Deep Learning for Object Detection and Segmentation in Videos
Toward an Integration With Domain Knowledge

Ilioudi, Athina; Dabiri, Azita ; Wolf, Ben J.; Schutter, Bart De

DOI
10.1109/ACCESS.2022.3162827

Publication date
2022

Document Version
Final published version

Published in
IEEE Access

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 8, 2022, accepted March 21, 2022, date of publication March 28, 2022, date of current version April 4, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3162827

Deep Learning for Object Detection and Segmentation in Videos: Toward an Integration With Domain Knowledge

ATHINA ILIOUDI, AZITA DABIRI, BEN J. WOLF, AND BART DE SCHUTTER, (Fellow, IEEE)
Delft Center for Systems and Control, Delft University of Technology, 2628 Delft, The Netherlands

Corresponding author: Athina Ilioudi (a.ilioudi@tudelft.nl)

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Programme under Grant 871295 (SeaClear), and in part by the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Programme under Grant 101018826-CLariNet.

ABSTRACT Deep learning has enabled the rapid expansion of computer vision tasks from image frames to video segments. This paper focuses on the review of the latest research in the field of computer vision tasks in general and on object localization and identification of their associated pixels in video frames in particular. After performing a systematic analysis of the existing methods, the challenges related to computer vision tasks are presented. In order to address the existing challenges, a hybrid framework is proposed, where deep learning methods are coupled with domain knowledge. An additional feature of this survey is that a review of the currently existing approaches integrating domain knowledge with deep learning techniques is presented. Finally, some conclusions on the implementation of hybrid architectures to perform computer vision tasks are discussed.

INDEX TERMS Computer vision, object detection, deep learning, theory-guided data science.

I. INTRODUCTION

Just as motion perception is essential to our visual system, allowing us to interpret the world, to detect the presence of creatures [25], and to avoid danger [34], video computer vision helps artificial intelligence agents to decipher their surrounding environment and to synthesize actionable information. Inspired by the human visual system and enabled by the latest advancements in deep learning (DL), novel video processing methods are emerging that achieve remarkable results and that seek to revolutionize how computer vision tasks are implemented. Yet, similarly to human perception, computer vision is quite prone to illusions.

The fast pace of DL breakthroughs in combination with the improvement in hardware capabilities in terms of computation power, memory capacity, and sensor resolution have accelerated the spread of data-driven methods over the conventional computer vision techniques. Contrary to classical techniques, DL reaches human-level accuracy, requires less expert analysis, and provides superior flexibility including allowing re-training whenever new data are available [115].

The associate editor coordinating the review of this manuscript and approving it for publication was Zhongyi Guo.

The objective of this work is to investigate the advancements of deep learning techniques for computer vision tasks in videos as well as their research perspectives to address their current weaknesses. More specifically, the contributions of our study are trifold:

- We present an analysis of the existing DL techniques for detection and segmentation of objects in videos.
- We present an overview of the challenges with the existing data-driven approaches.
- We outline new directions for research in video processing.

The paper is organized in seven sections. Section II presents an overview of necessary preliminary knowledge. Section III gives a comprehensive overview of DL-based video computer vision methods. In Section IV the current challenges are presented and analyzed. To address these challenges, Section V presents an overview of approaches that couple DL methods with domain knowledge. Section VI highlights the most prominent topics that are expected to draw major interest from the research community in the following years, and Section VII gives concluding remarks.

A list of abbreviations mentioned in this paper and their definitions are presented in Table 1.

TABLE 1. List of abbreviations.

Abbreviation	Definition
CNN	Convolutional Neural Network
DFF	Deep Feature Flow
DL	Deep Learning
DM	Dynamics Model
FFT	Fast Fourier Transform
FGFA	Flow-Guided Feature Aggregation
GAN	Generative Adversarial Network
GRU	Gated Recurrent Unit
IID	Independent and Identically Distributed
LSTM	Long Short-Term Memory
RBM	Restricted Boltzmann Machine
RCNN	Region-based Convolutional Neural Network
RNN	Recurrent Neural Network
SNN	Siamese Neural Network
SSD	Single-Shot Detector
VAE	Variational Auto-encoder
YOLO	You Only Look Once

II. PRELIMINARIES

In this section, we introduce the most typical tasks of computer vision and we present a brief, comparative analysis between deep learning and conventional techniques in the domain of computer vision, as well as an overview of basic deep learning methods such as convolutional neural networks, restricted Boltzmann machines, and auto-encoders, which constitute the core for DL architectures in computer vision.

A. COMPUTER VISION TASKS

Computer vision tasks can be categorised into 4 major fields: (1) semantic segmentation, (2) classification & localization, (3) object detection, and (4) instance segmentation. The task of semantic segmentation refers to the process of assigning a class label to every pixel in an image [72]. One of the shortcomings of this task is the fact that semantic segmentation does not differentiate between instances of the same class. On the other hand, the classification & localization task aims to predict the class of a specific object in an image and to draw a bounding box around the region of the classified object in an image [126]. This task refers to a single object. However, most images in real-world settings contain multiple objects of different shapes and sizes. Therefore, object detection [37] refers to a more general approach where a varying number of predicted objects for every input image can be extracted, since it is unknown how many objects are expected to be detected in each image.

Object detection systems strive to find every instance of an object and estimate the spatial extent of each one. Nevertheless, the detected objects are located just with bounding boxes.

The task of instance segmentation refers to the problem of detecting all the instances of a category in an image and marking the pixels that belong to each one of them [39]. Extending this task to the video domain results in simultaneous detection, segmentation, and tracking of the instances [121]. The instance segmentation task combines object detection, where individual objects are classified and localized with a bounding box, and semantic segmentation, where each pixel is classified into the given classes.

The task of object classification & localization is included in object detection. At the same time, in semantic segmentation, each pixel of an image is associated with a class label like a road, tree, pedestrian, etc. In other words, all objects of an image that belong to the same class are treated as a single entity. On the other hand, each object of the same class is treated as a distinct individual instance with instance segmentation. Hence, instance segmentation can be considered as a more elaborate implementation of semantic segmentation. Since all the computer vision tasks are similar, in this work mainly object detection and instance segmentation techniques will be examined, as they are the most dominant techniques required in extensive applications such as autonomous driving [69], video surveillance [100], face recognition [108], and robot navigation [120].

B. DEEP LEARNING VS. TRADITIONAL COMPUTER VISION TECHNIQUES

Traditional computer vision methods are based on hard-coded, rigid-rule algorithms to apply feature extraction on images [80]. Several algorithms have been developed to extract properties such as corners, edges, and regions of interest from images [2], [12], [40], [74], [88]. These algorithms showcase advantages such as transparency, in terms of allowing to trace back to all steps of how a decision was made, and performance that is independent of the training dataset. At the same time, however, they have been criticised to be inflexible, difficult to improve or adapt, and highly time-consuming to develop manually for each additional object to be detected [83]. Moreover, the performance of these methods significantly deteriorates when the number of classes to be detected increases. By contrast, DL utilizes massive data sets and numerous training cycles to learn how an object looks, following a process during which relevant features of an object of interest are extracted automatically. The DL architecture can then be implemented on previously unseen images and make accurate predictions. DL-based methods perform remarkably better than traditional methods, albeit with trade-offs regarding computational requirements and training time [83]. As a result, they have vastly replaced traditional computer vision techniques, thanks to their strong ability to be easily adjusted, to extract complex features in much more detail, and to be much more efficient in terms of accuracy and versatility [83]. Tremendous advancements in research have taken place in this domain, resulting in the development of numerous methods. The fundamental DL methods implemented on image computer vision applications are discussed in section II-C.

C. IMAGE-BASED DEEP LEARNING METHODS

1) CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks (CNNs) have been widely used in image processing applications over the past decades [62], [66], [133]. Their structure consists of a number of convolutional and pooling layers, stacked one after another [5]. The convolutional layer can be visualized as a square matrix W of weights, called kernel [87]. The kernel slides over the image looking for patterns and when it distinguishes a part of an image that is similar with its pattern, it returns a large positive value, otherwise, it returns a small value. The input image is represented as a pixel matrix with size length $\times$ width $\times$ number of color channels (i.e. an RGB image has 3 color channels).

The convolutional layer is utilized for feature extraction and the pooling layer to downsample the resolution of the convolutional layer output. In this way, a dimension reduction is accomplished, which reduces the number of necessary parameters in the next layer, resulting in a less complex architecture. During the training process, the training samples are fed through the CNN and the error with respect to the desired output is calculated. The error and its gradient are then backpropagated through the network layers and the weights are updated.

CNN-based image object detectors can be separated into two main categories [105], [127]:

- **Two-stage approach:** In the two-stage method, the first stage extracts region proposals and the second stage classifies those region proposals and determines the bounding boxes of the classified objects. In the region proposal part, sliding window techniques such as Deformable Part Models [20] are adopted. An additional region proposal technique, employed in region-based convolutional neural networks (R-CNNs) [27], is selective search [111]. R-CNNs extract around 2000 region proposals on each input image, which is a significantly reduced number of regions needed compared to other sliding window methods. At the second stage of this architecture, a CNN is used for object detection over the region proposals. The size of the proposed regions is arbitrary, while the CNN requires a fixed size input. Hence, a major drawback of R-CNNs is due to the fact that images need to be cropped or resized to accomplish the requirement for a fixed size input. Spatial pyramid pooling [31], [42], [64] is a method used in order to achieve a fixed-size output irrespective of the input image size. Hence, spatial pyramid pooling networks can be trained and tested on varying size images, which reduces overfitting of the model.
  Both R-CNNs and spatial pyramid pooling networks are particularly slow during training. Fast R-CNN [27] tries to solve this drawback by passing the original image through the CNN instead of using the region proposals. As a result, fast R-CNN is faster than R-CNN because the convolutional operation is implemented only once on the original image instead of 2000 times on the region proposals. Fast R-CNNs can train detection networks whose architecture involves multiple layers like VGG-16 [99], as they are 9 times faster compared to R-CNNs and 3 times faster than spatial pyramid pooling networks [105]. The drawback of the high time cost has been further addressed by faster R-CNNs [92]. In faster R-CNNs the time-consuming selective-search algorithm is replaced with a fully convolutional network that learns the region proposals of an image with arbitrary size. A major additional development of the previous R-CNNs is achieved by Mask-RCNNs [41]. Mask R-CNNs extend the previous architectures by labeling the pixels corresponding to each object instance. The Mask R-CNN inherits the region proposal network from faster R-CNNs and employs an additional branch that outputs a binary mask classifying whether or not a given pixel is part of an object. Two-stage approaches yield a high accuracy since each stage performs one specific task. However, in terms of real-time applications, two-stage approaches show weaknesses in computational time.
- **One-stage approach:** One-stage approaches skip the first stage of region proposal and simply run detection directly on the input image. This simpler architecture allows them to have faster inference. Some networks can achieve a processing speed of up to 150 frames per second (fps). There is a trade-off, however, in terms of accuracy. Notable one-stage methods are the ''you only look once'' (YOLO) network [91], which extracts class and bounding boxes predictions directly from an input image using a CNN and the single-shot detector (SSD) [71], which takes an input image and passes it through multiple convolutional layers with different sizes of filters.

2) RESTRICTED BOLTZMANN MACHINES

The Restricted Boltzmann Machine (RBM) is a two-layer undirected graphical model [6] that was introduced in 1986 [46]. It consists of a set of visible nodes and a set of hidden nodes. RBMs are in essence a variant of Boltzmann machines, but in RBMs there are no intralayer connections between the nodes in the visible layer and the hidden layer (i.e. no visible node is connected to any other visible node and no hidden node is connected to any other hidden node respectively). In this way, RBMs are easier to implement and more efficient in training compared to Boltzmann Machines. Their visible nodes receive the input, combine it with weights and a bias, and pass it to the hidden nodes. The value generated at the hidden nodes is combined accordingly with weights and a bias and the result is passed to the visible nodes to reconstruct the input.

If we consider the visible vector V, the hidden vector H, and the weight parameters α_i, b_i, w_{ij}, an RBM configuration can be assigned with an energy E given by [24]:

$$E(V, H) = -\sum_i \alpha_i v_i - \sum_j b_j h_j - \sum_{ij} v_i w_{ij} h_j. \quad (1)$$

Given this energy function, a probability P is assigned to every pair (V, H):

$$P(V, H) = \frac{1}{Z} e^{-E(V,H)}, \tag{2}$$

where Z is equal to the sum of the energy of all the pairs of visible and hidden vectors.

$$Z = \sum_{(V,H)} e^{-E(V,H)}. \tag{3}$$

For a given visible vector V, the probability that is assigned to the hidden node h_j is

$$P(h_j = 1|V) = \sigma\left(b_j + \sum_i v_i w_{ij}\right), \tag{4}$$

where $\sigma(\cdot)$ is the logistic sigmoid function [38]. For a hidden vector H the assigned probability of a visible node v_i is respectively:

$$P(v_i = 1|H) = \sigma\left(\alpha_i + \sum_j h_j w_{ij}\right). \tag{5}$$

The weight parameters are optimized with the aim to maximize the likelihood of the visible and hidden vectors (V, H).

The intuition behind RBMs is based on the association of a scalar energy to each combination of the variables of interest. Learning is achieved, therefore, by calculating the combination that has the lowest energy.

RBMs are useful for dimensionality reduction, classification, regression, and feature learning. However, due to the fact that RBMs consist of only two layers, the complexity of the data representation that they can achieve is limited [24]. For this reason, a number of extended, architectures has been developed. An example of such architecture is the Deep Belief Network [44], which consists of multiple stacked RBMs. Deep Belief Networks are used for feature extraction in many computer vision applications. Except for Deep Belief Networks, another RBM-based architecture is the Deep Boltzmann Machine [95], [96]. Deep Boltzmann Machines are similar to Deep Belief Networks, although the former have only undirected connections between their layers, which makes them more robust to noisy observations, while the latter have bidirectional connections in the last layer [104].

3) AUTO-ENCODERS

Auto-encoders [8], [45] refer to a specific type of neural networks that aim to compress the input image data into a lower-dimension (latent) representation and then reconstruct the original image from this representation. Their architecture consists of two main parts, namely, the encoder and the decoder. The encoder maps an input vector of images X into a compressed, lower dimensional vector Z, while the decoder part maps the latent variable Z to a reconstruction of the input image. The encoder and decoder mappings $\phi : \mathcal{X} \to \mathcal{Z}$ and $\psi : \mathcal{Z} \to \mathcal{X}$ are given by:

$$(\phi, \psi) = \underset{(\phi,\psi)}{\operatorname{argmin}} \ \|X - (\psi \circ \phi)(X)\|^2, \tag{6}$$

where the operator $\circ$ refers to the composition function: $\psi \circ \phi(X) = \psi(\phi(X))$. The autoencoder is trained with the objective to select the optimal encoder and decoder functions so that the minimum amount of information is required to encode the image in order to be regenerated on the decoder side.

III. DEEP LEARNING METHODS FOR DETECTION AND SEGMENTATION OF OBJECTS IN VIDEOS

Due to the similarity between video detection and image detection, some methods of image detection are often used for video detection. The methods described above can be extended to the video domain by running detection for each image in a sequence of frames [7]. In this way, however, the temporal correlation between frames is not taken into account. In addition, running a detection algorithm for each frame results in computational inefficiency since there might be feature extraction redundancies between sequential frames. Furthermore, in a video sequence, there might be poor-quality frames which could lead to low inference accuracy. One obvious reason that this extension is not trivial is due to the fact that a video sequence introduces an additional dimension; the temporal one. In other words, instead of being considered as a sequence of frames, a video should be rather regarded as a sequence of related frames.

Due to the complexity of video data and the computation cost for training, research has been limited in this field. However, more and more video-related research works have surfaced lately, due to the release of ImageNet VID [93] and other massive video datasets. Depending on the architecture, DL-based techniques for video object detection can be broadly diversified into six categories, namely (1) optical flow, (2) tracking, (3) long short-term memory, (4) gated recurrent unit, (5) self-attention mechanism, and (6) generative learning. In the following subsections a critical appraisal of these architecture paradigms is presented.

A. OPTICAL FLOW

One of the most fundamental concepts in video processing is optical flow. Optical flow was originally introduced in [25] referring to human perception and the changing pattern of light that reaches our eyes. In computer vision applications, optical flow refers to the problem of estimating the displacement vector for each pixel in subsequent image frames [48].

A key assumption in optical flow is brightness constancy. This practically means that a pixel at the position (x, y) of an image at time t moves to the position $(x+\Delta x, y+\Delta y)$ at time $t + \Delta t$ and the brightness $I(x, y, t)$ remains constant:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) \approx I(x, y, t). \tag{7}$$

The Taylor series expansion of the left-hand side of (7) is

$$\begin{aligned} & I(x+\Delta x, y+\Delta y, t+\Delta t) \\ & \quad = I(x,y,t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \ldots \\ & \quad \Rightarrow I(x+\Delta x, y+\Delta y, t+\Delta t) - I(x,y,t) \\ & \quad = I_x \Delta x + I_y \Delta y + I_t \Delta t, \end{aligned} \tag{8}$$

where I_x, I_y, I_t are the partial derivatives of the intensity function I with respect to x, y, and t respectively. Hence, if we substitute (7) into (8) we can derive:

$$\nabla I \cdot v^T + I_t = 0, \tag{9}$$

where $\nabla I = (I_x, I_y)$ and $v = (\frac{\Delta x}{\Delta t}, \frac{\Delta y}{\Delta t})$ are the components of the optical flow, and I_t is the temporal gradient of the intensity function.

Optical flow can be applied to estimate the motion of detected objects in video segments by assigning an optical flow vector to the pixels corresponding to the detected object.

Optical flow can be either ''sparse'' or ''dense''. Sparse optical flow estimates the flow vectors of some specific features, such as corners or edges of an object within an image frame. Dense optical flow, on the other hand, includes the flow vectors of all the pixels in an image frame. The latter method achieves higher accuracy than the former, although at the cost of increased computational requirements.

Recently, modern CNN architectures have been successfully used for optical flow estimation applications [18]. CNNs can be trained to run on pairs of images and to predict the optical flow field. These flow networks are employed in computer vision tasks for videos according to two different approaches. In the first approach, one neural network is responsible for the task of object detection and it is applied on sparse key frames. The extracted feature maps from these key frames are then propagated to the next frames with a flow network. This technique is called Deep Feature Flow (DFF) [132] and it achieves great computational efficiency due to the fact that it implements the object detection task only on key frames.

The second approach involving flow networks is known as flow-guided feature aggregation (FGFA) [131]. In FGFA, a feature extraction network is run on all individual frames to create the respective feature maps per frame. The inference at a reference frame is enhanced with an optical flow network that predicts the motion between the neighbor frames and the adjacent frames. The propagated feature maps from neighbor frames are aggregated with the feature map from the reference frame in an adaptive weighting method. FGFA achieves higher inference accuracy but at a higher computation time compared to DFF. For this reason, an impression network [43] is another proposed architecture that combines the two abovementioned techniques, with the objective to take advantage of both methods. Sparse key frame feature maps are then aggregated with other key frames feature maps and at the same time they are propagated to other non-key frames. The impression network overcomes DFF both in terms of accuracy and inference speed. It is also faster than FGFA although it achieves a slightly lower accuracy level. An alternative architecture, which outperforms FGFA, is proposed in [17], where a two-stream feature aggregation approach is integrated into a one-stage detector to achieve video object detection. In particular, the first stream applies optical flow to estimate the motion and to aggregate the features along the motion path, while the second stream predicts the features of the frame of interest by spatio-temporal sampling and aggregation of features from the adjacent frames. The final predictions result from blending the outcomes from the two streams.

B. TRACKING

Visual tracking can be described as the problem of estimating an unknown target trajectory over a sequence of image frames [78]. Traditional methods employ a variety of tracking algorithms, such as mean shift algorithm [14], particle filtering [30], and Kalman filtering [54]. With the advancements in data science in recent years, novel DL-based visual trackers have been developed.

Object tracking outperforms optical flow in accuracy [129]. This can be explained by the fact that tracking uses shared networks to achieve feature extraction for detection and tracking. Hence, the requirements in terms of computational power are limited and at the same time, the fusion between the two tasks is performed in a more straightforward way, which achieves higher accuracy compared to optical flow based models.

CNN is the first architecture that was adopted for DL-based visual tracking. In [19], a region-based fully convolutional neural network [15] is used for jointly performing detection and tracking in an integrated framework. The model is fed with a set of two consecutive image frames, from which the convolutional feature maps are computed. Object detection is run on each frame and a regressor is employed to compute the box transformation from one frame to the other. CNN-based object tracking models showcase some weaknesses in performance though, due to the scarcity of labeled data in terms of including sets of two consecutive frames, which are necessary for their training, as well as their speed limitations with respect to real-time applications [79].

A baseline approach presented in [121] extends the Mask R-CNN to include an additional tracking branch with an external memory for tracking object instances across frames. The proposed architecture extracts the classification, the bounding boxes, and the segmentation predictions of Mask R-CNN, and it takes into account the past frame information only for tracking. In this way, the task of instance segmentation is extended to videos. CrossVIS [122] presents a novel, cross-frame learning approach that uses the features of an instance in the current frame to segment the same instance in other frames. Crossover learning is integrated with the instance segmentation loss as an objective to obtain cross-frame instance segmentation consistency, achieving a

low computational cost. CrossVIS outperforms MaskTrack R-CNN [121] in terms of both accuracy and speed [122].

An additional DL-based method for tracking arbitrary objects involves Siamese Neural Networks (SNNs) [109]. SNNs have been extensively implemented on visual tracking applications in the past years [4]. An SNN is basically a two-stream network that takes as input pairs of the target and search image and outputs a similarity map. In other words, SNNs learn a function $f : (z, x) \rightarrow f(z, x)$ which compares an image z with a candidate image x returning a high score when the two images are similar with each other. The position tracking of an object can thus be determined by checking all possible locations and selecting the one that corresponds to an image with the highest similarity to the previous frame. SNNs can learn the function f from a training video dataset with labeled object trajectories and they are one of the most promising methods for object tracking due to their performance and efficiency.

Recurrent neural networks (RNNs) [28] are an alternative architecture employed in visual object tracking applications. RNNs can be considered to operate on a sequence that contains vectors $x(t)$ and each vector can describe e.g. an image frame from a video at time step t. In other words, an RNN is a neural network that is specialized for processing a sequence of values $x(1), \ldots, x(n)$, where n is the length of the sequence, in a similar way as a convolutional network is specialized for processing a tensor representing an image. The same update rule is applied to each part of the output, resulting in the sharing of parameters through a deep computational graph. RNN-based methods can be considered as a suitable method for visual object tracking since they take into account both spatial and temporal features of video frames [124]. The RNN-based methods aim to improve the tracking performance by utilizing temporal information such as past states of the target's position. However, their implementation is limited because their complex architecture involves a significant number of parameters that need to be determined [68].

C. LONG SHORT-TERM MEMORY

Although RNNs are naturally suited to time series data, like videos, their implementation suffers from various weaknesses. First of all, while they take into consideration information from the previous time stamp, their performance is deteriorated, when storing information for a longer time period [60]. Sometimes, certain information stored at long past time step might be required to accurately predict the current output. RNNs in that cases are incapable of utilizing such "long-term" dependencies. In addition, RNNs do not have the possibility to keep part of the past time stamp information and to discard the rest. An additional challenge in RNNs is that gradients propagated through the network tend to either vanish or explode because of the repetition of the weight matrix over all recurrent units. At the same time, optical flow techniques make use of temporal information only on two adjacent frames without using temporal information from other previous frames. Long short-term memory (LSTM) [47] is an improved type of RNN that is capable of utilizing long-term dependencies.

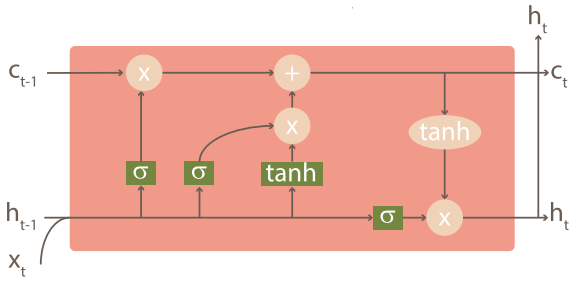

FIGURE 1. LSTM cell structure, adapted from [125].

The architecture of an LSTM cell is depicted in Figure 1. LSTMs are cells consisting of three parts which are known as gates. The first gate determines what part of the information coming from past time steps needs to be "remembered" or can be "forgotten". The second gate inputs information of the current time step to the cell. Finally, the third gate passes the updated information from the current time step to the next one. The first gate is called forget gate while the second and the third ones are called input and output gates respectively.

In the following equations $f(t)$, $i(t)$, $o(t)$ represent the forget, input and output gate vectors respectively, σ is the sigmoid function, $W^{(j)}$ and $b^{(j)}$ refer to the weights and biases corresponding to the j-th gate's neurons, $h(t-1)$ refers to the output of the previous cell at time stamp $t-1$, and $x(t)$ represents the input at time t [49].

- Forget gate

$$f(t) = \sigma\left(W^{(f)}[h(t-1), x(t)] + b^{(f)}\right) \tag{10}$$

- Input gate

$$i(t) = \sigma\left(W^{(i)}[h(t-1), x(t)] + b^{(i)}\right) \tag{11}$$

- Output gate

$$o(t) = \sigma\left(W^{(o)}[h(t-1), x(t)] + b^{(o)}\right) \tag{12}$$

Moreover, an additional vector $\bar{C}$ is used that modifies the cell's state C:

$$\bar{C}(t) = \tanh\left(W^{(c)}[h(t-1), x(t)] + b^{(c)}\right) \tag{13}$$

$$C(t) = f(t) \odot C(t-1) + i(t) \odot \bar{C}(t), \tag{14}$$

where the operator $\odot$ corresponds to the elementwise multiplication. The hidden state is equal to:

$$h(t) = o(t) \odot \tanh C(t). \tag{15}$$

LSTMs can maintain important information over a long sequence of data. [33] presents an extensive analysis of variants of LSTM as well as a review of the impact of the involved hyperparameters. In [75] an LSTM framework is developed as an extension to an SSD architecture in order to associate detected object instances across consecutive frames.

The proposed method outperforms other RNN architectures [110] and it can be applied online. However, the weakness of this approach is that the SSD architecture involved is pre-trained in advance and thus, the SSD features do not get updated in response to the output of the LSTMs. In [70], an approach is suggested where LSTM is used in combination with interleaving conventional feature extractors with extremely lightweight ones. The main advantage of this approach is that minimal computation is required to produce accurate detection. In other words, an interleaved model framework is proposed, where multiple feature extractors are run sequentially or concurrently. A memory mechanism is then proposed to aggregate these frame-level features. A modified LSTM cell is used in [130] to achieve faster results with low computational requirements. The proposed architecture connects fast single-image object detection frameworks in series with convolutional LSTM layers in order to propagate frame-level information over time. This architecture inputs one single frame of the video at a time and it is quite simple. Hence, it achieves reduced computational cost as well as enhanced inference speed.

D. GATED RECURRENT UNIT

Similarly to LSTMs, gated recurrent units (GRUs) [13] are another type of RNNs. However, GRUs have fewer parameters than LSTMs, since they only have two gates: the update gate and the reset gate. As seen in Figure 2, in contrast to LSTMs, a GRU cell does not have an output gate, as in the case of LSTMs, and they combine the input and the forget gate of LSTMs into the update gate. Due to their simplicity, GRUs are significantly faster rather than LSTMs.

The update and reset gates in a GRU cell are defined as in equations (16) and (17) respectively. In the following equations $z(t)$, $r(t)$ represent the update and reset gate vectors respectively, and $W^{(j)}$, $b^{(j)}$ refer to the weights and biases corresponding to the j-th gate's neurons [49].

- Update gate

$$z(t) = \sigma\left(W^{(z)}[h(t-1), x(t)] + b^{(z)}\right) \tag{16}$$

- Reset gate

$$r(t) = \sigma\left(W^{(r)}[h(t-1), x(t)] + b^{(r)}\right) \tag{17}$$

The update gate determines the amount of previous time-step information that passes along the next state, while the reset gate is responsible for deciding what part of the past information is neglected. After multiplying the input vector and the hidden state with the weights of the reset gate as presented in (17), the element-wise product between the reset gate and the previous time-step hidden state is calculated. Then, a non-linear activation function is applied to the result leading to the candidate hidden state:

$$\bar{h}(t) = \tanh\left(W^{h}[r(t) \odot h(t-1), x(t)] + b^{(h)}\right). \tag{18}$$

The hidden state then reads as:

$$h(t) = (1 - z(t)) \odot h(t-1) + z(t) \odot \bar{h}(t). \tag{19}$$

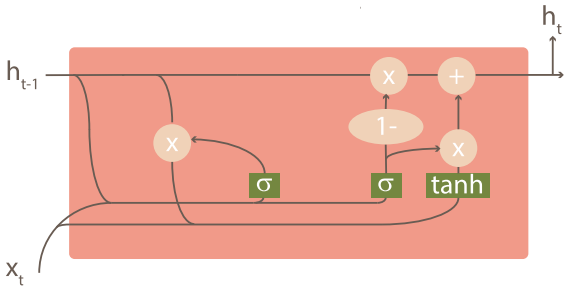

FIGURE 2. GRU structure, adapted from [125].

In [9] an SSD-based architecture is extended to multi-frame data. Convolutional GRUs are employed in order to fuse features across multiple frames and to enhance the accuracy of object detection. From a mathematical perspective, this architecture replaces the dot product operator in the standard gated recurrent unit definition in (16)-(18) with the convolution operator. As reported in [23], this approach improves the existing SSD architecture by 2.7 % in terms of the mean average precision on the KITTI dataset [22]. An additional example is provided in [110], where first a pseudo-labeler is trained on individual labeled frames. The pseudo-labeler assigns the labels to all video frames and then a recurrent architecture with GRUs is trained, which takes sequences of pseudo-labeled frames as input. The standard cost function used for the training of the RNN is augmented with an additional term to ensure the consistency across consecutive frames. In [112] a human activity recognition technique is proposed, where skip connections are introduced among GRU layers to ensure that even in a deep architecture with multiple layers, there is no vanishing gradient impact on the performance.

Both LSTM and GRU can ensure that important information is maintained along long time-series data. GRU is faster than LSTM in terms of training speed [123]. Their performance is comparable, although in small datasets, GRU slightly outperforms LSTM.

E. SELF-ATTENTION MECHANISM

RNNs, LSTMs, and GRUs have been widely adopted in sequence modeling applications. However, due to the fact that they process the data in a sequential manner, they do not allow for parallel computation, which could critically affect long sequences of frames, due to memory constraints limiting the batch size of samples during training.

Self-attention mechanism [58] relates different elements of a sequence to generate a representation of this sequence. Contrary to the architectures mentioned above, it supports parallel processing of sequential data. Originally it was proposed for machine translation [113] and then its application was extended to video data [26].

Three vectors are involved in the self-attention mechanism. These vectors are used for the representation of features (key vector), values (value vector), and the values to be

determined (query vector). Let us assume that we have a sequence of n elements $(x_1, x_2, \ldots, x_n)$ of $X \in \mathbb{R}^{n\times d}$, with d being the embedding dimension for the representation of each element [57]. We can then define three learnable weight matrices in order to transform the queries ($W^q \in \mathbb{R}^{n\times d_q}$), keys ($W^k \in \mathbb{R}^{n\times d_k}$) and values vectors ($W^v \in \mathbb{R}^{n\times d_v}$). In this way, the input X is first transformed with the weight matrices and projected onto $Q = XW^q$, $K = XW^k$, and $V = XW^v$. A similarity function is used to calculate the similarity between the query and the key vector. The self-attention layer outputs $\mathbf{Z} \in \mathbb{R}^{n\times d_v}$ which is equal to

$$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_q}}\right)V, \tag{20}$$

where softmax function is defined by

$$\text{softmax}(\mathbf{X})_i = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}, \tag{21}$$

for $i = 1 \ldots k$ and $\mathbf{X} = (x_1, \ldots, x_k) \in \mathbb{R}^k$. The self-attention determines the similarity between the key and the query vector by computing their dot product. The dot product is then normalized using softmax so that the sum of all the scores becomes equal to 1. Each element is then given by the weighted sum of all elements in the sequence. The weights in this case correspond to the attention scores. The most well-known, self-attention architecture is the transformer [113].

In [26] a transformer framework is developed to recognize and localize human actions in a video. A person feature is represented as the query (Q) and the features from adjacent video frames correspond to the key (K) and the values (V). A video instance segmentation architecture built upon transformers is proposed in [116]. Four modules are included in the developed architecture: a backbone CNN to extract features over the video frames, an encoder-decoder transformer that determines the similarity of features on pixel and instance level, an instance-sequence matching, and a segmentation module. The overall performance of this framework is competitive compared to the single-model approaches tested on the YouTube-VIS dataset [121], although it is somewhat lower in comparison to other complex CNN-based models [3].

In [35] a constrained self-attention architecture is proposed for video object detection that captures motion cues under the assumption that moving objects follow a continuous trajectory. An additional, self-attention based architecture is proposed in [36], which is applied in the temporal-spatial domain towards aligning two feature maps of consecutive frames. The proposed method features a low amount of parameters, while it achieves higher accuracy in comparison to optical flow-based methods such as DFF and FGFA. A related, efficient, and simplified architecture for video object detection via aggregating semantic features across frames is presented in [118]. Cosine similarity is implemented to compute the semantic similarities of the extracted proposals across frames, which are then aggregated accordingly. In [16] an object relation module is employed as part of a multi-stage architecture, in order to extract object relations in both spatial and temporal context. The relations are then further distilled with refined supportive object proposals and propagated across frames. Finally, in [98] an attention-based module is developed to learn long-range temporal relations between objects, in order to propagate the extracted features. The proposed architectures in [16], [118], and [98] outperform optical flow-based approaches in accuracy.

F. GENERATIVE LEARNING

The objective of generative learning is to approximate a complex, high-dimensional probabilistic distribution that generates a class of data, in order to generate similar data. Developing generative architectures to understand complicated data distributions has been a long-standing research problem [84]. Recent works in this area [29], [59] have provided a new set of generative algorithms that can efficiently generate video segments or extract features from them. The most outstanding generative algorithms are the variational autoencoders (VAEs) and generative adversarial networks (GANs).

- **Variational auto-encoders:** Their architecture resembles an auto-encoder, with the difference that their latent variable distribution is regularised during the training. VAEs stemmed from the limitation of auto-encoders to generate new, unseen data, due to the fact that the distribution of the latent variable is unknown. To alleviate this issue, VAEs are trained to learn the distribution of the latent variable, assuming that it follows a Gaussian distribution with a mean μ and variance σ^2 [50]. One example of a VAE-based architecture for video object detection is presented in [67], where a modified VAE architecture, built on top of a Mask R-CNN is proposed, in order to detect and to segment multiple instances in diverse videos. The proposed architecture outperforms MaskTrack R-CNN [121], because the MaskTrack R-CNN architecture depends entirely on the Mask R-CNN to perform predictions, resulting in difficulties to handle false negative proposals of the Mask R-CNN in highly diverse videos with occlusions, deformations, and pose variations of objects. By contrast, the architecture proposed in [67] merges a VAE with a Mask R-CNN network in a topology consisting of one encoder and three decoders. This results in three parallel branches that provide strong complements for predictions about bounding boxes and mask features, and they significantly reduce the number of false negatives in the Mask R-CNN module.
- **Generative adversarial networks:** Generative adversarial networks are built on the basis of a two-player, min-max game. The generator network G and the discriminator network D correspond to the first and the second player respectively. The generator's objective is to mislead the discriminator by generating natural-looking data (e.g. images, videos, etc.) from a random, latent vector z. The discriminator on the other

hand, tries to distinguish whether the data are real or fake (generated). The game is modeled as the following optimization problem:

$$\min_G \max_D (G, D) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z))]. \quad (22)$$

A generative adversarial approach is developed in [102], to randomly generate masks that correspond to object appearance variations in time. The masks are then applied to reduce overfitting via adaptively dropping out input features. The developed architecture identifies the mask that maintains the most robust features of the target objects over a long period of time. In [106] a GAN is trained on color and depth information in order to generate similar backgrounds to the test samples. The generated background samples are then subtracted from the given test samples to detect foreground moving objects. Finally, in [11] the encoder-decoder architecture of [82], which is limited to process information between only two adjacent frames, is extended with a GAN, to enforce temporal and spatial coherence of the generated object masks and to exploit information within a longer temporal window. The developed architecture exhibits similar accuracy as other state-of-the-art computer vision methods, while it is almost four times faster.

IV. CHALLENGES IN DEEP-LEARNING-BASED COMPUTER VISION

Despite the tremendous advances in deep learning and the fast pace of its breakthroughs over the last years, there are still challenges that prevent it from reaching its full potential. This section illustrates a set of major challenges related to computer vision tasks on video analysis with DL techniques.

DL-based methods have succeeded in achieving even human-level performance in complex, computer vision tasks. However, this is possible only when massive datasets are available for training. Data are the core of any DL-based process and hence their shortage is often responsible for poor performance. Large-scale amounts of data are not available for all video applications though.

The impact of data scarcity is further escalated by the stand-alone approach of DL. A typical workflow for developing a DL module consists of creating a training set of inputs associated with outputs and learning the relations between them. In this way, however, the architecture becomes free-standing and isolated from prior, useful knowledge. Hence, the DL performance is highly determined by the existence of big-volume datasets while at the same time, applications that are more related to common sense reasoning and less to categorization, cannot be sufficiently targeted with purely DL methods [76].

Generalizability is an additional major challenge concerning the performance of a data-driven model trained on one dataset when applied to other datasets. When training deep neural networks with high complexity and numerous parameters, the cost function might have multiple minima, which minimize the training error but may not generalize well to unseen data. The presence of noise and outliers in the training dataset is an additional reason for poor generalizability. Generalizability is also deteriorated due to the weakness of DL methods to deal with hierarchical structures, since DL modules tend to fail when generalization depends on compositional processes [63].

At the same time, although correlation does not imply causation, they do not seem to be distinguishable for DL. Numerous neural network architectures have surfaced over the last decades that are highly capable of discovering complex correlations in data, yet they lack in reasoning about cause-effect relations or environment changes.

Finally, deep learning has delivered new, highly performing approaches in computer vision tasks, whose dominance, however, remains inversely proportional to their explanatory power. Rationalizing the output of data-driven techniques is a critical issue since more and more data-driven systems are adopted in safety-critical and high impact applications.

V. INTEGRATING DEEP LEARNING WITH DOMAIN KNOWLEDGE

A. MOTIVATION

A prudent approach to address the abovementioned challenges is to expand the current methods and to merge them with principles that govern the dynamic behavior of systems over the time, enabling an adaptation to new, unseen scenarios. Combining DL-based techniques with equation-based dynamic models (DMs) in a complementary way, or in other words, integrating common sense understanding into artificial intelligence constitutes a particularly interesting challenge for computer vision systems.

Enabling data-driven vision systems to understand the principles that govern the behavior of objects is essential for the development of autonomous systems that understand observed scenarios and have the ability to adopt these principles to a never seen situation. Leveraging domain knowledge to identify equation-based models that describe how the properties of objects and entities change over time and embedding them into DL techniques can lead to novel, highly robust, and performing architectures. Such models could be developed for instance from well-known first principles in order to describe how an object moves and they could be coupled with DL methods forming a hybrid computer vision architecture. It is straightforward to conclude that hybrid architectures are more efficient compared to purely data-driven or model-based techniques as they harness the benefits of both disciplines. Hybrid methods that combine scientific domain knowledge with data-driven models allow for accurate inference even with imperfect models and limited amounts of data.

The integration of the two disciplines in a hybrid architecture can be realized either by infusing mathematical rules to

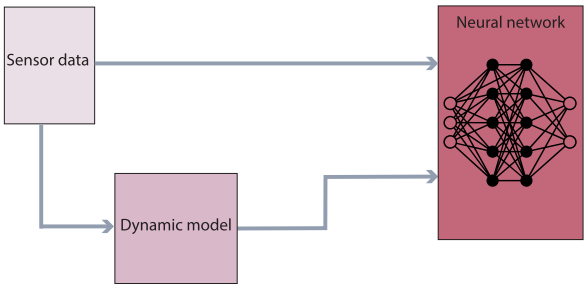

FIGURE 3. Hybrid architecture. The dynamic model could refer to a first-principle model or any other mathematical or computer model that is derived from domain knowledge and that describes how the properties of objects and entities change over time.

a DL architecture or by combining the operation of the two separate modules in a complementary manner. An advantage of this second version of a hybrid architecture is the fact that an easy and straightforward recalibration of the DM module is feasible if a bidirectional interaction between the two modules is enabled. More specifically, the DL module, which can be re-trained incrementally when new data become available, can also enable the recalibration of the DM module. This results in a hybrid architecture which is highly flexible and easily adaptable to different scenarios.

Hybrid architectures merging data-driven techniques with domain knowledge, such as from physics have been recently developed, introducing a novel research field which is still in its infancy [55], [90]. As a result, their applications are limited mainly to topics related to climate science and geology. Their expansion to other disciplines like computer vision tasks remains a challenging research topic but would undoubtedly contribute towards addressing the abovementioned impediments in purely data-driven methods.

B. HYBRID ARCHITECTURES

A taxonomy of four general classes for integrated data-driven and model-based techniques can be derived. This classification is based on the level at which the integration takes place [55], [90]. More specifically, the four classes are: (1) preprocessing level, (2) initialization, (3) design of architecture, and (4) regularization. This section presents an analysis of these different methodologies.

1) DATA PREPOSSESSING LEVEL

Data preprocessing is essential in all data-driven techniques before passing the data through the DL module. The reason is straightforward: the quality of data determines the information that can be extracted and hence, it directly influences the learning process of the DL algorithm. As a result, it is vital that we apply a preprocessing technique before passing the data through the DL model.

The concept of data preprocessing is a major area in the field of deep learning. There are three main steps involved in data preprocessing, that is: (1) data cleaning, (2) data transformation, and (3) data reduction. Data cleaning refers to the handling of missing data as well as, to noise removal. Data transformation may include normalization of the data, band-pass filtering, downsampling, and feature selection. When the input involves time-series signals, the data can be converted to the frequency domain via the fast Fourier transform (FFT). This implementation can be implemented in anomaly detection such as e.g. in the bearings of a rotating machine [94]. Finally, reducing the dimension of the feature set is another technique widely applied when preprocessing the data. A thorough analysis of the data preprocessing techniques is presented in [21].

2) INITIALIZATION

One important design choice when building a neural network architecture, is related to the parameter initialization [117]. Iterative optimization algorithms such as gradient descent are used during the process of training a neural network in order to estimate the network's parameters. In this process, an initial value for the parameters is required as a first step to start the optimization process. Quite often the initialization of the parameters is done based on a random distribution. Random initialization though can make the optimization algorithm that is employed for the calculation of the network weights to converge to local minima or saddle points.

An approach towards this issue would be to use a technique called transfer learning [85]. The basic idea of transfer learning is based on pretraining a neural network on a simpler, related problem. This pretraining task takes place under the assumption that a big quantity of data is available. This pretrained neural network can then be implemented as the initial state for the training of the original problem as it is closer to the optimal parameters value than random initialization. Transfer learning is a widely used technique in complex DL applications such as natural language processing and computer vision. However, the performance of this technique is highly dependent on the availability of big-scale data. An alternative approach is to employ domain-specific knowledge to assist the selection of the initial values of the parameters involved [55]. In this way, first-principle models can be used to generate approximate simulations for the initialization of the parameters of the neural network. Domain knowledge can ensure a reliable initialization of the parameters, which can assist in achieving generalizable, interpretable, and physically consistent architectures.

3) DESIGN OF ARCHITECTURE

Data-driven techniques have made a major impact at realizing highly performing systems for solving hard problems related to pattern recognition, prediction, etc. However, a major impediment in their wide adoption in critical applications is their "black box" nature since our understanding of their complexity is limited. Hence, domain knowledge can be infused in a DL architecture to ensure its interpretability.

One possible approach for this integration is to infuse the output of the equation-based model f_{DM} as input to the DL module f_{DL}, i.e. $f_{\mathrm{hybrid}} : (X, P_{\mathrm{DM}}, P_{\mathrm{DL}}) \rightarrow Y$ where X is the input, Y is the output, P_{DM}, P_{DL} the parameters of the

dynamic model and the DL model respectively, and f_{hybrid} the composition of the two functions, $f_{\text{hybrid}} = f_{\text{DL}} \circ f_{\text{DM}}$ [90].

Two main categories of architectures can result from merging DL with dynamic models, founded on prior domain knowledge. In the first category, the output of the model is fed through the DL module at the first or at an additional layer. In the second category, the model is embedded into the DL module. Many architectures with respect to the first class have surfaced lately in the field of climate and geology applications. In [52], [56], the output of a physics-based model is provided as an additional input feature to the DL module in an application related to predicting the temperature of a lake based on the depth. In [86], a physics-based neural network architecture is used in order to simulate broadband earthquake ground motions. The DL module is used to predict the ground motion in the short term, including transient effects, which are particularly complex to model mathematically. The DM module is then used to simulate the response in a long-term period.

In the second class, the DM module is embedded into the DL module architecture. An example of this class is a physics-based model with an RNN including LSTMs [101] where the sensor data as well as the DM generated output are ingested as input to the RNN architecture.

4) REGULARIZATION

Deep neural networks can involve numerous parameters. However, when no large amounts of data are available, deep neural networks tend to overfit or, in other words, they fail to discover the underlying relationship described by the training data and hence they cannot extrapolate to observed data outside the training set. One way to handle this issue is to apply physical constraints on the loss function of the neural network. Several regularization techniques have been developed in this way, to prevent neural networks from overfitting. This is achieved by applying penalties to layer parameters, and by integrating these penalties in the loss function that is minimized during training. The loss function in that case will be of the following form [117]:

$$f_{\text{Loss}} = f_{\text{Trn}}(Y, \hat{Y}) + \lambda R(W) + \gamma f_{\text{Phy}}(\hat{Y}), \tag{23}$$

where f_{Trn} corresponds to a function that represents the error between the predicted value $\hat{Y}$ and the true value Y. This function can be for example the mean squared error or cross entropy. In addition, λ represents a hyperparameter determining the weight of the regularization term $R(W)$. The first two terms of (23) describe the standard loss function used when training a neural network. The additional term f_{Phy} corresponds to the physics-based constraint and it aims to ensure the consistency of the trained system with first-principle laws or dynamic models. The weight of this function is represented by the hyperparameter γ. Given the true value Y, the following is considered as the general optimization problem to solve for (23):

$$\underset{w}{\text{argmin}} \; f_{\text{Trn}}(Y, \hat{Y}) + \lambda R(W) + \gamma f_{\text{Phy}}(\hat{Y}). \tag{24}$$

By introducing model-based constraints in the loss function for the train of DL modules, scientific consistency is achieved, which is essential for training generalizable models. In addition, the physics-based loss function f_{Phy} requires no labeled data which allows the training of the DL module to be expanded to non-labeled data. A plethora of implementations that impose physics-based constraints on the training of DL modes has surfaced recently [81], [103], [107]. In [56] a physics-based loss function is used for the training of a temperature lake predictor. The loss function encompasses a constraint resulting from the relationship between the temperature, the density, and the depth of the lake water. In this way, the trained predictor achieves enhanced generalizability while at the same time consistency with first-principle laws is ensured for the results. In [51], the application of lake temperature prediction is extended to include temporal physical processes. More specifically, a physics-based RNN is developed that involves energy conservation constraints. Standard LSTM models store specific information at each time step, which feeds to the next time step. However, when the models are trained on data from specific seasons or from multiple years, it is difficult to generalize to data from different time periods since the time profiles vary significantly between each other. By including the energy flux changes, however, which determine the temperature changes, the architecture can successfully predict the lake temperature, even on unseen data. Another example is given in [53], where the data-driven model is penalized with the equation describing the time evolution of waves in order to identify the location of underwater obstacles from acoustic measurements. In this way, the accuracy of the model outside the training dataset is enhanced. Finally, [10] presents a case where multiple physics-based terms are present in a loss function. These might be competing loss terms with multiple local minima and correspond to different physics equations that need to be minimized together. Hence, an approach is presented where the contribution of each term is adaptively tuned during the training phase in order to improve the generalizability of the developed architecture.

C. HYBRID ARCHITECTURE IMPLEMENTATION IN COMPUTER VISION

Integrating useful domain knowledge into DL-based computer vision tasks is essential to build robust, generalizable systems and to compensate for the lack of large-volume training data. An example of such a hybrid architecture is proposed in [103], where the height of a free-falling object is estimated on each frame of a video by training a CNN to detect and track objects obeying to free-falling laws of physics. The training of this CNN is based on a loss function in which first-principle laws are encoded. In [1], physics are blended with DL in the framework of a two-stage encoder with the aim to recover the shape of an object based on polarized photos. In [61] an LSTM architecture is combined with a dynamics model in order to acquire a

proposal distribution over an object's state. Finally, in [119], a generative vision system is proposed for estimating physical features of objects by integrating the output of a multi-physics simulation engine in the loop.

Integrating DL techniques with domain knowledge is a recently introduced research topic [55], [90]. As a result, using domain knowledge to derive first-principle models or on a broader perspective, any dynamic mathematical or computer model [73] that describes how the properties of objects and entities change over time (Figure 3), and merging them with existing DL architectures constitute an especially promising research task to address the challenges of DL in computer vision.

VI. OUTLOOK: FUTURE DIRECTIONS IN DEEP LEARNING FOR OBJECT DETECTION AND SEGMENTATION IN VIDEOS

Deep learning has brought a catalytic effect in the field of computer vision for video analysis. Although nobody knows with certainty how DL will evolve over the coming decades, it is expected that much of the future research will revolve around the following critical areas [32], [77], [114]:

- **Out-of-distribution generalization:** Future computer vision systems should be able to make accurate predictions not only in a known context but also for data with different distributions than the ones learned from the training samples. The main reason behind the difficulty of DL systems to accurately generalize and predict on unseen data is caused by the fundamental assumption that training and test data are independent and identically distributed (IID) [97], [128]. In many real-life cases however, the IID assumption is hardly satisfied. The ability to generalize under distribution shifts is of critical significance, and hence, the investigation of out-of-distribution generalization is expected to attract enormous research interest in the academic field.
- **Deep learning systems with causal structures:** Causality is expected to be a central strand of DL research in the coming years [89]. Developing DL systems that can represent causal relationships can increase their safety and reliability, and introducing a causal understanding of basic concepts in DL methods could certainly be the key to achieve robustness in complex real-world environments.
- **Effective representation learning with few or no labeled data:** While techniques for representation learning when massive labeled datasets are available have become remarkably powerful, various challenges remain in the case of limited labeled data. Developing approaches for addressing the issue of labeled data scarcity is an emerging popular direction of research.
- **Adaptation in time-varying environments:** Adapting to time-varying environments and other dynamic-behavior-related problems has been under examination for many years and it is expected to gain massive attention by the DL research community over the coming years. Allowing integration of new knowledge online and at the same time being capable of preserving the knowledge learned during previous interactions are only a few of the desirable features of future vision mechanisms.
- **Multi-modal learning:** Ultimately, major emphasis in research is expected to be placed upon developing methods that can process and link information combining modalities from various architectures [65], [76], since unimodal DL methods seem to fail to fulfill all the desirable future DL capabilities. In particular, combined architectures that integrate DL modules with domain knowledge could provide a suitable answer to most research questions arising from the DL directions listed above.

VII. CONCLUSION

In this paper a study is presented about detection and segmentation of objects applied to video segments. A review of the currently existing techniques has been presented as well as the major challenges that data-driven techniques face. Then an extension of the data-driven techniques to a hybrid architecture that fuses data-driven techniques with equation-based models describing the dynamic behavior of objects and entities over time has been proposed in order to address issues like data scarcity, generalizability, and interpretability of the purely data-driven architectures. Finally, a survey of the current developments in hybrid architectures has been presented. We hope that this work will assist in better understanding the current status of DL in computer vision for video analysis as well as in presenting interesting directions as guidelines for future work.

REFERENCES

[1] Y. Ba, A. Ross Gilbert, F. Wang, J. Yang, R. Chen, Y. Wang, L. Yan, B. Shi, and A. Kadambi, "Deep shape from polarization," 2019, *arXiv:1903.10210*.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Computer Vision–(ECCV)*. Berlin, Germany: Springer, 2006, pp. 404–417.

[3] G. Bertasius and L. Torresani, "Classifying, segmenting, and tracking object instances in video with mask propagation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9739–9748.

[4] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional Siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV Workshops)*, 2016, pp. 850–865.

[5] D. Bhatt, C. Patel, H. Talsania, J. Patel, R. Vaghela, S. Pandya, K. Modi, and H. Ghayvat, "CNN variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, Oct. 2021.

[6] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. New York, NY, USA: Springer-Verlag, 2006.

[7] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "YOLACT++: Better real-time instance segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 2, pp. 1108–1121, Feb. 2022.

[8] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biol. Cybern.*, vol. 59, nos. 4–5, pp. 291–294, Sep. 1988.

[9] A. Broad, M. Jones, and T. Y. Lee, "Recurrent multi-frame single shot detector for video object detection," in *Proc. BMVC*, 2018, pp. 1–14.

[10] M. Elhamod, J. Bu, C. Singh, M. Redell, A. Ghosh, V. Podolskiy, W.-C. Lee, and A. Karpatne, "CoPhy-PGNN: Learning physics-guided neural networks with competing loss functions for solving eigenvalue problems," 2020, *arXiv:2007.01420*.

[11] S. Caelles, A. Pumarola, F. Moreno-Noguer, A. Sanfeliu, and L. Van Gool, "Fast video object segmentation with spatio-temporal GANs," 2019, *arXiv:1903.12161*.
[12] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *Computer Vision–(ECCV)*. Berlin, Germany: Springer, 2010, pp. 778–792.
[13] K. Cho, B. van Merrienboer, C. C. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014.
[14] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2000, pp. 142–149.
[15] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," 2016, *arXiv:1605.06409*.
[16] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, and T. Mei, "Relation distillation networks for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Seoul, South Korea, Oct. 2019, pp. 7022–7031.
[17] J. Deng, Y. Pan, T. Yao, W. Zhou, H. Li, and T. Mei, "Single shot video object detector," *IEEE Trans. Multimedia*, vol. 23, pp. 846–858, 2021.
[18] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. V. D. Smagt, D. Cremers, and T. Brox, "FlowNet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2758–2766.
[19] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3057–3065.
[20] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[21] S. García, S. Ramírez-Gallego, J. Luengo, J. M. Benítez, and F. Herrera, "Big data preprocessing: Methods and prospects," *Big Data Anal.*, vol. 1, no. 1, pp. 1–22, Dec. 2016.
[22] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.
[23] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.
[24] T. Georgiou, Y. Liu, W. Chen, and M. Lew, "A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision," *Int. J. Multimedia Inf. Retr.*, vol. 9, no. 3, pp. 135–170, Sep. 2020.
[25] J. J. Gibson, *The Ecological Approach to Visual Perception*. Boston, MA, USA: Houghton Mifflin, 1979.
[26] R. Girdhar, J. Joao Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 244–253.
[27] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (Adaptive Computation and Machine Learning Series). Cambridge, MA, USA: MIT Press, 2016.
[29] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, vol. 2. Cambridge, MA, USA: MIT Press, 2014, pp. 2672–2680.
[30] N. Gordon, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc. F-Radar Signal Process.*, vol. 140, no. 6, pp. 107–113, Apr. 1993.
[31] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2005, pp. 1458–1465.
[32] H. S. Greenwald and C. K. Oertel, "Future directions in machine learning," *Frontiers Robot. AI*, vol. 3, p. 79, Jan. 2017.
[33] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space Odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
[34] R. L. Gregory, *Eye and Brain: The Psychology of Seeing*. New York, NY, USA: McGraw-Hill, 1978.
[35] Y. Gu, L. Wang, Z. Wang, Y. Liu, M.-M. Cheng, and S.-P. Lu, "Pyramid constrained self-attention network for fast video salient object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 10869–10876.
[36] C. Guo, B. Fan, J. Gu, Q. Zhang, S. Xiang, V. Prinet, and C. Pan, "Progressive sparse local attention for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3909–3918.
[37] D. Hall, F. Dayoub, J. Skinner, H. Zhang, D. Miller, P. Corke, G. Carneiro, A. Angelova, and N. Sunderhauf, "Probabilistic object detection: Definition and evaluation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 1020–1029.
[38] J. Han and C. Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *From Natural to Artificial Neural Computation*. Berlin, Germany: Springer, 1995, pp. 195–201.
[39] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Simultaneous detection and segmentation," 2014, *arXiv:1407.1808*.
[40] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Alvey Vis. Conf.*, 1988, pp. 147–151.
[41] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020.
[42] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2014.
[43] C. Hetang, H. Qin, S. Liu, and J. Yan, "Impression network for video object detection," 2017, *arXiv:1712.05896*.
[44] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 1960.
[45] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[46] G. E. Hinton and T. J. Sejnowski, *Learning and Relearning in Boltzmann Machines*. Cambridge, MA, USA: MIT Press, 1986, pp. 282–317.
[47] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[48] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, nos. 1–3, pp. 185–203, Aug. 1980.
[49] B. J. Hou and Z. H. Zhou, "Learning with interpretable structure from gated RNN," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2267–2279, Jul. 2020.
[50] A. Jabbar, X. Li, and B. Omar, "A survey on generative adversarial networks: Variants, applications, and training," *ACM Comput. Surv.*, vol. 54, no. 8, pp. 1–49, Nov. 2022.
[51] X. Jia, J. Willard, A. Karpatne, J. Read, J. Zwart, M. S. Steinbach, and V. Kumar, "Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles," in *Proc. SIAM Int. Conf. Data Mining*, May 2019, pp. 558–566.
[52] X. Jia, J. Willard, A. Karpatne, J. S Read, J. A. Zwart, M. Steinbach, and V. Kumar, "Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles," 2020, *arXiv:2001.11086*.
[53] A. Kahana, E. Turkel, S. Dekel, and D. Givoli, "Obstacle segmentation based on the wave equation and deep learning," *J. Comput. Phys.*, vol. 413, Jul. 2020, Art. no. 109458.
[54] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.
[55] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, "Theory-guided data science: A new paradigm for scientific discovery from data," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 10, pp. 2318–2331, Jun. 2017.
[56] A. Daw, A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (PGNN): An application in lake temperature modeling," 2017, *arXiv:1710.11431*.
[57] S. Khan, M. Naseer, M. Hayat, S. Waqas Zamir, F. Shahbaz Khan, and M. Shah, "Transformers in vision: A survey," 2021, *arXiv:2101.01169*.
[58] Y. Kim, C. Denton, L. Hoang, and A. M. Rush, "Structured attention networks," 2017, *arXiv:1702.00887*.
[59] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, Apr. 2014.
[60] J. F. Kolen and S. C. Kremer, "Gradient flow in recurrent nets: The difficulty of learning LongTerm dependencies," in *A Field Guide to Dynamical Recurrent Networks*. IEEE Press, 2001, pp. 237–243, doi: 10.1109/9780470544037.ch14.
[61] J. Kossen, K. Stelzner, M. Hussing, C. Voelcker, and K. Kersting, "Structured object-aware physics prediction for video modeling and planning," in *Proc. Int. Conf. Learn. Represent.*, 2020.
[62] A. Kumar and S. Srivastava, "Object detection system based on convolution neural networks using single shot multi-box detector," *Proc. Comput. Sci.*, vol. 171, pp. 2610–2617, Jan. 2020.

[63] B. M. Lake and M. Baroni, "Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks," in *Proc. ICML*, 2018, pp. 2879–2888.
[64] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2006, pp. 2169–2178.
[65] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, Feb. 2015.
[66] K. Li, W. Ma, U. Sajid, Y. Wu, and G. Wang, "Object detection with convolutional neural network," 2019, *arXiv:1912.01844*.
[67] C.-C. Lin, Y. Hung, R. Feris, and L. He, "Video instance segmentation tracking with a modified VAE architecture," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13144–13154.
[68] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*.
[69] D. Liu, Y. Cui, Y. Chen, J. Zhang, and B. Fan, "Video object detection for autonomous driving: Motion-aid feature calibration," *Neurocomputing*, vol. 409, pp. 1–11, Oct. 2020.
[70] M. Liu, M. Zhu, M. White, Y. Li, and D. Kalenichenko, "Looking fast and slow: Memory-guided mobile video object detection," 2019, *arXiv:1903.10172*.
[71] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Computer Vision (ECCV)*. Cham, Switzerland: Springer, 2016, pp. 21–37.
[72] X. Liu, Z. Deng, and Y. Yang, "Recent progress in semantic image segmentation," *Artif. Intell. Rev.*, vol. 52, no. 2, pp. 1089–1106, Aug. 2019.
[73] L. Ljung, *System Identification: Theory for the User*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1999.
[74] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2, Sep. 1999, pp. 1150–1157.
[75] Y. Lu, C. Lu, and C.-K. Tang, "Online video object detection using association LSTM," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2363–2371.
[76] G. Marcus, "Deep learning: A critical appraisal," 2018, *arXiv:1801.00631*.
[77] G. Marcus, "The next decade in AI: Four steps towards robust artificial intelligence," 2020, *arXiv:2002.06177*.
[78] S. Mojtaba Marvasti-Zadeh, L. Cheng, H. Ghanei-Yakhdan, and S. Kasaei, "Deep learning for visual tracking: A comprehensive survey," 2019, *arXiv:1912.00535*.
[79] M. Müller, A. Bibi, S. Giancola, S. Al-Subaihi, and B. Ghanem, "TrackingNet: A large-scale dataset and benchmark for object tracking in the wild," 2018, *arXiv:1803.10794*.
[80] W. K. Mutlag, S. K. Ali, Z. M. Aydam, and B. H. Taher, "Feature extraction methods: A review," in *Proc. J. Phys., Conf.*, Jul. 2020, vol. 1591, no. 1, Art. no. 012028.
[81] M. Amin Nabian and H. Meidani, "Physics-driven regularization of deep neural networks for enhanced engineering design and analysis," 2018, *arXiv:1810.05547*.
[82] S. W. Oh, J.-Y. Lee, K. Sunkavalli, and S. J. Kim, "Fast video object segmentation by reference-guided mask propagation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 7376–7385.
[83] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh, "Deep learning vs. traditional computer vision," in *Proc. Comput. Vis. Conf. (CVC)*. Cham, Switzerland: Springer, 2020, pp. 128–144.
[84] A. Oussidi and A. Elhassouny, "Deep generative models: Survey," in *Proc. Int. Conf. Intell. Syst. Comput. Vis. (ISCV)*, Apr. 2018, pp. 1–8.
[85] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[86] R. Paolucci, F. Gatti, and M. Infantino, "Broadband ground motions from 3D physics-based numerical simulations using artificial neural networks," *Bull. Seismolog. Soc. Amer.*, vol. 108, no. 3A, pp. 1272–1286, Feb. 2018.
[87] C. Patel, D. Bhatt, U. Sharma, R. Patel, S. Pandya, K. Modi, N. Cholli, A. Patel, U. Bhatt, M. A. Khan, S. Majumdar, M. Zuhair, K. Patel, S. A. Shah, and H. Ghayvat, "DBGC: Dimension-based generic convolution block for object recognition," *Sensors*, vol. 22, no. 5, p. 1780, Feb. 2022.
[88] C. I. Patel, S. Garg, T. Zaveri, and A. Banerjee, "Top-down and bottom-up cues based moving object detection for varied background video sequences," *Adv. Multimedia*, vol. 2014, pp. 1–20, Jan. 2014.
[89] J. Pearl and D. Mackenzie, *The Book Why: The New Sci. Cause Effect*, 1st ed. New York, NY, USA: Basic Books, 2018.
[90] R. Rai and C. K. Sahu, "Driven by data or derived through physics? A review of hybrid physics guided machine learning techniques with cyber-physical system (CPS) focus," *IEEE Access*, vol. 8, pp. 71050–71073, 2020.
[91] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.
[92] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
[93] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F. F. Li, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[94] M. Sadoughi and C. Hu, "Physics-based convolutional neural network for fault diagnosis of rolling element bearings," *IEEE Sensors J.*, vol. 19, no. 11, pp. 4181–4192, Jun. 2019.
[95] R. Salakhutdinov and G. Hinton, "Deep Boltzmann machines," in *Proc. 12th Int. Conf. Artif. Intell. Statist.*, vol. 5, Clearwater Beach, FL, USA, Apr. 2009, pp. 448–455.
[96] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep Boltzmann machines," in *Proc. AISTATS*, 2010, pp. 693–700.
[97] Z. Shen, J. Liu, Y. He, X. Zhang, R. Xu, H. Yu, and P. Cui, "Towards out-of-distribution generalization: A survey," 2021, *arXiv:2108.13624*.
[98] M. Shvets, W. Liu, and A. Berg, "Leveraging long-range temporal relationships between proposals for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9755–9763.
[99] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[100] S. Singh, A. Prasad, K. Srivastava, and S. Bhattacharya, "Object motion detection methods for real-time video surveillance: A survey with empirical evaluation," in *Smart Systems and IoT: Innovations in Computing*. Singapore: Springer, 2020, pp. 663–679.
[101] S. K. Singh, R. Yang, A. Behjat, R. Rai, S. Chowdhury, and I. Matei, "PI-LSTM: Physics-infused long short-term memory network," in *Proc. 18th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2019, pp. 34–41.
[102] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. Lau, and M.-H. Yang, "VITAL: Visual tracking via adversarial learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8990–8999.
[103] R. Stewart and S. Ermon, "Label-free supervision of neural networks with physics and domain knowledge," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2576–2582.
[104] H. Suk, *An Introduction to Neural Networks and Deep Learning*. Amsterdam, The Netherlands: Elsevier, Jan. 2017, pp. 3–24.
[105] F. Sultana, A. Sufian, and P. Dutta, "A review of object detection models based on convolutional neural network," in *Intelligent Computing: Image Processing Based Applications*. Singapore: Springer, 2020, pp. 1–16.
[106] M. Sultana, A. Mahmood, S. Javed, and S. Ki Jung, "Unsupervised RGBD video object segmentation using GANs," 2018, *arXiv:1811.01526*.
[107] L. Sun, H. Gao, S. Pan, and J.-X. Wang, "Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data," *Comput. Methods Appl. Mech. Eng.*, vol. 361, Apr. 2020, Art. no. 112732.
[108] X. Sun, P. Wu, and S. C. H. Hoi, "Face detection using deep learning: An improved faster RCNN approach," *Neurocomputing*, vol. 299, pp. 42–50, Jul. 2018.
[109] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1420–1429.
[110] S. Tripathi, Z. Lipton, S. Belongie, and T. Nguyen, "Context matters: Refining object detection in video with recurrent neural networks," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 1–12.
[111] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.
[112] A. Ullah, K. Muhammad, W. Ding, V. Palade, I. U. Haq, and S. W. Baik, "Efficient activity recognition using lightweight CNN and DS-GRU network for surveillance applications," *Appl. Soft Comput.*, vol. 103, May 2021, Art. no. 107102.

[113] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2017, pp. 6000–6010.
[114] R. Verschae and J. Ruiz-del-Solar, "Object detection: Current and future directions," *Frontiers Robot. AI*, vol. 2, p. 29, Nov. 2015.
[115] J. Wang, E. Sezener, D. Budden, M. Hutter, and J. Veness, "A combinatorial perspective on transfer learning," in *Proc. Adv. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2020, pp. 918–929.
[116] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, "End-to-end video instance segmentation with transformers," 2020, *arXiv:2011.14503*.
[117] J. Willard, X. Jia, S. Xu, M. Steinbach, and V. Kumar, "Integrating scientific knowledge with machine learning for engineering and environmental systems," 2020, *arXiv:2003.04919*.
[118] H. Wu, Y. Chen, N. Wang, and Z.-X. Zhang, "Sequence level semantics aggregation for video object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9216–9224.
[119] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. Tenenbaum, "Galileo: Perceiving physical object properties by integrating a physics engine with deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, Red Hook, NY, USA: Curran Associates, 2015, pp. 1–9.
[120] C. Xie, Y. Xiang, A. Mousavian, and D. Fox, "Unseen object instance segmentation for robotic environments," 2020, *arXiv:2007.08073*.
[121] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5187–5196.
[122] S. Yang, Y. Fang, X. Wang, Y. Li, C. Fang, Y. Shan, B. Feng, and W. Liu, "Crossover learning for fast online video instance segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 8043–8052.
[123] S. Yang, X. Yu, and Y. Zhou, "LSTM and GRU neural network performance comparison study: Taking yelp review dataset as an example," in *Proc. Int. Workshop Electron. Commun. Artif. Intell. (IWECAI)*, Jun. 2020, pp. 98–101.
[124] T. Yang and A. B. Chan, "Recurrent filter learning for visual tracking," 2017, *arXiv:1708.03874*.
[125] Y. Yu, X. Si, C. Hu, and Z. Jianxun, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019.
[126] D. Zhang, J. Han, G. Cheng, and M.-H. Yang, "Weakly supervised object localization and detection: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Apr. 20, 2021, doi: 10.1109/TPAMI.2021.3074313.
[127] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.
[128] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. Change Loy, "Domain generalization in vision: A survey," 2021, *arXiv:2103.02503*.
[129] H. Zhu, H. Wei, B. Li, X. Yuan, and N. Kehtarnavaz, "A review of video object detection: Datasets, metrics and methods," *Appl. Sci.*, vol. 10, no. 21, p. 7834, Nov. 2020.
[130] M. Zhu and M. Liu, "Mobile video object detection with temporally-aware feature maps," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5686–5695.
[131] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, "Flow-guided feature aggregation for video object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 408–417.
[132] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," 2016, *arXiv:1611.07715*.
[133] I. Ševo and A. Avramović, "Convolutional neural network based automatic object detection on aerial images," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 5, pp. 740–744, May 2016.

ATHINA ILIOUDI received the joint M.S. degree in smart electrical networks and systems from the KTH Royal Institute of Technology and the Eindhoven University of Technology, in 2018. She is currently pursuing the Ph.D. degree with the Delft Center for Systems and Control, Delft University of Technology. Her research interests include deep learning methods with first-principle modeling techniques and physics informed neural networks for computer vision applications.

AZITA DABIRI received the Ph.D. degree from the Automatic Control Group, Chalmers University of Technology, in 2016. She was a Postdoctoral Researcher with the Department of Transport and Planning, TU Delft, from 2017 to 2019. In 2019, she received an ERCIM Fellowship and a Marie Curie Individual Fellowship, which allowed her to perform research at the Norwegian University of Technology (NTNU) as a Postdoctoral Researcher. In 2020, she joined the Delft Center for Systems and Control, TU Delft, as an Assistant Professor. Her research interests include integration of model-based and learning-based control.

BEN J. WOLF received the Ph.D. degree *(cum laude)* in artificial intelligence from the University of Groningen, The Netherlands, in 2020, on the topic of hydrodynamic imaging. He is currently a Postdoctoral Researcher at the Delft Center for Systems and Control, Delft University of Technology. His research interests include machine learning, neural networks, robotics, and hydrodynamic sensing.

BART DE SCHUTTER (Fellow, IEEE) received the Ph.D. degree *(summa cum laude)* in applied sciences from Katholieke Universiteit Leuven, Belgium, in 1996.

He is currently a Full Professor and the Head of Department at the Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands. His current research interests include reinforcement learning, learning-based control, multi-level and multi-agent control, and control of hybrid systems. He is a Senior Editor of the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and an Associate Editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL.