Unidentified Video Objects: A Benchmark for Dense, Open-World Segmentation

Weiyao Wang Matt Feiszli Heng Wang Du Tran
Facebook AI Research
{weiyaowang,mdf,hengwang,trandu}@fb.com

Abstract

*Current state-of-the-art object detection and segmentation methods work well under the closed-world assumption. This closed-world setting assumes that the list of object categories is available during training and deployment. However, many real-world applications require detecting or segmenting novel objects, i.e., object categories never seen during training. In this paper, we present, UVO (**U**nidentified **V**ideo **O**bjects), a new benchmark for open-world class-agnostic object segmentation in videos. Besides shifting the focus to the open-world setup, UVO is significantly larger, providing approximately 6 times more videos compared with DAVIS, and 7 times more mask (instance) annotations per video compared with YouTube-VO(I)S. UVO is also more challenging as it includes many videos with crowded scenes and complex background motions. We also demonstrated that UVO can be used for other applications, such as object tracking and super-voxel segmentation. We believe that UVO is a versatile testbed for researchers to develop novel approaches for open-world class-agnostic object segmentation, and inspires new research directions towards a more comprehensive video understanding beyond classification and detection.*

1. Introduction

During daily activities, humans routinely encounter novel objects, *e.g.*, unfamiliar birds or unknown flowers; despite this unfamiliarity, people have no problem perceiving them as distinct object instances. Even cinematic examples like UFOs will be identified as independent things. Many real-world applications such as object search [27, 30], instance registration [50], human-object interaction modeling [14] and human activity understanding [6] require such open-world prediction abilities, *e.g.*, exhaustively detecting or segmenting objects (both known and unknown), to accomplish their tasks. Open-world is also the natural setting for applications like embodied AI (*e.g.*, robotics, autonomous driving) and augmented-reality assistants, which will encounter novel situations regularly.

(a) (b) (c)

Figure 1: **State-of-the-art object detection/segmentation methods do not work well in open-world settings.** We evaluated (a) Mask R-CNN trained on COCO and (b) Google AI cloud API on UVO videos and found both methods fail to segment many objects that have not been seen in training. (c) Real-world applications require segmenting all objects that appear in the videos, even unseen objects. Mask R-CNN works well only on predefined categories and fails to recognize objects (*e.g.*, barbell) or confuses non-object with objects in the taxonomy (refrigerator). Google cloud object detector offers stronger detection results but still misses all gym equipment in the background. (c) UVO is designed to detect/segment all objects regardless of the categories and beyond.

In contrast, the current state-of-the-art methods are designed for closed-world detection and segmentation. Mask R-CNN [17], when trained on COCO [24] with a closed-world taxonomy of 80 object categories, cannot segment novel objects (see Figure 1a); the model can only be expected to segment the classes it is trained for. Similarly, an industry publicly-available object detector (capable of detecting 550+ objects) [1] still cannot detect many objects (see Figure 1b). From a methodology perspective, open-world object segmentation is a challenging problem due to its open-taxonomy nature. Though extensive research has been done to develop either supervised top-down approaches [17, 34, 33] or unsupervised bottom-up approaches [39, 21, 15] for object detection/segmentation in the closed-world setting, we find that simple modifications of existing approaches do not work well for the open-world segmentation problem. On one hand, adopting a top-down approach, *e.g.*, Mask R-CNN, to class-agnostic object segmentation by replacing its multi-class loss with a

binary foreground versus background loss performs poorly on unseen classes (as shown in Table 6). This is because the top-down approaches are strongly biased toward contextual cues from seen classes [35]. On the other hand, unsupervised approaches, *e.g.*, GBH [15], rely on pixel grouping using local cues, *e.g.*, colors and/or motions, which have no notion about semantic object boundaries, thus also perform poorly (see Figure 8). We believe that open-world object segmentation is a challenging yet important problem that needs to be addressed by approaches that are conceptually different from existing methods.

Open-world object segmentation also provides opportunities for long video modeling and more complex prediction tasks. Current video understanding approaches [36, 5, 38, 12, 37] cannot scale well to long videos due to GPU memory constraints and are not designed for complex prediction tasks beyond classification and detection. Grouping pixels into semantic entities (including unknown classes) can provide plausible alternatives for long-term video modeling and flexible prediction tasks, *e.g.*, reasoning about objects and their interactions, even when their types are unknown, possibly by applying graph convolutional networks [42, 43], knowledge graphs [49, 13], or attention-based models [41, 9] on top of entities instead of pixel-based CNNs which are memory-intensive.

Due to its broad applications, open-world object segmentation is an important problem to study. Current datasets are often constructed with predefined taxonomy in a closed-world manner. Removing object labels in existing datasets will not make them suitable for open-world segmentation: to evaluate detectors in open-world, a dataset needs to contain exhaustive annotations on all objects; otherwise, a model detecting un-annotated objects is not rewarded and can even be penalized. Ideally, the dataset should be annotated in a bottom-up fashion: annotators watch videos, spot and mask all objects. To our knowledge, no existing dataset provides such exhaustive annotations at scale for videos or images [1]. Bottom-up annotation pipelines are absent in videos and are rare in images [2]. In this paper, we make the first step forward in addressing this problem by constructing a new benchmark for open-world object segmentation, and providing a comprehensive set of baselines with in-depth analysis to understand the benchmark and problem. Our contributions are:

- A method for constructing open-world object segmentation datasets using object interpolation and tracking, which is **4x** more efficient than the baseline.
- We introduce, **U**nidentified **V**ideo **O**bjects (UVO), a new benchmark for open-world class-agnostic object segmentation. UVO focuses on open-world and has approximately **6x** more videos compared with DAVIS [29], and **7x** more mask (instance) annotations per video compared with YouTube-VO(I)S [47, 48].
- We provide a comprehensive set of baselines to understand our proposed tasks and benchmark. We believe that UVO is a versatile testbed for open-world object segmentation and will inspire new research directions toward more complex video understanding tasks beyond classification and detection.

[1] LVIS [16] is a federated dataset of multiple small shards; each contains exhaustive annotations w.r.t. a subset of classes within the shard but not inter-shard: *e.g.*, "person" is not annotated in many images.

[2] LVIS is bottom-up in federated setting; ADE20k [51] is smaller scale.

2. Related Work

Open-world object recognition and detection. Open-world problems have been studied in the context of recognition [3, 25]: given a closed-world training dataset, how to actively identify new object categories? To handle novel objects, previous works explicitly differentiate unknown from known, such as by spotting outliers in the embedding space. There are also previous studies on open-world object mask prediction. Pinheiro *et al*. [31] trained a class-agnostic mask predictor. Hu *et al*. [19] proposed an approach for predicting unknown object masks using known object bounding boxes. More recently, Jaiswal *et al*. [20] proposed an adversarial framework to learn out-of-taxonomy objects. Dhamija *et al*. [8] discussed the difficulties in open-world object detection. Despite the aforementioned works, we still lack a dedicated dataset on this topic. This inspires us to create UVO to facilitate more extensive research efforts on open-world object detection and segmentation.

Related datasets. Object detection and segmentation have been the focus of computer vision for the past decades. Much of the progress we have achieved so far is built upon pioneering datasets: BSDS [26], Caltech101 [23], PASCAL-VOC [11], COCO [24], ADE20k [51], LVIS [16], *etc*. Many recent datasets extend the task from image to video: DAVIS [29], YouTube-VOS (YTVOS) [47], MOTS [40], YouTube-VIS (YTVIS) [48], TAO [7]. UVO is inspired by the aforementioned datasets, but have several key features differentiating itself from previous ones. Existing datasets typically rely on fixed taxonomies, such as "salient objects" (DAVIS) or a list of object categories (YTVOS/YTVIS) (see Table 1). On the contrary, UVO is taxonomy-free and provides exhaustive annotations for all the objects in the open world. As a result, our dataset contains significantly more instances per video compared to previous video object segmentation datasets.

3. Open-World Object Segmentation

Compared with the traditional object segmentation task, the open-world setup requires the model to segment all the entities or objects *class-agnostically* and *exhaustively*. These requirements ensure the model detects unseen cat-

Dataset	Videos / Frames	Taxonomy	Objects per video	Ann. fps
DAVIS [32]	150/11k	"salient"	2.99	24
YTVOS [47]	4453/120k	94 classes	1.64	6
YTVIS [48]	2883/78k	40 classes	1.68	6
$\text{UVO}_{\mathcal{D}}$	1024/93k	open-world	13.52	30
$\text{UVO}_{\mathcal{S}}$	10337/31k			1

Table 1: **Compare UVO with popular datasets.** UVO is the largest compared with popular datasets in terms of the number of annotated frames and object masks. UVO has no predefined taxonomy, but all objects are exhaustively annotated, resulting in a significantly larger number of annotated objects per video.

egories during testing and thus can potentially learn more generic representations of the visual world. We detail the process of building the dataset and provide an in-depth analysis about its characteristics in the following sections.

3.1. An overview of UVO

We introduce *UVO* for open-world object segmentation which contains real-world videos with exhaustive object mask annotations. Video object masks are costly to annotate, and no existing dataset provides such annotations at large-scale. Real-world videos often contain dozens of object instances (see Figure 2), and it takes a trained annotator 45 minutes per frame to densely annotate all the object masks and link them across frames. We build a semi-automatic approach to accelerate the process and reduce the annotation cost. This section provides an overview of UVO.

Since annotators are asked to mask out **all** objects, UVO is more exhaustively annotated compared with existing dataset (Table 1). Different from YouTube-VOS and DAVIS, we do not specify which object to annotate. Unlike YouTube-VIS, we do not have a predefined taxonomy for objects. As a result, UVO provides 13.52 object annotations per video on average, which is 7x more than Youtube-VIS and Youtube-VOS and 4x more than DAVIS. The number of objects per video follows a long-tail distribution shown in Figure 2. In some extreme cases, the number of object instances in a video can be over 100 (Figure 2).

UVO consists of two subsets: $\text{UVO}_{\mathcal{S}}$ containing 10,337 videos sparsely annotated at 1fps and $\text{UVO}_{\mathcal{D}}$ containing 1,024 videos densely annotated at 30fps. $\text{UVO}_{\mathcal{S}}$ is used for frame-level open-world segmentation while $\text{UVO}_{\mathcal{D}}$ is used for video-level open-world segmentation and tracking. $\text{UVO}_{\mathcal{S}}$ can optionally be used for pre-training video-level models (Table 4). We further divide the two subsets into train, validation, and test splits. To simulate the open-world setting, we use the 400 labeled video action classes to define splits (202 for training, 101 for validation, and 97 for test). We release the train and validation splits with ground truth to facilitate research and development, release the test split without ground truth, and provide a server for evaluation.

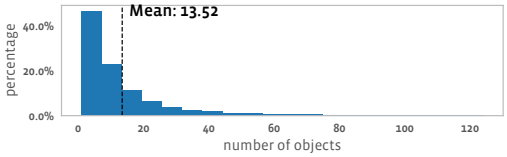


Figure 2: **Distribution of the number of objects per video**. The distribution is long-tail, with a mean of 13.52 and a median of 8. In some extreme cases, we observe over 100 object instances in a videos.

3.2. Dataset annotation

Data source. We adopt videos from Kinetics-400 [22] for UVO, which contains 10-second 30fps YouTube video clips labeled with human actions. There are several advantages of using Kinetics videos. First, Kinetics videos are human-centric and contain diverse sets of human actions and human-object interactions. In addition, Kinetics videos are sampled from YouTube with a wide variety of sources: by both professionals and amateurs, by both cameras and mobile devices, third-person or egocentric. Furthermore, Kinetics videos cover very diverse object categories, appearances, and motions, including challenging real-world cases with occlusion and camera motion.

Annotation guideline. Since the definition of "object" is ambiguous, we follow the common approach in object proposal literature: [45, 10, 53, 39]: objects are defined as things not belonging to background or stuff. We clarify the definition of background and stuff with annotators through examples: grass, sky, floor, *etc*. The difference between objects and stuff has been discussed previously [18, 2, 4]. On granularity, we ask the annotators to choose the coarse end of the object definition: err towards the coarsest possible segmentation that produces meaningful segments. We identified a few major ambiguities during the annotator training process and addressed them individually:

- Group of objects (connected objects). A group of objects could be marked as one if they stay together through the whole video, such as a stack of bowling balls and a crowd of static people. If an object leaves the group, the object needs to be segmented on its own.
- Accessories of humans. An object may have been constantly connected to a human throughout the video. We use a criteria on interaction to decide when to split or merge. For example, in Figure 1, a person working out with a barbell, both the barbell and the sneakers are connected to the person. The person is interacting with the barbell, so it is segmented as an individual object; while the sneakers remain static on the person's feet, so they are annotated as the person.
- Objects in the mirror. We make it explicit that mirrored objects by a reflective surface are not annotated.

With the guidelines provided, annotators maintained a consistent definition on objects. As UVO videos are not an-

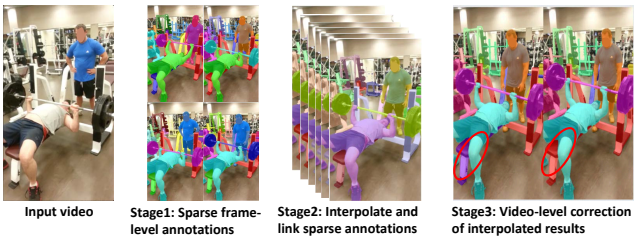


Figure 3: **Annotation pipeline overview**. We propose a semi-automated pipeline to accelerate the annotation process. We first annotate the videos sparsely (*e.g.*, 1fps), then propagate the masks to the next frames densely. The propagated masks are then corrected by annotators.

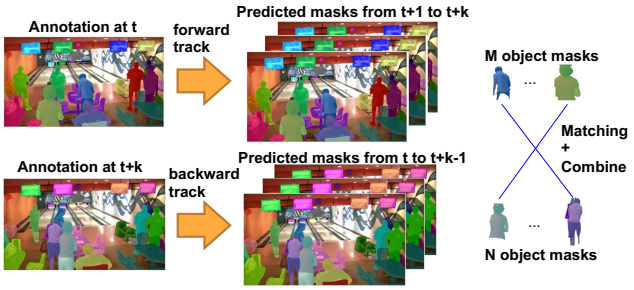


Figure 4: **Propagating masks from sparsely annotated frames**. We interpolate all the object masks between two sparsely annotated frames t and $t+k$. For an un-annotated frame, we generate its annotations by forward and backward tracking the annotations from frame t and $t+k$, matching and combining these two sets of annotations.

notated with category labels, our object masks can be considered as "anonymous-semantic" objects. Our open-world object segmentation task can be interpreted as grouping pixels/voxels into "anonymous-semantic" objects or entities.

Annotation frequency. Annotating object masks is time consuming. On average, each frame takes 16.3 minutes to annotate without linking the object masks over time. To make the annotation more tractable, UVO is split into a dense set, *i.e.*, UVO$_\mathcal{D}$, with videos densely annotated at 30fps and a sparse set, UVO$_\mathcal{S}$, with videos sparsely annotated at 1fps. For UVO$_\mathcal{D}$, we randomly select 3-second clips from 1200 videos sampled from Kinetics validation set with considerations to balance each action class. We remove videos with no obvious object or too many shot changes, and end up with 1,024 videos. This sums up to 93k frames with temporally dense annotations. For UVO$_\mathcal{S}$, we sample 12k videos from Kinetics training set, remove videos following the same procedure as UVO$_\mathcal{D}$, and annotate them sparsely at 1fps. We end up with 10.3k videos of 31k frames annotated. Table 1 summarizes the statistics of UVO.

Quality assurance. We perform a two-stage quality assurance check: frame-level and video-level. On frame-level, auditors (experts) are asked to check whether the mask quality is high, *e.g.*, examining object boundaries. Frames with poor annotations found by the auditors are corrected by the annotators. On video-level, we render each individual object mask on the video, which are checked by 5 annotators for temporal consistency. Videos including any object mask without majority votes of the 5 annotators are sent back for correction.

Scratch-frame	Scratch-video	Copy-paste	Our pipeline
16.3	45.0	30.7	11.0

Table 2: **Annotation time per frame (in minutes) in different settings.** Both scratch-frame and scratch-video have no initialization of object masks. Scratch-video requires linking objects across the frames, whereas scratch-frame does not. Copy-paste copies the masks from the sparsely annotated frames as an initialization for dense annotations. With our proposed pipeline, we are able to cut down annotation time by 4x.

3.3. Efficient annotation using mask propagation

As shown in Table 2, it takes 45 minutes per frame to annotate all object masks and link them across frames. To speed up the annotation, we detail our proposed semi-automated annotation pipeline (Figure 3) in this section.

Sparse frame-level segmentation. Given a video, annotators preview the clip and annotate all the masks at 1 fps. We only ask the annotators to associate each object mask with a unique index and do not require them to link masks over time. This significantly reduces the annotation time from 45 minutes/frame to 16.3 (scratch-frame in Table 2).

Propagating sparsely annotated masks. We interpolate the sparsely annotated masks to all the frames. To this end, we use Space-Time Memory Network (STM) [28] to track the object masks through frames. For each un-annotated frame, we consider both forward and backward tracking, *i.e.*, tracking from the closest earlier frame and the closest later frame. Since objects in sparse annotations may not be linked, forward and backward tracking of the same frame may not match. We formulate a maximal bipartite matching problem (*i.e.*, mapping M forward tracked objects to N backward tracked objects) and solve it with Hungarian Matching using cues including overlap (IoU), mask size, object color histogram and object center distance. To combine the forward and backward predictions, we weight their logits by their temporal distances to the un-annotated frame. Our pipeline is summarized in Figure 4.

Video-level correction We send interpolated masks to annotators for manual correction. To assist the annotation, we present both the target frame (frame to correct) and the closest frame with sparse annotations. This helps to guide the annotator, ensure temporal consistency and correct linking errors. The proposed pipeline significantly reduces annotation time from 45 (naive frame annotation and linking) and 30.7 (copy-paste) to 11 minutes per frame (Table 2).

Potential algorithmic bias by STM Since we use STM to accelerate our annotation framework (Figure 3), we conduct the following experiment to understand if there is any

Figure 5: **Examples of UVO**. UVO videos are exhaustively annotated with masks regardless of object categories. UVO features a wide-range of videos (*e.g.*, third-person/egocentric, professional/amateur, crowded/sparse objects) making it a challenging benchmark. Best viewed in color.

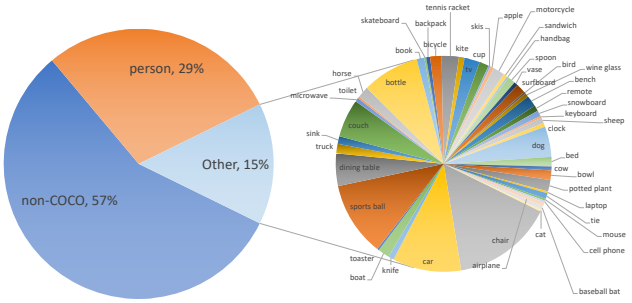


Figure 6: **Distribution of object categories**. Our dataset contains 57% objects not belonging to any of the 80 COCO categories. Due to the nature of Kinetics as a human action recognition dataset, our dataset contains 29% instances labeled as person. Our objects cover 51 COCO categories.

bias caused by STM. We replace STM by copy-paste in our annotation pipeline while keeping all other components the same (annotation time is presented in Table 2). We compare these masks (copy-paste pipeline) with UVO masks (STM pipeline) to understand the algorithmic bias. Given two sets of annotated masks (on the same set of videos), we evaluate STM and found minimal difference in performance: $\mathcal{J}$-score is $+0.2$ (74.3 for copy-pasting vs. 74.5 of UVO), $\mathcal{F}$-score is -0.4 (79.1 for copy-pasting vs. 78.7 of UVO). This indicates that there is no significant difference between UVO masks and copy-pasting masks. We note that using our pipeline with copy-pasting is very close to manually annotate every frame. We, therefore, believe that UVO has very little algorithmic bias/ advantage towards STM.

3.4. Look into UVO

Open-worldness Previous datasets use predefined taxonomies, *e.g.*, MSCOCO has 80 object categories. It is natural to ask how well a predefined taxonomy can cover real-world scenarios. To understand the open-worldness of UVO, we further label the object instances with 80 COCO categories and one additional category to catch all non-COCO classes with a random subset of 300 videos from UVO. Figure 6 presents the distribution of object categories in UVO. Since Kinetics focuses on human actions, human covers 29 % of object instances. Besides human, about 15 % of object instances are includes in the COCO taxonomy. 57 % object instances do not belong to any of the 80 COCO classes. This indicates a limited coverage of a well predefined taxonomy. Many non-COCO objects are less common but are not rare, *e.g.*, ski poles, cable crossover, pliers, dog leashes, pain patch, dried seaweed. Many of these are not covered by 1.2k LVIS [16] taxonomy neither. There are also many truly "unknown" objects: one can spot the object but fails to identify what the object is.

Diverse objects and camera motions. Besides object categories, there are other attributes that can impact the performance of a video object segmentation algorithm. In the image domain, datasets are often evaluated by dividing objects into different sizes (large, medium and small). For objects in video, motion is an important attribute. We analyze two types of motion: camera motion and object motion. We extract camera motion (rotation and translation) with an off-the-shelf camera pose estimator [52], and find YTVIS and UVO have a similar distributions of camera motions.

For an object instance, we decouple its motion into two types: disappear/ appear and independent object motion. We study the lifespan of an object and compare with YTVIS (Figure 7a). UVO features a wider distributions of object lifespans (more frequent disappearing and appearing), while YTVIS focuses on objects appeared throughout the entire video. For object motion, we compute pair-wise mask IoU between two timestamps (Figure 7b). Our dataset has a broader distribution on maskIoU and on average has smaller

IoU (larger motion). We further decouple maskIoU into size-change (*e.g.*, occlusion, shrinking/expanding) and velocity (distance between object mask center). UVO provides a larger range of motions and on average has more significant motion, shown in Figure 7c,d.

4. Experiments

4.1. Baselines and implementation details

We use UVO$_\mathcal{S}$ for frame-level segmentation and UVO$_\mathcal{D}$ for video-level segmentation and tracking experiments, unless stated otherwise.

Frame-level segmentation: Mask R-CNN [17]. Mask R-CNN is a high-performance two-stage model with an RPN [34] to generate object proposals and an ROIAlign operation on top of the proposed bounding boxes for classification, box regression, and mask segmentation. Following the common practices, we use COCO17 [24] for pre-training. We also include experiments on LVIS [16], an expansion of COCO with 1.2k categories.

Video-level segmentation: MaskTrack R-CNN [48]. Besides detecting and segmenting objects in each frame, video models also need to correctly link objects across the frames and predict their space-time masks. MaskTrack R-CNN adds an additional tracking head on top of Mask R-CNN, and combines tracking head predictions, class assignments and mask IoU to link objects. We use YouTube-VIS [48] (YTVIS) dataset for pre-training and inference. Since YTVIS does not release its validation set annotations and its validation server does not provide class-agnostic evaluation, we split its training data into a training split (1938 videos) and a held-out validation split (300 videos). For training with UVO$_\mathcal{S}$, we generate 15fps pseudo-groundtruth through interpolation using STM.

Bottom-up segmentation: Hierarchical Graph-Based Video Segmentation (GBH) [15]. Besides the aforementioned top-down methods [17, 48], bottom-up approaches can also be used for unsupervised object segmentation. We adopt the GBH super-voxel algorithm as a baseline. GBH builds a hierarchy of segmentations by progressively grouping similar (super-)voxels. GBH is a non-parametric approach, and we show that in an open-world scenario, it can serve as a competitive baseline.

Object tracking: Space-Time Memory network (STM) [28]. Tracking (a.k.a semi-supervised video object segmentation) aims at segmenting objects at video level with ground-truth masks provided at the first frame. We adopt the state-of-the-art STM as a baseline on UVO$_\mathcal{D}$. STM uses memory to store features from previous frames and attention to match the current and previous frames for tracking and segmenting objects.

Implementation details. We use the popular object detection framework Detectron2 [44] for experiments with class-agnostic Mask R-CNN and MaskTrack R-CNN. We use the open-sourced inference model from [28] for the tracking experiments. For super-voxel, we re-implement GBH for better efficiency following the original paper [15] and LIBSVX [46]. All models are trained with synchronous SGD with momentum using 8 GPUs. Hyper-parameters follow the settings in the original paper of each model. We use average precision (AP) and average recall (AR) at 100 proposals to measure the performance of frame and video level segmentation, and $\mathcal{J}$-score (IoU, region-similarity) and $\mathcal{F}$-score (contour accuracy) for tracking.

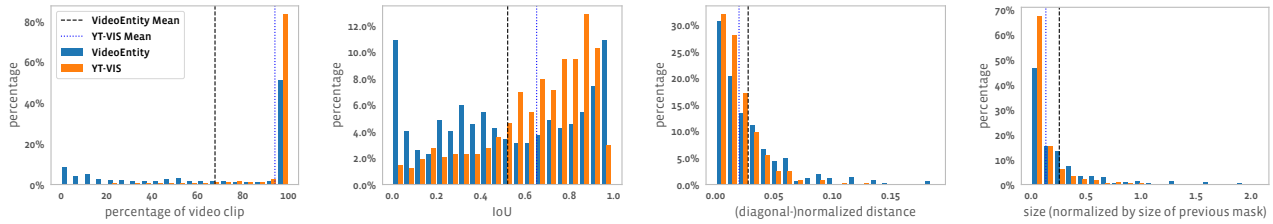
4.2. Results and analysis

UVO is compatible and complementary with existing datasets. To show the complementary and compatibility of UVO with related instance segmentation datasets, such as COCO and YTVIS, we (pre-)train models on related datasets and optionally finetune on UVO, and cross-evaluate on the datasets (Table 3,4). On both frame and video level evaluations, UVO is more challenging: models trained on COCO or YTVIS suffer from a significant performance drop. Finetuning models on UVO offers good gains, but is still lower than the performance of previous datasets. A larger taxonomy (LVIS) also performs poorly on UVO, and pre-training on LVIS performs slightly worse than COCO, possibly due to lower performance on "person" class (similar results are found in [7]). On video instance segmentation, finetuning without YTVIS gives better results than finetuning with YTVIS, possibly due to the smaller taxonomy and object sparsity in YTVIS.

UVO is also compatible with previous datasets. Finetuning on UVO offers 8% gain on AR$_{100}$ when evaluating on COCO (Table 3), and only suffers a small drop (4.2%) on YTVIS (Table 4). Note that a major portion of YTVIS instances are non-human animals (eg. ape, leopard), which are rare or even absent in Kinetics400 videos. In addition, UVO$_\mathcal{S}$ is an alternative to the commonly used COCO pre-training: pretraining on UVO$_\mathcal{S}$ offers competitive performance on UVO and YTVIS.

On tracking, we evaluate STM on UVO$_\mathcal{D}$ and compare with DAVIS16 (single-object) [29], DAVIS17 [32] (multi-object) and YouTube-VOS [47] (YTVOS) (Table 5). UVO is notably harder than existing datasets: performance is lower except for $\mathcal{J}$-score for unseen objects in YTVOS.

Open-world detection and segmentation are challenging. We evaluate the effect of open-world vs. closed-world in detection and segmentation. To simulate an open-world effect on COCO and YTVIS, we follow the common practice [31] by splitting the classes into two sets: classes overlapped with VOC [11] and classes unique to COCO/YTVIS (non-VOC). By training only on VOC classes and test on all or non-VOC classes, we can understand how well a detector performs on unseen (open-world)



(a) Object lifespan. (b) Object Mask IoU every 5 frames. (c) Object center distance every 5 frames. (d) Object size change every 5 frames.

Figure 7: **Comparing object motion statistics with YouTube-VIS.** UVO has a wider object lifespan (a), which indicates more frequent object appearing and disappearing. UVO objects have a more diverse distributions of motion magnitudes (bcd), also larger motions on average .

Train	AR	AP	AR	AP	AR	AP
	COCO		UVO$_S$val		UVO$_S$test	
COCO	49.6	37.0	41.0	19.5	36.7	19.0
UVO$_S$	47.8	17.4	37.2	17.8	34.4	17.4
COCO+UVO$_S$	57.6	27.6	44.9	25.0	41.4	23.6
	LVIS		UVO$_S$val		UVO$_S$test	
LVIS	37.3	15.4	33.8	5.5	29.2	4.3
LVIS + UVO$_S$	21.6	4.2	43.1	22.2	39.4	21.1

Table 3: **UVO frame-level results of Mask R-CNN, cross-evaluated on COCO and LVIS.** Because UVO contains many objects not in COCO categories, performance of model trained on COCO, when tested on UVO$_S$, drops significantly. Finetuning on UVO$_S$ improves performance by 4–5%, though still lower than training and evaluating on only COCO. We note that finetuning on UVO$_S$ significantly improves recall (+8%) on COCO. Models trained on a large-taxonomy (LVIS) also performs poorly on UVO$_S$.

Train	AR	AP	AR	AP	AR	AP
	YTVIS		UVO$_D$val		UVO$_D$test	
Initialization by ImageNet+COCO pre-trained						
YTVIS	41.0	34.7	9.3	7.6	7.3	6.6
UVO$_D$	31.6	22.2	17.4	11.2	12.5	7.7
YTVIS+UVO$_D$	36.8	26.4	15.0	9.6	11.7	7.2
UVO$_S$+UVO$_D$	35.5	24.5	20.9	13.2	16.2	9.9
Initialization by ImageNet pre-trained						
UVO$_S$+UVO$_D$	26.9	15.6	17.5	10.7	13.3	7.5
UVO$_S$+YTVIS	41.6	32.0	7.5	5.6	6.8	5.5

Table 4: **UVO video-level results with MaskTrack R-CNN, cross-evaluated with YTVIS.** Performance of MaskTrack R-CNN drops significantly when evaluated on UVO$_D$ due to unseen objects (open-world) and more complex background motions (sec 3.4). UVO$_S$ also provides a good alternative to COCO-pretraining for both UVO and YTVIS.

objects. It is worth noting that COCO contains all 20 VOC classes, and YTVIS contains 10 VOC classes.

Results are presented in Table 6. When trained in the closed-world, and tested under the open-world setting, all models suffer a significant drop in performance. This suggests that existing detectors are bounded to detect only in-taxonomy objects and are not capable of performing open-world detection. In addition, we observe that when enlarging taxonomy (VOC to COCO, COCO to UVO), performance on in-taxonomy (seen) objects may decrease slightly. This suggests that open-world detection may involve a per-class performance trade-off when increasing taxonomy size.

Train	Test	$\mathcal{J}$-score	$\mathcal{F}$-score
YTVOS+DAVIS	DAVIS16	0.887	0.899
	DAVIS17	0.792	0.843
YTVOS	YTVOSseen	0.797	0.842
	YTVOSunseen	0.728	0.809
YTVOS+DAVIS	UVO$_D$val	0.737	0.751
	UVO$_D$test	0.701	0.750

Table 5: **UVO tracking results with STM, cross-evaluated with DAVIS and YTVOS.** The overall performance on UVO$_D$ has a significant drop compared to other datasets, and is closer to the unseen scenario in YTVOS.

Data (trained categories)	All	Seen	Unseen
Mask R-CNN			
COCO(VOC)	32.8(-16.8)	51.3(+0.3)	8.8(-39.4)
COCO+UVO(COCO)	20.5(-20.8)	49.5(+0.8)	8.9(-23.6)
MaskTrack R-CNN			
YTVIS(VOC)	28.8(-12.2)	41.4(+1.2)	20.3(-21.8)
COCO+UVO(COCO)	12.6(-4.6)	30.2(+1.2)	3.5(-6.1)

Table 6: **Detecting and segmenting objects in the open-world is more challenging.** Performance is measured by AR100. Numbers in the bracket indicates difference compared to training on all annotated objects (regardless of class). In all settings, training on only a subset of categories decreases AR significantly for unseen classes/ overall performance, but may slightly improve AR on seen classes. UVO results are trained and evaluated on a subset of 300 videos with object categories labeled.

A possible alternative: bottom-up super-voxel. Unlike top-down approaches, bottom-up approaches, such as super-voxel algorithms, are by nature taxonomy free. They are typically non-parametric and do not rely on labeled data to train, and therefore, is a natural baseline for open-world problems. We evaluate GBH [15] on UVO and YTVIS.

Since GBH provides an over-segmentation of a video, metrics in object detection and segmentation, such as AP/AR, are not directly applicable. On the other hand, super-voxel metrics such as under segmentation error, segmentation accuracy (SA3D) and boundary recall dis-

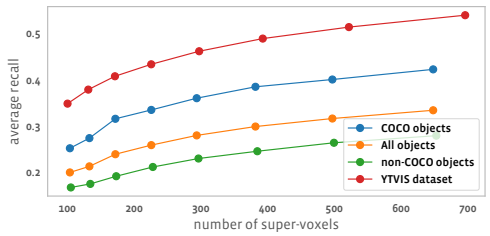


Figure 8: **GBH performance on UVO (all, COCO, non-COCO) and YTVIS**. At an average of 103 super-voxel proposals, approximate AR on UVO is 19.8%, 2.6% higher than AR100 of top-down approach MaskTrack R-CNN. In addition, the gap between unknown and COCO objects is much smaller compared to top-down approach (8.4% vs. 19.4%). Even on COCO objects, bottom-up approach is only 4% behind. For YTVIS, at an average 100 proposals, approximate AR is 34.7%, 5.9% higher than only training with VOC objects and 6.3% lower than training on all objects. UVO results are trained and evaluated on a subset of 300 videos with object categories labeled.

tance [46] are not comparable with AP/AR.

To quantitatively compare bottom-up approach with top-down approach, we adopt an approximate AR for super-voxel algorithms. AR is computed as the average of true-positive detections at multiple IoU thresholds divided by total objects. We make two relaxations. First, we relax the one-to-one mapping in object assignment due to over-segmentation in super-voxels. Specifically, a super-voxel is assigned to the object that has the largest intersection (similar to SA3D); an object, therefore, may be assigned multiple super-voxels. The combination of the assigned super-voxels provides an object-level prediction. Second, with no ranking score, it is not possible to take a cut-off using a fixed number of proposals for each video (*e.g.*, AR100 allows maximal 100 proposals per video). We compute ARk, where k is the average number of super-voxels for a dataset. The goal of this comparison is not to suggest super-voxel algorithms are stronger or weaker compared with top-down methods (*e.g.*, MaskTrack R-CNN); but rather to provide a possible alternative baseline for open-world problems.

On UVO, super-voxel algorithms achieve 19.8% for approximate AR103, 2.6% higher than top-down MaskTrack R-CNN AR100 (Figure 8). The gap between non-COCO objects to COCO objects is much smaller than MaskTrack R-CNN. We note that the metric used in top-down and bottom up is not directly comparable since multiple super-voxels can be assigned to one detection. On YTVIS, GBH achieves 34.7% for approximate AR100, 5.9% higher than open-world YTVIS (train on VOC classes and test on all) and 6.3% lower than closed-world YTVIS (Table 6).

Is temporally dense annotation necessary? Temporally dense annotations (30fps) are costly to obtain and YTVIS circumvents the problem by annotating sparsely (6fps). We examine the choice in the context of UVO by training and evaluating on a down-sampled version at 6fps.

Train data	Test data	AP	AR100
30fps	30fps	9.3	17.2
	6fps	6.7	14.1
6fps	30fps	7.2	14.5
	6fps	6.6	14.8
1fps + interpolation	30fps	7.2	15.0
1fps + interplt. + 2x data	30fps	8.4	16.0

Table 7: **Evaluating temporally dense video segmentation requires dense annotations; training with sparsely annotation can also achieve competitive performances.** A sparsely annotated evaluation data cannot tell the difference between the first two models, while there is a 2.7% gap on AR when test on densely annotated data. On the other hand, for training, training with 1fps ground-truth plus interpolated masks for intermediate frames gives competitive performance compared to 6fps. By trading-off annotation density and annotating additional data at 1fps, we can further improve the performance and closer the gap with 30fps data. *Experiments ran on 600 videos subset of UVO$_\mathcal{D}$.*

Models trained at 6fps and 30fps have only a minor difference evaluating on 6fps, but a 2.7% gap on AR 100 evaluating at 30fps (Table 7): temporally dense segmentation benefits from temporally dense annotations in evaluation.

For training, from a cost-efficient perspective, we may potentially trade-off high fps annotation with lower fps to scale for more videos. We examine this choice by using 1fps tracking-free annotation (no linking of objects) with intermediate data generated by STM [28] (similar as UVO$_\mathcal{S}$). This mirrors the mask propagation step in our annotation pipeline (sec 3.3). Training on the same number of videos with 1fps plus interpolated masks, the model is able to achieve similar performance compared to 6fps annotated data (Table 7). By trading-off sparsity for more videos and using 2x number of videos in training data, we are able to further closen the gap with 30fps annotations: sparse annotation could be sufficient for training.

5. Conclusion

We have presented UVO, a new benchmark for open-world object segmentation–an unsolved challenging yet important problem with various real-world applications. Compared with current benchmarks, UVO is not only different in the open-world problem setup but also multiple times larger in terms of size and annotations. We believe that UVO will enable more comprehensive video understanding research such as long-term video modeling and complex video understanding tasks.

Acknowledgement. We would like to thank Abhijit Ogale, Mike Zheng Shou, Dhruv Mahajan, Kristen Grauman, Lorenzo Torresani, Manohar Paluri, Rakesh Ranjan, and Federico Perazzi for their valuable feedback on the dataset; Jiabo Hu, Haoqi Fan, and William Wen for engineering support; Sally Yoo, Yasmine Babaei, and Eric Alamillo for supporting on annotation logistics; and our annotators for their hard work.

References

[1] Google cloud model cards object detection. https://modelcards.withgoogle.com/object-detection. Accessed: 2021-03-16. 1

[2] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 73–80. IEEE, 2010. 3

[3] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1893–1902, 2015. 2

[4] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1209–1218, 2018. 3

[5] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 2

[6] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, 2018. 1

[7] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. Tao: A large-scale benchmark for tracking any object. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 436–454, Cham, 2020. Springer International Publishing. 2, 6

[8] A. R. Dhamija, M. Günther, J. Ventura, and T. E. Boult. The overlooked elephant of object detection: Open set. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1010–1019, 2020. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2

[10] I. Endres and D. Hoiem. Category-independent object proposals with diverse ranking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):222–234, 2014. 3

[11] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015. 2, 6

[12] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, 2019. 2

[13] Pallabi Ghosh, Nirat Saini, Larry S. Davis, and Abhinav Shrivastava. All about knowledge graphs for actions, 2020. 2

[14] Georgia Gkioxari, Ross Girshick, Piotr Dollar, and Kaiming He. Detecting and recognizing human-object interactions. In *CVPR*, 2018. 1

[15] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2141–2148, 2010. 1, 2, 6, 7

[16] A. Gupta, P. Dollár, and R. Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5351–5359, 2019. 2, 5, 6

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, pages 2980–2988, 2017. 1, 6

[18] Geremy Heitz and Daphne Koller. Learning spatial context: Using stuff to find things. In *European conference on computer vision*, pages 30–43. Springer, 2008. 3

[19] R. Hu, P. Dollár, K. He, T. Darrell, and R. Girshick. Learning to segment every thing. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4233–4241, 2018. 2

[20] Ayush Jaiswal, Yue Wu, Pradeep Natarajan, and Premkumar Natarajan. Class-agnostic object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 919–928, January 2021. 2

[21] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 1

[22] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017. 3

[23] Li Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006. 2

[24] Tsung-Yi Lin, M. Maire, Serge J. Belongie, James Hays, P. Perona, D. Ramanan, Piotr Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1, 2, 6

[25] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019. 2

[26] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001. 2

[27] J. Meng, J. Yuan, J. Yang, G. Wang, and Y. Tan. Object instance search in videos via spatio-temporal trajectory discovery. *IEEE Transactions on Multimedia*, 18(1):116–127, 2016. 1

[28] S. W. Oh, J. Y. Lee, N. Xu, and S. J. Kim. Space-time memory networks for video object segmentation with user guidance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. 4, 6, 8

[29] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 724–732, 2016. 2, 6

[30] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. 1

[31] Pedro O. Pinheiro, Ronan Collobert, and Piotr Dollár. Learning to segment object candidates. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 1990–1998, Cambridge, MA, USA, 2015. MIT Press. 2, 6

[32] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbelaez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. 04 2017. 3, 6

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 1

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28, pages 91–99. Curran Associates, Inc., 2015. 1, 6

[35] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't judge an object by its context: Learning to overcome contextual bias. In *CVPR*, 2020. 2

[36] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015. 2

[37] Du Tran, Heng Wang, Lorenzo Torresani, and Matt Feiszli. Video classification with channel-separated convolutional networks. In *ICCV*, 2019. 2

[38] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018. 2

[39] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. Segmentation as selective search for object recognition. In *ICCV*, pages 1879–1886, 2011. 1, 3

[40] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. Mots: Multi-object tracking and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[41] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2

[42] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, 2018. 2

[43] Yang Wang, Gedas Bertasius, Tae-Hyun Oh, Abhinav Gupta, Minh Hoai, and Lorenzo Torresani. Supervoxel attention graphs for long-range video modeling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 155–166, January 2021. 2

[44] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[45] B. Xiong, S. Jain, and K. Grauman. Pixel objectness: Learning to segment generic objects automatically in images and videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2677–2692, nov 2019. 3

[46] Chenliang Xu and Jason Corso. Libsvx: A supervoxel library and benchmark for early video processing. *International Journal of Computer Vision*, 119, 09 2016. 6, 8

[47] N. Xu, L. Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and T. Huang. Youtube-vos: A large-scale video object segmentation benchmark. *ArXiv*, abs/1809.03327, 2018. 2, 3, 6

[48] L. Yang, Y. Fan, and N. Xu. Video instance segmentation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5187–5196, 2019. 2, 3, 6

[49] Fang Yuan, Zhe Wang, Jie Lin, Luis Fernando D'Haro, Kim Jung Jae, Z. Zeng, and V. Chandrasekhar. End-to-end video classification with knowledge graphs. *ArXiv*, abs/1711.01714, 2017. 2

[50] Haoyang Zhang and Xuming He. Deep free-form deformation network for object-mask registration. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[51] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, 2017. 2

[52] Tinghui Zhou, M. Brown, Noah Snavely, and D. Lowe. Unsupervised learning of depth and ego-motion from video. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6612–6619, 2017. 5

[53] C. L. Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 3