A Survey on Deep Learning Technique for Video Segmentation

Tianfei Zhou, Fatih Porikli, Fellow IEEE, David J. Crandall, Senior Member IEEE, Luc Van Gool, Wenguan Wang, Senior Member IEEE

Abstract—Video segmentation—partitioning video frames into multiple segments or objects—plays a critical role in a broad range of practical applications, from enhancing visual effects in movie, to understanding scenes in autonomous driving, to creating virtual background in video conferencing. Recently, with the renaissance of connectionism in computer vision, there has been an influx of deep learning based approaches for video segmentation that have delivered compelling performance. In this survey, we comprehensively review two basic lines of research — generic object segmentation (of unknown categories) in videos, and video semantic segmentation — by introducing their respective task settings, background concepts, perceived need, development history, and main challenges. We also offer a detailed overview of representative literature on both methods and datasets. We further benchmark the reviewed methods on several well-known datasets. Finally, we point out open issues in this field, and suggest opportunities for further research. We also provide a public website to continuously track developments in this fast advancing field: https://github.com/tfzhou/VS-Survey.

Index Terms—Video Segmentation, Video Object Segmentation, Video Semantic Segmentation, Deep Learning

1 INTRODUCTION

VIDEO segmentation — identifying the key objects with some specific properties or semantics in a video scene — is a fundamental and challenging problem in computer vision, with numerous potential applications including autonomous driving, robotics, autonomous surveillance, social media, augmented reality, movie production, and video conferencing.

The problem has been addressed using various traditional computer vision and machine learning techniques, including hand-crafted features (e.g., histogram statistics, optical flow, etc.), heuristic prior knowledge (e.g., visual attention mechanism [1], motion boundaries [2], etc.), low/mid-level visual representations (e.g., super-voxel [3], trajectory [4], object proposal [5], etc.), and classical machine learning models (e.g., clustering [6], graph models [7], random walks [8], support vector machines [9], random decision forests [10], markov random fields [11], conditional random fields [12], etc.). Recently, deep neural networks, and Fully Convolutional Networks (FCNs) [13] in particular, have led to remarkable advances in video segmentation. These deep learning-based video segmentation algorithms are significantly more accurate (and sometimes even more efficient) than traditional approaches.

With the rapid advance of this field, there is a huge body of new literature being produced. However, most existing surveys predate the modern deep learning era [14], [15], and often take a narrow view, such as focusing only on video foreground/background segmentation [16], [17]. In this paper, we offer a state-of-the-art review that addresses the wide area of video segmentation, especially to help new researchers enter this rapidly-developing field. We systematically introduce recent advances in video seg-
mentation, spanning from task formulation to taxonomy, from algorithms to datasets, and from unsolved issues to future research directions. We cover crucial aspects including task categories (i.e., foreground/background separation vs semantic segmentation), inference modes (i.e., automatic, semi-automatic, and interactive), and learning paradigms (i.e., supervised, unsupervised, and weakly supervised), and we try to clarify terminology (e.g., background subtraction, motion segmentation, etc.). We hope that this survey helps accelerate progress in this field.

This survey mainly focuses on recent progress in two major branches of video segmentation, namely video object segmentation (Fig. 1(a-e)) and video semantic segmentation (Fig. 1(f-h)), which are further divided into eight sub-fields. Even after restricting our focus to deep learning-based video segmentation, there are still hundreds of papers in this fast-growing field. We select influential work published in prestigious journals and conferences. We also include some non-deep learning video segmentation models and relevant literature in other areas, e.g., visual tracking, to give necessary background. Moreover, in order to promote the development of this field, we provide an accompanying webpage which catalogs algorithms and datasets addressing video segmentation: https://github.com/tfzhou/VS-Survey.

Fig. 2 shows the structure of this survey. Section §2 gives some brief background on taxonomy, terminology, study history, and related research areas. We review representative papers on deep learning algorithms and video segmentation datasets in §3 and §4, respectively. Section §5 conducts performance evaluation and analysis, while §6 raises open questions and directions. Finally, we make concluding remarks in §7.

2 BACKGROUND

In this section, we first formalize the task, categorize research directions, and discuss key challenges and driving factors in §2.1. Then, §2.2 offers a brief historical background covering early work and foundations, and §2.3 establishes linkages with relevant fields.

2.1 Problem Formulation and Taxonomy

Formally, let $X$ and $Y$ denote the input space and output segmentation space, respectively. Deep learning-based video segmentation solutions generally seek to learn an ideal video-to-segment mapping $f^*: X \rightarrow Y$.

2.1.1 Video Segmentation Category

According to how the output space $Y$ is defined, video segmentation can be broadly categorized into two classes: video object (foreground/background) segmentation, and video semantic segmentation.

- **Video Foreground/Background Segmentation (Video Object Segmentation, VOS).** VOS is the classic video segmentation setting and refers to segmenting dominant objects (of unknown categories). In this case, $Y$ is a binary, foreground/background segmentation space. VOS is typically used in video analysis and editing application scenarios, such as object removal in movie editing, content-based video coding, and virtual background creation in video conferencing. It typically is not concerned with the exact semantic categories of the segmented objects.

- **Video Semantic Segmentation (VSS).** As a direct extension of image semantic segmentation to the spatio-temporal domain, VSS aims to extract objects within predefined semantic categories (e.g., car, building, pedestrian, road) from videos. Thus, $Y$ corresponds to a multi-class, semantic parsing space. VSS serves as a perception foundation for many application fields, such as robot sensing, human-machine interaction, and autonomous driving, which require high-level understanding of the physical environment.

**Remark.** VOS and VSS share some common challenges, such as fast motion and object occlusion. However, due to differences in application scenarios, many challenges are different. For instance, VOS often focuses on human created
media, which often have large camera motion, deformation, and appearance changes. VSS instead often focuses on applications like autonomous driving, which requires a good trade off between accuracy and latency, accurate detection of small objects, model parallelization, and cross-domain generalization ability.

### 2.1.2 Inference Modes for Video Segmentation

VOS methods can be further classified into three types: automatic, semi-automatic, and interactive, according to how much human intervention is involved during inference.

- **Automatic Video Object Segmentation (AVOS).** AVOS, or unsupervised video segmentation or zero-shot video segmentation, performs VOS in an automatic manner, without any manual initialization (Fig. 1(a-b)). The input space $\mathcal{X}$ refers to the video domain $\mathcal{V}$ only. AVOS is suitable for video analysis but not for video editing that requires segmenting arbitrary objects or their parts flexibly; a typical application is virtual background creation in video conferencing.

- **Semi-automatic Video Object Segmentation (SVOS).** SVOS, also known as semi-supervised video segmentation or one-shot video segmentation [18], involves limited human inspection (typically provided in the first frame) to specify the desired objects (Fig. 1(c)). For SVOS, $\mathcal{X} = \mathcal{V} \times \mathcal{M}$, where $\mathcal{V}$ indicates the video space and $\mathcal{M}$ refers to human input. Typically the human input is an object mask in the first video frame, in which case SVOS is also called pixel-wise tracking or mask propagation. Other forms of human input include bounding boxes and scribbles [8]. From this perspective, language-guided video object segmentation (LVOS) is a sub-branch of SVOS, in which the human input is given as linguistic descriptions about the desired objects (Fig. 1(e)). Compared to AVOS, SVOS is more flexible in defining target objects, but requires human input. SVOS is typically applied in a user-friendly setting (without specialized equipment), such as video content creation in mobile phones. One of the core challenges in SVOS is how to fully utilize target information from limited human intervention.

- **Interactive Video Object Segmentation (IVOS).** IVOS models are designed to operate automatically once the target has been identified, while systems for IVOS incorporate user guidance throughout the analysis process (Fig. 1(d)). IVOS can obtain high-quality segments and works well for computer-generated imagery and video post-production, where tedious human supervision is possible. IVOS is also studied in the graphics community as video cutout. The input space $\mathcal{X}$ for IVOS is $\mathcal{V} \times S$, where $S$ typically refers to human scribbling. Key challenges include: 1) allowing users to easily specify segmentation constraints; 2) incorporating human specified constraints into the segmentation algorithm; and 3) giving quick response to the constraints.

In contrast to VOS, VSS methods typically work in an automatic mode (Fig. 1(f-h)), i.e., $\mathcal{X} = \mathcal{V}$. Only a few early methods address the semi-automatic setting, called label propagation [19].

**Remark.** The terms “unsupervised” and “semi-supervised” are conventionally used in VOS to specify the amount of human interaction involved during inference. But they are easily confused with “unsupervised learning” and “semi-supervised learning.” We urge the community to replace these ambiguous terms with “automatic” and “semi-automatic.”

### 2.1.3 Learning Paradigms for Video Segmentation

Deep learning-based video segmentation models can be grouped into three categories according to the learning strategy they use to approximate $f^*$: supervised, unsupervised, and weakly supervised.

- **Supervised Learning Methods.** Modern video segmentation models are typically learned in a fully supervised manner, requiring $N$ input training samples and their desired outputs $y_n := f^*(x_n)$, where $\{(x_n, y_n)\}_n \subset \mathcal{X} \times \mathcal{Y}$. The standard method for evaluating learning outcomes follows an empirical risk/loss minimization formulation:

$$\hat{f} \in \text{arg min}_{f \in \mathcal{F}} \frac{1}{N} \sum_{n=1}^{N} \varepsilon(f(x_n), z(x_n)),$$

where $\mathcal{F}$ denotes the hypothesis (solution) space, and $\varepsilon : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ is an error function that evaluates the estimate $f(x_n)$ against video segmentation related prior knowledge $z(x_n) \in \mathcal{Z}$. To make $\hat{f}$ a good approximation of $f^*$, current supervised video segmentation methods directly use the desired output $y_n$, i.e., $z(x_n) := f^*(x_n)$, as the prior knowledge, with the price of requiring vast amounts of well-labeled data.

- **Unsupervised (Self-supervised) Learning Methods.** When only data samples $\{(x_n)\}_n \subset \mathcal{X}$ are given, the problem of approximating $f^*$ is known as unsupervised learning. Unsupervised learning includes fully supervised learning methods in which the methods do not need any labels at all, as well as self-supervised learning methods in which networks are explicitly trained with automatically-generated pseudo labels without any human annotations [20]. Almost all existing unsupervised learning-based video segmentation models are self-supervised learning methods, where the prior knowledge $\mathcal{Z}$ refers to pseudo labels derived from intrinsic properties of video data (e.g., cross-frame consistency). We thus use “unsupervised learning” and “self-supervised learning” interchangeably.

- **Weakly-Supervised Learning Methods.** In this case, $\mathcal{Z}$ is typically a more easily-annotated domain, such as tags, bounding boxes, or scribbles, and $f^*$ is approximated using a finite number of samples from $\mathcal{X} \times \mathcal{Z}$.

**Remark.** So far, deep supervised learning-based methods are dominant in the field of video segmentation. However, exploring the task in an unsupervised or weakly supervised setting is more appealing, not only because it alleviates the annotation burden of $\mathcal{Y}$, but because it inspires an in-depth understanding of the nature of the task by exploring $\mathcal{Z}$.

### 2.2 History and Terminology

Digital image segmentation has been studied for at least 50 years, starting with the Roberts operator [21] for identifying object boundaries. Since then, numerous algorithms for image segmentation have been proposed, and many are extended to the video domain. The field of video segmentation has evolved quickly and undergone great change.

Earlier attempts focus on video over-segmentation, i.e., partitioning a video into space-time homogeneous, perceptually distinct-regions. Typical approaches include hierarchical video segmentation [7], temporal superpixel [22], and super-voxels [3], based on the discontinuity and similarity of pixel intensities in a particular location, i.e., separating pixels

---

1. We omit the regularization term for brevity.
 according to abrupt changes in intensity or grouping pixels with similar intensity together. These methods are instructive for early stage video preprocessing, but cannot solve the problem of object-level pattern modeling, as they do not provide any principled approach to flatten the hierarchical video decomposition into a binary segmentation [2], [9].

To extract foreground objects from video sequences, background subtraction techniques emerged beginning in the late 70s [23], and became popular following the work of [24]. They assume that the background is known a priori, and that the camera is stationary [25], [26] or undergoes a predictable, parametric 2D [27] or 3D motion with 3D parallax [28]. These geometry-based methods fit well for specific application scenarios such as surveillance systems [9], [26], but they are sensitive to model selection (2D or 3D), and cannot handle non-rigid camera movements.

Another group of video segmentation solutions tackled the task of motion segmentation, i.e., finding objects in motion. Background subtraction can also be viewed as a specific case of motion segmentation. However, most motion segmentation models are built upon motion analysis [29], [30], factorization [31], and/or statistical [32] techniques that comprehensively model the characteristics of moving scenes without prior knowledge of camera motion. Among the big family of motion segmentation algorithms, trajectory segmentation attained particular attention [4], [33]–[36]. Trajectories are generated through tracking points over multiple frames and can represent long-term motion patterns, serving as an informative cue for segmentation. Though impressive, motion-based methods heavily rely on the accuracy of optical flow estimation and can fail when different parts of an object exhibit heterogeneous motions.

To overcome these limitations, the task of extracting generic objects from unconstrained video sequences, i.e., AVOS, has drawn increasing research interest [37]. Several methods [5], [38]–[40] explored object hypotheses or proposals [41] as middle-level object representations. They generate a large number of object candidates in every frame and cast the task of segmenting video objects as an object region selection problem. The main drawbacks of the proposal-based algorithms are the high computational cost [17] and complicated object inference schemes. Some others explored heuristic hypotheses such as visual attention [1] and motion boundary [2], but easily fail in scenarios where the heuristic assumptions do not hold.

As argued earlier, an alternative to the above unattended solutions is to incorporate human-marked initialization, i.e., SVOS. Older SVOS methods often rely on optical flow [8], [42]–[44] and share a similar spirit with object tracking [45], [46]. In addition, some pioneering IVOS methods were proposed to address high-quality video segmentation under extensive human guidance, including rotoscoping [47], [48], scribble [8], [49]–[52], contour [53], and points [54]. Significant engineering is typically needed to allow IVOS systems to operate at interactive speeds. In short, SVOS and IVOS pay for the improved flexibility and accuracy: they are infeasible at large scale due to their human-in-the-loop nature.

In the pre-deep learning era, relatively few papers [12], [55]–[58] considered VSS due to the complexity of the task. The approaches typically relied on supervised classifiers such as SVMs and video over-segmentation techniques.

Overall, traditional approaches for video segmentation, though giving interesting results, are constrained by hand-crafted features and heavy engineering. But deep learning brought the performance of video segmentation to a new level, as we will review in §3.

2.3 Related Research Areas

There are several research fields closely related to video segmentation, which we now briefly describe.

- **Visual Tracking.** To infer the location of a target object over time, current tracking methods usually assume that the target is determined by a bounding box in the first frame [59]. However, in more general tracking scenarios, and in particular the cases studied in early tracking methods, diverse object representations are explored [60], including centroids, skeletons, and contours. Some video segmentation techniques, such as background subtraction, are also merged into older trackers [61], [62]. Hence, visual tracking and video segmentation encounter some common challenges (e.g., object/camera motion, appearance change, occlusion, etc.), fostering their mutual collaboration.

- **Image Semantic Segmentation.** The success of end-to-end image semantic segmentation [63]–[65] has sparked the rapid development of VSS. Rather than directly applying image semantic segmentation techniques frame by frame, recent VSS systems explore temporal continuity to increase both accuracy and efficiency. Nevertheless, image semantic segmentation techniques continue to serve as a foundation for advancing segmentation in video.

- **Video Object Detection.** To generalize object detection in the video domain [112], video object detectors incorporate temporal cues over the box- or feature-level. There are many key technical steps and challenges, such as object proposal generation, temporal information aggregation, and cross-frame object association, that are shared between video object detection and (instance-level) video segmentation.

3 Deep Learning-based Video Segmentation

3.1 Deep Learning-based VOS Models

VOS extracts generic foreground objects from video sequences with no concern for semantic category recognition. Based on how much human intervention is involved in inference, VOS models can be divided into three classes (§2.1.2): automatic (AVOS, §3.1.1), semi-automatic (SVOS, §3.1.2), and interactive (IVOS, §3.1.3). Moreover, although language-guided video object segmentation (LVOS) falls in the broader category of SVOS, LVOS methods are reviewed alone (§3.1.4), due to the specific multi-modal task setup.

3.1.1 Automatic Video Object Segmentation (AVOS)

Instead of using heuristic priors and hand-crafted features to automatically execute VOS, modern AVOS methods learn generic video object patterns in a data-driven fashion. We group landmark efforts based on their key techniques.

- **Deep Learning Module based Methods.** In 2015, Fragkiadaki et al. [113] made an early effort that learns a multi-layer perceptron to rank proposal segments and infer foreground objects. In 2016, Tsai et al. [43] proposed a joint optimization framework for AVOS and optical flow estimation.
with a naive use of deep features from a pre-trained classification network. Later methods [70], [72] learn FCNs to predict initial, pixel-level foreground estimates from frame images [72], [114] or optical flow fields [70], while several post-processing steps are still needed. Basically, these primitive solutions largely rely on traditional AVOS techniques; the learning ability of neural networks is under-explored.

- **Pixel Instance Embedding based Methods.** A group of AVOS models has been developed to make use of stronger deep learning descriptors [74], [77] – instance embeddings – learned from image instance segmentation data [115]. They first generate pixel-wise instance embeddings, and select representative embeddings which are clustered into foreground and background. Finally, the labels of the sampled embeddings are propagated to the other ones. The clustering and propagation can be achieved without video specific supervision. Though using fewer annotations, these methods suffer from a fragmented and complicated pipeline.

- **End-to-end Methods with Short-term Information Encoding.** End-to-end model designs became the mainstream in this field. For example, convolutional recurrent neural networks (RNNs) were used to learn spatial and temporal visual patterns jointly [78], [89]. Another big family is built upon two-stream networks [66], [68], [69], [75], [90], [93], [94], wherein two parallel streams are built to extract features from raw image and optical flow, which are further fused for segmentation prediction. Two-stream methods make explicit use of appearance and motion cues, at the cost of optical flow computation and vast learnable parameters. These end-to-end methods improve accuracy and show the advantages of applying neural networks to this task. However, they only consider local context within very limited time span; they stack appearance and/or motion information from a few successive frames as input, ignoring relations among distant frames. Although RNNs are usually adopted, their internal hidden memory creates the inherent limits in modeling longer-term dependencies [116].

- **End-to-end Methods with Long-term Context Encoding.** Current leading AVOS models use global context over long time spans. In a seminal work [83], Lu et al. proposed a Siamese architecture-based model that extracts features for arbitrary frame pairs and captures cross-frame context by calculating pixel-wise feature correlations. During inference, for each test frame, context from several other frames (within the same video) is aggregated to locate objects. A contemporary work [92] exploited a similar idea but only used the first frame as reference. Several papers [101], [107] extended [83] by making better use of information from multiple frames [89], [175], [176], encoding spatial context [105], and incorporating temporal consistency to improve representation power and computation efficiency [96], [104].

- **Un-/Weakly-supervised based Methods.** Only a handful of methods learn to perform AVOS from unlabeled or weakly labeled data. In [87], static image salient object segmentation and dynamic eye fixation data, which are more easily acquired compared with video segmentation data, are used to learn video generic object patterns. In [98], visual patterns are learned through exploring several intrinsic properties of video data at multiple granularities, i.e., intra-frame saliency, short-term visual coherence, long-range semantic correspondence, and video-level discriminativeness. In [86], an adversarial contextual model is developed to segment moving objects without any manual annotation, achieved by minimizing the mutual information between the motions of an object and its context. This method is further enhanced in [108] by adopting a bootstrapping strategy and enforcing temporal consistency. In [109], motion is exclusively exploited to discover moving objects, and a Transformer-based model is designed and trained by self-supervised flow reconstruction using unlabeled video data.

### Table 1

<table>
<thead>
<tr>
<th>Year</th>
<th>Method</th>
<th>Pub.</th>
<th>Core Architecture</th>
<th>Instance</th>
<th>Flow</th>
<th>Training Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IVO [87]</td>
<td>ICCV</td>
<td>Two-Stream FCN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>LVO [88]</td>
<td>ICCV</td>
<td>Two-Stream FCN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>LMP [89]</td>
<td>ICCV</td>
<td>FCN</td>
<td>Object</td>
<td>✓</td>
<td>FTID [71]</td>
</tr>
<tr>
<td></td>
<td>NRE [90]</td>
<td>ICCV</td>
<td>FCN</td>
<td>Object</td>
<td>✓</td>
<td>Youtube-Objects [72]</td>
</tr>
<tr>
<td></td>
<td>RIC [91]</td>
<td>CVPR</td>
<td>FCN + RNN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>MBBN [93]</td>
<td>ICCV</td>
<td>FCN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>PFDR [94]</td>
<td>RNN</td>
<td>FCN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>MOT [95]</td>
<td>IJCAI</td>
<td>Two-Stream FCN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td>2018</td>
<td>RyBC [96]</td>
<td>CVPR</td>
<td>RNN</td>
<td>Instance</td>
<td>✓</td>
<td>Youtube-Objects [72]</td>
</tr>
<tr>
<td></td>
<td>AGC [99]</td>
<td>CVPR</td>
<td>FCN</td>
<td>Object</td>
<td>✓</td>
<td>FTID [71] + DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>ACGN [100]</td>
<td>ICCV</td>
<td>FCN + GNN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17] + FMMS [36]</td>
</tr>
<tr>
<td></td>
<td>MGA [101]</td>
<td>ICCV</td>
<td>Two-Stream FCN</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>I2MEM [102]</td>
<td>ICCV</td>
<td>Siamese FCN + Co-attention</td>
<td>Object</td>
<td>✓</td>
<td>DAVIS16 [17]</td>
</tr>
<tr>
<td></td>
<td>MAII [104]</td>
<td>CVPR</td>
<td>FCN</td>
<td>Object</td>
<td>✓</td>
<td>OnLVa [99]</td>
</tr>
<tr>
<td></td>
<td>FNet [108]</td>
<td>AAII</td>
<td>Siamese FCN</td>
<td>Object</td>
<td>✓</td>
<td>Youtube-Objects [72]</td>
</tr>
</tbody>
</table>

Note: The table lists some of the reviewed AVOS methods, including their year, method name, publication, core architecture, instance handling, flow, and training dataset.
and associating instances over different frames. In an early attempt [80], Ventura et al. delivered a recurrent network-based model that consists of a spatial LSTM for per-frame instance discovery and a temporal LSTM for cross-frame instance association. This method features an elegant model design, while its representation ability is too weak to enumerate all the object instances and to capture complex interactions between instances over the temporal domain. Thus later methods [175], [178], [179] strengthen the two-step pipeline through: i) employing image instance segmentation models (e.g., Mask R-CNN [180]) to detect object candidates, and ii) leveraging tracking/re-identification techniques and manually designed rules for instance association. Foreground/background AVOS techniques [83], [89] are also used to filter out nonsalient candidates [175], [179]. More recent methods, e.g., [116], generate object candidates first and obtain corresponding tracklets via advanced SVOS techniques. Overall, current instance-level AVOS models follow the classic tracking-by-detection paradigm, involving several ad-hoc designs. There is still considerable room for further improvement in accuracy and efficiency.

### 3.1.2 Semi-automatic Video Object Segmentation (SVOS)

Deep learning-based SVOS methods mainly focus on the first-frame mask propagation setting. They are categorized by their utilization of the test-time provided object masks.

- **Online Fine-tuning based Methods.** Following the one-shot principle, this family of methods [18], [129], [132], [138] trains a segmentation model separately on each given object mask in an online fashion. Fine-tuning methods essentially exploit the transfer learning capabilities of neural networks and often follow a two-step training procedure: i) **offline pre-training:** learn general segmentation features from images and video sequences, and ii) **online fine-tuning:** learn task-specific representations from test-time supervision. The idea of fine-tuning was first introduced in [18], where only the initial image-mask pair is used for training an online, one-shot, but merely appearance-based FCN model. Then, in [122], more pixel samples in the unlabeled frames are mined as online training samples to better adapt to further changes over time. As [18], [122] have no notion of individual objects, [132] further incorporates instance segmentation models (e.g., Mask R-CNN [180]) during inference. While elegant through their simplicity, fine-tuning methods have several weaknesses: i) pre-training is fixed and not optimized for subsequent fine-tuning, ii) hyperparameters of online fine-tuning are often excessively hand-crafted and fail to generalize between test cases, iii) the common existing fine-tuning setups suffer from high test runtimes (up to 1,000 training iterations per segmented object online [18]). The root cause is that these approaches choose to encode all the target-related cues (i.e., appearance, mask) into network
parameters implicitly. Towards efficient and automated fine-tuning, some recent methods [128], [138], [153] turn to meta learning techniques, i.e., optimize the fine-tuning policies [138], [153] (e.g., generic model initialization, learning rates, etc.) or even directly modify network weights [128].

- **Propagation-based Methods.** Two recent lines of research – built upon mask propagation and template matching techniques respectively – try to refrain from the online optimization to deliver compact, end-to-end SVOS solutions. In particular, propagation-based methods use the previous frame mask to infer the current mask [117], [119], [120]. For example, Jampani et al. [120] propose a bilateral network for long-range video-adaptive mask propagation. Perazzi et al. [117] approach SVOS by employing a modified FCN, where the previous frame mask is considered as an extra input channel. Follow-up work adopts optical flow guided mask alignment [129], heavy first-frame data augmentation [124], and multi-step segmentation refinement [130]. Others apply re-identification to retrieve missing objects after prolonged occlusions [134], design a reinforcement learning agent that tackles SVOS as a conditional decision-making process [131], or propagate masks in a spatiotemporal MRF model to improve temporal coherency [125]. Some researchers leverage location-aware embeddings to sharpen the feature [135], or directly learn sequence-to-sequence mask propagation [95]. Advanced tracking techniques are also exploited in [126], [140], [158], [157]. Propagation-based methods are found to easily suffer from error accumulation due to occlusions and drifts during mask propagation. Conditioning propagation on the initial frame-mask pair [127], [141], [148] seems a feasible solution to this. Although target-specific mask is explicitly encoded into the segmentation network, making up for the deficiencies of fine-tuning methods to a certain extent, propagation-based methods still embed object appearance into hidden network weights. Clearly, such implicit target-appearance modeling strategy hurts flexibility and adaptivity (while [142] is an exception – a generative model of target and background is explicitly built to aid mask propagation).

- **Matching-based Methods.** This type of methods, might the most promising SVOS solution so far, constructs an embedding space to memorize the initial object embeddings, and classifies each pixel’s label according to their similarities to the target object in the embedding space. Thus the initial object appearance is explicitly modeled, and test-time fine-tuning is not needed. The earliest effort in this direction can be tracked back to [121]. Inspired by the advance in visual tracking [190], Yoon et al. [121] proposed a Siamese network to perform pixel-level matching between the first frame and upcoming frames. Later, [126] proposed to learn an embedding space from the first-frame supervision and pose VOS as a task of pixel retrieval: pixels are simply their respective nearest neighbors in the learned embedding space. The idea of [121] is also explored in [133], while it computes two matching maps for each upcoming frame, with respect to the foreground and background annotated in the first frame. In [139], pixel-level similarities computed from the first frame and from the previous frame are used as a guide to segment succeeding frames. Later, many matching-based solutions were proposed [145], [191], perhaps most notably Oh et al., who propose a space-time memory (STM) model to explicitly store previously computed segmentation information in an external memory [149]. The memory facilitates learning the evolution of objects over time and allows for comprehensive use of past segmentation cues even over long period of time. Almost all current top-leading SVOS solutions [159], [163] are built upon STM; they improve the target adaptation ability [100], [158], [164], incorporate local temporal continuity [162], [172], [173], explore instance-aware cues [171], and develop more efficient memory designs [154], [161], [165], [168]. Recently, [170] introduced a Transformer [192] based model, which performs matching-like computation through attending over a history of multiple frames. In general, matching-based solutions enjoy the advantage of flexible and differentiable model design as well as long-term correspondence modeling. On the other hand, feature matching relies on a powerful and generic feature embedding, which may limit its performance in challenging scenarios.

It is also worth mentioning that, as an effective technique for target-specific model learning, online learning is applied by many propagation [95], [117], [125], [130], [140] and matching [121], [134], [145] methods to boost performance.

- **Box-initialization based Methods.** As pixel-wise annotations are time-consuming or even impractical to acquire in realistic scenes, some work has considered the situation where the first-frame annotation is provided in the form of a bounding box. Specifically, in [143], Siamese trackers are augmented with a mask prediction branch. In [156], reinforcement learning is introduced to make decisions for target updating and matching. Later, in [167], an outside memory is utilized to build a stronger Siamese track-segmenter.

- **Un-/Weakly-supervised based Methods.** To alleviate the demand for large-scale, pixel-wise annotated training samples, several un-/weakly-supervised learning-based SVOS solutions were recently developed. They are typically built as a reconstruction scheme (i.e., each pixel from a ‘query’ frame is reconstructed by finding and assembling related pixels from adjacent frame(s)) [136], [160], [193], and/or adopt a cycle-consistent tracking paradigm (i.e., pixels/patches are encouraged to fall into the same location after one cycle of forward and backward tracking) [98], [150], [152], [174].

- **Other Specific Methods.** Other papers make specific contributions that deserve a separate look. In [147], Zeng et al. extract mask proposals per frame and formulate the matching between object templates and proposals in a differentiable manner. Instead of using only the first frame annotation, [144] learns to select the best frame from the whole video for user interaction, so as to boost mask propagation. In [166], Li et al. introduce a forward-backward data flow based cycle consistency mechanism to improve both traditional SVOS training and offline inference protocols, through mitigating the error propagation problem. To accelerate processing speed, a dynamic network [169] is proposed to selectively allocate computation source for each frame according to the similarity to the previous frame.

### 3.1.3 Interactive Video Object Segmentation (IVOS)

AVOS, without any human involvement, loses flexibility in segmenting arbitrary objects of user interest. SVOS addsetting has gained increasing attention. Unlike classic models [48], [49], [52] requiring extensive and professional user intervention, recent deep learning-based IVOS solutions usually work with multiple rounds of scribble supervision,
tionally considers first-frame annotations, but easily fails in challenging scenes without human feedback. Moreover, the first-frame annotations are typically detailed masks, which are tedious to acquire; 79 seconds per instance for coarse polygon annotations of COCO [123], and much more for higher quality. Thus performing VOS in the interactive space to minimize the user’s effort. In this scenario [194], the user draws scribbles on a selected frame and an algorithm computes the segmentation maps for all video frames in a batch process. For refinement, user intervention and segmentation are repeated. This round-based interaction [183] is useful for consumer-level applications and rapid prototyping for professional usage, where efficiency is the main concern. One can control the segmentation quality at the expense of time, as more rounds of interaction will provide better results.

- **Interaction-propagation based Methods.** The majority of current studies [188], [189] follow an interaction-propagation scheme. In the preliminary attempt [181], IVOS is achieved by a simple combination of two separate modules: an interactive image segmentation model [195] for producing segmentation based on user annotations; and a SVOS model [18] for propagating masks from the user-annotated frames to the others. Later, [183] devised a more compact solution, with also two modules for interaction and propagation, respectively. However, the two modules are internally connected through intermediate feature exchanging, and also externally connected, i.e., each of them is conditioned on the other’s output. In [185], a similar model design is also adopted, however, the propagation part is specifically designed to address local mask tracking (over adjacent frames) and global propagation (among distant frames), respectively. However, these techniques [181], [185] have to start a new feed-forward computation in each interaction round, making them inefficient as the number of rounds grows. A more efficient solution was developed in [184]. The critical idea is to build a common encoder for discriminative pixel embedding learning, upon which two small network branches are added for interactive segmentation and mask propagation, respectively. Thus the model extracts pixel embeddings for all frames only once (in the first round). In the following rounds, the feed-forward computation is only made within the two shallow branches.

- **Other Methods.** Chen et al. [182] propose a pixel embedding learning-based model, applicable to both SVOS and IVOS. With a similar idea of [126], IVOS is formulated as a pixel-wise retrieval problem, i.e., transferring labels to each pixel according to its nearest reference pixel. This model supports different kinds of user input, such as masks, clicks and scribbles, and can provide immediate feedback after user interaction. In [186], an interactive annotation tool is proposed for VOS. The annotation has two phases: annotating objects with tracked boxes, and labeling masks inside these tracks. Box tracks are annotated efficiently by approximating the trajectory using a parametric curve with a small number of control points which the annotator can interactively correct. Segmentation masks are corrected via scribbles which are propagated through time. In [187], a reinforcement learning framework is exploited to automatically determine the most valuable frame for interaction.

### 3.1.4 Language-guided Video Object Segmentation (LVOS)

LVOS is an emerging area, dating back to 2018 [196], [197]. Although there have already existed some efforts [247] in the intersection of language and video understanding, none of them addresses pixel-level video-language reasoning. Most efforts in LVOS are made around the theme of efficient alignment between visual and linguistic modalities. According to the multi-modal information fusion strategy, existing models can be divided into three groups.

- **Dynamic Convolution-based Methods.** The first initiate was proposed in [196] that applies dynamic networks [248] for visual-language relation modeling. Specifically, convolution filters, dynamically generated from linguistic query, are used to adaptively transform visual features into desired segments. In the same line of work, [199], [200] incorporate spatial context into filter generation. However, as indicated by [198], linguistic variation of input description may greatly impact sentence representation and subsequently make dynamic filters unstable, causing inaccurate segmentation. For example, “car in blue is parked on the grass” and “blue car standing on the grass” have the same meaning but different generated filters, leading to poor performance.

- **Capsule Routing-based Methods.** In [201], both video and textual inputs are encoded through capsules [249], which are considered effective in modeling visual/textual entities. Then, dynamic routing is applied over the video and text capsules for visual-textual information integration.

- **Attention-based Methods.** Neural attention technique is also widely adopted in the filed of LVOS [197], [202], [204], [250], [251], for fully capturing global visual/textual context. In [198], vision-guided language attention and language-guided vision attention were developed to capture visual-textual correlations. In [203], two different attentions are learned to ground spatial and temporal relevant linguistic cues to static and dynamic visual embeddings, respectively.

### 3.2 Deep Learning-based VSS Models

Video semantic segmentation aims to group pixels with different semantics (e.g., category or instance membership),...
where different semantics result in different types of segmentation tasks, such as (instance-agnostic) video semantic segmentation (VSS, §3.2.1), video instance segmentation (VIS, §3.2.2) and video panoptic segmentation (VPS, §3.2.3).

### 3.2.1 (Instance-agnostic) Video Semantic Segmentation (VSS)

Extending the success of deep learning-based image semantic segmentation techniques to the video domain has become a research focus in computer vision recently. To achieve this, the most straightforward strategy is the naive application of an image semantic segmentation model in a frame-by-frame manner. But this strategy completely ignores temporal continuity and coherence cues provided in videos. To make better use of temporal information, research efforts in this field are mainly made along two lines.

- **Efforts towards More Accurate Segmentation.** A major stream of methods exploits cross-frame relations to boost the prediction accuracy. They typically first apply the very same segmentation algorithms to each frame independently. Then they add extra modules on top, e.g., optical flow-guided feature aggregation [212], [213], [219], and sequential network based temporal information propagation [216], to gather multi-frame context and get better results. For example, in some pioneer work [207], [209], after performing static semantic segmentation for each frame individually, optical flow [209] or 3D CRF [207] based post processing is applied for gaining temporally consistent segments. Later, [220] jointly learns CNN-based per-frame segmentation and CRF-based spatio-temporal reasoning. In [213], features warped from previous frames with optical flow are combined with the current frame features for prediction. These methods require additional feature aggregation modules, which increase the computational costs during the inference. Recently, [223] proposes to only incorporate flow-guided temporal consistency into the training phase, without bringing any extra inference cost. But its processing speed is still bounded to the adopted per-frame segmentation algorithms, as all features must be recomputed at each frame. For these methods, the utility in time-sensitive application areas, such as mobile and autonomous driving, is limited.

- **Efforts towards Faster Segmentation.** Yet another complementary line of work tries to leverage temporal information to accelerate computation. They approximate the expensive per-frame forward pass with cheaper alternatives, i.e., reusing the features in neighbouring frames. In [205], parts of segmentation networks are adaptively executed across frames, thus reducing the computation cost. Later methods use keyframes to avoid processing each frame, and then propagate the outputs or the feature maps to other frames. For instance, [214] employs optical flow to warp the features between the keyframe and non-key frames. Adaptive keyframe selection is later exploited in [211], [218], further enhanced by adaptive feature propagation [217]. In [221], Jain et al. use a large, strong model to predict the keyframe and use a compact one in non-key frames. Keyframe-based methods have different computational loads between keyframes and non-key frames, causing high maximum latency and unbalanced occupation of computation resources that may decrease system efficiency [224]. Additionally, the spatial misalignment of other frames with respect to the keyframes is challenging to compensate for and often leads to different quantity results between keyframes and non-key frames. In [231], a temporal consistency guided knowledge distillation technique is proposed to train a compact network, which is applied to all frames. In [224], several weight-sharing sub-networks are distributed over sequential frames, whose extracted shallow features are composed for final segmentation. This trend of methods indeed speeds

### Table 5: Summary of essential characteristics for reviewed VSS methods (§3.2.2). Flow indicates whether optical flow is used.

<table>
<thead>
<tr>
<th>Year</th>
<th>Method</th>
<th>Pub</th>
<th>Seg. Level</th>
<th>Core Architecture</th>
<th>Flow</th>
<th>Technical Feature</th>
<th>Training Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015</td>
<td>Ladvik</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN + SegNet</td>
<td>✓</td>
<td>Keyframe Selection</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>SFO</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>IJF</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td>2016</td>
<td>IJF</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>SFO</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>VT</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td>2017</td>
<td>FRC</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>PEAK</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>NetWrac</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>DFF</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>BBF</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>IF</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>LVS</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>DVM</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>EUN</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>GCRF</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td>2018</td>
<td>Asnchr</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>MDOT</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>MaskTrack-R-CNN</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>EPL</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td>2019</td>
<td>TDNet</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>MaskProp</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>VPS</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>MOTSNet</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>MVAE</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>ETC</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>Smpmask</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>STEmig</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>Naive-Stc</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td>2020</td>
<td>GameVL</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>TraDeS</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>SG-Net</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>ViTeR</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>SDO</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>Sano</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>VIPDeepLab</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>BBN</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>SemiNet</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>Propose-Reduce</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
<tr>
<td></td>
<td>CrossVS</td>
<td>CVPR</td>
<td>Semantic</td>
<td>FCN</td>
<td>✓</td>
<td>Temporal Feature Aggregation</td>
<td>Cityscapes [206]</td>
</tr>
</tbody>
</table>

Indicates whether optical flow is used.

Efforts towards Faster Segmentation. Yet another complementary line of work tries to leverage temporal information to accelerate computation. They approximate the expensive per-frame forward pass with cheaper alternatives, i.e., reusing the features in neighbouring frames. In [205], parts of segmentation networks are adaptively executed across frames, thus reducing the computation cost. Later methods use keyframes to avoid processing each frame, and then propagate the outputs or the feature maps to other frames. For instance, [214] employs optical flow to warp the features between the keyframe and non-key frames. Adaptive keyframe selection is later exploited in [211], [218], further enhanced by adaptive feature propagation [217]. In [221], Jain et al. use a large, strong model to predict the keyframe and use a compact one in non-key frames. Keyframe-based methods have different computational loads between keyframes and non-key frames, causing high maximum latency and unbalanced occupation of computation resources that may decrease system efficiency [224]. Additionally, the spatial misalignment of other frames with respect to the keyframes is challenging to compensate for and often leads to different quantity results between keyframes and non-key frames. In [231], a temporal consistency guided knowledge distillation technique is proposed to train a compact network, which is applied to all frames. In [224], several weight-sharing sub-networks are distributed over sequential frames, whose extracted shallow features are composed for final segmentation. This trend of methods indeed speeds
up inference, but still with the cost of reduced accuracy.
• Semi-/Weakly-supervised based Methods. Away from these main battlefields, some researchers made efforts to learn VSS under annotation efficient settings. In [215], classifier heatmaps are used to learn VSS from image tags only. [222], [234] use both labeled and unlabeled video frames to learn VSS. They propagate annotations from labeled frames to other unlabeled, neighboring frames [222], or alternatively train teacher and student networks with groundtruth annotations and iteratively generated pseudo labels [234].

3.2.2 Video Instance Segmentation (VIS)
In 2019, Yang et al. extended image instance segmentation to the video domain [82], which requires simultaneous detection, segmentation and tracking of instances in videos. This task is also known as multi-object tracking and segmentation (MOTS) [210]. Based on the patterns of generating instance sequences, existing frameworks can be roughly categorized into four paradigms: i) track-detect, ii) clip-match, iii) propose-reduce, iv) segment-as-a-whole. Track-detect methods detect and segment instances for each individual frame, followed by frame-by-frame instance tracking [82], [210], [228], [230], [232], [235]–[237], [252]. For example, in [82], [210], [253], Mask R-CNN [180] is adapted for VIS/MOTS by adding a tracking branch for cross-frame instance association. Alternatively, [237] models spatial attention to describe instances, tackling the task from a novel single-stage yet elegant perspective. Clip-match methods divide an entire video into multiple overlapped clips, and perform VIS independently for each clip through mask propagation [226] or spatial-temporal embedding [233]. Final instance sequences are generated by merging neighboring clips. Both of the two paradigms need two independent steps to generate a complete sequence. They both generate multiple incomplete sequences (i.e., frames or clips) from a video, and merge (or complete) them by tracking/matching at the second stage. Intuitively, these paradigms are vulnerable to error accumulation in the process of merging sequences, especially when occlusion or fast motion exists. To address these limitations, a propose-reduce paradigm is proposed in [244]. It first samples several key frames and obtains instance sequences by propagating the instance segmentation results from each key frame to the entire video. Then, the redundant sequence proposals of the same instances are removed. This paradigm not only discards the step of merging incomplete sequences, but also achieves robust results considering multiple key frames. However, these three types of methods still need complex heuristic rules to associate instances and/or multiple steps to generate instance sequences. The segment-as-a-whole paradigm [238] is more elegant; it poses the task as a direct sequence prediction problem using Transformer [192].

Almost all VIS models are built upon fully supervised learning, while [242], [243] are the exceptions. Specifically, in [242], motion and temporal consistency cues are leveraged to generate pseudo-labels from tag labeled videos for weakly supervised VIS learning. In [243], a semi-supervised embedding learning approach is proposed to learn VIS from pixel-wise annotated images and unlabeled videos.

3.2.3 Video Panoptic Segmentation (VPS)
Very recently, Kim et al. extended image panoptic segmentation to the video domain [227], which aims at a holistic segmentation of all foreground instance tracklets and background regions, and assigning a semantic label to each pixel. They adapt an image panoptic segmentation model [254] for VPS, by adding two modules for temporal feature fusion and cross-frame instance association, respectively. Later, temporal correspondence was explored in [240] through learning coarse segment-level and fine pixel-level matching. Qiao et al. [241] propose to learn monocular depth estimation and video panoptic segmentation jointly.

4 Video Segmentation Datasets
Several datasets have been proposed for video segmentation over the past decades. Fig. 3 shows example frames from twenty commonly used datasets. We summarize their essential features in Table 6 and give detailed review below.

4.1 VOS Datasets
4.1.1 AVOS/SVOS/IVOS Datasets
• Youtube-Objects is a large dataset of 1,407 videos collected from 155 web videos belonging to 10 object categories (e.g., dog, cat, plane, etc.). VOS models typically test the generalization ability on a subset [256] having totally 126 shots with 20,647 frames that provides coarse pixel-level fore-/background annotations on every 10th frames.
• FBMS$_{59}$ [36] consists of 59 video sequences with 13,860 frames in total. However, only 720 frames are annotated for fore-/background separation. The dataset is split into 29 and 30 sequences for training and evaluation, respectively.
• DAVIS$_{16}$ [17] has 50 videos (30 for train set and 20 for val set) with 3,455 frames in total. For each frame, in addition to high-quality fore-/background segmentation annotation, a set of attributes (e.g., deformation, occlusion, motion blur, etc.) are also provided to highlight the main challenges.
• DAVIS$_{18}$[81] contains 150 videos, i.e., 60/30/30/30 videos for train/val/test-dep/test-challenge sets. Its train and val sets are extended from the respective sets in DAVIS$_{16}$. There are 10,459 frames in total. DAVIS$_{17}$ provides instance-level annotations to support SVOS. Then, DAVIS$_{18}$ challenge[194] provides scribble annotations to support IVOS. Moreover, as the original annotations of DAVIS$_{17}$ are biased towards the SVOS scenario, DAVIS$_{19}$ challenge[177] re-annotates val and test-dep sets of DAVIS$_{17}$ to support AVOS.
• YouTube-VOS [95] is a large-scale dataset, which is split into a train (3,471 videos), val (507 videos), and test (541 videos) set, in its newest 2019 version. Instance-level precise annotations are provided every five frames in a 30FPS frame rate. There are 94 object categories (e.g., person, snake, etc.) in total, of which 26 are unseen in train set.
Remark. Youtube-Objects, FBMS$_{59}$ and DAVIS$_{16}$ are used for instance-agnostic AVOS and SVOS evaluation. DAVIS$_{17}$ is unique in comprehensive annotations for instance-level AVOS, SVOS as well as IVOS, but its scale is relatively small. YouTube-VOS is the largest one but only supports SVOS benchmarking. There also exist some other VOS datasets, such as SegTrackv1 $^{[45]}$ and SegTrackv2 $^{[76]}$, but they were less used recently, due to the limited scale and difficulty.

4.1.2 LVOS Datasets
• A2D Sentence $^{[196]}$ augments A2D $^{[257]}$ with phrases. It contains 3,782 videos, with 8 action classes performed by 7 actors. In each video, 3 to 5 frames are provided with segmentation masks. It contains 6,655 sentences describing acting and their actions. The dataset is split into 3,017/737 for train/test, and 28 unlabeled videos are ignored $^{[198]}$.
• J-HMDB Sentence $^{[196]}$ is built upon J-HMDB $^{[258]}$. It is comprised of 928 short videos with 928 corresponding sentences describing 21 different action categories. DAVIS$_{17}$-RVOs $^{[197]}$ extends DAVIS$_{17}$ by collecting referring expressions for the annotated objects. 90 videos from train and val sets are annotated with more than 1,500 referring expressions. They provide two types of annotations, which describe the highlighted object: 1) based on the entire video (i.e., full-video expression) and 2) using only the first frame of the video (i.e., first-frame expression).
• Refer-Youtube-VOS $^{[202]}$ includes 3,975 videos from YouTube-VOS $^{[98]}$, with 27,899 language descriptions of target objects. Similar to DAVIS$_{17}$-RVOs $^{[197]}$, both full-video and first-frame expression annotations are provided.

Remark. To date, A2D Sentence and J-HMDB Sentence are the main test-beds. However, the phrases are not produced with the aim of reference, but description, and limited to only a few object categories corresponding to the dominant actors’ performing a salient ‘action’ $^{[202]}$. But newly introduced DAVIS$_{17}$-RVOs and Refer-Youtube-VOS show improved difficulties in both visual and linguistic modalities.

4.2 VSS Datasets
• CamVis $^{[208]}$ is composed of 4 urban scene videos with 11-class pixelwise annotations. Each video is annotated every 30 frames. The annotated frames are usually grouped into 467/100/233 for train/val/test $^{[207]}$.
• CityScapes $^{[206]}$ is a large-scale VSS dataset for street views. It has 2,975/500/1,525 snippets for train/val/test, captured at 17FPS. Each snippet contains 30 frames, and only the 20$^{th}$ frame is densely labelled with 19 semantic classes. 20,000 coarsely annotated frames are also provided.
• NYUDv2 $^{[225]}$ contains 518 indoor RGB-D videos with high-quality ground-truths (every 10$^{th}$ video frame is labelled). There are 795 training frames and 654 testing frames being rectified and annotated with 40-class semantic labels.
• VSPW $^{[255]}$ is a recently proposed large-scale VSS dataset. It addresses video scene parsing in the wild by considering diverse scenarios. It consists of 3,536 videos, and provides pixel-level annotations for 124 categories at 15FPS. The train/val/test sets contain 2,806/343/387 videos with 198,244/24,502/28,887 frames, respectively.
• YouTube-VIS $^{[82]}$ is built upon YouTube-VOS $^{[95]}$ with instance-level annotations. Its newest 2021 version has 3,859 videos (2,985/421/453 for train/val/test) with 40 semantic categories. It provides 232K high-quality annotations for 8,171 unique video instances.
• KITTI MOTS $^{[210]}$ extends the 21 training sequences of KITTI tracking dataset $^{[259]}$ with VIS annotations – 12 for training and 9 for validation, respectively. The dataset contains 8,008 frames with a resolution of 375 × 1242, 26,899 annotated cars and 11,420 annotated pedestrians.
• MOTSChallenge $^{[210]}$ annotates 4 of 7 training sequences of MOTSChallenge$^{2017}$ $^{[260]}$. It has 2,862 frames with 26,894 annotated pedestrians and presents many occlusion cases.
• BDD100K $^{[229]}$ is a large-scale dataset with 100K driving videos (40 seconds and 30FPS each) and supports various tasks, including VSS and VIS. For VSS, 7,000/1,000/2,000 frames are densely labelled with 40 semantic classes for train/val/test. For VIS, 90 videos with 8 semantic categories are annotated by 129K instance masks – 60 training videos, 10 validation videos, and 20 testing videos.
• OVIS $^{[246]}$ is a new challenging VIS dataset, where object occlusions usually occur. It has 901 videos and 296K high-quality instance masks for 25 semantic categories. It is split into 607 training, 140 validation and 154 test videos.
• VIPER-VPS $^{[227]}$ re-organizes VIPER $^{[261]}$ into the video panoptic format. VIPER, extracted from the GTA-V game engine, has annotations of semantic and instance segmentations for 10 thing and 13 stuff classes on 254K frames of ego-centric driving scenes at 1080 × 1920 resolution.
• CityScapes-VPS $^{[227]}$ is built upon CityScapes$^{[206]}$. Dense panoptic annotations for 8 thing and 11 stuff classes for
5 PERFORMANCE COMPARISON

Next we tabulate the performance of previously discussed algorithms. For each of the reviewed fields, the most widely used dataset is selected for performance benchmarking. The performance scores are gathered from the original articles, unless specified. For the running speed, we obtain the FPS for most methods by running their codes on a RTX 2080Ti GPU. For a small set of methods whose implementations are not well organized or publicly available, we directly borrow the values from the corresponding papers. Despite this, it is essential to remark the difficulty when comparing runtime. As different methods are with different code bases and levels of optimization, it is hard to make completely fair runtime comparison [17, [63], [262]], the values are only provided for reference.

5.1 Object-level AVOS Performance Benchmarking

5.1.1 Evaluation Metrics

Presently, three metrics are frequently used [17] to measure how object-level AVOS methods perform on this task:

- **Region Jaccard** $\mathcal{J}$ is calculated by the intersection-over-union (IoU) between the segmentation results $\hat{Y} \in \{0, 1\}^{w \times h}$ and the ground-truth $Y \in \{0, 1\}^{w \times h}$: $\mathcal{J} = \frac{|\hat{Y} \cap Y|}{|\hat{Y} \cup Y|}$, which computes the number of pixels of the intersection between $\hat{Y}$ and $Y$, and divides it by the size of the union.

- **Boundary Accuracy** $\mathcal{F}$ is the harmonic mean of the boundary precision $P_e$ and recall $R_e$. The value of $\mathcal{F}$ reflects how well the segment contours $c(\hat{Y})$ match the ground-truth contours $c(Y)$. Usually, the value of $P_e$ and $R_e$ can be computed via bipartite graph matching [129], then the boundary accuracy $\mathcal{F}$ can be computed as: $\mathcal{F} = 2P_eR_e/(P_e + R_e)$.

- **Temporal Stability** $\mathcal{T}$ is informative of the stability of segments. It is computed as the pixel-level cost of matching two successive segmentation boundaries. The match is achieved by minimizing the shape context descriptor [264] distances between matched points while preserving the order in which the points are present in the boundary polygon. Note that $\mathcal{T}$ will compensate motion and small deformations, but not penalize inaccuracies of the contours [17].

5.1.2 Results

We select DAVIS16 [17], the most widely used dataset in AVOS, for performance benchmarking. Table 7 presents the results of those reviewed AVOS methods DAVIS16 val set. The current best solution, RTNet [107], reaches 83.6 region similarity $\mathcal{J}$, significantly outperforming the earlier deep learning-based methods, such as SFL [68], proposed in 2017.

5.2 Instance-level AVOS Performance Benchmarking

5.2.1 Evaluation Metrics

In instance-level AVOS setting, region Jaccard $\mathcal{J}$, boundary accuracy $\mathcal{F}$, and $\mathcal{J} \& \mathcal{F}$ – the mean of $\mathcal{J}$ and $\mathcal{F}$ – are used for evaluation [177]. Each of the annotated object trajectories will be matched with one of predicted tracklets according to $\mathcal{J} \& \mathcal{F}$, using bipartite graph matching. For a certain criterion, the final score will be computed between each ground-truth object and its optimal assignment.

5.2.2 Results

Regarding instance-level AVOS, we take into account DAVIS17 [81] in which the vast majority of methods are evaluated. From Table 8 we can find that UnOVOsT [178] is the top scorer, with 67.9 $\mathcal{J}$ at the time of this writing.
TABLE 10
Quantitative IVOS results on DAVIS$_{17}$ [81] in terms of AUC and $\mathcal{F}$.

<table>
<thead>
<tr>
<th>Method</th>
<th>AUC</th>
<th>$\mathcal{F}@60s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>FASTtmu [156]</td>
<td>63.3</td>
<td>67.6</td>
</tr>
<tr>
<td>GIEL [171]</td>
<td>73.0</td>
<td>76.1</td>
</tr>
<tr>
<td>MSN [165]</td>
<td>82.0</td>
<td>82.9</td>
</tr>
<tr>
<td>RMNet [173]</td>
<td>84.9</td>
<td>85.4</td>
</tr>
</tbody>
</table>

TABLE 11
Quantitative LVOS results on A2D Sentence [196] in terms of Precision@$K$, mAP and IoU.

<table>
<thead>
<tr>
<th>Method</th>
<th>mAP$@0.5$</th>
<th>IoU$@0.5$</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2DS [196]</td>
<td>50.0</td>
<td>23.1</td>
<td>-</td>
</tr>
<tr>
<td>CMSANet [204]</td>
<td>46.7</td>
<td>13.6</td>
<td>-</td>
</tr>
<tr>
<td>AAN [198]</td>
<td>55.7</td>
<td>20.7</td>
<td>6.3</td>
</tr>
<tr>
<td>VT-Capsule [206]</td>
<td>52.6</td>
<td>4.5</td>
<td>-</td>
</tr>
<tr>
<td>CDNet [199]</td>
<td>60.7</td>
<td>23.5</td>
<td>4.5</td>
</tr>
<tr>
<td>PolarRPE [200]</td>
<td>63.4</td>
<td>32.2</td>
<td>8.3</td>
</tr>
<tr>
<td>CST [203]</td>
<td>65.4</td>
<td>33.3</td>
<td>9.9</td>
</tr>
</tbody>
</table>

5.4 IVOS Performance Benchmarking

5.4.1 Evaluation Metrics

Area under the curve (AUC) and Jaccard at 60 seconds ($\mathcal{F}@60s$) are two widely used IVOS evaluation criteria [194].

- **AUC** is designed to measure the overall accuracy of the evaluation. It is computed over the plot Time vs Jaccard. Each sample in the plot is computed considering the average time and the average Jaccard for a certain interaction.

- **$\mathcal{F}@60$** measures the accuracy with a limited time budget (60 seconds). It is achieved by interpolating the Time vs Jaccard at 60 seconds. This evaluates which an IVOS method can obtain in 60 seconds.

5.4.2 Results

DAVIS$_{17}$ [81] is also widely used for IVOS performance benchmarking. Results summarized in Table 10 that the method proposed by Cheng et al. [189] is the top one.

5.5 LVOS Performance Benchmarking

5.5.1 Evaluation Metrics

As [196], overall IoU, mean IoU and precision are adopted.

- **IoU**: overall IoU is computed as total intersection area of all test data over the total union area, while mean IoU refers to average over IoU of each test sample.

- **Precision**: Precision@$K$ is computed as the percentage of test samples whose IoU scores are higher than a threshold $K$. Precision at five thresholds ranging from 0.5 to 0.9 and mean average precision (mAP) over 0.5:0.05:0.95 are reported.

5.5.2 Results

A2D Sentence [196] is arguably the most popular dataset in LVOS. Table 11 gives the results of six recent methods on A2D Sentence test set. It shows clear improvement trend from the first LVOS model [196] proposed in 2018, to recent complicated solution [203]. For runtime comparison, all the methods are tested on a video clip of 16 frames with resolution $512 \times 512$ and a textual sequence of 20 words.

5.6 VSS Performance Benchmarking

5.6.1 Evaluation Metrics

IoU metric is the most widely used metric in VSS. Moreover, in Cityscapes [206] – the gold-standard benchmark dataset in this field, two IoU scores, IoU$_{category}$ and IoU$_{class}$, defined over two semantic granularities, are reported. Here, ‘category’ refers to high-level semantic categories (e.g., vehicle, human), while ‘class’ indicates more fine-grained semantic classes (e.g., car, bicycle, person, rider). In total, [206] considers 19 classes, which are further grouped into 8 categories.

5.6.2 Results

Table 12 summarizes the results of eleven VSS approaches on Cityscapes [206] val set. As seen, EFC [223] performs the best currently, with 83.5% in terms of IoU$_{class}$.

5.7 VIS Performance Benchmarking

5.7.1 Evaluation Metrics

As in [82], precision and recall metrics are used for VIS performance evaluation. Precision at IoU thresholds 0.5 and 0.75, as well as mean average precision (mAP) over 0.5:0.05:0.95 are reported. Recall@N is defined as the maximum recall given N segmented instances per video. These two metrics are first evaluated per category and then averaged over the category set. The IoU metric is similar to region Jaccard $\mathcal{J}$ used in instance-level AVOS ($\mathcal{S}$).

5.7.2 Results

Table 13 gathers VIS results for on YouTube-VIS [82] val set, showing that Transformer-based architecture, i.e., VisTR [238], and redundant sequence proposal based solution Propose-Reduce [244], greatly improve the state-of-the-art.
TABLE 14
Quantitative VPS results on Cityscapes-VPS [227] (§5.8.2) test in term of VPQ. Each cell shows VPO\textsuperscript{k} / VPQ\textsuperscript{k}-Thing / VPQ\textsuperscript{k}-Stuff.

<table>
<thead>
<tr>
<th>Method</th>
<th>mAP\textsuperscript{0}</th>
<th>mAP\textsuperscript{1}</th>
<th>mAP\textsuperscript{5}</th>
<th>Recall\textsuperscript{0} @10%</th>
<th>Recall\textsuperscript{1} @10%</th>
<th>Recall\textsuperscript{5} @10%</th>
<th>FPS\textsuperscript{k}</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPS [227]</td>
<td>36.4 / 39.0 / 67.7</td>
<td>57.9 / 46.5 / 68.1</td>
<td>54.8 / 41.1 / 63.4</td>
<td>52.6 / 36.5 / 62.9</td>
<td>57.4 / 45.8 / 64.8</td>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td>SiamTrack (240)</td>
<td>63.8 / 94.9 / 66.6</td>
<td>58.2 / 47.2 / 65.9</td>
<td>56.0 / 45.2 / 64.4</td>
<td>54.7 / 40.2 / 63.2</td>
<td>57.8 / 47.5 / 65.0</td>
<td>4.5</td>
<td></td>
</tr>
<tr>
<td>ViP-DeepLab (241)</td>
<td>86.9 / 61.6 / 73.5</td>
<td>62.9 / 51.0 / 70.5</td>
<td>59.9 / 40.6 / 68.8</td>
<td>56.2 / 42.1 / 68.4</td>
<td>62.5 / 50.2 / 70.3</td>
<td>5.0</td>
<td></td>
</tr>
</tbody>
</table>

TABLE 13
Quantitative VIS results on YouTube-VIS [92] the val (§5.7.2) in terms of Precision@K, mAP, Recall@N and IoU.

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision@5</th>
<th>Precision@10</th>
<th>Precision@50</th>
<th><a href="mailto:mAP@0.5IoU">mAP@0.5IoU</a></th>
<th><a href="mailto:mAP@0.75IoU">mAP@0.75IoU</a></th>
<th>Precision@20</th>
<th>Precision@50</th>
<th>FPS\textsuperscript{k}</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBN [142]</td>
<td>27.2</td>
<td>61.2</td>
<td>12.3</td>
<td>13.6</td>
<td>10.3</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>MaskTrack R-CNN [92]</td>
<td>51.1</td>
<td>32.6</td>
<td>31.0</td>
<td>35.5</td>
<td>30.3</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SipMask [232]</td>
<td>53.0</td>
<td>33.3</td>
<td>33.5</td>
<td>36.9</td>
<td>32.5</td>
<td>24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STEm-Seg [233]</td>
<td>55.8</td>
<td>37.9</td>
<td>34.4</td>
<td>41.6</td>
<td>34.6</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CrossVIS [245]</td>
<td>57.3</td>
<td>39.7</td>
<td>36.0</td>
<td>42.0</td>
<td>36.6</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SemiTrack [241]</td>
<td>61.1</td>
<td>39.8</td>
<td>36.9</td>
<td>44.5</td>
<td>38.3</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MaskProp [226]</td>
<td>-</td>
<td>45.6</td>
<td>-</td>
<td>-</td>
<td>42.8</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompFeat [235]</td>
<td>56.0</td>
<td>38.6</td>
<td>33.1</td>
<td>40.3</td>
<td>35.3</td>
<td>117</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TraDiS [236]</td>
<td>52.6</td>
<td>32.8</td>
<td>29.1</td>
<td>36.6</td>
<td>32.6</td>
<td>26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SG-Net [227]</td>
<td>57.1</td>
<td>39.6</td>
<td>35.9</td>
<td>43.0</td>
<td>36.3</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ViTR [238]</td>
<td>64.0</td>
<td>45.0</td>
<td>38.3</td>
<td>44.9</td>
<td>40.1</td>
<td>58</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Propose-Reduce [244]</td>
<td>71.6</td>
<td>51.8</td>
<td>46.3</td>
<td>56.0</td>
<td>47.6</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.8 VPS Performance Benchmarking
5.8.1 Evaluation Metrics
In [227], the panoptic quality (PQ) metric used in image-based segmentation is modified as video panoptic quality (VPQ) to adapt to video panoptic segmentation.

- **VPQ:** Given a snippet $V^{t:t+k}$ with time window $k$, true positive (TP) is defined by $TP = \{(u, \hat{u}) \in U \times \hat{U} : IoU(u, \hat{u}) > 0\}$ where $U$ and $\hat{U}$ are the set of the ground-truth and predicted masks, respectively. False Positives (FP) and False Negatives (FN) are defined accordingly. After accumulating $TP_c$, $FP_c$, and $FN_c$ on all the clips with window size $k$ and class $c$, we define $VPQ = \frac{1}{N_{t=0}} \sum_{c=0}^{C-1} \sum_{i=1}^{N_t} \frac{TP_c}{TP_c + \frac{1}{2} FP_c + \frac{1}{2} FN_c}$, when $k=1$, $VPQ^1$ is equivalent to PQ. For evaluation, $VPQ^k$ is reported over $k \in \{0, 5, 10, 15\}$ and finally, $VPQ = \frac{1}{5} \sum_{k=0}^{5} VPQ^k$.

5.8.2 Results
Cityscapes-VPS [227] is chosen for testing VPS methods. As shown in Table 14, ViP-DeepLab [241] is the top one.

5.9 Summary
From the results, we can draw several conclusions. The most important of them is related to reproducibility. Across different video segmentation areas, many methods do not describe the setup for the experimentation or do not provide the source code for implementation. Some of them even do not release segmentation masks. Moreover, different methods use various datasets and backbone models. These make fair comparison impossible and hurt reproducibility.

Another important fact discovered thanks to this study is the lack of information about execution time and memory use. Many methods particularly in the fields of AVOS, LVOS, and VPS, do not report execution time and almost no paper reports memory use. This void is due to the fact that many methods focus only on accuracy without any concern about running time efficiency or memory requirements. However, in many application scenarios, such as mobile devices and self-driving cars, computational power and memory are typically limited. As benchmark datasets and challenges serve as a main driven factor behind the fast evolution of segmentation techniques, we encourage organizers of future video segmentation datasets to give this kind of metrics its deserved importance in benchmarking.

Finally, performance on some extensively studied video segmentation datasets, such as DAVIS16 [17] in AVOS, DAVIS17 [81] in SVOS, A2D Sentence [196] in LVOS, have nearly reached saturation. Though some new datasets are proposed recently and claim huge space for performance improvement, the dataset collectors just gather more challenging samples, without necessarily figuring out which exact challenges have and have not been solved.

6 Future Research Directions
Based on the reviewed research, we list several future research directions that we believe should be pursued.

- **Long-Term Video Segmentation:** Long-term video segmentation is much closer to practical applications, such as video editing. However, as the sequences in existing datasets often span several seconds, the performance of VOS models over long video sequences (e.g., at the minute level) are still unexamined. Bringing VOS into the long-term setting will unlock new research lines, and put forward higher demand of the re-detection capability of VOS models.

- **Open World Video Segmentation:** Despite the obvious dynamic and open nature of the world, current VSS algorithms are typically developed in a closed-world paradigm, where all the object categories are known as a priori. These algorithms are often brittle once exposed to the realistic complexity of the open world, where they are unable to efficiently adapt and robustly generalize to unseen categories. For example, practical deployments of VSS systems in robotics, self-driving cars, and surveillance cannot afford to have complete knowledge on what classes to expect at inference time, while being trained in-house. This calls for smarter VSS systems, with a strong capability to identify unknown categories in their environments [265].

- **Cooperation across Different Video Segmentation Sub-fields:** VOS and VSS face many common challenges, e.g., object occlusion, deformation, and fast motion. Moreover, there are no precedents for modeling these tasks in a unified framework. Thus we call for closer collaboration across different video segmentation sub-fields.

- **Annotation-Efficient Video Segmentation Solutions:** Though great advances have been achieved in various video segmentation tasks, current top-leading algorithms are built on fully-supervised deep learning techniques, requiring a huge amount of annotated data. Though semi-supervised, weakly supervised and unsupervised alternatives were explored in some literature, annotation-efficient solutions receive far less attention and typically show weak performance, compared with the fully supervised ones. As the high temporal correlations in video data can provide additional cues for supervision, exploring existing annotation-efficient techniques in static semantic segmentation in the area of video segmentation is an appealing direction.

- **Adaptive Computation:** It is widely recognized that there exist high correlations among video frames. Though such data redundancy and continuity are exploited to reduce...
the computation cost in VSS, almost all current video segmentation models are fixed feed-forward structures or work alternatively between heavy and light-weight modes. We expect more flexible segmentation model designs towards more efficient and adaptive computation [266], which allows network architecture change on-the-fly – selectively activating part of the network in an input-dependent fashion.

- **Neural Architecture Search:** Video segmentation models are typically built upon hand-designed architectures, which may be suboptimal for capturing the nature of video data and limit the best possible performance. Using neural architecture search techniques to automate the design of video segmentation networks is a promising direction.

### 7 CONCLUSION

To our knowledge, this is the first survey to comprehensively review recent progress in video segmentation. We provided the reader with the necessary background knowledge and summarized more than 150 deep learning models according to various criteria, including task settings, technique contributions, and learning strategies. We also presented a structured survey of 20 widely used video segmentation datasets and benchmarking results on 7 most widely-used ones. We discussed the results and provided insight into the shape of future research directions and open problems in the field. In conclusion, video segmentation has achieved notable progress thanks to the striking development of deep learning techniques, but several challenges still lie ahead.

### References


P. Hu, F. Caba, O. Wang, Z. Lin, S. Sclaroff, and F. Perazzi, “Temporally distributed networks for fast video semantic seg-


