detection approaches (i.e., PC). At a recall rate of 80%, our best performing model detects one false positive activity every 50 min of training. On average, we at least double the performance of self-supervised approaches for spatial segmentation. Additionally, we show that our approach is robust to various environmental conditions (e.g., moving shadows). We also benchmark the framework on other datasets (i.e., Kinetics-GEBD, TAPOS) from different domains to demonstrate its generalizability. The data and code are available on our project page: https://aix.eng.usf.edu/research_automated_ethogramming.html

Keywords Spatio-Temporal Event Segmentation · Spatial Object Localization · Streaming Input · Automated Ethogramming

Communicated by Helge Rhodin.

✉ Ramy Mounir
ramy@usf.edu

Ahmed Shahabaz
shahabaz@usf.edu

Roman Gula
rgula@miiz.eu

Jörn Theuerkauf
jtheuer@miiz.eu

Sudeep Sarkar
sarkar@usf.edu

[1] Computer Science and Engineering, University of South Florida, Tampa, USA

[2] Museum and Institute of Zoology, Polish Academy of Sciences, Warsaw, Poland

1 Introduction

Much effort in computer vision science has been devoted toward tracking, modeling, and understanding the behavior of humans (Petrovich et al., 2021; Kocabas et al., 2021; Dwivedi et al., 2021; Hesse et al., 2020; Kocabas et al., 2020); however, fewer works explore the same tasks for other species (Testard et al., 2021; Hayden et al., 2021). The scarcity of annotation and difficulty of collecting data has a significant contribution to the slow progress of the field (Hayden et al., 2021). Recent developments in digital video and recording technology opened new opportunities to study animal behavior in the wild. Not only do these recent developments increase the volume of data, but also the speed at which data is generated. Monitoring video systems can generate data at faster rates than those can be handled by AI systems, leading to the notion of fast data (Lam et al., 2012). Batch (offline) learning systems fail to process fast data, where–possibly infinite–items arrive in a temporal sequence (Sahoo et al., 2018), due to the need to access old data. Revisiting old data not only wastes computational resources but also requires that the data be stored, which is impractical for large volumes of data and hinders the scalability of the learning systems. For streaming input applications (e.g., monitoring and surveillance), storage costs can exceed computational costs. The goal is to process and summarize streaming input data to store only useful high-level information, such as detected events and their objects' attributes, while ignoring spatial and temporal background clutter.

Stream Learning (SL) aims to build adaptive models of the streaming data (Benczúr et al., 2018). These models must be updated after every datapoint without access to any past data. Additionally, the SL systems need to account for distribution shifts by adapting to the changes in the streaming data. Our training scheme completely disregards data points after being processed by the network. Training and inference are done simultaneously, alleviating the need for epoch-based training in order to appeal to practical applications and reduce training time. We address the distribution shift problem by building adaptable *event models* which contain a feature representation of the event and its objects (i.e., bird), allowing us to effectively segment the object from its surrounding at every frame–despite changes in illumination and environmental conditions.

Ethology is the scientific and objective study of animal behavior, from which stems the term *Ethogram*. Ethogramming defines and categorizes animal behaviors in a completely objective manner. It is important to avoid subjectivity and bias when describing animal behavior in ethograms; behavior definitions must be based on "mechanical actions that are observed rather than on any intentionality, motivating the expression of that behavior" Rose and Riley (2021). The required objectivity in the behavior descriptions makes computer vision algorithms a suitable candidate for automating ethograms. Ethogramming can ideally be automated by correlating low-level attributes (e.g., location, motion patterns, time-of-day) to high-level behaviors (e.g., incubation) Hayden et al. (2021). Our goal is to detect the low-level attributes used in describing the behavior of animals in ethograms, such that these attributes can then be used to generate and localize objective behavior labels automatically.

Events are ubiquitous; they are the building blocks of videos. Events can be present at various time scales, whereas actions are a type of event that occurs at a smaller temporal scale. For example, the event of a bird feeding its chick can be composed of multiple feeding actions, where *feeding* here refers to the activity. To truly understand videos, algorithms must be capable of detecting and segmenting significant events from background noise. The detected events can then be encoded in an embedding space for representation learning (Pan et al., 2021; Lin et al., 2021), annotated and classified (Arnab et al., 2021; Liu et al., 2021; Yan et al., 2022; Li et al., 2021), or even used for video summarization tasks (Apostolidis et al., 2021; Ji et al., 2019; Fajtl et al., 2018; Zhu et al., 2020). The task of detecting prominent events becomes even more important when processing long videos for wildlife monitoring or even video monitoring of other contexts (Corona et al., 2021). To perceive events, one must not rely on the noisy low-level features but instead build algorithms that detect the dynamics and patterns of high-level features. These high-level features should be sufficient to capture object-level representations; the designed architectures should ideally capture these features' temporal evolution over time.

Very few publications show event detection and localization performance on video spanning several days, mainly due to the challenges arising from storing, processing, and evaluating on large datasets. Researchers have mainly focused on event-centric datasets by using trimmed videos of actions, thus eliminating the need to segment events. This is analogous to removing background from images and using only masks of objects to train an object classifier. Although this makes the task easier on the learning model, it becomes harder to generalize to real-world applications. Real-world datasets contain more challenging scenarios of untrimmed events (e.g., empty segments) and noisy low-level changes in the foreground (e.g., occlusions, shadows). Most wildlife monitoring studies use camera traps with motion triggers to store images of animals or record short videos of animals in controlled environments – eliminating real-world lighting variations and environmental conditions. We address the issues mentioned above by collecting and annotating a ten-day continuous monitoring dataset of a nest of the Kagu; a flightless bird of New Caledonia. The dataset is sampled at 25 FPS, offering more than 23 million frames and more than 253 h of video footage. Events are not trimmed to allow researchers to evaluate the models' performances on more realistic and challenging scenarios. Ideally, algorithms will have to deal with overfitting, catastrophic forgetting, and sparsity of events to maintain high performance in spatial and temporal event segmentation tasks.

Our framework follows key ideas from the perceptual prediction line of work in cognitive psychology (Zacks et al., 2001; Zacks & Tversky, 2001; Zacks & Swallow, 2007; Loschky et al., 2020). Research has shown that "event segmentation is an ongoing process in human perception, which helps form the basis of memory and learning" (Radvansky & Zacks, 2014; Zacks et al., 2007). Humans can identify event boundaries, in a purely bottom-up fashion, using a perceptual predictive model that predicts future states based on the current perceived sensory information. Experiments have shown that the human perceptual system identifies event boundaries based on the appearance and motion cues in the video (Zacks, 2004; Speer et al., 2003; Radvansky et al., 2011). Our model implements this perceptual predictive framework and introduces a motion-weighted loss function to allow for the localization and processing of motion cues.

As shown in Fig. 1, our approach uses a feature encoding network to transform low-level perceptual information into a higher-level feature representation. The model is trained to predict the future perceptual encoded input and signal an event if the prediction is significantly different from the future perceived features (i.e., unpredictable features). The prediction error signal is used to flag events and train the model to predict better features. Prediction occurs after aggregating higher-level features with a recurrent cell; the hidden state incorporates a higher-level representation of the movement cues within frames. We utilize an attention mechanism on the backbone features to provide a more focused prediction error signal and, more importantly, spatially segment the frames. The error signal is used for temporal event segmentation, while an attention map segments a frame spatially. Our approach takes a significant step toward an eventual system that can detect adverse behavioral events without requiring a dataset of such behaviors. We aim to facilitate the goal of automated ethogramming; detecting and characterizing behaviors of animals objectively without prior domain information or low-level annotations.

Our **key contributions** can be summarized as:

- Introducing a stream learning framework capable of detecting and localizing prominent events without prior knowledge of the target events.
- Collecting and annotating a remarkably long wildlife video monitoring dataset (over 23 M frames) with frame-level bounding box annotation and behavioral categories. For all frames containing the bird (i.e., moving or stationary), a single frame-level bounding box is provided to highlight its location.
- Providing extensive evaluation of our framework's performance with respect to other traditional and state-of-the-art approaches on multiple domains and datasets.

2 Relevant Work

This section presents a literature review of relevant works. We begin by introducing supervised approaches targeting event segmentation and boundary detection in Sect. 2.1, followed by a review of self-supervised event segmentation approaches in Sect. 2.2. We also provide a review of other approaches that share a similar predictive learning intuition for training models without explicit supervision (Sect. 2.3). Additionally, a short review of attention-based models and continual learning approaches can be found in Sects. 2.4 and 2.5, respectively. We conclude by presenting animal-related computer vision works and comparison of the Kagu video monitoring dataset to other animal datasets provided in different formats (e.g., images, videos, features) in Sects. 2.6 and 2.7.

2.1 Fully Supervised Approaches

Supervised temporal event segmentation uses direct labeling (of frames) to segment videos into smaller constituent events. Fully supervised models are heavily dependent on a vast amount of training data to achieve good segmentation results. Different model variations and approaches have been tested, such as using an encoder-decoder temporal convolutional network (ED-TCN) Lea et al. (2017) or a spatiotemporal CNN model (Lea et al., 2016). To alleviate the need for expensive direct labeling, weakly supervised approaches (Richard et al., 2017; Bojanowski et al., 2014; Ding & Xu, 2018; Huang et al., 2016) have emerged with an attempt to use metadata (such as captions or narrations) to guide the training process without the need for explicit training labels (Malmaud et al., 2015; Alayrac et al., 2016). However, such metadata are not always available as part of the dataset, making weakly supervised approaches inapplicable to most practical applications. UnweaveNet (Price et al., 2022) introduces the task unweaving, which defines each activity as a single thread and builds a thread bank to segment and label parts of activities in an untrimmed video. UnweaveNet uses a supervised transformer architecture to learn temporal dependencies between clips.

Supervised boundary detection approaches use boundary labels as a supervised learning signal for the model. For example, Lea et al. (2017, 2016); Lin et al. (2018); Shao et al. (2020); Shou et al. (2021) use a binary classifier head

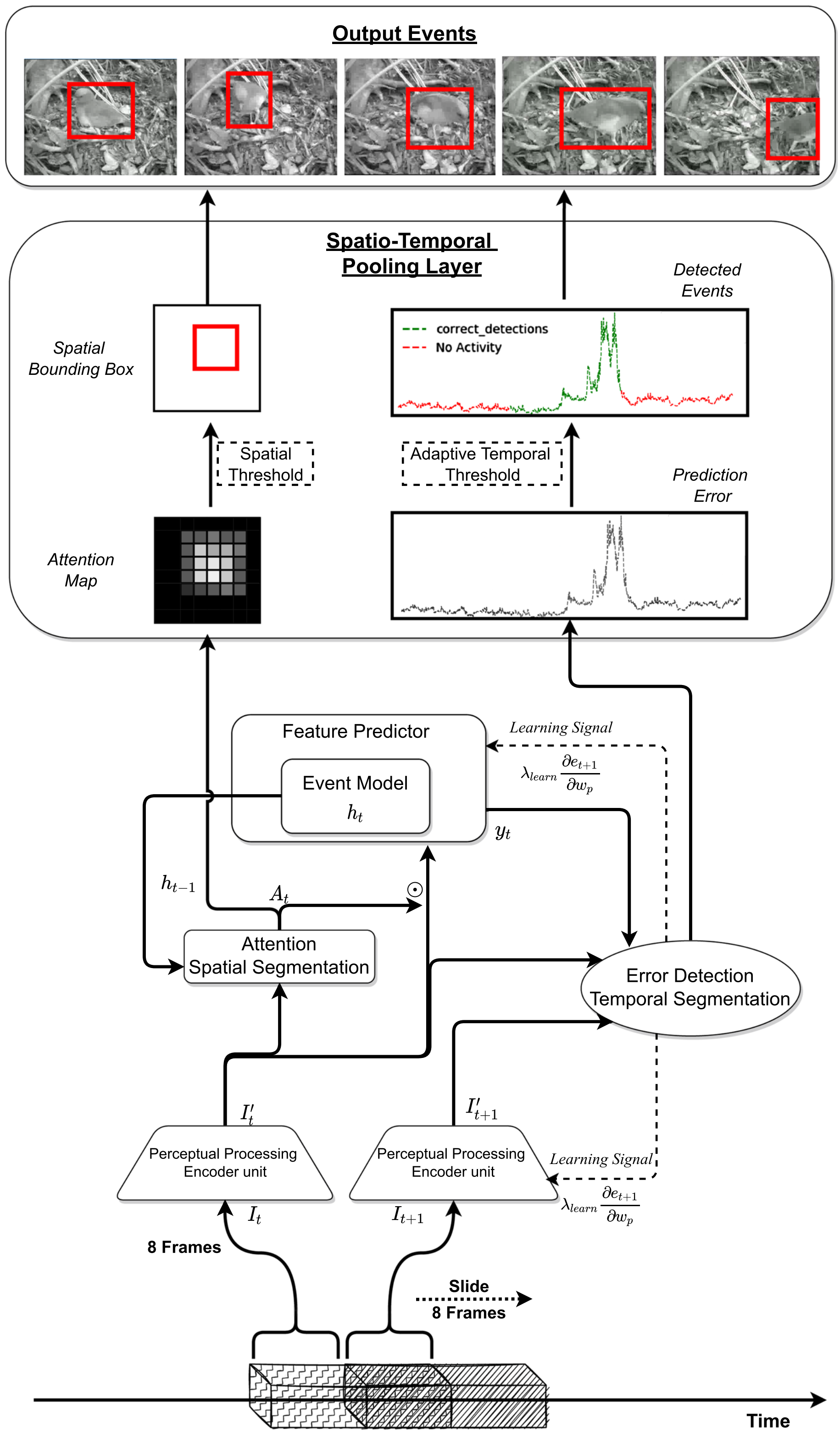

Output Events
Spatio-Temporal Pooling Layer
Spatial Bounding Box
Detected Events
correct_detections
No Activity
Spatial Threshold
Adaptive Temporal Threshold
Prediction Error
Attention Map
Feature Predictor
Event Model
h_t
Learning Signal
$\lambda_{learn} \frac{\partial e_{t+1}}{\partial w_p}$
y_t
h_{t-1}
A_t
$\odot$
Attention Spatial Segmentation
Error Detection Temporal Segmentation
I'_t
I'_{t+1}
Perceptual Processing Encoder unit
Perceptual Processing Encoder unit
Learning Signal
$\lambda_{learn} \frac{\partial e_{t+1}}{\partial w_p}$
I_t
I_{t+1}
8 Frames
Slide 8 Frames
Time

Fig. 1 Overview of our full architecture. The perceptual processing unit encodes current frames and future frames into a grid feature representations. An attention operation is applied to the current features to spatially segment the event objects. The predictor combines the event model representation with the current features to predict the future features. Error in the prediction is used as a learning signal for the trainable weights. The spatio-temporal pooling layer receives as input spatial localization map and prediction error signal and outputs the detected events

on the extracted features to classify the state of every frame (boundary/non-boundary). The undesirable need for direct labeling gave rise to a family of approaches categorized as "weakly-supervised" methods. Similar to event segmentation, weakly supervised approaches (Ding & Xu, 2018; Huang et al., 2016) make use of certain metadata, such as video narrations or temporal ordering of frames, to provide a learning signal.

2.2 Self-supervised Approaches

Self-supervised temporal event segmentation attempts to completely eliminate the need for annotations (Sener & Yao, 2018),[48]. Many approaches rely heavily on higher-level features clustering of frames to sub-activities (Bhatnagar et al., 2017; VidalMata et al., 2021). The performance of the clustering algorithms in unsupervised event segmentation is proportional to the performance of the embedding/encoding model that transforms frames into higher-level feature representations. Clustering algorithms can be computationally expensive depending on the number of frames to be clustered. Recent work (Aakur & Sarkar, 2019) uses a self-supervised perceptual predictive model to detect event boundaries; we improve upon this model to include an attention unit (other differences are discussed in Sec. 5.1.3), which helps the model focus on the main event-causing object and allows for locating it in each frame. Other work (Metcalf & Leake, 2017) uses a self-supervised perceptual prediction model that is refined over a significant amount of reinforcement learning iterations. Recently (Shou et al., 2021) proposed the use of context embedding difference before and after the target frame to classify the state of each frame (boundary/non-boundary). Despite using a backbone pretrained (Han et al., 2019) on the same domain (Kinetics400), we still outperform (Shou et al., 2021) using our online training approach.

Recent work (Aakur & Sarkar, 2020) has used the prediction loss, with the assistance of pretrained region proposal networks (RPNs) and multi-layer LSTM units, to localize actions. We eliminate the need for RPNs and multi-layer LSTM units by extracting Bahdanau Bahdanau et al. (2014) attention weights prior to the LSTM prediction layer, which allows our model to localize objects of interest, even when stationary. From our experiments, we found that the prediction loss attention tends to fade away as moving objects become stationary, which makes its attention map more similar to results extracted from background subtraction or optical flow. In contrast, our model successfully attends to moving and stationary objects despite variations in environmental conditions, such as moving shadows and lighting changes, as presented in the supplementary videos.

2.3 Predictive Models

Frame predictive models have attempted to provide accurate predictions of the next frame in a sequence (Lotter et al., 2017; Wang et al., 2018; Qiu et al., 2019; wichers et al., 2018; Finn et al., 2016); however, these models focus on predicting future frames in raw pixel format. Such models may generate a prediction loss that only captures frame motion difference with a limited understanding of higher-level features that constitutes event boundaries.

High-level predictive models (Adeli et al., 2021; Huang et al., 2019; Ivanovic & Pavone, 2019; Kosaraju et al., 2019; Salzmann et al., 2020; Mohamed et al., 2020; Wang et al., 2021; Lee et al., 2017; Liang et al., 2019) implement recurrent-based architecture to predict high-level attributes, such as future trajectories and activity classes. In addition to the useful predictions made by the trained models, such models are expected to output useful intermediate representations. However, predicting a small subset of the high-level features (e.g., human trajectories) does not result in a better representation when compared to models trained to extract and predict high-level features directly from RGB images.

Recent representation learning models (Doersch et al., 2015; Grill et al., 2020; Chen & He, 2021; Caron et al., 2021; Zbontar et al., 2021; Bardes et al., 2022) create two transformations of a single image and forces one to predict the other. Such a setting can be thought of as a predictive learning approach applied to images. The goal of these models is to learn a useful representation of images, where distances in the embedding space resemble distances in visual/semantic space. Most works have not yet considered applying the knowledge learned from training these image-based architectures to video-based applications. Our approach implements the predictive model on the temporal dimension of data while using changes between frames as transformations to avoid model collapse.

2.4 Attention Units and Transformers

Attention units were first introduced as a method to eliminate the bottleneck representation in encoder-decoder architectures (Bahdanau et al., 2014; Luong et al., 2015); it gives the decoder a way to weight the inputs (based on their representations) before decoding. The transformer architecture is based on the attention unit and has been extensively applied to NLP tasks (Xu et al., 2015; Vaswani et al., 2017; Devlin et al., 2019; Yang et al., 2019; Zaheer et al., 2020; Radford et al., 2018, 2019; Brown et al., 2020; Liu et al., 2019). The success of transformers in NLP has triggered the rapid increase of the transformers' usage in the vision domain as joint CNN-Transformer architectures (Carion et al., 2020; Girdhar et al., 2019; Kamath et al., 2021; Guan et al., 2022;

Chen et al., 2021), and as stand-alone vision transformer architectures (Dosovitskiy et al., 2020; Arnab et al., 2021; Zhang et al., 2021; Caron et al., 2021)-to list a few.

2.5 Continual and Lifelong Learning

The continual learning task, alternatively known as class incremental learning, focuses on avoiding catastrophic forgetting by evaluating the model's ability to retain the performance of classifying old classes after training on new classes (Mittal et al., 2021; Choi et al., 2021; Sun et al., 2021). Models have unlimited access to the current classes (or tasks) but cannot retrain on previous classes. One popular solution is to learn a generative function of each class and use it to generate class instances to continue training on old classes after moving to new classes. The datasets used in such tasks usually consist of image classes or short videos. For example, the Core50 dataset (Lomonaco & Maltoni, 2017) is comprised of 15-second videos. Additionally, the training schemes used in these approaches are not appropriate for streaming input applications; these models use offline batch-based learning. Incremental learning approaches attempt to *simulate* real-life datasets by not shuffling traditional datasets (e.g., ImageNet, Cifar, MNIST) and training sequentially on class labels to evaluate for catastrophic forgetting. We provide a real-life dataset and a true benchmark containing most of the challenges currently being artificially simulated in continual and incremental learning datasets.

2.6 Computer Vision for Animals

Machine learning models, specifically deep learning approaches, have been rising in popularity among ethologists. Recent models have targeted pose estimation (Fangbemi et al., 2020; Zhang et al., 2021), tracking (McIntosh et al., 2020; Ratnayake et al., 2021; Pedersen et al., 2020) and other tasks (Sarfati et al., 2020; Clapham et al., 2020). However, most approaches rely heavily on full supervision and manually annotated datasets. Similarly, a few works (Tuia et al., 2022; Akçay et al., 2020; Ferreira et al., 2020; Hawkins & DuRant, 2020; Mathis & Mathis, 2020; Valletta et al., 2017; Wang, 2019) have used manual annotations to train deep learning architectures and facilitate behavioral analysis of animals. In contrast to these approaches, we propose a stream learning model that learns and adapts to the data in an online manner without the need for task-specific labels.

2.7 Animal Datasets

The field of animal video monitoring and analysis has recently started growing in popularity in the vision community (Günel et al., 2019; Li et al., 2020; Silvia Zuffi, 2019; Zuffi et al., 2018, 2017; Kanazawa et al., 2016; Sanakoyeu et al., 2020). However, most works focus on image-level species classification and, more recently, pose and keypoint estimation. The types of annotations provided with datasets limit the available tasks to be tackled by researchers. As can be seen from Table 1, large-scale datasets targeting event localization are almost nonexistent-limiting the research in the field of animal behavior analysis. The majority of datasets collect animal images through camera traps which are motion activated. The collected images are then manually filtered to remove false positives triggered by wind or lighting conditions (e.g., moving clouds and falling leaves). The animals detected in camera trap datasets are then manually labeled for classification (i.e., species (Swanson et al., 2015; Welinder et al., 2010) or ID annotations (Li et al., 2019; Holmberg et al., 2009)), localization (i.e., bounding box (Ferrari et al., 2018; Li et al., 2019; Holmberg et al., 2009; Welinder et al., 2010)[123], masks (Welinder et al., 2010), point annotations (Kellenberger et al., 2021)), or pose estimation (Yao et al., 2021; Li et al., 2019).

Video data of continuous monitoring of animal behavior in their natural habitat are hard to find; very few annotated datasets (Burgos-Artizzu et al., 2012; Sun et al., 2021; Lorbach et al., 2018) provide a collection of short videos captured in a controlled environment with event/behavior labels. The scarcity of extensive wildlife video datasets, targeting events, and behavioral analysis, led to the collection and annotation of our dataset. Our dataset features ten consecutive days (11 nights) of continuous monitoring of a Kagu nest. We provide two types of annotations; bounding boxes for spatial segmentation and frame-level event labels for temporal event segmentation. Both types of annotations are provided for the whole dataset (23 million frames). In addition to the dataset's scale, length, and quantity of annotations, our dataset is collected in the wild, in a non-invasive manner.

3 Approach

This section introduces the technical details of our cognitively-inspired stream learning framework. We first discuss the inspiration from cognitive theories in Sect. 3.1. The proposed architecture, summarised in Fig. 2, can be divided into several individual components. We explain the role of each component, starting with the encoder network and attention unit in Sects. 3.2 & 3.3, followed by a discussion on the recurrent predictive layer in Sect. 3.4. We conclude by introducing the loss functions (Sect. 3.5) used for self-supervised learning as well as the output pooling layer (Sect. 3.6). Full pseudocode is provided in Algorithm 1.

Table 1 Review of wildlife datasets

Title	Type	Length	Classes	Labelled frames	Annotations	Location
Mouse Resident-Intruder Burgos-Artizzu et al. (2012)	RGB Video	88 hrs	13	8M	Action	Controlled Env
Mouse Social Interaction Sun et al. (2021)	RGB Video	–	3	1M	Behavior	–
RatSI Lorbach et al. (2018)	RGB Video	1.75 hrs	8	160K	Action	Controlled Env
Boxes on Bees and Pollen Noninvasive bee tracking in videos (2021)	RGB Video	–	1	5K	B.Box	–
Conservation Drones Bondi et al. (2020)	UAV TIR Videos	–	2	62K	B.Box	Southern Africa
Snapshot Serengeti Swanson et al. (2015)	RGB Images	–	48	322.7K	Species	Tanzania
Caltech Camera Traps Ferrari et al. (2018)	RGB Images	–	21	243K	B.Box	Southwestern USA
Open Monkey Yao et al. (2021)	RGB Images	–	4	195K	Pose	–
Amur Tiger Re-id Li et al. (2019)	RGB Images	–	1	9.5K	B.Box / Pose / 92 IDs	10 China Zoos
Whale Shark ID Holmberg et al. (2009)	RGB Images	–	1	7.7K	B.Box / ID	Western Australia
Caltech-UCSD Birds 200 Welinder et al. (2010)	RGB Images	–	200	6K	B.Box / Mask / Species	–
NOAA Arctic Seals 2019 [123]	Aerial Images	–	–	14K	B.Box	Alaska, USA
Seabirds W. Africa Kellenberger et al. (2021)	Aerial GeoTIFF Image	–	6	1	21K Point Annot	West Africa
Animals with Attributes2 Xian et al. (2019)	Feature Rep	–	50	37.3K	Numeric attributes	–
Livestock Behavior Patterns Rodriguez-Baena et al. (2020)	Acceleration	12 days	2	–	Activity	Farmhouse
Nest of the Kagu	**Continuous Video**	***252.8 hrs***	**5**	**23 M**	**B.Box / Event**	**New Caledonia**

Bold highlights our dataset

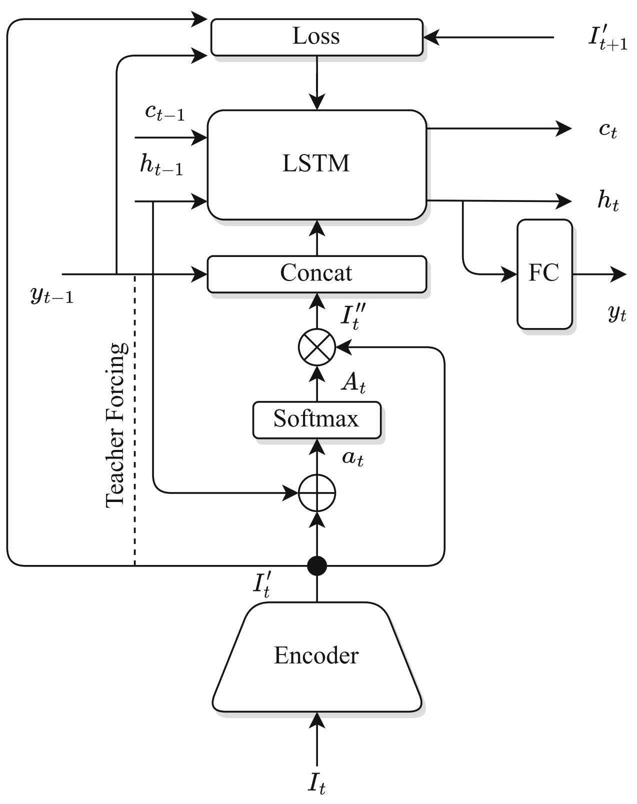


Fig. 2 The architecture of the self-supervised learning, perceptual prediction algorithm. Input frames from each time instant are encoded into high-level features using a deep-learning stack, followed by an attention overlay that is based on inputs from previous time instant, which is input to an LSTM. The training loss is composed of the predicted and computed features from current and next frames. $\oplus$ and $\otimes$ denote element-wise addition and multiplication operations, whereas I, y, h, c, and A represent input image, prediction, hidden state, cell state, and attention map, respectively. The teacher forcing connection is discussed in the implementation details section

3.1 Cognitive Inspiration

Our proposed framework is heavily inspired by cognitive psychology theories, more specifically, the Event Segmentation Theory (EST). The theory is developed by Zacks and Swallow (2007) based on findings from cognitive neuroscience experiments. EST introduces a cognitive learning theory, where making predictions of future inputs plays a central role in both, learning and event segmentation. The theory posits that humans maintain a stable representation of the current event called the "event model". The event model and the current perceptual inputs are used to predict the future perceptual input. Failing to predict the next input is an indicator that the current event has changed, and the event model cannot be used to predict inputs received from a different event, which results in perceiving an event boundary. Thus, event segmentation does not require conscious attention; instead, it emerges as a side effect of the ongoing perceptual prediction process. The predictive learning approach allows for

Algorithm 1 Temporal Event Segmentation Model with Attention-based Spatial Event Localization. The input is an untrimmed/streaming video $\mathbb{I}$, which is a set of frame blocks $\{I_1, ..., I_t, I_{t+1}, ...I_T\}$, each of the size N frames. The output is a boolean set of event predictions $\mathbb{B} = \{b_1, b_2, ...b_{T-1}\} \in \{0, 1\}^{T-1}$.

```
Input: Video frames $\{I_1, ..., I_t, I_{t+1}, ...I_T\} \in \mathbb{R}^{TxNxCxWxH}$
Output: Event prediction values $\mathbb{B} = \{b_1, b_2, ...b_{T-1}\}$
 1: procedure ATTENTION($I'_t, h_{t-1}$)
 2:     $a_t \leftarrow linear(tanh(linear(h_{t-1}) + linear(I'_t)))$
 3:     $A_t \leftarrow softmax(a_t)$
 4:     $I''_t \leftarrow A_t \odot I'_t$
 5:     return $I''_t$
 6: end procedure

 7: procedure SEGMENT($I_t, I_{t+1}, h_{t-1}, y_{t-1}$)
 8:     $I'_t \leftarrow$ ENCODER($I_t$)
 9:     $I'_{t+1} \leftarrow$ ENCODER($I_{t+1}$)
10:     $I''_t \leftarrow$ ATTENTION($I'_t, h_{t-1}$)
11:     $h_t \leftarrow$ LSTM($h_{t-1}, linear(concat(I''_t, y_{t-1}))$)
12:     $y_t \leftarrow$ DECODER($h_t$)
13:     $e_t \leftarrow \|(I'_{t+1} - y_t)^{\odot 2} \odot (I'_{t+1} - I'_t)^{\odot 2}\|^2$
14:     $b_t \leftarrow$ GATE($e_t$)
15:     return $h_t, b_t, y_t$
16: end procedure

17: $(h_{t-1}, y_{t-1}) \leftarrow (0, 0)$
18: for $\{I_t, I_{t+1}\} \in \{I_1, I_2\}, \{I_2, I_3\}, ..., \{I_{T-1}, I_T\}$ do
19:     $h_t, b_t, y_t \leftarrow$ SEGMENT($I_t, I_{t+1}, h_{t-1}, y_{t-1}$)
20:     $(h_{t-1}, y_{t-1}) \leftarrow (h_t, y_t)$
21:     $\mathbb{B}.append(b_t)$
22: end for
```

detecting generic events without providing a definition or a description of the target event.

The mismatch between the predicted future inputs and the actual future inputs drives the segmentation process to group events into discrete time intervals. EST is a continual perceptual process that segments a continuous stream of multimodal sensory input into a discrete and coherent set of events. The prediction error is also used as a training signal for the predictor to finetune the predictions within the same event. Our approach uses an LSTM unit as a predictor where the hidden state builds an event representation over time. We also use a CNN backbone to extract features, resembling the perceptual processing unit in EST. We study the prediction error to detect and localize events in a stream of RGB images. Additionally, we utilize the event model representation inside the LSTM to spatially localize the objects in the input feature representation.

3.2 Input Encoding

The raw input images are transformed from pixel space into a higher-level feature space by utilizing an encoder (CNN) model. This encoded feature representation allows the network to extract features of higher importance to the task being learned. We denote the output of the CNN layers by $I'_t = f(I_t, \theta_e)$ where θ_e is the learnable weights and biases parameters and I_t is the input image at time t. The encoder network transforms an input image with dimensions $I_t \in \mathbb{R}^{3\times W\times H}$ to output features with dimensions $I'_t \in \mathbb{R}^{M\times N\times N}$, where $N \times N$ is the spatial dimensions and M is the feature vector length.

3.3 Attention Unit

In this framework, we utilize Bahdanau attention (Bahdanau et al., 2014) to spatially localize the event in each processed frame. The attention unit receives as an input the encoded features and outputs a set of attention weights (A_t) with dimensions $N \times N$. The hidden feature vectors (h_{t-1}) from the prediction layer of the previous time step are used to calculate the output set of weights using Eq. 1, expressed visually in Fig. 2.

$$A_t = \gamma(\,FC(\varphi(FC(h_{t-1}) + FC(I'_t)))\,) \tag{1}$$

where φ represents hyperbolic tangent (tanh) function, and γ represents a softmax function. The weights (A_t) are then multiplied by the encoded input feature vectors (I'_t) to generate the masked feature vectors (I''_t). The attention function uses the object representation embedded in h_{t-1} to filter the features I_t and remove the noisy background features. In other words, the attention function ensures that the LSTM model receives only the object (i.e., bird) features when making its prediction. Interestingly, as we will see in the experimental section, this change does not have a significant quantifiable effect on the overall temporal segmentation performance due to the sparsity of events in the proposed dataset, however, it does enable spatial segmentation. We have experimented with other datasets (i.e., Kinetics) in different domains, where multiple objects can be present. Results (Fig. 18) show that the model can localize multiple objects simultaneously.

3.4 Prediction Layer

The process of future prediction requires a layer capable of storing a flexible internal state (event model) of the previous frames. For this purpose, we use a recurrent layer, specifically long-short term memory cell (LSTM) Hochreiter and Schmidhuber (1997), which is designed to output a future prediction based on the current input and a feature representation of the internal state. More formally, the LSTM cell can be described using the function $h_t = g(I''_t, W_{lstm}, h_{t-1})$, where h_t and h_{t-1} are the output hidden state and previous hidden state respectively, I''_t is the attention-masked input features at time step t and W_{lstm} is a set of weights and biases vectors controlling the internal state of the LSTM. The input to the LSTM can be formulated as:

$$FC([I''_t, y_{t-1}]) \tag{2}$$

where I''_t is the masked encoded input feature vector and y_{t-1} is defined as $FC(h_{t-1})$. The notation [.] represents vector concatenation.

3.5 Loss Function

The perceptual prediction model aims to train a model capable of predicting the feature vectors of the next time step. We define two different loss functions; prediction loss and motion-weighted loss.

Prediction Loss This function is defined as the L2 Euclidean distance loss between the output prediction y_t and the next frame encoded feature vectors I'_{t+1}.

$$e^p_t = \|(I'_{t+1} - y_t)\|^2 \tag{3}$$

Motion Weighted Loss This function aims to extract the motion-related feature vectors from two consecutive frames to generate a motion-dependent mask, which is applied to the prediction loss. The motion-weighted loss function allows the network to benefit from motion information in higher-level feature space rather than pixel space. This function is formally defined as:

$$e^{mw}_t = \|(I'_{t+1} - y_t) \otimes (I'_{t+1} - I'_t)\|^2 \tag{4}$$

where $\otimes$ denotes an element-wise multiplication operation.

3.6 Spatio-Temporal Pooling Layer

The spatio-temporal pooling layer is a decision-based function that receives as input a spatial attention map and continuous prediction error signal. The pooling layer generates a bounding box from the attention map and detects events from the error signal. The output of the layer is the detected events segmented spatially and temporally. We define pooling, in this context, as going from noisy frame-level representation of prediction error to event-level classification. Ideally, more layers of prediction and pooling can be built on top of this pooled event-level representation to create a hierarchy of event models (Zacks & Swallow, 2007; Hawkins et al., 2017; Hawkins & Ahmad, 2016).

3.6.1 Bounding Box generation

We use the attention map output from Eq. 1 to spatially localize the objects in the frames. However, it can be further processed to generate bounding boxes, which are more important to ethogramming. As shown in Fig. 1, the model outputs an 8 × 8 attention map, which we resize to the original image size using bilinear interpolation. Then we apply MinMax scaling to the attention map and threshold it. The result is a binary map for which we calculate the bounding rectangle. We evaluate the performance of the attention maps and the generated bounding boxes in Sect. 5.2.2.

3.6.2 Activity Gating

The Activity gating function receives, as an input, the error signal defined in Sect. 3.5, and applies a thresholding function to classify each frame. The threshold function applies a threshold (ϕ) to each loss value resulting in a binary signal (i.e., $\{0, 1\}^T$). Groups of frames with loss above the threshold value are detected as events. Additionally, we define the parameter ψ as the gap (in frames) between two consecutive events, below which the two events will be merged as one. The performance of our approach with different thresholds, and model ablations, are provided in Sect. 5.1.2.

Simple Threshold The simple threshold function applies a constant threshold value to the prediction error signal $\mathbf{e} \in \mathbb{R}^T$. Each loss value e_t is compared against the threshold ϕ as shown in Eq. 6.

$$\mathbf{e} = \{e_1 \dots e_t \dots e_T\} \in \mathbb{R}^T \tag{5}$$

$$f(e_t, \phi) = \begin{cases} 1, & \text{if } e_t \geq \phi \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Adaptive Threshold A simple threshold does not adapt to changes in the loss values over time; it is expected for the loss values to decrease as the model learns to predict better future features. Therefore, we propose to use an adaptive threshold function that defines the threshold value ϕ as an offset above the smoothed error signal $\mathbf{e}_s$. First, we compute the smoothed values $\mathbf{e}_s$ as shown in Eq. 7. The smoothing function acts as a low-pass filter to attenuate high-frequency noise in the raw prediction signal $\mathbf{e}$. Then, we use Eq. 8 to calculate the difference between the raw prediction signal and the smoothed signal. Finally, we apply a threshold ϕ to each value e_{d_t} in $\mathbf{e}_d$ using Eq. 9.

$$\begin{aligned} \mathbf{e}_s &= \{e_{s_1} \dots e_{s_t} \dots e_{s_T}\} \in \mathbb{R}^T \\ &= \mathbf{e} \circledast [\{\frac{1}{n} \dots \frac{1}{n}\} \in \mathbb{R}^n] \end{aligned} \tag{7}$$

$$\mathbf{e}_d = \mathbf{e} - \mathbf{e}_s \tag{8}$$

$$f(e_{d_t}, \phi) = \begin{cases} 1, & \text{if } e_{d_t} \geq \phi \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where $\circledast$ represents a 1D convolution operation.

3.7 Implementation Details

In our experiments, we use an Inception V3 (Szegedy et al., 2016) encoding model (pretrained on the ImageNet dataset) to transform input images from raw pixel representation to a higher-level feature representation. We freeze the encoder's parameters (weights and biases) and remove the last layer. We resize the input image to $3 \times 299 \times 299$. The output of the Inception V3 model is a $2048 \times 8 \times 8$ feature tensor, which we reshape to 2048×64 feature vectors. Each 2048 feature vector requires one LSTM cell for future feature prediction. In other words, the encoded input frame (I'_t) is provided with 64 LSTM cells (sharing the same weights W_{LSTM}), each processing a 2048 features vector (hidden state size) simultaneously. During a single optimization step, the model receives eight current frames and predicts eight future frames. The next training step slides the model 8 frames into the future. Hidden states of the LSTM are copied (i.e., stop gradient operation) to initialize the LSTMs of future training steps (i.e., stateful LSTM). We use a 0.4 drop rate (recurrent dropout) on the hidden states to prevent overfitting, which may easily occur due to the stateful LSTM nature of the model and the dataset size. LSTMs' hidden states are initialized to zero. Teacher forcing (Lamb et al., 2016) approach is utilized by concatenating the weighted encoded input image (I''_t) with the encoded input image (I'_t) instead of concatenating it with its prediction from the previous time step (y_{t-1}). Adam optimizer is used with a learning rate of $1e^{-8}$ for the gradient descent algorithm. We do not use data augmentations. The dataset is divided into four equal portions and simultaneously trained on four Nvidia GTX 1080 GPUs.

4 Nest monitoring of the Kagu

4.1 Dataset Overview

We used a dataset of videos from nest monitoring of the Kagu Gula et al. (2010). The dataset consists of around ten days (253 h) of continuous monitoring sampled at 25 frames per second. We fully annotated the entire dataset (23 M frames) with spatial localization labels in the form of a tight bounding box. Additionally, we provide temporal event segmentation labels as five unique bird activities: {Feeding, Pushing leaves, Throwing leaves, Walk-In, Walk-Out}. The feeding event represents the period of time when the birds feed the chick. The nest-building events (pushing/throwing leaves) occur when the birds work on the nest during incubation. Pushing leaves is a nest-building behavior during which the birds form a crater by pushing leaves with their legs toward the edges of the nest while sitting on the nest. Throwing leaves is another nest-building behavior during which the birds throw leaves with the bill towards the nest while being, most of the time, outside the nest. Walk-in and walk-out events represent the transitioning events from an empty nest to incubation or brooding, and vice versa. We also provide five additional labels that are based on time-of-day and lighting conditions: {Day, Night, Sunrise, Sunset, Shadows}. While our approach ignores the lighting conditions and focuses only on object-centric (bird) activities, we include these labels to evaluate and contrast the different performances under different conditions. Ideally, a model with a robust representation of the bird should perform consistently well during all lighting conditions. A good model learns to extract useful features and inhibit background features. Figure 3 shows a sample of images from the dataset.

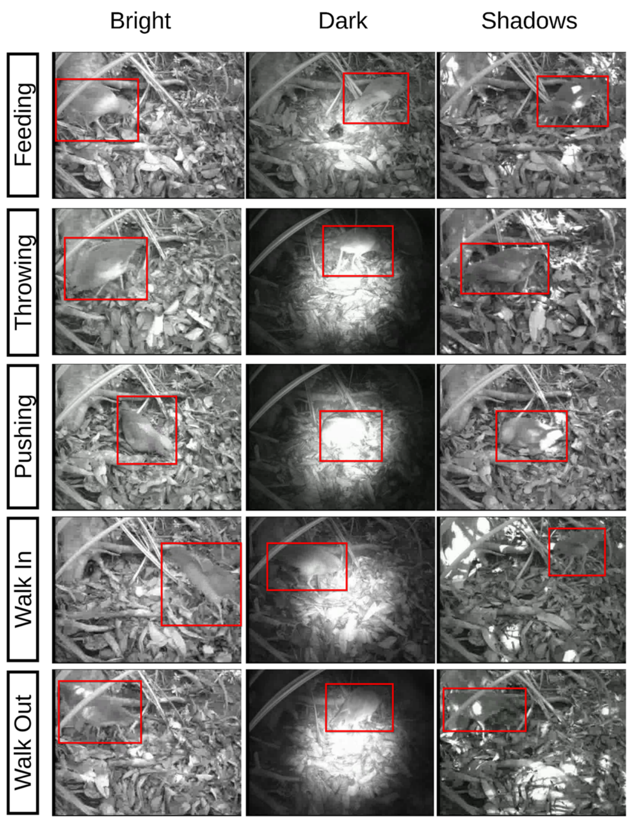


Fig. 3 Samples of images from the Nest of the Kagu dataset at different behavioral categories and lighting conditions. Red bounding box shows the groundtruth label of the bird location (Color figure online)

4.2 Dataset Statistics

We present the distributions of event counts and their durations in Fig. 4. Both event counts and event durations are further categorized by Day and Night conditions. As can be seen from the stacked bar charts, the Walk-in, Walk-out, and Feeding contribute the highest number of events, whereas the Throwing event has the highest total duration. Most events have a high day-to-night ratio, except for the Throwing event, where around 40% of the event duration occurs during twi-

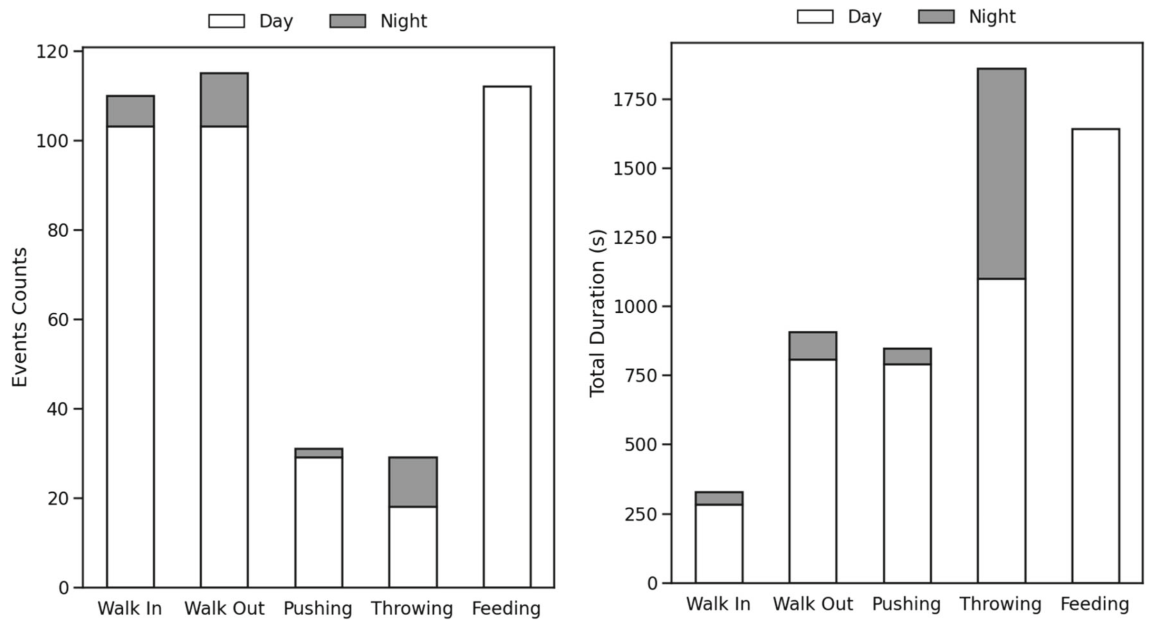


Fig. 4 Proportion of behavioral categories (subcategorized for day and night) of event counts **(left)** and event duration **(right)**

light periods. Feeding event occurs only in the daytime. It can be seen that both count and duration plots show the same day-to-night ratio for each event type, which means that the duration of each event at night equals its duration at day. In other words, the average duration of each event does not change based on the time of day.

Figure 5 shows the durations of different event types and bird states (stationary inside frame, moving inside frame, outside frame). As shown in the right subfigure, event duration vary significantly based on its type. Throwing events can span over two minutes, whereas short events (e.g., Walk-In/Out) only take a few seconds. This variability in durations makes the dataset more challenging to activity detection algorithms. The left subfigure introduces a different kind of challenge-sparsity of events. We show that the dataset contains a total duration of around one hour of motion/activities (for the whole dataset), while for the rest of the dataset, the bird is either stationary in the nest or outside of the image frame. The sparsity of events simulates real-life conditions; online algorithms should be robust against the resulting challenges-unbalanced dataset, catastrophic forgetting, etc.

4.3 Annotation Protocol

We have manually annotated the dataset with temporal events, time-of-day/lighting conditions, and spatial bounding boxes without relying on any object detection/tracking algorithms. The temporal annotations were initially created by experts who study the behavior of the Kagu bird and later refined to improve the precision of the temporal boundaries. Additional labels, such as lighting conditions, were added during the refinement process. The spatial bounding box annotations of 23 M frames were created manually using professional video editing software (Davinci Resolve). We attempted to use available data annotation software tools, but they did not work for the scale of our video (10 days of continuous monitoring). We resorted to video editing software, which helped us annotate and export bounding box masks as videos. The masks were then post-processed to convert annotations from binary mask frames to bounding box coordinates for storage. It is worth noting that the video editing software allowed us to linearly interpolate between keyframes of the bounding boxes annotations, which helped save time and effort when the bird's motion is linear. Both temporal and spatial annotations were verified by two volunteer graduate students. The process of creating spatial and temporal annotations took approximately two months.

4.4 Validation and Test Splits

We split the full 10-day dataset into a validation and test set. The validation set is chosen as the first two days of the dataset (20% of the total frames), while the remaining eight days are used to create the test set. The validation set can be used to tune hyperparameters, such as segmentation thresholds, for adjusting segmentation granularity. The tuned parameters are then used for evaluation on the test set. As shown in Fig. 6, less than 10% of the total number of events occur in the first two days. Additionally, the feeding event occurs only after the chick has hatched on day 8, which also affects the frequency of walking in and out of the nest. The significant changes in event statistics (counts and durations) between

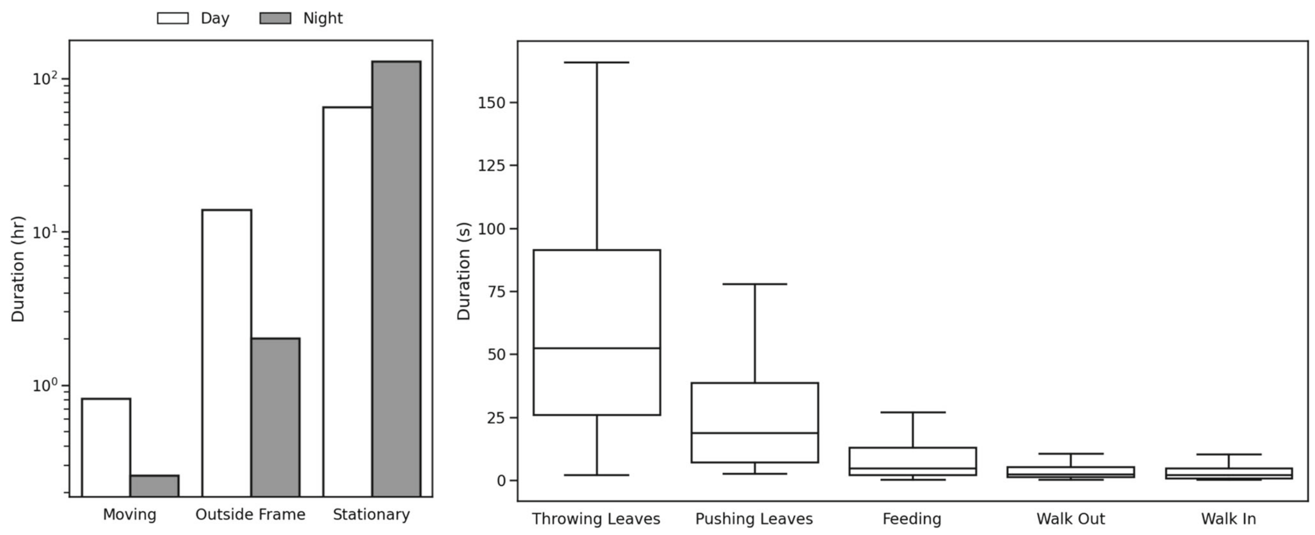


Fig. 5 Event statistics. (**Left**): Total durations of bird states plotted in log scale and categorized by Day and Night conditions. (**Right**): Box-and-Whisker-Plots for the duration of behavioral categories

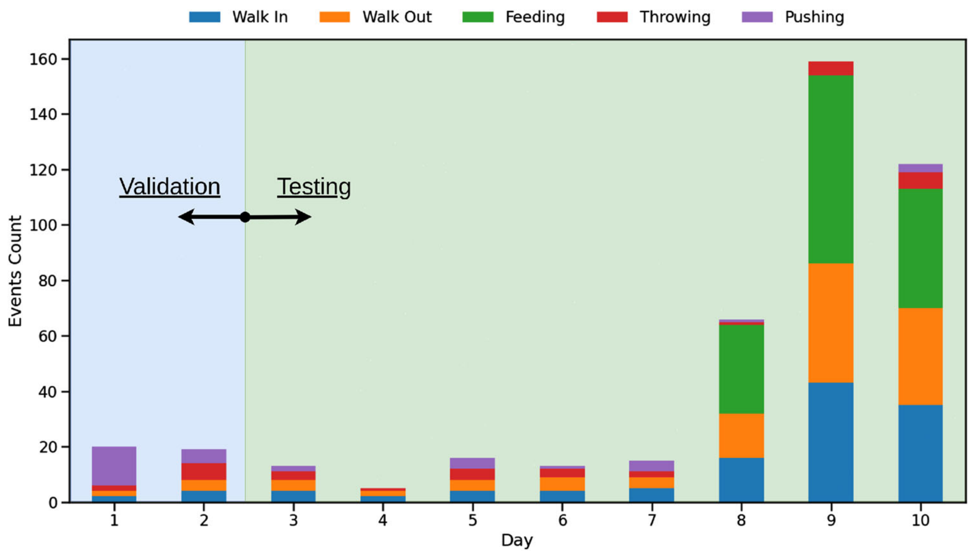


Fig. 6 Stack plot of event counts for each day of the proposed 10-day dataset categorized by the event type. The validation split is chosen as the first two days

the validation and test splits provide a challenging setup for learning algorithms.

5 Experimental Evaluation

In this section, we present the results of our experiments for our approach defined in Sect. 3. We divide the results section into temporal segmentation and spatial segmentation. In both sections, we explain the evaluation metrics used to quantify the performance, discuss model variations and conclude by presenting quantitative and qualitative results on the Kagu monitoring dataset. We also provide comparisons with state-of-the-art approaches on other datasets in different domains in Sect. 5.3

5.1 Temporal Segmentation

Temporal segmentation is the task of segmenting events in untrimmed videos. The goal is to detect and localize events such that important activities (e.g., bird walking out) are accurately trimmed to their start and end boundaries. We present two different evaluation approaches to temporal event segmentation; activity detection and boundary detection. Activity detection targets the detection of the full event and distinguishes between the start and end boundaries. Boundary detection only considers the detection of event boundaries as a separation between events. As discussed in Sect. 5.1.1, the evaluation of activity detection is based on one-to-one mapping, with IoU maximization objective, of the predicted events and the ground truth events, whereas the boundary detection is evaluated based on the distance between the pre-

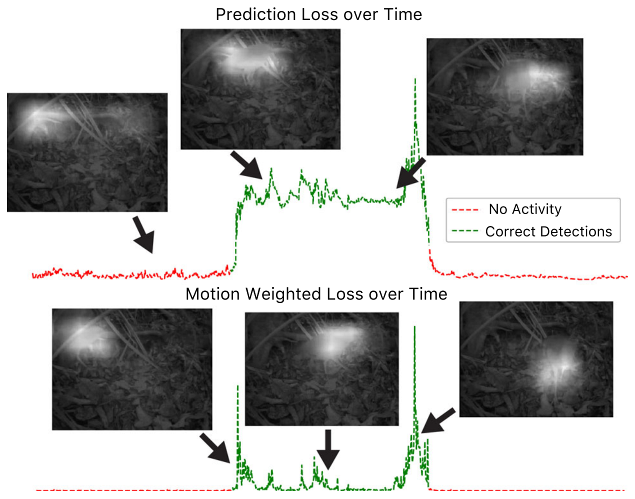


Fig. 7 Plots of the two kinds of errors before, during, and after an activity: **(top):** feature prediction loss over the frames, **(bottom):** motion weighted feature prediction loss over the frames. Some selected frames are shown and overlaid with corresponding attention maps (after bilinear interpolation resizing)

dicted boundaries and the ground truth boundaries. Figure 7 shows an example of temporal segmentation for an event; the event is detected by thresholding the error signal.

5.1.1 Evaluation Metrics

Our evaluation metrics are tied directly to the quality of ethogramming. Accurately localizing the events and their boundaries can lead to higher ethogramming quality since the behavior attributes can only be found within the boundaries of the event. It is important to increase the recall rate (reduce missed detections) and decrease false positives to ensure all important events are detected and analyzed while erroneous and irrelevant activities are disregarded. We use three different evaluation metrics to assess the temporal segmentation performance of our approach. Conceptually, frame-level evaluation captures the ability of a model to correctly classify whether a frame belongs to an event. Unlike frame-level, activity-level evaluation focuses on the ability to capture the existence and location of full events within an untrimmed video. The boundary distance metric measures the temporal precision of the predicted boundaries to the groundtruth boundaries. We discuss the evaluation metrics in more detail in the following paragraphs.

Frame Level The frame-level evaluation of temporal segmentation measures the ability of an algorithm to classify whether each frame belongs to an event. The recall value in frame-level ROC is calculated as the ratio of true positive frames (event present) to the number of positive frames in the annotations dataset, while the false positive rate is expressed as the ratio of the false positive frames to the total number of negative frames (event not present) in the annotation dataset. Frame window size (ψ) is defined as the maximum joining window size between events; a high ψ value can cause separate detected events to merge, which decreases the overall performance. The threshold value (ϕ) is varied to obtain a single ROC line, while varying the frame window size (ψ) results in a different ROC line.

Activity Level The Activity level evaluation measures the ability of an algorithm to detect events. We utilize the Hungarian matching (Munkres assignment) algorithm to achieve one-to-one mapping between the ground truth labeled events and the detected events. The recall is defined as the ratio of the number of correctly detected events (overlapping frames) to the total number of ground truth events. For the activity level ROC chart, the recall values are plotted against the false positive rate per minute, defined as the ratio of the total number of false-positive detected events to the total duration of the dataset in minutes. The false-positive rate per minute evaluation metric is also used in the ActEV TRECVID challenge [139]. Frame window size value (ψ) is varied to obtain a single ROC line, while varying the threshold value (ϕ) results in a different ROC line.

Boundary Distance In addition to activity detection evaluation, we also evaluate the ability of our approach to detect generic boundaries of events. We quantify the performance of boundary detection by applying one-to-one (Hungarian) matching between the detected boundaries and the annotated boundaries. The distances, in seconds, between the resulting matches are calculated and thresholded. If the distance for a boundary is lower than the specified threshold value, it is considered a true positive (TP). As the threshold increases, more TPs occur, resulting in higher recall and precision values and vice versa. By using Hungarian matching, we make sure that the algorithm is penalized for duplicated detections by reducing its precision value.

5.1.2 Activity Detection Results

Model Variations Different variations of our framework (section 3) have been evaluated to quantify the effects of individual components on the overall performance. In our experiments, we tested the base model, which trains the perceptual prediction framework-including the attention unit-using the prediction loss function for backpropagation of the error signal. We refer to the base model as *LSTM+ATTN*. We also experimented with the effect of removing the attention unit, from the model architecture, on the overall segmentation performance; results of this variation are reported under the model name *LSTM*. Further testing includes using the motion-weighted loss for backpropagation of the error signal. We refer to the motion-weighted model

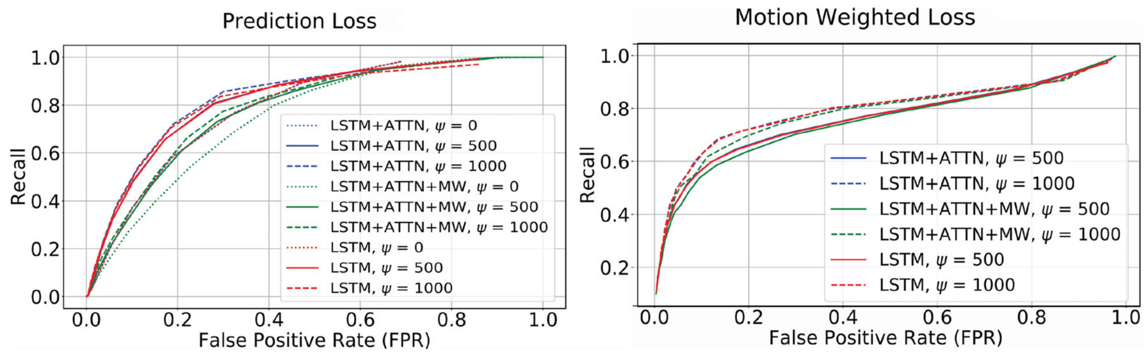


Fig. 8 Frame-level event segmentation ROCs when activities are detected based on simple thresholding of the prediction and motion weighted loss signals. Plots are shown for different ablation studies

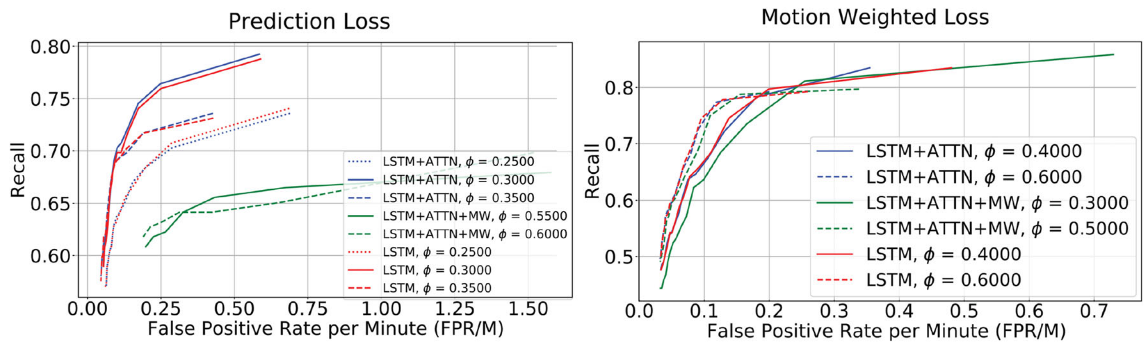


Fig. 9 Activity-level event segmentation ROCs when activities are detected based on simple thresholding of the prediction and motion weighted loss signals. Plots are shown for different ablation studies

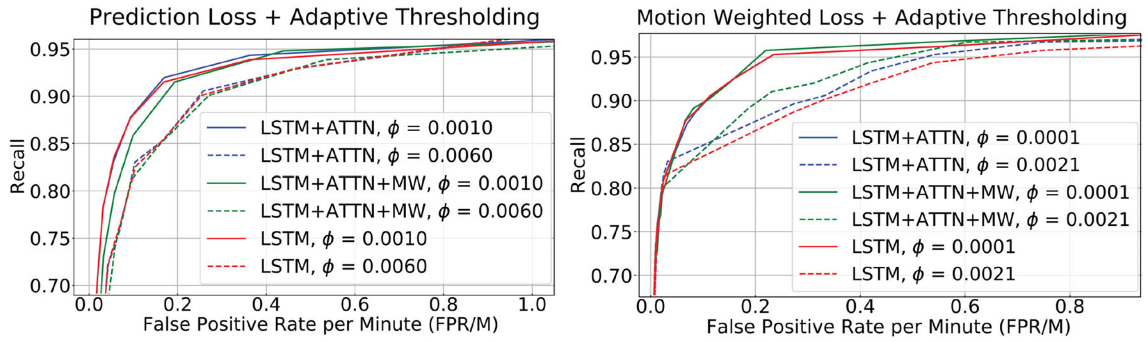


Fig. 10 Activity-level event segmentation ROCs when activities are detected based on adaptive thresholding of the prediction and motion weighted loss signals. Plots are shown for different ablation studies

as *LSTM+ATTN+MW*. Each of the models has been tested extensively; frame-level segmentation results are shown in Fig. 8, while activity-level results are provided in Figs. 9 and 10. Ablations are evaluated on the full ten-days dataset.

Comparing the results shown in Figs. 9 & 10 indicates a significant increase in overall performance when using an adaptive threshold for loss signal gating. The efficacy of adaptive thresholding is evident when applied to activity-level event segmentation. Comparing the results (LSTM & LSTM+ATTN) show that the model can effectively generate attention maps for spatial segmentation without impacting the temporal segmentation performance.

Results Discussion We tested three different models, *LSTM*, *LSTM+ATTN*, and *LSTM+ATTN+MW*, for frame level and activity level event segmentation. Simple and adaptive gating functions (Sect. 3.6.2), were applied to prediction and motion-weighted loss signals (Sect. 3.5) for frame-level and activity-level experiments. For each model, we vary parameters such as the threshold value ϕ and the frame window size ψ to achieve the ROC charts presented in Figs. 8, 9 & 10.

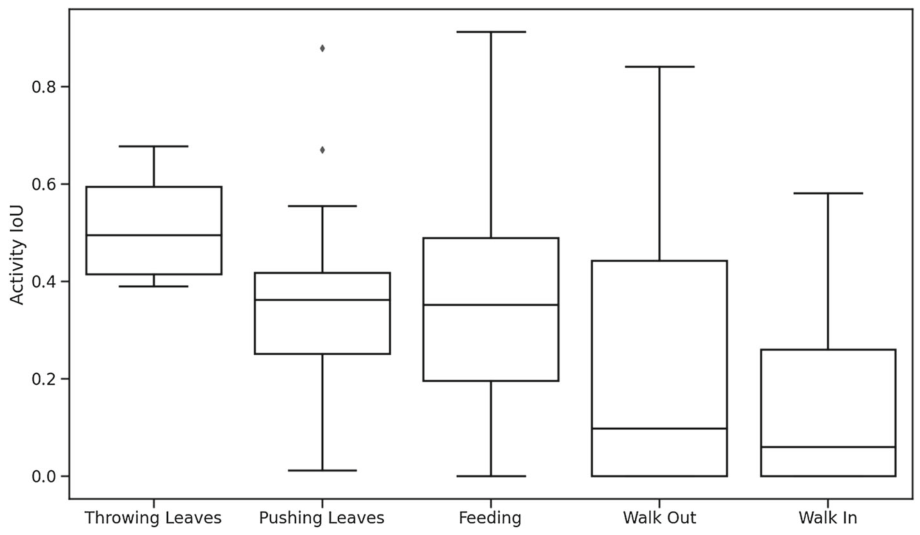


Fig. 11 Activity IoU for the best performing Activity-level temporal segmentation model. Results are categorized by event type

It is to be noted that thresholding a loss signal does not necessarily imply that the model was trained to minimize this particular signal. In other words, loss functions used for back-propagating the error to the models' learnable parameters are identified only in the model name; however, thresholding experiments have been conducted on different types of loss signals, regardless of the backpropagating loss function used for training.

The best performing model, for frame level segmentation, (*LSTM+ATTN,*$\psi = 1000$) is capable of achieving {40%, 60%, 70%} frame recall value at {5%, 10%, 20%} frame false positive rate respectively. Activity level segmentation can recall {80%, 90%, 95%} of the activities at {0.02, 0.1, 0.2} activity false positive rate per minute, respectively, for the model (*LSTM+ATTN+MW,*$\phi = 0.0001$) as presented in Fig. 10. A 0.02 false positive activity rate per minute can also be interpreted as one false activity detection every 50 min of training (for detecting 80% of the groundtruth activities). While the attention mechanism does not offer a clear improvement in temporal segmentation, it allows us to localize the object in every frame. Spatial segmentation was not possible without the attention mechanism added to the LSTM baseline. Other modifications, such as motion-weighted loss and adaptive thresholding, contribute more to a significant improvement in temporal segmentation performance.

We further inspect the performance of the best-performing activity-level model by presenting the IoU between the detected events and the ground truth events–categorized by event type in Fig. 11. Based on the results, it can be seen that there is a correlation between the event durations and the activity overlap. Long events (e.g., Throwing) result in higher IoU, with the detected event, than short events (e.g., Walk-In).

5.1.3 Boundary Detection Results

Even though our approach targets the detection of full activities, we can evaluate the performance of our approach on the boundary detection task by converting each detected activity into two boundaries. Using boundaries, we can quantify the performance of our approach by calculating the distance of each detected boundary to the ground truth boundary. We evaluate the performance at varying distance thresholds and report the results (with baseline comparisons) in Table 2. All segmentation thresholds are tuned on the validation set and used for evaluation on the test set.

Baselines *Scene detect [140]* is a popular online tool for shot boundary detection. Scenedetect analyzes the video for changes in average frame brightness/intensity and applies a threshold to detect boundaries. Lower threshold values result in more boundaries, which increases recall and decreases precision.

Uniform is a simple baseline created by using equally separated boundaries as predictions. Varying the frequency of predicted boundaries results in moving to different positions on the Precision-Recall line. We use the frequency resulting in the best F1 score on the validation set for evaluation on the test set.

PredictAbility (PA-DPC) Shou et al. (2021) is a baseline created by using the ability of the model to predict future frames. PA uses Dense Predictive Coding (DPC) model (Han et al., 2019) as a backbone and provides state-of-the-art results for self-supervised generic event boundary detection. This baseline calculates the prediction loss for a given frame by computing the difference between past and future context embeddings. The context is defined as five frames before and after the target frame. DPC uses a 3D ResNet architecture as the backbone, which is used to calculate the context embeddings. The PA model proposes event boundaries at the local

Table 2 Temporal segmentation results on Kagu dataset for unsupervised event boundary detection methods at different distance thresholds

Metric	Approach	Distance threshold (seconds)									
		10	50	100	150	200	250	300	350	400	450
Precision	SceneDetect*	0.095	0.095	0.143	0.143	0.143	0.143	0.238	0.238	0.238	0.238
	Uniform	0.005	0.019	0.030	0.044	0.055	0.069	0.078	0.088	0.098	0.103
	PA-DPC Shou et al. (2021)	0.005	0.016	0.026	0.035	0.049	0.058	0.063	0.067	0.075	0.087
	PA-DPC-FT Shou et al. (2021)	0.002	0.009	0.012	0.012	0.012	0.014	0.014	0.014	0.014	0.014
	LSTM+AL Aakur and Sarkar (2019)	0.030	0.042	0.052	0.059	0.062	0.067	0.069	0.073	0.079	0.088
	KNN	0.120	0.201	0.233	0.254	0.265	0.273	0.281	0.283	0.289	0.294
	Ours	**0.167**	**0.304**	**0.361**	**0.369**	**0.386**	**0.411**	**0.416**	**0.439**	**0.449**	**0.460**
Recall	SceneDetect*	0.003	0.003	0.004	0.004	0.004	0.004	0.007	0.007	0.007	0.007
	Uniform	0.013	0.054	0.084	0.121	0.152	0.192	0.218	0.244	0.271	0.287
	PA-DPC Shou et al. (2021)	0.006	0.018	0.029	0.040	0.056	0.066	0.072	0.077	0.075	0.087
	PA-DPC-FT Shou et al. (2021)	0.001	0.006	0.009	0.009	0.009	0.010	0.010	0.010	0.010	0.010
	LSTM+AL Aakur and Sarkar (2019)	0.042	0.057	0.072	0.081	0.086	0.092	0.095	0.100	0.109	0.121
	KNN	0.055	0.092	0.106	0.116	0.121	0.125	0.128	0.130	0.132	0.134
	Ours	**0.108**	**0.196**	**0.232**	**0.237**	**0.248**	**0.264**	**0.268**	**0.282**	**0.289**	**0.296**
F1	SceneDetect*	0.005	0.005	0.008	0.008	0.008	0.008	0.013	0.013	0.013	0.013
	Uniform	0.007	0.028	0.045	0.064	0.080	0.102	0.115	0.129	0.144	0.152
	PA-DPC Shou et al. (2021)	0.006	0.017	0.027	0.038	0.053	0.062	0.067	0.072	0.080	0.092
	PA-DPC-FT Shou et al. (2021)	0.001	0.007	0.010	0.010	0.010	0.011	0.011	0.011	0.011	0.011
	LSTM+AL Aakur and Sarkar (2019)	0.035	0.048	0.061	0.068	0.072	0.077	0.080	0.084	0.092	0.102
	KNN	0.076	0.126	0.146	0.159	0.166	0.171	0.176	0.178	0.181	0.185
	Ours	**0.131**	**0.238**	**0.283**	**0.289**	**0.302**	**0.321**	**0.326**	**0.344**	**0.351**	**0.360**

Bold highlights best results
FT denotes fine-tuned on the Kagu dataset
* SceneDetect's threshold parameter is tuned on the full dataset, otherwise the performance is zero

maxima of the error signal. The Laplacian of Gaussian (LoG) is applied to the 1D temporal error signal. Peaks are detected at the negative to positive zero crossings of the derivative of the LoG signal. In our experiments, we found that thresholding the error signal and applying a 1D non-max suppression operation provides better results (reported) on our proposed dataset. *PA-DPC-FT* Shou et al. (2021) is a finetuned version of PA-DPC, where the DPC model (Han et al., 2019) is fine-tuned on 1000 short videos ($\approx$ 10 seconds each) extracted from the Kagu dataset.

LSTM+AL Aakur and Sarkar (2019) is a predictive learning baseline that tackles the problem of event boundary detection by learning a future prediction function on high-level frame features. LSTM+AL detects event boundaries as peaks in the prediction loss signal. Similar to our proposed approach, LSTM+AL is heavily inspired by the "Event Segmentation Theory" introduced by Zacks and Swallow (2007). However, in addition to the added attention module, there exist several important distinctions between the two approaches. We summarize the main differences in the following points:

- Our approach processes a sliding window of 16 frames (8 frames predicting 8 frames) with a stride of 8 frames, allowing the representation to be more robust by predicting farther into the future. LSTM+AL (Aakur & Sarkar, 2019) uses a single frame to predict a single future frame, limiting the representation and prediction capacity of the network.
- Our approach uses motion-weighted loss to learn events from motion cues, whereas LSTM+AL (Aakur & Sarkar, 2019) only uses the prediction loss peaks to detect boundaries.
- Our approach freezes the encoder weights and only allows the prediction network weights to be modified. According to recent findings (Vasconcelos et al., 2022), "an extreme form of knowledge preservation-freezing the classifier-initialized backbone- consistently improves many different detection models, and leads to considerable resource savings.". We also find that freezing the encoder network leads to better performance and prevents model collapse.
- Our approach does not use the adaptive learning trick introduced in LSTM+AL (Aakur & Sarkar, 2019) to keep

the prediction loss in a specific range. Instead, we use adaptive thresholding to detect events.

- LSTM+AL (Aakur & Sarkar, 2019) focuses on detecting boundaries by detecting peaks in the prediction loss, whereas our approach detects full events by thresholding the motion-weighted prediction loss signal.

KNN is an unsupervised online learning baseline that aims to adapt to changes in the data distributions. We design an unsupervised KNN algorithm where every frame in the dataset is compared to the five nearest neighbors in a moving window of the past four seconds (100 frames). The average (over the nearest five neighbors) cosine similarity score is stored for every frame. We threshold the similarity scores and apply 1D non-max suppression to remove duplicate boundary detections in a window of 1 s.

Results discussion We report the evaluation results of boundary detection in Table 2. The F1 score metric is used to compare the overall performance of our method to other baselines. We additionally provide precision and recall values to analyze and compare the behavior of each detector with varying distance thresholds. Based on the F1 results, it can be seen that we at least double the performance of all the other methods at all thresholds.

SceneDetect [140] is designed to detect shot boundaries based on illumination changes; therefore, it fails when applied to a dataset of continuous monitoring. It results in higher precision values than the other baselines because there is a higher chance of what is perceived as a shot boundary to be also classified as an activity boundary. However, the precision values are matched with a significantly lower recall rate demonstrating its inability to detect most of the boundaries.

Uniform results in high recall rates and low precision values, which demonstrate the ability to detect a high percentage of the ground truth boundaries, but only when sampling predicted boundaries at high frequency (shown by the low precision values). This baseline is implemented to demonstrate the irregular distribution of activities over time, causing simple, equally spaced boundary predictions to fail-especially at lower distance thresholds.

PA-DPC uses a pretrained backbone to detect boundaries based on the difference between context embeddings. The PA baseline, surprisingly, fails at detecting boundaries and results in average performance with respect to the other baselines. The F1 scores are slightly lower than the Uniform baseline. The low performance of this baseline can be attributed to the difference between the pretraining domain (kinetics 400) and the animal monitoring domain (Kagu dataset). A substantial difference between our approach and PA-DPC lies in the continuous predictive learning and adaptation on the target dataset, which cannot be done with PA-DPC by merely comparing the context embeddings. The results highlight the importance of the proposed online stream learning approach. Other results comparing our method to PA-DPC on in-domain datasets (e.g., GEBD-Kinetics) are discussed in Sect. 5.3.

PA-DPC-FT attempts to bridge the gap in domains between the Kinetics dataset and the Kagu dataset by finetuning the DPC architecture on a subset of our proposed Kagu dataset. Results show that training on the Kagu dataset did not increase performance; our approach still outperforms it by a significant margin. It is important to note that contrastive approaches such as DPC rely heavily on the diversity of labels and features within and across video samples, making the Kagu videos a challenging dataset for learning useful representation in a contrastive manner.

LSTM-AL shows a slight improvement in performance over PA-DPC-FT; however, the performance is low compared to our approach. The low performance can be attributed to the differences discussed earlier in the baselines section.

KNN reports approximately double the performance of LSTM-AL. The results highlight the importance of online training and adaptation for processing streaming data. However, the KNN approach is still significantly outperformed by our predictive framework indicating that the similarity between frame features does not contain enough information to detect event boundaries.

5.2 Spatial Segmentation

Spatial segmentation is the task of detecting and localizing the object of interest in each frame; the foreground (i.e., bird) is segmented from the background (i.e., nest). We use the Bhadanau attention map (Sect. 3.3) to spatially localize the bird.

5.2.1 Evaluation Metrics

Successful spatial segmentation of the events not only depends on the ability of the algorithm to spatially locate the event-causing objects but also detect their presence inside the frame. To generate high quality ethograms, the algorithm must be evaluated on event detection and localization tasks. The bird's location, velocity, and motion patterns in the nest can be correlated with specific behavior. Therefore, the accurate localization of events is essential for automating ethograms.

Evaluating Attention Maps Although the ground truth annotations are given in a bounding box format, most of our baselines output a heat map–or a mask–indicating the predicted location of the bird in each frame. The *probability sum* is a metric designed to evaluate the success of the mask in predicting the location of the object. First, the predicted mask is converted into a probability distribution using the Softmax

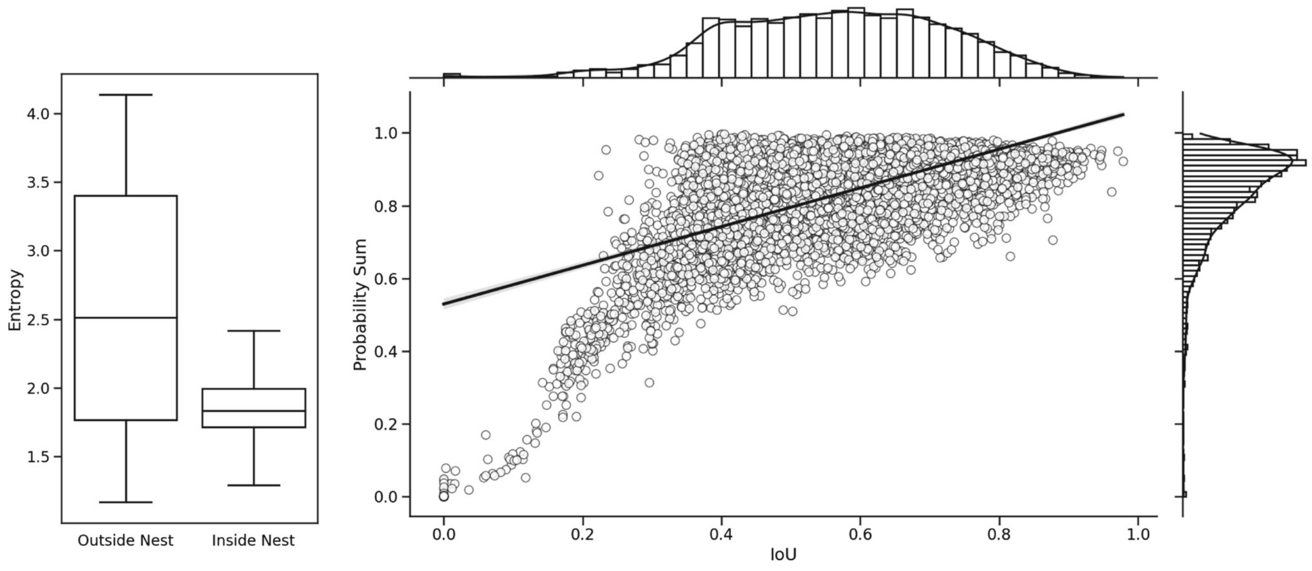


Fig. 12 Spatial segmentation evaluation metrics. **(Left):** Entropy distributions for when the bird is inside and outside of the nest. **(Right):** Joint distributions of Probability Sum and IoU metrics

function. The probability sum is then calculated by adding all the probability values located inside the ground truth bounding box. Applying the Softmax function provides means for penalizing the mask for predicting high values outside of the ground truth bounding box. A high probability sum value indicates a high percentage of mask values inside the bounding box; however, it does not guarantee a high overlap.

Generating Bounding Boxes We generate bounding boxes by thresholding the predicted mask to detect the biggest contour for which we find the bounding rectangle. The *IoU metric* measures the overlap between the two bounding boxes normalized by the union of their areas. We use the Jaccard Index (IoU) to quantify the overlap of the predicted bounding box with the ground truth bounding box. As can be seen in Fig. 12 (right), there is a correlation between the Probability Sum values and the calculated IoU values; however, we also expect to see some values with high Probability Sum and relatively lower IoU representing detections of parts of the bird (e.g., bill) inside a larger ground truth bounding box.

Bird Detection Both, Probability Sum and IoU, metrics merely quantify the accuracy of the true positive detections (i.e., when the bird is in the frame); however, we can also quantify the ability of the approaches to detect when the bird is outside of the frame. The average precision (AP) uses the detector's confidence as an indicator of whether an object is present. In our approach, we use the entropy of the probability distribution of the attention map to indicate the presence of the bird. As shown in Fig. 12 (left), when the bird is in the nest, entropy values are low, indicating that the model is more confident in its predictions; however, when the bird is not in the nest, attention becomes more spatially distributed resulting in higher entropy values. A Precision-Recall chart can be generated at different Probability Sum/IoU thresholds by calculating the true positives, false positives, and false negatives at each entropy threshold. The AP is calculated by taking the average of 11-recall points (Everingham et al., 2010) on the precision-recall chart. We report the AP values at different Probability Sum/IoU thresholds in Table 4. The entropy $\mathcal{H}$ is calculated using the Eq. 10.

$$\mathcal{H}(P) = -\sum_{i=1}^{k} p_i \log(p_i) \tag{10}$$

5.2.2 Spatial Localization Results

Baselines We evaluated the performance of five different baselines on the spatial segmentation task. Some of the baselines are traditional computer vision approaches (e.g., Optical Flow) that do not require any training data, while others (i.e., DINO) are trained on ImageNet in a self-supervised manner.

Optical Flow outputs the magnitude and direction of the pixels' motion patterns. We only use the magnitude of the distance traveled by each pixel to predict the bird's location. In addition to using the magnitude for localization, we also use the maximum magnitude as a confidence score to detect if the bird is not inside the nest, which is necessary for Average Precision computation.

*Background Subtraction*Zivkovic and Van Der Heijden (2006) shares the same goal (i.e., motion detection) with Optical Flow, but instead of comparing two consecutive frames, it builds a background model of the scene over time and compares it to the current frame. The background

Table 3 Probability Sum evaluation for different approaches at various illumination conditions

Approach	Probability Sum at different conditions					
	Day	Night	Shadows	Sunrise	Sunset	Avg
Optical Flow	0.147	0.047	0.190	0.181	0.153	0.144
Background Subtraction Zivkovic and Van Der Heijden (2006)	0.339	0.593	0.163	0.330	0.432	0.371
Pretrained Inception Szegedy et al. (2016)	0.131	0.139	0.129	0.131	0.140	0.134
DINO Caron et al. (2021)	0.361	0.354	0.273	0.411	0.472	0.374
Ours	**0.858**	**0.779**	**0.784**	**0.708**	**0.864**	**0.799**

Bold highlights best results

model is constantly updated by new images to ensure continuous adaptation to the changing background features. The resulting foreground maps consisted of many disconnected contours. We applied ellipse kernel convolutions to dilate the foreground map and join the disconnected–but close–contours before finding the largest contour for bounding box calculation.

Fixed Boundingbox is based on the premise that the bird stays most of the time sitting in the nest. We design a baseline that assumes a stationary bounding box placed at the center of the nest at all times. This simple baseline uses priors about the behavior of the bird in the dataset and cannot be blindly applied to other datasets.

Pretrained Inception is the same backbone that we use for our approach (Sect. 3.2). The Inception-V3 (Szegedy et al., 2016) backbone is pretrained on ImageNet to extract useful features. We compare the feature grids extracted from every two consecutive images to calculate a feature difference grid. The difference in pretrained features should, in theory, inhibit background features (e.g., shadows) and highlight important bird features. This baseline is designed to test whether a representation of the bird is embedded in the pretrained weights of the backbone or is trained in an online learning manner in the LSTM and attention weights of our approach.

DINO-ImageNet Caron et al. (2021) is a recent self-supervised vision transformer model (ViT (Dosovitskiy et al., 2020)), designed specifically for representation learning. The DINO model is trained on ImageNet from scratch, and the output representations achieve competitive classification results on ImageNet. DINO also utilizes the ViT model to visualize the attention as a heat map. We use the attention maps from DINO as a baseline for spatial segmentation. *DINO-Landmarks v2* uses weights from the ViT model pretrained on the Google Landmarks v2 dataset (Weyand et al., 2020). *DINO-FT-{N}E* refers to a DINO model pretrained on ImageNet and finetuned on 1000 randomly sampled images from the Kagu dataset for N epochs.

iBOT Zhou et al. (2021) is another self-supervised approach to representation learning in images. It uses the masked language model pretraining paradigm to perform masked prediction with an online tokenizer. Similar to DINO, we extract the output attention from the last layer of the ViT architecture (Dosovitskiy et al., 2020) and use it as a spatial segmentation baseline on the Kagu dataset.

Results Discussion We evaluate our results against the baselines and show that our approach outperforms all the baselines. Table 3 shows the Probability Sum performances at different conditions (e.g., day, night). Table 4 presents the Average Precision performances at different probability sum and IoU thresholds. We also provide additional results to further categorize the performance by the bird's state and the type of activity in Figs. 13 and 14, respectively.

Table 3 only considers the frames where the bird is present inside the frame and presents the average Probability Sum for each of the baselines. Based on the results, it can be seen that the shadows pose the biggest challenge to many of the baselines, except for Optical Flow, which is more affected by the low illumination–and less movement–conditions at night. The sharp shadows present in the Kagu dataset can become problematic to object detectors and traditional motion trackers because the shadows can cause severe changes to the look and texture of both the object and the background. A robust and adaptable representation of the bird is a must to overcome these challenges; our approach learns a robust representation by attending to the object and its motion cues and adapting to the changing background through continuous online training. On average, our approach doubles the performance of SOTA and traditional methods while providing stable and robust localization performance across all the different illumination and shadow conditions.

The Average Precision (AP) results reported in Table 4 consider not only the localization performance but also evaluate the detection performance and the ability of the different models to predict if the bird is not inside the nest. For a detection to be counted as true positive, both the detection score (Entropy or Magnitude) and localization score (Prob. Sum. or IoU) must exceed the specified threshold values. Varying the detection threshold results in a Precision-Recall chart which is summarized by the AP value; however, each column reports the results at different localization thresholds. Our approach significantly outperforms the other baselines

Table 4 Comparison of Average Precision performance at different localization metrics' thresholds for traditional and self-supervised spatial segmentation approaches

Threshold	Approach	Threshold Value								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Probability Sum	Optical Flow	0.228	0.129	0.089	0.069	0.056	0.047	0.040	0.033	0.025
	Background Subtraction Zivkovic and Van Der Heijden (2006)	0.750	0.611	0.508	0.425	0.353	0.285	0.255	0.172	0.122
	Pretrained Inception Szegedy et al. (2016)	0.753	0.109	0.092	0.091	0.091	0.000	0.000	0.000	0.000
	DINO-ImageNet Caron et al. (2021)	0.855	0.872	0.590	0.309	0.102	0.033	0.010	0.003	0.002
	DINO-Landmarks v2 Caron et al. (2021)	0.844	0.766	0.563	0.200	0.020	0.000	0.000	0.000	0.000
	DINO-FT-20E Caron et al. (2021)	0.064	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	DINO-FT-100E Caron et al. (2021)	0.038	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	iBOT Zhou et al. (2021)	0.858	0.842	0.797	0.684	0.435	0.178	0.045	0.009	0.002
	Ours	**0.876**	**0.876**	**0.875**	**0.874**	**0.869**	**0.860**	**0.821**	**0.716**	**0.392**
IoU	Optical Flow	0.121	0.085	0.054	0.033	0.018	0.009	0.003	0.001	0.000
	Background Subtraction Zivkovic and Van Der Heijden (2006)	0.389	0.198	0.132	0.093	0.065	0.046	0.030	0.013	0.002
	Fixed Boundingbox	0.838	0.835	0.826	0.610	0.281	0.098	0.023	0.001	0.000
	Pretrained Inception Szegedy et al. (2016)	0.840	0.605	0.314	0.192	0.130	0.101	0.013	0.006	0.001
	DINO-ImageNet Caron et al. (2021)	0.686	0.558	0.420	0.225	0.083	0.021	0.004	0.001	0.000
	iBOT Zhou et al. (2021)	0.787	0.753	0.666	0.467	0.224	0.078	0.013	0.001	0.000
	Ours	**0.876**	**0.874**	**0.855**	**0.758**	**0.569**	**0.376**	**0.188**	**0.053**	**0.004**

Bold highlights best results
FT denotes fine-tuned on the Kagu datset

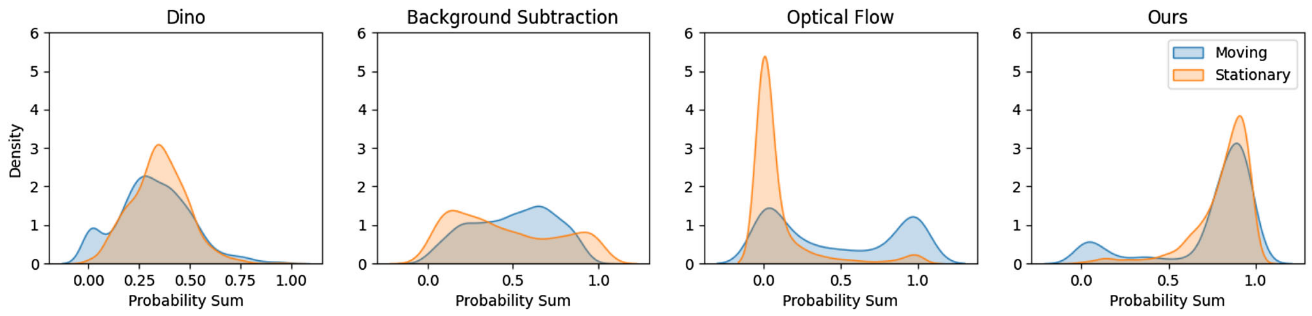
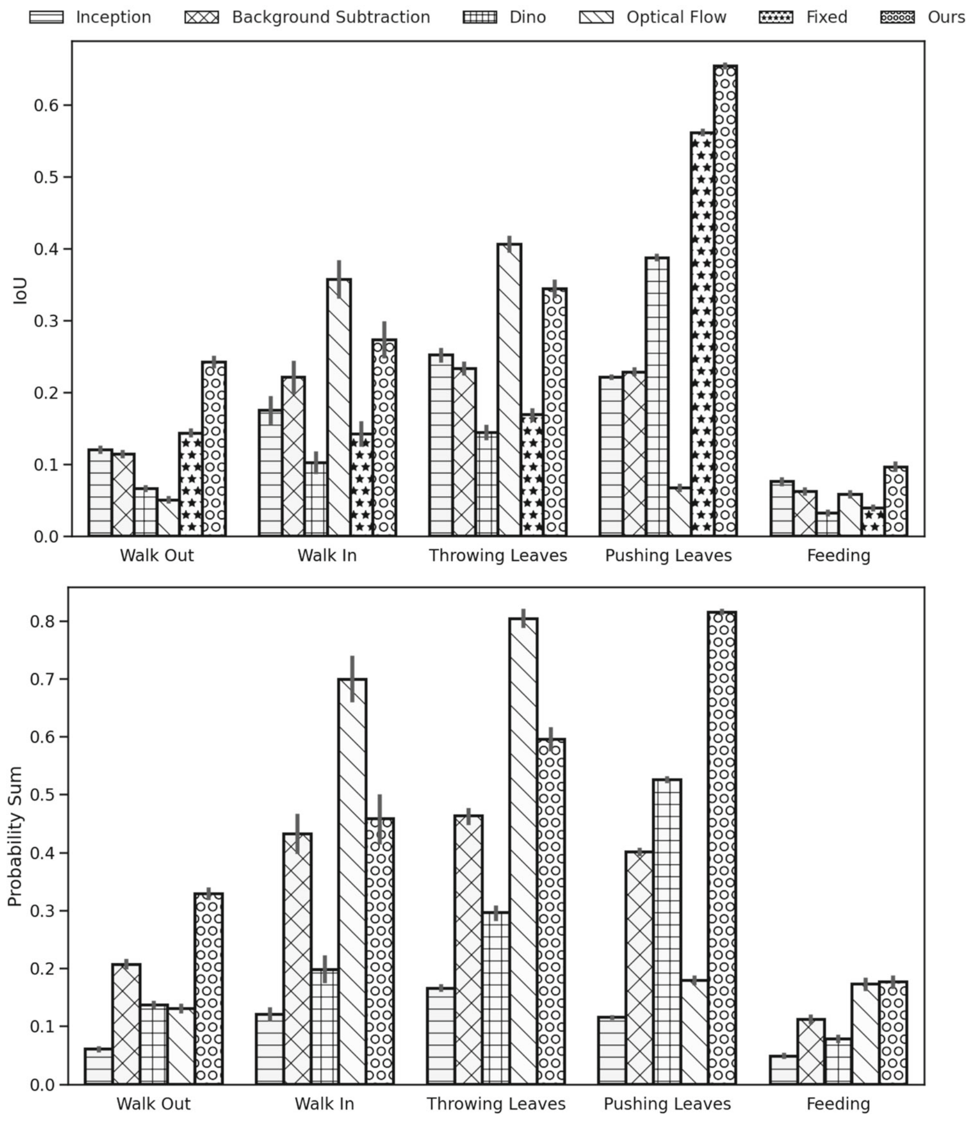


Fig. 13 The density function of the probability sum for several spatial segmentation approaches. Each plot is further categorized by the motion state of the bird

Fig. 14 Spatial segmentation performance comparison for differnt methods. **(top)**: IoU metric. **(bottom)**: Probability Sum metric



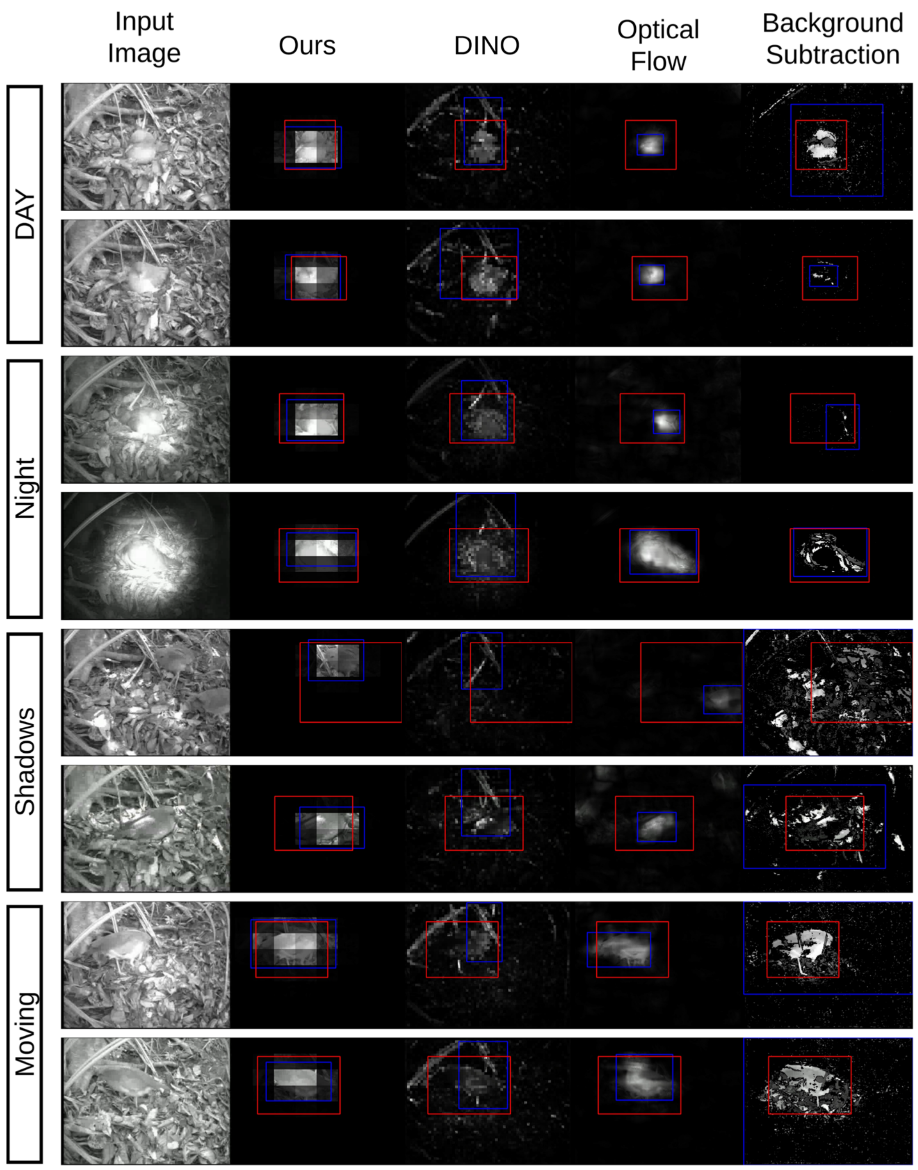


Fig. 15 Qualitative comparison of the spatial segmentation task. Red bounding boxes indicate groundtruth labels, while blue boxes indicate predicted location of the bird

at all localization thresholds. Both Optical Flow and Background Subtraction result in a low performance which can be attributed to the extended periods of time when the bird is stationary in the nest. Background Subtraction (Zivkovic & Van Der Heijden, 2006) results show a big difference between Prob. Sum. and IoU, which can be interpreted as a very localized prediction (low entropy) on part of the bird inside the ground truth bounding box. On the contrary, Pretrained Inception predicts high entropy masks resulting in low Prob. Sum. scores and a large predicted bounding box. Outperforming the pretrained Inception weights highlights the efficacy of our approach in learning a robust representation of the bird, which is not embedded in the Inception pretrained weights. The fixed bounding box baseline is designed based on prior knowledge of the dataset and the bird's location; it performs better than the other baselines but is still outperformed by our approach. We noticed that the DINO baseline consistently attends to a tree branch in the background, which affects the qualitative results by consistently producing average localization scores. Results from DINO highlight the importance of continuous domain adaptation and online training. Using the Google Landmarks V2 (Weyand et al., 2020) weights instead of ImageNet results in lower performance, which is likely due to the lower domain overlap of the Kagu dataset with the Landmarks dataset than with ImageNet. Finetuning the DINO architecture (pretrained on ImageNet) shows a significant decrease in performance, which could result from model collapse. DINO (Caron et al., 2021) relies on the similarity between crops of the same image and requires the main object to occupy most of the image (otherwise, the crop will not contain the object). Our dataset can prove to be challenging for crop-based contrastive approaches like BYOL (Grill

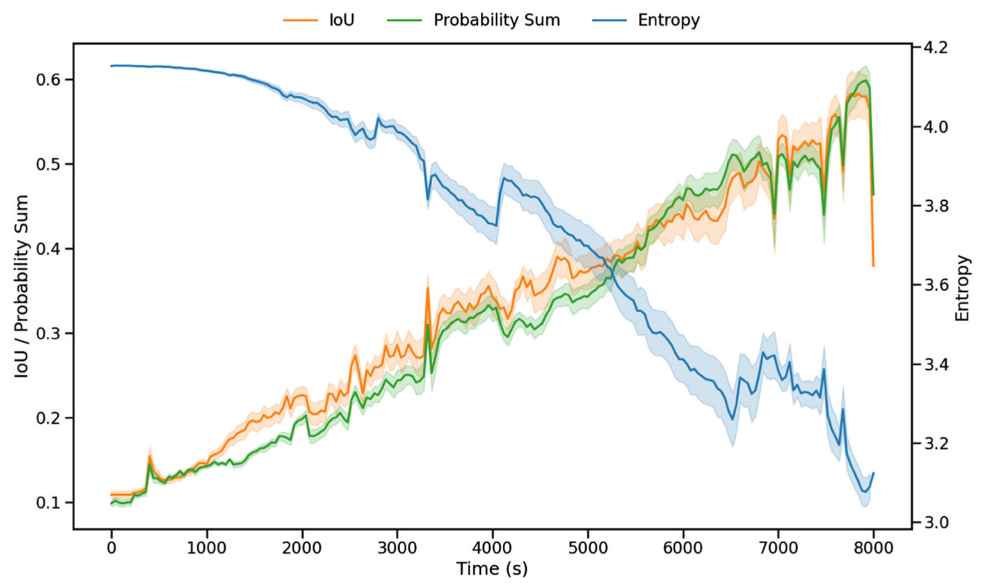


Fig. 16 Localization metrics and entropy values during the first two hours of training. As expected, IoU and Prob.Sum. increase while the entropy of the attention map decreases, indicating more localized predictions. The experiment is repeated four times at different dataset starting points. Results are shown as average lines with widths (in faded color) indicating deviation from the averages

Table 5 Quantitative results on the validation set of TAPOS and Kinetics-GEBD datasets

Supervision	Approach	Rel.Dis.@5%	
		TAPOS	Kinetics
Full	ISBA Ding and Xu (2018)	0.106	–
	TCN Lea et al. (2016); Lin et al. (2018)	0.237	0.588
	CTM Huang et al. (2016)	0.244	–
	TransParser Shao et al. (2020)	0.289	–
	BMN Lin et al. (2019)	–	0.186
	BMN-StartEnd Shou et al. (2021)	–	0.491
	TCN-TAPOS Shou et al. (2021)	–	0.464
	PC Shou et al. (2021)	**0.522**	0.625
	Kang et al. (2021)	–	**0.813**
None	SceneDetect Shou et al. (2021)	0.035	0.275
	PA-Random Shou et al. (2021)	0.158	0.336
	PA-DPC Shou et al. (2021)	0.360	0.396
	Ours	**0.562**	**0.672**

Bold highlights best results
Results are reported as F1 scores and shown at 0.05 Rel.Dis. threshold

et al., 2020) and DINO (Caron et al., 2021) because the object of interest (Kagu bird) occupies, on average, less than 10% of the image. iBOT (Zhou et al., 2021) performs significantly better than DINO on our proposed Kagu dataset, yet it is still outperformed by our approach.

Figure 13 can provide better insights into the performance of each baseline and their differences. We show the density function for the Probability Sum scores over the entire dataset and further categorize the results by the motion state of the bird. It can be seen that Optical Flow performs much worse when the bird is stationary (shown as a high peak close to zero Prob. Sum score). Background Subtraction performs slightly better when the bird is moving, but the performance is almost uniformly distributed over the entire range of Prob. Sum. DINO shows both moving and stationary peaks at approximately 0.4; both distributions have average variances. Our approach shows two high peaks (low variance) close to 1.0 Prob.Sum score. Having similar moving and stationary distributions is desirable behavior for object detectors because it shows the robustness of the detector to the location and motion state of the object of interest. Figure 15 provides qualitative examples to support the findings presented in Fig. 13. We show that, despite our low resolution (8×8 grid) attention, our bird representation is more accurate and robust to different lighting and environmental conditions when compared to DINO and traditional approaches. Results show that DINO is always distracted by other features (e.g., tree branch), which explains the 0.4 Prob.Sum score presented in

Table 6 F1 validation results on Kinetics-GEBD for various supervised and unsupervised GEBD methods at different Rel.Dis. thresholds

Supervision	Approach	Rel.Dis. threshold									
		0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Full	BMN Lin et al. (2019)	0.186	0.204	0.213	0.220	0.226	0.230	0.233	0.237	0.239	0.241
	BMN-StartEnd Shou et al. (2021)	0.491	0.589	0.627	0.648	0.660	0.668	0.674	0.678	0.681	0.683
	TCN-TAPOS Shou et al. (2021)	0.464	0.560	0.602	0.628	0.645	0.659	0.669	0.676	0.682	0.687
	TCN Lea et al. (2016); Lin et al. (2018)	0.588	0.657	0.679	0.691	0.698	0.703	0.706	0.708	0.710	0.712
	PC Shou et al. (2021)	**0.625**	**0.758**	**0.804**	**0.829**	**0.844**	**0.853**	**0.859**	**0.864**	**0.867**	**0.870**
None	SceneDetect Shou et al. (2021)	0.275	0.300	0.312	0.319	0.324	0.327	0.330	0.332	0.334	0.335
	PA-Random Shou et al. (2021)	0.336	0.435	0.484	0.512	0.529	0.541	0.548	0.554	0.558	0.561
	PA-DPC Shou et al. (2021)	0.396	0.488	0.520	0.534	0.544	0.550	0.555	0.558	0.561	0.564
	Ours	**0.672**	**0.768**	**0.793**	**0.804**	**0.809**	**0.812**	**0.814**	**0.815**	**0.816**	**0.818**

Bold highlights best results

Table 7 F1 validation results on TAPOS for various supervised and unsupervised GEBD methods at different Rel.Dis. thresholds

Supervision	Approach	Rel.Dis. threshold									
		0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Full	ISBA Ding and Xu (2018)	0.106	0.170	0.227	0.265	0.298	0.326	0.348	0.369	0.382	0.396
	TCN Lea et al. (2016); Lin et al. (2018)	0.237	0.312	0.331	0.339	0.342	0.344	0.347	0.348	0.348	0.348
	CTM Huang et al. (2016)	0.244	0.312	0.336	0.351	0.361	0.369	0.374	0.381	0.383	0.385
	TransParser Shao et al. (2020)	0.289	0.381	0.435	0.475	0.500	0.514	0.527	0.534	0.540	0.545
	PC Shou et al. (2021)	**0.522**	**0.595**	**0.628**	**0.646**	**0.659**	**0.665**	**0.671**	**0.676**	**0.679**	**0.683**
None	SceneDetect Shou et al. (2021)	0.035	0.045	0.047	0.051	0.053	0.054	0.055	0.056	0.057	0.058
	PA-Random Shou et al. (2021)	0.158	0.233	0.273	0.310	0.331	0.347	0.357	0.369	0.376	0.384
	PA-DPC Shou et al. (2021)	0.360	0.459	0.507	0.543	0.567	0.579	0.592	0.601	0.609	0.615
	Ours	**0.562**	**0.617**	**0.644**	**0.668**	**0.682**	**0.692**	**0.700**	**0.708**	**0.715**	**0.718**

Bold highlights best results

Fig. 13. Traditional approaches (i.e., Optical Flow and Background Subtraction) are heavily affected by the bird's motion state and environmental conditions (e.g., shadows).

In addition to the motion state of the bird, we investigate the performance of all approaches categorized by the event type in Fig. 14. Results show that our approach outperforms the other baselines for all event types except for "Walk-In" and "Throwing", where Optical Flow performs better due to the bird being located at the far edges of the nest. Optical Flow detects motion toward the edges of the frame more accurately than our approach due to the scarcity of training examples with the bird moving close to the edge of the frame/nest.

It is challenging to quantify the performance of models that learn in an online manner from streaming input, mainly due to the absence of explicit training split. Therefore, it is necessary to monitor the model's performance and training progress during the initial phase of training (i.e., the first few hours of data). Ideally, the model should not be able to localize the bird at the beginning of the training; however, it should start to more accurately and confidently locate the bird as the training progresses. During the first few hours, we expect to see an increase in the localization metrics and a decrease in entropy–indicating higher performance and confidence in the predictions as training progresses. Figure 16 displays the IoU and Prob.Sum. localization metrics and the entropy of the attention map plotted for the first two hours of streaming data. Results show a significant increase in localization performance, as well as a decrease in entropy–indicating more localized predictions. We repeat the experiment starting at four different starting times in the dataset; the lines report the averages, while the widths of the lines (shown with faded color) report the deviations from the averages.

5.3 Applicability to other Vision Domains

To test the validity and generalizability of our approach in other domains, we evaluate its performance on other datasets and report state-of-the-art results compared to both supervised and self-supervised generic event boundary detection (GEBD) approaches. In addition to evaluating the performance on a different domain, we test the robustness of our approach to model variations. In this section, we experi-

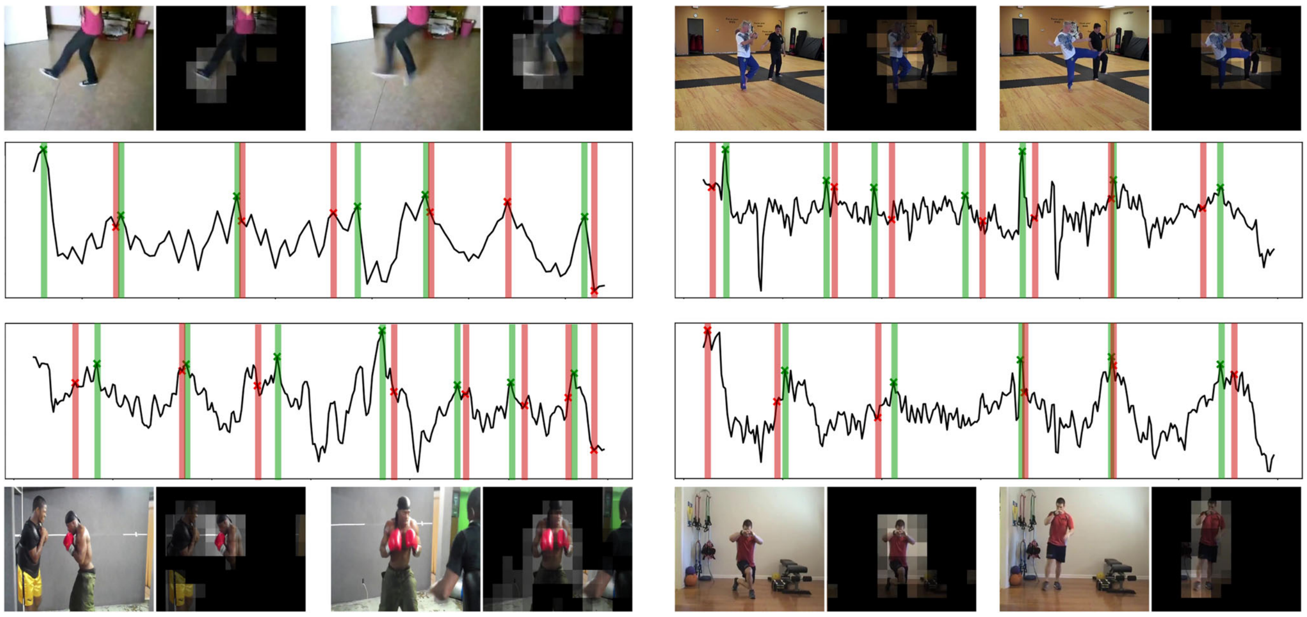


Fig. 17 Qualitative results demonstrating the efficacy of our approach at detecting event boundaries and localizing actions within input frames. Middle plots show the gradient of error signal for 4 different examples. Green vertical lines show the location of the predicted boundaries relative to the annotated boundaries presented as red vertical lines. Action localization qualitative results are presented as an overlay of transparency map (Color figure online)

ment with using a transformer architecture (instead of an LSTM) as the prediction layer. We use a transformer encoder architecture to both encode temporal patches of eight images and decode patch predictions of the next eight images. The transformer is a drop-in replacement to the LSTM recurrent architecture; both models are used for prediction and have the same input and output dimensions.

5.3.1 Datsets

TAPOS The TAPOS (Shao et al., 2020) dataset contains 21 action categories of Olympics sports videos collected from public resources (e.g., YouTube). 16,294 video instances are provided, with an average duration of 9.4 s per instance. The total number of boundaries for all instances is 33K, and the number of instances per class varies from 200 to 1,600. A single annotation is released for each video instance.

Kinetics-GEBD The Kinetics-GEBD (Shou et al., 2021) dataset contains 55,351 videos with 1,498K generic boundaries collected from the Kinetics400 dataset (originally from YouTube). The average duration of each video is 10 s at 30 FPS (300 Frames total). The dataset is divided evenly between training, validation, and test splits. Five annotations per video are released. Unlike other datasets, Kinetics-GEBD provides *generic* boundaries, not only boundaries caused by action change. The main two boundary causes are action change and shot change, which constitute 63.6% and 19.0% of the dataset, respectively. Consistency scores between annotators are provided, and inconsistent annotations are disregarded.

5.3.2 Results Discussion

The performance of our approach is mainly quantified in Table 5. We use the evaluation protocol in Shou et al. (2021) to test the performance of our approach compared to other baselines. The relative distance metric requires normalizing the boundary distance by the total duration of the video instance. We compare our self-supervised architecture to other supervised and self-supervised approaches. The results are reported at 5% Rel.Dis. threshold value for both TAPOS and Kinetics-GEBD datasets. It is clear from the results that our approach significantly outperforms all the other self-supervised approaches and performs competitively with the supervised state-of-the-art on both datasets. Our approach achieved a 0.733 F1 score on the Kinetics-GEBD test set.

Finally, we provide F1 scores at different thresholds for TAPOS and kinetics-GEBD datasets in Tables 6 and 7, respectively. The tables compare the F1 scores to other approaches at different Rel.Dis. thresholds. For the TAPOS dataset (Table 7), the results show that our approach consistently outperforms all other approaches across all thresholds. However, for the Kinetics-GEBD (Table 6), our approach doubles the performance of all other self-supervised approaches and performs on par with the best-supervised approach when considering the whole range of thresholds.

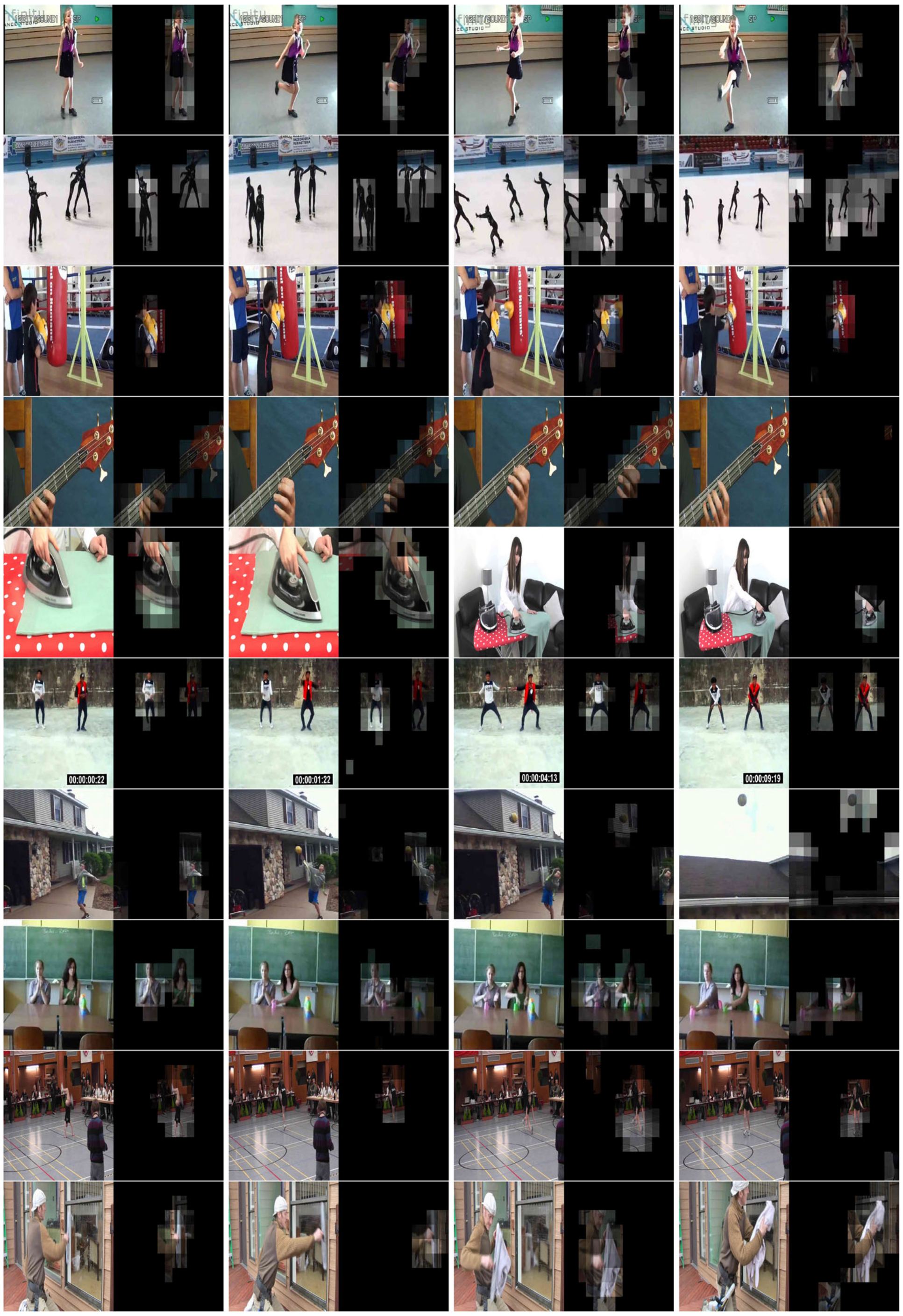


Fig. 18 Additional qualitative results for action localization from the Kinetics-GEBD dataset. Results are displayed as transparency map overlayed on the input RGB images

We present qualitative results (Fig. 17) in the form of prediction error plots annotated with ground truth and prediction timestamps. We show that the predicted boundaries (shown in green lines), detected at the peaks of the prediction error signal, appear close to the ground truth boundaries (red vertical lines), which results in high precision. It is also clear that the number of predicted boundaries is relatively close to the ground truth boundaries resulting in a high recall rate. The figure also shows the corresponding action localization result for each plot. Action localization results are presented as alpha/transparency channels overlayed on top of the input images; the visible regions of the image highlight the most dominant action. We provide qualitative results showing videos during which the camera is moving when our approach performs reasonably well on both segmentation and localization tasks. This demonstrates the effectiveness of the predictive approach in learning to predict and ignore camera motion when calculating the total loss. More action localization results are provided in Fig. 18.

5.4 Limitations

We have demonstrated the ability of our approach to work on different vision domains, as well as provide adapting representations of objects that are robust to environmental and lighting conditions. However, there exists a few limitations worth discussing. Our approach does not label its outputs; while it can temporally detect events and spatially localize the objects in frames, it cannot label these events or objects. This results in not being able to handle instance segmentation when multiple objects are causing the perceived event. The algorithm outputs a single union mask of the objects, as shown in Fig. 18, to represent all objects causing the event. Additionally, the detected temporal events are currently unlabeled, which results in not being able to count the instances of a specific event or provide a summary of the different types of events present in the video. Our approach uses motion cues as a learning signal. It does not require the object to be always moving (as required by traditional approaches), but it uses motion cues to learn the representation of the objects causing the event; the learned representations are used to localize the objects even when they are not moving.

6 Conclusion

We demonstrate a stream learning approach to temporal event segmentation. Our framework can effectively segment a long sequence of activities (video) into a set of individual events. We introduce a novel approach to extracting attention results from unsupervised temporal event segmentation networks. Gating the loss signal with different threshold values can result in segmentation at different granularities. We introduce a new remarkably long wildlife monitoring dataset that is more realistic and challenging for event segmentation tasks; our dataset provides a true benchmark containing most of the challenges currently being artificially simulated in continual and incremental learning datasets. We evaluate our approach on spatial and temporal segmentation on the Kagu video monitoring dataset and compare our performance to traditional and state-of-the-art methods. Additionally, our approach is tested on datasets from other domains (i.e., action). We demonstrated the effectiveness of our approach in understanding the higher-level spatiotemporal features required for practical event segmentation. Further employing this approach for automated ethogramming will facilitate progress in the study of animal behavior by reducing the manual labor of dataset annotations.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-023-01781-2.

Acknowledgements This research was supported by the US National Science Foundation Grants CNS 1513126 and IIS 1956050. The bird video dataset used in this paper was made possible through funding from the Polish National Science Centre (Grant NCN 2011/01/M/NZ8/03344 and 2018/29/B/NZ8/02312). Province Sud (New Caledonia) issued all permits required for data collection. The authors would like to thank Sujal Vijayaraghavan and Caio da Silva for their help with verifying the Kagu dataset annotations.

References

Aakur, S. N., & Sarkar, S. (2019). A perceptual prediction framework for self supervised event segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1197–1206).

Aakur, S., & Sarkar, S. (2020). Action localization through continual predictive learning. In *European conference on computer vision* (pp. 300–317). Springer

ActEV: Activities in Extended Video. https://actev.nist.gov/

Adeli, V., Ehsanpour, M., Reid, I., Niebles, J.C., Savarese, S., Adeli, E., & Rezatofighi, H. (2021). Tripod: Human trajectory and pose dynamics forecasting in the wild. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*.

Akçay, H. G., Kabasakal, B., Aksu, D., Demir, N., Öz, M., & Erdoğan, A. (2020). Automated bird counting with deep learning for regional bird distribution mapping. *Animals, 10*(7), 1207.

Alayrac, J.-B., Bojanowski, P., Agrawal, N., Sivic, J., Laptev, I., & Lacoste-Julien, S. (2016). Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4575–4583).

Apostolidis, E., Balaouras, G., Mezaris, V., & Patras, I. (2021). Combining global and local attention with positional encoding for video summarization. In *2021 IEEE international symposium on multimedia (ISM)* (pp. 226–234). IEEE

Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., & Schmid, C. (2021). Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)* (pp. 6836–6846).

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473

Bardes, A., Ponce, J., & LeCun, Y. (2022). VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International conference on learning representations*.

Benczúr, A.A., Kocsis, L., & Pálovics, R. (2018). Online machine learning in big data streams. arXiv preprint arXiv:1802.05872

Bhatnagar, B.L., Singh, S., Arora, C., & Jawahar, C. (2017). CVIT, K. Unsupervised learning of deep feature representation for clustering egocentric actions. In *IJCAI* (pp. 1447–1453).

Bojanowski, P., Lajugie, R., Bach, F., Laptev, I., Ponce, J., Schmid, C., & Sivic, J. (2014). Weakly supervised action labeling in videos under ordering constraints. In *European conference on computer vision* (pp. 628–643). Springer

Bondi, E., Jain, R., Aggrawal, P., Anand, S., Hannaford, R., Kapoor, A., Piavis, J., Shah, S., Joppa, L., Dilkina, B., & Tambe, M. (2020). Birdsai: A dataset for detection and tracking in aerial thermal infrared videos. In *WACV*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., & Amodei, D. (2020). Language models are few-shot learners. In *Advances in neural information processing systems* (vol. 33) (NeurIPS)

Burgos-Artizzu, X. P., Dollár, P., Lin, D., Anderson, D. J., & Perona, P. (2012). Social behavior recognition in continuous video. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 1322–1329). https://doi.org/10.1109/CVPR.2012.6247817

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European conference on computer vision* (pp. 213–229). Springer

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*.

Chen, X., & He, K. (2021). Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 15750–15758).

Chen, Z., Xie, L., Niu, J., Liu, X., Wei, L., & Tian, Q. (2021). Visformer: The vision-friendly transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*.

Choi, Y., El-Khamy, M., & Lee, J. (2021). Dual-teacher class-incremental learning with data-free generative replay. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR) workshops* (pp. 3543–3552).

Clapham, M., Miller, E., Nguyen, M., & Darimont, C. T. (2020). Automated facial recognition for wildlife that lack unique markings: A deep learning approach for brown bears. *Ecology and Evolution, 10*(23), 12883–12892.

Corona, K., Osterdahl, K., Collins, R., & Hoogs, A. (2021). Meva: A large-scale multiview, multimodal video dataset for activity detection. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 1060–1068).

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., Solorio, T. (eds.) *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*

Ding, L., & Xu, C. (2018). Weakly-supervised action segmentation with iterative soft boundary assignment. In *IEEE/CVF conference on computer vision and pattern recognition*.

Ding, L., & Xu, C. (2018). Weakly-supervised action segmentation with iterative soft boundary assignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6508–6516).

Doersch, C., Gupta, A., & Efros, A. A. (2015) Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision* (pp. 1422–1430).

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*.

Dwivedi, S. K., Athanasiou, N., Kocabas, M., & Black, M. J. (2021). Learning to regress bodies from images using differentiable semantic rendering. In *Proceedings of international conference on computer vision (ICCV)* (pp. 11250–11259).

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision, 88*(2), 303–338.

Fajtl, J., Sokeh, H. S., Argyriou, V., Monekosso, D., & Remagnino, P. (2018). Summarizing videos with attention. In *Asian conference on computer vision* (pp. 39–54). Springer

Fangbemi, A.S., Lu, Y.F., Xu, M.Y., Luo, X.W., Rolland, A., & Raissi, C. (2020). Zoobuilder: 2D and 3D pose estimation for quadrupeds using synthetic data. arXiv preprint arXiv:2009.05389

Ferrari, V., Hebert, M., Sminchisescu, C., & Weiss, Y. (eds.) (2018). Computer Vision-ECCV 2018-15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part XVI. Lecture Notes in Computer Science

Ferreira, A. C., Silva, L. R., Renna, F., Brandl, H. B., Renoult, J. P., Farine, D. R., Covas, R., & Doutrelant, C. (2020). Deep learning-based methods for individual recognition in small birds. *Methods in Ecology and Evolution, 11*(9), 1072–1085.

Finn, C., Goodfellow, I., & Levine, S. (2016). Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems* (pp. 64–72).

Garcia del Molino, A., Lim, J.-H., Tan, A.-H.: Predicting visual context for unsupervised event segmentation in continuous photo-streams. In *Proceedings of the 26th ACM international conference on multimedia*.

Girdhar, R., Carreira, J., Doersch, C., & Zisserman, A. (2019). Video action transformer network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 244–253)

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems, 33*, 21271–21284.

Guan, T., Wang, J., Lan, S., Chandra, R., Wu, Z., Davis, L., & Manocha, D. (2022). M3detr: Multi-representation, multi-scale, mutual-relation 3D object detection with transformers. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*.

Gula, R., Theuerkauf, J., Rouys, S., & Legault, A. (2010). An audio/video surveillance system for wildlife. *European Journal of Wildlife Research, 56*, 803–807. https://doi.org/10.1007/s10344-010-0392-y

Günel, S., Rhodin, H., Morales, D., Campagnolo, J., Ramdya, P., & Fua, P. (2019). Deepfly 3D, a deep learning-based approach for 3D limb and appendage tracking in tethered, adult *Drosophila*. *eLife*

Han, T., Xie, W., & Zisserman, A. (2019). Video representation learning by dense predictive coding. In *Proceedings of the IEEE/CVF international conference on computer vision workshops* (pp. 0–0).

Hawkins, J., & Ahmad, S. (2016). Why neurons have thousands of synapses, a theory of sequence memory in neocortex. *Frontiers in Neural Circuits, 10*, 23.

Hawkins, J., Ahmad, S., & Cui, Y. (2017). A theory of how columns in the neocortex enable learning the structure of the world. *Frontiers in Neural Circuits, 11*, 81.

Hawkins, W. D., & DuRant, S. E. (2020). Applications of machine learning in behavioral ecology: Quantifying avian incubation behavior and nest conditions in relation to environmental temperature. *PLoS ONE, 15*(8), 0236925.

Hayden, B. Y., Park, H. S., & Zimmermann, J. (2021). Automated pose estimation in primates. *American Journal of Primatology, 23*, 348.

Hesse, N., Pujades, S., Black, M. J., Arens, M., Hofmann, U., & Schroeder, S. (2020). Learning and tracking the 3D body shape of freely moving infants from RGB-D sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 42*(10), 2540–2551. https://doi.org/10.1109/TPAMI.2019.2917908

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Holmberg, J., Norman, B., & Arzoumanian, Z. (2009). Estimating population size, structure, and residency time for whale sharks rhincodon typus through collaborative photo-identification. *Endangered Species Research, 7*(1), 39–53.

Huang, Y., Bi, H., Li, Z., Mao, T., & Wang, Z. (2019). Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 6272–6281).

Huang, D.-A., Fei-Fei, L., & Niebles, J. C. (2016). Connectionist temporal modeling for weakly supervised action labeling. In *ECCV*.

Huang, D.-A., Fei-Fei, L., & Niebles, J. C. (2016). Connectionist temporal modeling for weakly supervised action labeling. In *European conference on computer vision* (pp. 137–153). Springer

Ivanovic, B., & Pavone, M. (2019). The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 2375–2384).

Ji, Z., Xiong, K., Pang, Y., & Li, X. (2019). Video summarization with attention-based encoder-decoder networks. *IEEE Transactions on Circuits and Systems for Video Technology, 30*(6), 1709–1717.

Kamath, A., Singh, M., LeCun, Y., Misra, I., Synnaeve, G., & Carion, N. (2021). Mdetr-modulated detection for end-to-end multi-modal understanding. In *2021 IEEE/CVF international conference on computer vision (ICCV)*.

Kanazawa, A., Kovalsky, S., Basri, R., & Jacobs, D. (2016). Learning 3D deformation of animals from 2d images. *Computer Graphics Forum, 35*(2), 365–374. https://doi.org/10.1111/cgf.12838

Kang, H., Kim, J., Kim, K., Kim, T., & Kim, S. J. (2021). Winning the cvpr'2021 kinetics-gebd challenge: Contrastive learning approach. arXiv preprint arXiv:2106.11549

Kellenberger, B., Veen, T., Folmer, E., & Tuia, D. (2021). 21 000 birds in 4.5 h: Efficient large-scale seabird detection with machine learning. *Remote Sensing in Ecology and Conservation*

Kocabas, M., Athanasiou, N., & Black, M. J. (2020). VIBE: Video inference for human body pose and shape estimation. In *Proceedings IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 5252–5262). IEEE. https://doi.org/10.1109/CVPR42600.2020.00530

Kocabas, M., Huang, C.-H. P., Hilliges, O., & Black, M. J. (2021). PARE: Part attention regressor for 3D human body estimation. In *Proceedings of international conference on computer vision (ICCV)* (pp. 11127–11137).

Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I., Rezatofighi, H., & Savarese, S. (2019). Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *Advances in neural information processing systems*

Lam, W., Liu, L., Prasad, S., Rajaraman, A., Vacheri, Z., & Doan, A. (2012). Muppet: Mapreduce-style processing of fast data. arXiv preprint arXiv:1208.4175

Lamb, A. M., Goyal, A. G. A. P., Zhang, Y., Zhang, S., Courville, A. C., & Bengio, Y. (2016). Professor forcing: A new algorithm for training recurrent networks. In *Advances in neural information processing systems* (pp. 4601–4609).

Lea, C., Flynn, M. D., Vidal, R., Reiter, A., & Hager, G. D. (2017). Temporal convolutional networks for action segmentation and detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 156–165).

Lea, C., Reiter, A., Vidal, R., & Hager, G.D. (2016) Segmental spatiotemporal cnns for fine-grained action segmentation. In *ECCV*.

Lea, C., Reiter, A., Vidal, R., & Hager, G. D. (2016). Segmental spatiotemporal cnns for fine-grained action segmentation. In *European conference on computer vision* (pp. 36–52). Springer

Lee, N., Choi, W., Vernaza, P., Choy, C. B., Torr, P. H., & Chandraker, M. (2017). Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 336–345).

Li, S., Gunel, S., Ostrek, M., Ramdya, P., Fua, P., & Rhodin, H. (2020). Deformation-aware unpaired image translation for pose estimation on laboratory animals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13158–13168).

Li, S., Li, J., Tang, H., Qian, R., & Lin, W. (2019). Atrw: A benchmark for amur tiger re-identification in the wild. arXiv preprint arXiv:1906.05586

Li, Y., Wu, C.-Y., Fan, H., Mangalam, K., Xiong, B., Malik, J., & Feichtenhofer, C. (2021). Improved multiscale vision transformers for classification and detection. arXiv preprint arXiv:2112.01526

Liang, J., Jiang, L., Niebles, J. C., Hauptmann, A. G., & Fei-Fei, L. (2019). Peeking into the future: Predicting future person activities and locations in videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5725–5734).

Lin, Y., Guo, X., & Lu, Y. (2021). Self-supervised video representation learning with meta-contrastive network. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 8239–8249).

Lin, T., Liu, X., Li, X., Ding, E., & Wen, S. (2019). Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE/CVF international conference on computer vision*

Lin, T., Zhao, X., Su, H., Wang, C., & Yang, M. (2018). Bsn: Boundary sensitive network for temporal action proposal generation. In *European conference on computer vision*.

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al. (2021). Swin transformer v2: Scaling up capacity and resolution. arXiv preprint arXiv:2111.09883

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692

Lomonaco, V., & Maltoni, D. (2017). Core50: A new dataset and benchmark for continuous object recognition. In *Conference on robot learning* (pp. 17–26). PMLR

Lorbach, M., Kyriakou, E. I., Poppe, R., van Dam, E. A., Noldus, L. P. J. J., & Veltkamp, R. C. (2018). Learning to recognize rat social behavior: Novel dataset and cross-dataset application. *Journal of Neuroscience Methods, 300*, 166–172.

Loschky, L. C., Larson, A. M., Smith, T. J., & Magliano, J. P. (2020). The scene perception and event comprehension theory (spect) applied to visual narratives. *Topics in Cognitive Science, 12*(1), 311–351.

Lotter, W., Kreiman, G., & Cox, D. (2017). Deep predictive coding networks for video prediction and unsupervised learning. In *International conference on learning representations*.

Luong, T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 conference on empirical methods in natural language processing*.

Malmaud, J., Huang, J., Rathod, V., Johnston, N., Rabinovich, A., & Murphy, K. (2015). What's cookin'? interpreting cooking videos using text, speech and vision. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Mathis, M. W., & Mathis, A. (2020). Deep learning tools for the measurement of animal behavior in neuroscience. *Current Opinion in Neurobiology, 60*, 1–11.

McIntosh, D., Marques, T. P., Albu, A. B., Rountree, R., & De Leo, F. (2020). Movement tracks for the automatic detection of fish behavior in videos. arXiv preprint arXiv:2011.14070

Metcalf, K., & Leake, D. (2017). Modelling unsupervised event segmentation: Learning event boundaries from prediction errors. In *CogSci*.

Mittal, S., Galesso, S., & Brox, T. (2021) Essentials for class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR) workshops* (pp. 3513–3522).

Mohamed, A., Qian, K., Elhoseiny, M., & Claudel, C. (2020). Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 14424–14432).

NOAA Arctic Seals. https://lila.science/datasets/arcticseals

Noninvasive bee tracking in videos: Deep learning algorithms and cloud platform design specifications (2021)

Pan, T., Song, Y., Yang, T., Jiang, W., & Liu, W. (2021). Videomoco: Contrastive video representation learning with temporally adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11205–11214).

Pedersen, M., Haurum, J. B., Bengtson, S. H., & Moeslund, T. B. (2020). 3D-zef: A 3D zebrafish tracking benchmark dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2426–2436).

Petrovich, M., Black, M. J., & Varol, G. (2021). Action-conditioned 3D human motion synthesis with transformer VAE. In *Proceedings of international conference on computer vision (ICCV)* (pp. 10985–10995).

Price, W., Vondrick, C., & Damen, D. (2022) Unweavenet: Unweaving activity stories. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13770–13779).

Qiu, J., Huang, G., & Lee, T. S. (2019). A neurally-inspired hierarchical prediction network for spatiotemporal sequence learning and prediction. arXiv preprint arXiv:1901.09002

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog, 1*(8), 9.

Radvansky, G. A., Krawietz, S. A., & Tamplin, A. K. (2011). Walking through doorways causes forgetting: Further explorations. *Quarterly Journal of Experimental Psychology, 64*(8), 1632–1645.

Radvansky, G. A., & Zacks, J. M. (2014). *Event cognition*. Oxford University Press.

Ratnayake, M. N., Dyer, A. G., & Dorin, A. (2021). Tracking individual honeybees among wildflower clusters with computer vision-facilitated pollinator monitoring. *PLoS ONE, 16*(2), 0239504.

Richard, A., Kuehne, H., & Gall, J. (2017) Weakly supervised action learning with rnn based fine-to-coarse modeling. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 754–763).

Rodriguez-Baena, D. S., Gomez-Vela, F. A., García-Torres, M., Divina, F., Barranco, C. D., Daz-Diaz, N., Jimenez, M., & Montalvo, G. (2020). Identifying livestock behavior patterns based on accelerometer dataset. *Journal of Computational Science, 41*, 101076. https://doi.org/10.1016/j.jocs.2020.101076

Rose, P. E., & Riley, L. M. (2021). Conducting behavioural research in the zoo: A guide to ten important methods, concepts and theories. *Journal of Zoological and Botanical Gardens, 2*(3), 421–444. https://doi.org/10.3390/jzbg2030031

Sahoo, D., Pham, Q., Lu, J., & Hoi, S. C. H. (2018). Online deep learning: Learning deep neural networks on the fly. In *Proceedings of the twenty-seventh international joint conference on artificial intelligence, IJCAI-18*.

Salzmann, T., Ivanovic, B., Chakravarty, P., & Pavone, M. (2020) Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16* (pp. 683–700). Springer

Sanakoyeu, A., Khalidov, V., McCarthy, M. S., Vedaldi, A., & Neverova, N. (2020). Transferring dense pose to proximal animal classes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5233–5242).

Sarfati, R., Hayes, J. C., Sarfati, É., & Peleg, O. (2020). Spatio-temporal reconstruction of emergent flash synchronization in firefly swarms via stereoscopic 360-degree cameras. *Journal of The Royal Society Interface, 17*(170), 20200179.

Scenedetect: Video Scene Cut Detection and Analysis Tool. https://github.com/Breakthrough/PySceneDetect

Sener, F., & Yao, A. (2018). Unsupervised learning and segmentation of complex activities from video. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 8368–8376).

Shao, D., Zhao, Y., Dai, B., & Lin, D. (2020). Intra- and inter-action understanding via temporal action parsing. In *IEEE Conference on computer vision and pattern recognition (CVPR)*

Shou, M.Z., Lei, S.W., Wang, W., Ghadiyaram, D., & Feiszli, M. (2021). Generic event boundary detection: A benchmark for event segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*

Silvia Zuffi, T. B.-W. M. J. B., & Angjoo, K. (2019). Three-d safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *The IEEE international conference on computer vision (ICCV)*.

Speer, N. K., Swallow, K. M., & Zacks, J. M. (2003). Activation of human motion processing areas during event perception. *Cognitive, Affective, and Behavioral Neuroscience, 3*(4), 335–345.

Sun, J. J., Karigo, T., Chakraborty, D., Mohanty, S. P., Wild, B., Sun, Q., Chen, C., Anderson, D. J., Perona, P., Yue, Y., & Kennedy, A. (2021). The multi-agent behavior dataset: Mouse dyadic social interactions. arXiv preprint arXiv:2104.02710

Sun, W., Zhang, J., Wang, D., Geng, Y.-a., & Li, Q. (2021). Ilcoc: An incremental learning framework based on contrastive one-class classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR) workshops* (pp. 3580–3588).

Swanson, A., Kosmala, M., Lintott, C., Simpson, R., Smith, A., & Packer, C. (2015). Snapshot serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna. *Scientific Data, 2*(1), 1–14.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2818–2826).

Testard, C., Tremblay, S., & Platt, M. (2021). From the field to the lab and back: Neuroethology of primate social behavior. *Current Opinion in Neurobiology, 68*, 76–83.

Tuia, D., Kellenberger, B., Beery, S., Costelloe, B. R., Zuffi, S., Risse, B., Mathis, A., Mathis, M. W., van Langevelde, F., Burghardt, T., et al. (2022). Perspectives in machine learning for wildlife conservation. *Nature Communications, 13*(1), 1–15.

Valletta, J. J., Torney, C., Kings, M., Thornton, A., & Madden, J. (2017). Applications of machine learning in animal behaviour studies. *Animal Behaviour, 124*, 203–220.

Vasconcelos, C., Birodkar, V., & Dumoulin, V. (2022). Proper reuse of image classification features improves object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13628–13637).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you

need. In *Advances in neural information processing systems* (pp. 5998–6008).
VidalMata, R. G., Scheirer, W. J., Kukleva, A., Cox, D., & Kuehne, H. (2021). Joint visual-temporal embedding for unsupervised learning of actions in untrimmed sequences. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision (WACV)* (pp. 1238–1247).
Wang, C., Cai, S., & Tan, G. (2021). Graphtcn: Spatio-temporal interaction modeling for human trajectory prediction. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 3450–3459).
Wang, Y., Gao, Z., Long, M., Wang, J., & Yu, P. S. (2018). Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. arXiv preprint arXiv:1804.06300
Wang, G. (2019). Machine learning for inferring animal behavior from location and movement data. *Ecological Informatics, 49*, 69–76.
Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., & Perona, P. (2010). Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology
Weyand, T., Araujo, A., Cao, B., & Sim, J. (2020). Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2575–2584).
wichers, N., Villegas, R., Erhan, D., & Lee, H. (2018). Hierarchical long-term video prediction without supervision. In *Proceedings of the 35th international conference on machine learning*.
Xian, Y., Lampert, C. H., Schiele, B., & Akata, Z. (2019). Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 41*(9), 2251–2265. https://doi.org/10.1109/TPAMI.2018.2857768
Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning* (pp. 2048–2057).
Yan, S., Xiong, X., Arnab, A., Lu, Z., Zhang, M., Sun, C., & Schmid, C. (2022). Multiview transformers for video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3333–3343).
Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems* (pp. 5754–5764).
Yao, Y., Mohan, A., Bliss-Moreau, E., Coleman, K., Freeman, S. M., Machado, C. J., Raper, J., Zimmermann, J., Hayden, B. Y., & Park, H. S. (2021). Openmonkeychallenge: Dataset and benchmark challenges for pose tracking of non-human primates. *bioRxiv*
Zacks, J. M. (2004). Using movement and intentions to understand simple events. *Cognitive Science, 28*(6), 979–1008.
Zacks, J. M., Speer, N. K., Swallow, K. M., Braver, T. S., & Reynolds, J. R. (2007). Event perception: A mind-brain perspective. *Psychological Bulletin, 133*(2), 273.
Zacks, J. M., & Swallow, K. M. (2007). Event segmentation. *Current Directions in Psychological Science, 16*(2), 80–84.
Zacks, J. M., & Tversky, B. (2001). Event structure in perception and conception. *Psychological Bulletin, 127*(1), 3.
Zacks, J. M., Tversky, B., & Iyer, G. (2001). Perceiving, remembering, and communicating structure in events. *Journal of Experimental Psychology: General, 130*(1), 29.
Zaheer, M., Guruganesh, G., Dubey, K.A., Ainslie, J., Alberti, C., Ontañón, S., Pham, P., Ravula, A., Wang, Q., Yang, L., & Ahmed, A. (2020). Big bird: Transformers for longer sequences. In *Advances in neural information processing systems* (vol. 33) (NeurIPS)
Zbontar, J., Jing, L., Misra, I., LeCun, Y., & Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning* (pp. 12310–12320). PMLR
Zhang, L., Dunn, T., Marshall, J., Olveczky, B., & Linderman, S. (2021). Animal pose estimation from video data with a hierarchical von mises-fisher-gaussian model. In *International conference on artificial intelligence and statistics* (pp. 2800–2808). PMLR
Zhang, Y., Li, X., Liu, C., Shuai, B., Zhu, Y., Brattoli, B., Chen, H., Marsic, I., & Tighe, J. (2021). Vidtr: Video transformer without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*.
Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A., & Kong, T. (2021). ibot: Image bert pre-training with online tokenizer. In *International conference on learning representations*.
Zhu, W., Lu, J., Li, J., & Zhou, J. (2020). Dsnet: A flexible detect-to-summarize network for video summarization. *IEEE Transactions on Image Processing, 30*, 948–962.
Zivkovic, Z., & Van Der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters, 27*(7), 773–780.
Zuffi, S., Kanazawa, A., & Black, M.J. (2018). Lions and tigers and bears: Capturing non-rigid, 3D, articulated shape from images. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3955–3963).
Zuffi, S., Kanazawa, A., Jacobs, D., & Black, M.J. (2017). 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conferences on computer vision and pattern recognition (CVPR)*.