Open-world Object Detection and Tracking

Achal Dave

CMU-RI-TR-21-17

May 19, 2021

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:

Deva Ramanan	*Carnegie Mellon University* (chair)
Katerina Fragkiadaki	*Carnegie Mellon University*
Kris Kitani	*Carnegie Mellon University*
Cordelia Schmid	*INRIA*
Ross Girshick	*Facebook AI Research*

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

Copyright © 2021 Achal Dave. All rights reserved.

Abstract

Computer vision today excels at recognizing narrow slices of the real world: our models seem to accurately detect objects like cats, cars, or chairs in benchmark datasets. However, deploying models requires that they work in the open world, which includes arbitrary objects in diverse settings. Current methods struggle on both axes: they recognize only a few classes, and struggle in settings that differ from the training distribution. A model that addresses these challenges can serve as a fundamental building block for downstream applications, including recognizing actions, manipulating objects, and navigating around obstacles. This thesis presents our work in building *robust* models for detecting and tracking *any* object, especially ones with few or even no training examples.

We start by exploring how traditional models, which recognize only a small set of object classes, generalize to the real world. We show that current methods are extremely sensitive: even subtle changes in the input image or test distribution can lead to drops in accuracy. Our systematic evaluations show that models — even ones trained for robustness to adversarial or synthetic corruptions — often correctly classify one frame of a video, but fail on a perceptually similar nearby frame. A similar phenomenon applies even to small distribution shifts arising from natural variation between datasets. Finally, we present an approach for addressing an extreme form of generalization to object appearance: detecting fully occluded objects.

Next, we explore generalization to large or infinite vocabularies, which contain rare and never-before-seen classes. Since current datasets are largely limited to a small, closed-world set of objects, we first present a large vocabulary benchmark for measuring progress in detection and tracking. We show that current evaluations do not suffice for large vocabulary benchmarks, and present alternative metrics that appropriately evaluate progress in this setting. Finally, we present approaches which leverage advances in closed-world recognition to build accurate, generic detectors and trackers for *any* object.

Acknowledgments

I am privileged to have spent my Ph.D. surrounded by the most supportive colleagues, friends and family one could ask for.

I am grateful to my advisor, Deva Ramanan, for his mentorship. Deva taught me to trust my hunches, to believe in 'wacky' ideas, and to value communication as a fundamental component of the scientific pursuit. I also thank my other close collaborators and mentors: Olga Russakovsky, who taught me to celebrate the small victories, and to enjoy shrinking a paper to 8 pages; Pavel Tokmakov, who showed me the value of optimism in research, and of long, (sometimes uncomfortably) fast walks; Ross Girshick, who taught me what experimental rigor truly means, including the value of simple configuration systems and well-chosen spreadsheet fonts. I thank my committee, Deva, Katerina Fragkiadaki, Kris Kitani, Cordelia Schmid, and Ross for their mentorship and support. I am grateful to Alyosha Efros, Stella Yu, Mehul Nariyawala and Navneet Dalal for helping me fall in love with computer vision. I thank our lab for asking the hard questions in group meetings and helping refine my research, and the broader Smith Hall community for helpful discussions and advice. I thank the staff at CMU for their support, including Suzanne Muth (especially for advice during Starbucks trips), Hadley Pratt, Christine Downey, and Stephanie Matvey. I am thankful to my other collaborators, who made research fun and fulfilling: Nicholas Carlini, Piotr Dollár, Tarasha Khurana, Alexander Kirillov, Shu Kong, Bastian Leibe, Yang Liu, Jonathon Luiten, Aljoša Ošep, Neehar Peri, Rebecca Roelofs, Cordelia Schmid, Ludwig Schmidt, Vaishaal Shankar, Rohan Taori, Laura Leal-Taixé, and Idil Esen Zulfikar. Finally, thanks to one more faithful collaborator, my 2015 Macbook, for surviving the last many years of spills, drops, and rainy walks home.

Outside work, I thank my friends and family for keeping me sane. Humphrey Hu, Karthik Lakshmanan, and Kumar Shaurya Shankar hosted me during visit days, and showed me just how wonderful Pittsburgh and CMU are. I am grateful to Sourish Chaudhuri, for convincing me that I'd love Pittsburgh. My first office mates, Nicholas Gisolfi and Micah Corah, helped me start the Ph.D. on the right foot. Jingyan Wang made walks home feel shorter. Rick Goldstein renewed my love for board games. Rohit Girdhar helped keep my swing dancing skills alive. Peiyun Hu, Mengtian (Martin) Li, Aayush Bansal and Ravi Teja Mullapudi kept research and life discussions in our office lively. Tarasha Khurana made me a better mentor and collaborator. Shu Kong, Ishan Misra dispensed invaluable

wisdom during our (frequent) coffee breaks. Christopher (Kit) Ham, Gunnar Atli Sigurdsson, Nicholas Rhinehart, Allison Del Giorno, Kenneth (Kenny) Marino, Senthil Purushwalkam, and Nadine Chang kept Super Smash Brothers alive in Smith Hall. Puneet Puri and Anirudh Vemula were wonderful roommates, introducing me to new restaurants, recipes and TV shows. Dhruv Saxena made going to the gym feel like a fun break, and made it possible to win trivia nights; Xuning Yang helped me discover Pittsburgh's best pizza spots; Rosario Scalise's infectious spirit made the department feel a little closer, and a little smaller; Cara Bloom taught me the value of asking for help (avoiding a potentially catastrophic battery replacement in my trusty Honda Civic); Christine Baek brought Berkeley and the bay area to Pittsburgh. Senthil, Kenny, and Nadine made Smith hall feel like home: From video games to paper exchanges, coffee breaks to long dinners, gym trips to heated whiteboard discussions, they made coming into lab feel like a fun adventure, where research just happened along the way. From the other side of the country, through tens of thousands of messages in a half-decade long group chat, Vaishaal Shankar and Steve Yadlowsky helped celebrate the highs and soften the lows of the Ph.D. roller coaster. I looked forward waking up to hundreds of messages in the chat, whether they were about a recent arXiv submission or a particularly tasty burrito. Traveling with Vaishaal, Steve, Merra Kurubalan, Nielsen Hermanto, and Radhika Kannan helped me start every year of the Ph.D. with a fresh mind. A host of other friends made the past few years unforgettable, including Matt Barnes, Ankit Bhatia, Xinlei Chen, Shushman Choudhury, Micah Corah, Vishal Dugar, Pragna Mannam, Adithya Murali, Ishan Nigam, Devin Schwab, Tim Mueller Sim, Arun Venkatraman, Jacob Walker, and many others. Thank you all for making Pittsburgh feel like home.

My partner and my best friend, Radhika, made me want to travel, to dance, and to be spontaneous. No one else could have made a crowded, overnight journey on Megabus, or a significantly less crowded fall from a 20,000 feet high plane seem enticing. Thank you for always being there, and for patiently entertaining me when I just couldn't stop talking about research. I thank my grandparents and extended family for all their love and support from afar. My sister, Mitali (the real doctor in the family), was always on-call for any ailment: from navigating the transition to graduate school, to reassuring me when a WebMD self-diagnosis did not suffice. Finally, I owe a deep gratitude to mom and dad — Devangi and Dushyant Dave — who left a support system far wider than mine, in a country thousands of miles away, for me and my sister. Thank you.

Funding

This work was supported by the CMU Argo AI Center for Autonomous Vehicle Research, the National Science Foundation (NSF), Intelligence Advanced Research Projects Activity (IARPA), and the Intel Science and Technology Center for Visual Cloud Systems (ISTC-VCS).

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

People have the astounding ability to operate in staggeringly diverse environments. Children, for example, have no trouble playing with new toys, in new houses or playgrounds, with minimal supervision. Adults routinely drive through new routes and cities, tracking and planning around other people, cars and a vast variety of potential obstacles. This thesis focuses on replicating just one aspect of this ability in computer vision models: *reliable perception in the open-world*, which includes arbitrary objects in diverse contexts.

Although computer vision systems have significantly improved in recognition accuracy on benchmark datasets, they fall short on both of the above axes. They recognize only a few object classes, and struggle in settings that differ from the training distribution. Deploying models in real applications requires addressing both challenges. Robotic agents, for example, must be able to detect *any* obstacle, even ones that haven't been seen before, in varied contexts. This thesis presents our work on building robust models for detecting and tracking any object, especially ones with few or even no training examples. We tackle this challenge in three parts:

Generalizing to appearance changes. We start by analyzing how closed-world models, which are limited to small object vocabularies, generalize to the real world. Surprisingly, even in this limited setting, models are extremely sensitive: small changes to the appearance or context of objects can significantly degrade accuracy. To systematically analyze this, we develop a benchmark for assessing robustness to natural perturbations collected semi-automatically from videos [201]. We extend this

work to evaluate model robustness to a variety of *distribution shifts* [211], where we evaluate models on data that varies in data collection procedure, object appearance, or object context. We evaluate hundreds of models, and find that nearly all models – even ones trained to be robust to adversarial or synthetic corruptions – lack such natural robustness, with the exception of models trained on orders of magnitude more data. We also consider a natural extreme of robustness to object appearance [123]: detection in the face of complete occlusions, using temporal context and scene structure.

Scaling to large vocabularies. While models and datasets for recognizing a few classes have matured, large-vocabulary recognition of thousands of classes remains challenging. Most work, particularly in the video domain, evaluates on only a handful of classes in datasets with limited scene diversity. In the video domain, we collected a dataset for Tracking Any Object (TAO) [47], which contains tracking annotations for hundreds of object classes, spanning over 17,000 objects in nearly 3,000 videos. TAO allows evaluating multi-object trackers, person-specific trackers, and user-initialized trackers on a level playing field, and our analysis highlights important avenues for future improvements in object tracking. In addition to dataset limitations, we found that even evaluation strategies for small-vocabulary recognition do not generalize well to large-vocabulary settings [48]. To address this evaluation limitation, which arises due to challenges in score calibration, we proposed a fix to current evaluations and a calibration strategy for improving current image-based detectors.

Open-world detectors and trackers. In the limit, vision systems must recognize objects from an infinitely large vocabulary, detecting even objects that have never been seen before. Detecting and segmenting individual objects, regardless of their category, is crucial for applications ranging including action detection, robotic interaction, or video editing. We present an approach that leverages motion cues and synthetic training data to recognize such generic objects – even if the objects are missing from the training data [45]. A key insight here is that advances in category-specific detectors can be used to improve category-agnostic detection. We leveraged this insight again to build an approach to convert a category-specific object detector into a category-agnostic, object-specific detector (i.e. a tracker) efficiently [46], leading to significant improvements in video object tracking and segmentation.

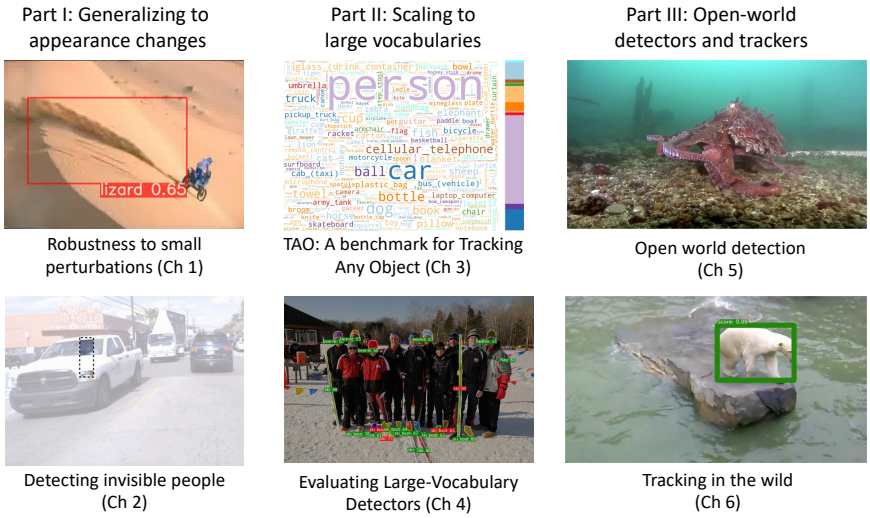


Figure 1.1: Current vision models tend to be limited to a few classes, and struggle to generalize to settings differing from training. This thesis tackles these challenges in three parts: (I) generalizing to appearance changes, like subtle perturbations or significant occlusions; (II) scaling to large vocabularies of hundreds or thousands of objects, and (III) detecting any object in the open-world.

1.1 Overview

This thesis is presented in three parts, exploring (I) model generalization to changes in object appearance or context, (II) large vocabulary recognition of hundreds to thousands of classes, (III) generic detectors and trackers for *any* object.

Part I: Generalizing to appearance changes. We start by analyzing how current models for limited vocabularies generalize to real world settings. Although these models achieve high accuracy on average in benchmark datasets, they remain sensitive to subtle changes in the input image or test distribution. Our work shows that even small changes in the orientation of an object in an image or in the test set can significantly degrade model accuracy. In the machine learning community, this sensitivity of models has typically been analyzed on images perturbed by an adversary [24, 81], or by hand-designed synthetic corruptions [56, 65, 92, 95]. However, these benchmarks rely on synthetically modifying images, serving at best as proxies for evaluating generalization to the real world. In this part, we systematically evaluate and address model robustness to *natural* perturbations and distribution shifts.

One result of this lack of robustness is a frustrating phenomenon when applying image recognition systems to videos: models correctly recognize objects in one frame, but fail to do so in the very next frame. In Chapter 3, we present a new benchmark for assessing robustness to such temporal perturbations found in videos. In our evaluation, we compare a model's accuracy on a single video frame with its worst case accuracy on nearby frames. Our results show a significant, consistent gap between these accuracies across various models. In [211], we extended this work to evaluate robustness to a number of *distribution shifts*, where the test distribution varies slightly from the training distribution. Unfortunately, our work shows that nearly all models – even ones trained to be robust to adversarial or synthetic corruptions – lack such natural robustness. Fortunately, our work also highlights some important exceptions: models trained on large, web-scale datasets (*e.g.* [182]) do confer some amount of natural robustness. However, further improving the robustness of these models, while reducing the amount of training data required, remains an ongoing challenge.

While we have so far considered robustness to subtle changes, such as partial occlusions, we next consider a natural extreme: recognition under complete occlusions. Object detection in online applications, such as self-driving vehicles, fundamentally requires object permanence: the ability to reason about even invisible objects. In Chapter 4, we re-purpose tracking benchmarks and propose new metrics for the task of detecting invisible objects, focusing on the illustrative case of people. We treat this as a short-term forecasting task, and incorporate scene structure from depth estimators to build an accurate, probabilistic approach for detecting invisible people from arbitrary, monocular videos.

Part II: Scaling to large vocabularies. The previous part focuses on model generalization to changes in object appearance and context. These models are largely limited to a narrow range of object classes, such as people or cars. Objects in the real world, however, span orders of magnitude more classes. To accommodate such diverse objects, we now turn to recognition of large class vocabularies.

A primary concern in large-vocabulary recognition is the lack of appropriate benchmarks for training and evaluating models. Most datasets, especially for video-based tasks, focus on a limited set of classes, such as people, vehicles, or animals. To address this, we first present TAO, a dataset for Tracking Any Object, in Chapter 5. TAO contains over 17,000 labeled object tracks, spanning hundreds of classes in

nearly 3,000 videos. TAO allows evaluating multi-object trackers, person-specific trackers, and user-initialized (or 'single-object') trackers on a level playing field. Our analysis reveals important shortfalls in current trackers, highlighting avenues for future research. For example, our experiments show that person tracking does work fairly well. However, these advances have not generalized to other classes, like hand-held objects or electronics, which have been overlooked in prior datasets.

Next, we consider appropriate strategies for evaluating large-scale recognition systems. In Chapter 6, we show that scaling to hundreds or thousands of classes which vary in recognition difficulty and rarity results in subtle, significant evaluation issues. Our analysis shows that current evaluations, in fact, are over-fit to the small-vocabulary regime. Specifically, the standard detection evaluation (Average Precision, or AP) results in a gameable metric that encourages miscalibration of scores across categories. We present a modification to address this issue, and a calibration strategy to improve large-vocabulary detectors.

Part III: Open-world detectors and trackers. Scaling to larger vocabularies allows recognizing a broad, but finite, array of objects. Many applications require vision systems to go beyond such fixed vocabularies, so as to detect any object, even ones which were absent from the training data. Autonomous navigation, for example, requires detecting never-before-seen obstacles and debris, while efficient video editing requires segmenting arbitrary objects and parts with minimal user input.

Defining the notion of a generic object can be ambiguous. A key insight in our work is relying on external cues to delineate objects. For example, Chapter 7 presents a detector for generic, moving objects, using motion as a cue to group pixels into objects. Our work shows that diverse, synthetic data suffices for building accurate motion-based detectors. Combining these detectors with models trained on limited real image datasets results in a generic model that strongly improves over prior work.

In Chapter 8, we tackle tracking arbitrary objects specified by a user (*e.g.*, human annotator) or external signal (*e.g.*, motion). A key insight in our approach in Chapter 7 is that advances in category-specific detectors can be used to improve category-agnostic detection. We leverage this insight again, presenting a strategy for converting a category-specific object detector into a category-agnostic, object-specific detector (i.e. a tracker) efficiently. The result is an accurate tracker which improves over prior work in video object tracking with precise, pixel-level segmentation.

Chapter 2

Background

Although recent work in computer vision focuses on class-specific object models, there is a rich history in the literature of building generic recognition models that work for arbitrary objects. Our work builds upon this literature, incorporating techniques from recent advances in class-specific detection. In this section, we provide a brief background of attempts at tackling open-world recognition in the literature.

Background subtraction. Perhaps the simplest approach for open-world detection and tracking is background subtraction. One can *model* the background of any scene, and remove this background component to mark generic, foreground objects. Various strategies exist for background subtraction. Jain et al. [106] simply computes the difference between successive frames to generate segmentation masks for moving objects. Wren et al. [240] models the background with a Gaussian distribution over color values for each pixel, marking as foreground pixels that deviate from the distribution. Stauffer and Grimson [207] extends this to more general scenes where even background pixels may have a multi-modal distribution, by representing the background with a Gaussian mixture model per pixel. While such approaches suffice for stationary cameras, they can struggle in the presence of camera motion. Additionally, while modeling the background allows for segmenting foreground objects, it does not provide instance-level segmentations around individual objects. Such instance-level segmentations can be useful for downstream applications, which may require, for example, forecasting the trajectory of individual objects.

Motion segmentation. Motion segmentation addresses some of the challenges

faced by background subtraction approaches. Rather than modeling the background of a scene captured by a static camera, motion segmentation aims to group pixels that move together. This allows for instance-level segmentation of arbitrary objects, even in the presence of camera motion. An early work in this direction from Shi and Malik [202] proposed treating this task as a spatio-temporal grouping problem, a philosophy espoused by a number of more recent approaches, including Brox and Malik [30], Grundmann et al. [83], Keuper et al. [119], as well as Ochs et al. [165]. In summary, these approaches track each pixel in a video individually (using optical flow estimators), encode the motion information of a pixel in a compact descriptor, and then obtain an instance segmentation by clustering the pixels based on motion similarity. These approaches differ in the strategy used for tracking, encoding, and clustering pixels. Unfortunately, while these approaches show promising results on small-scale benchmarks, they struggle in the wild (see Chapter 7), as they rely heavily on hand-designed heuristics for each stage. More recently, there has been an attempt to incorporate convolutional neural networks which can learn from large training datasets to improve the detection and segmentation of moving objects. Fragkiadaki et al. [72], for example, train a CNN to detect (but not segment) moving objects, and combines these detections with clustered pixel trajectories to derive segmentations. Bideau et al. [23] proposed to combine a heuristic-based motion segmentation method (from Bideau and Learned-Miller [22], Narayana et al. [163]) with a CNN trained for semantic segmentation for the task of moving object segmentation. Xie et al. [246] introduced a deep learning approach for motion segmentation that segments and tracks moving objects using a recurrent neural network. In Chapter 7, we build upon this line of work, and show how to re-purpose advances in class-specific detection to build a simple, learned approach for segmenting moving objects. Our work is perhaps most similar to that of Fragkiadaki et al. [72], but differs in that we predict segmentation masks per frame and track them over time, while [72] predicts bounding boxes and clusters pixel trajectories to create segmentations.

Object proposal generation. Methods for generating object proposals similarly aim to localize generic objects, independent of their class. Since these approaches operate on static images, they focus on appearance cues (color, boundaries, etc.) and priors. Russell et al. [195] discovers objects in collections of images, by generating segmentations in individual images and clustering these segmentations across the

dataset. More common are methods for generating class-agnostic proposals from single images, either in the form of a bounding box (as in Alexe et al. [4]) or a segmentation mask (as in Carreira and Sminchisescu [34], Endres and Hoiem [55]). These methods relied largely on *bottom-up* cues for localizing objects, and do not take advantage of appearance priors that can be *learned* from object detection datasets. More recent approaches leverage deep neural networks to learn such appearance priors, either as a standalone proposal generator (Kuo et al. [129], Pinheiro et al. [177]), as a part of an image-level segmentation method (Pham et al. [176]), or as part of a class-specific detection system (Erhan et al. [57], Ren et al. [190]). For a more thorough analysis of proposal generation methods, we refer the reader to Hosang et al. [94]. In Chapter 7, we leverage ideas from these top-down proposal generation methods to build an accurate motion-based, generic object segmentation system. Importantly, Chapter 7 focuses on segmenting *moving* objects, and can thus leverage synthetic data to detect generic objects based on their motion.

Open-world recognition. The term open-world was introduced, to the best of our knowledge, in Bendale and Boult [13] for the classification setting. This thesis focuses on the detection and tracking counterparts, which have also been explored concurrently in Joseph et al. [113] (detection) and Liu et al. [146], Wang et al. [229] (tracking). A related line of work aims to approximate open-world recognition by scaling up closed-world vocabularies. Redmon and Farhadi [187], for example, object detectors to thousands of classes by leveraging weak, image-level supervision. Hu et al. [96] similarly extends instance segmentation methods to larger vocabularies via bounding box supervision. Gupta et al. [88] relabels the popular MS COCO [142] dataset with over a thousand classes. Farhadi et al. [64] propose to use an attribute-based vocabulary, rather than a class-based one, which may allow recognizing properties of never-before-seen objects. While scaling or modifying vocabularies does not address the open-world task (as there may always be some objects outside any finite vocabulary), it may suffice for many applications. Chapter 5 follows this philosophy, introducing a large-scale benchmark for tracking a large vocabulary of closed-world *and* open-world classes. Chapter 6 highlights challenges and provides solutions for scaling approaches and evaluations to such large vocabularies.

User-initialized tracking. In the video domain, user-initialized tracking (which

includes tasks such as single-object tracking and video object segmentation) has long tackled the task of tracking generic objects. This resolves the issue of finding generic objects, by relying on user input. Initially, these approaches relied on simple appearance and motion models, such as appearance features in the KLT tracker (from Lucas and Kanade [149], Shi and Tomasi [203], Tomasi and Kanade [216]) or smoothness assumptions on object motion, as in Rangarajan and Shah [184], Sethi and Jain [199]. We refer the reader to the excellent survey from Yilmaz et al. [256] for a more detailed review of varying approaches for generic user-initialized tracking, including such generic appearance and motion models, state estimation techniques, and varying object representations. While these approaches relied on generic models for any object, subsequent work showed that tailoring appearance and motion models to specific classes can lead to improved tracking. The introduction of improved object detectors (*e.g.*, the Dalal-Triggs detector [42] and detectors based on pictorial models, particularly the Deformable Parts Model from Felzenszwalb et al. [68]) led to an increased interest in class-specific appearance modeling. Leibe et al. [136] is an early work in this direction, leveraging strong, single frame object detectors as a fundamental input for tracking pedestrians and vehicles in 3D from stereo cameras. Ramanan et al. [183] similarly relies on an accurate, generic person detector (based on pictorial structures [67, 69]) and adapting it to specific people in the (monocular) video. Andriluka et al. [6] continues in this direction, using more accurate detectors that allow for tracking in more diverse scenes. To our knowledge, Andriluka et al. [6] popularized the term 'tracking-by-detection,' by which this series of approaches is known today. Similar efforts have tailored motion models for specific classes, most commonly for people. Agarwal and Triggs [2] and Pavlovic et al. [171] represent human poses with a fixed, parametric model containing chains of limb segments, and learn dynamical models of motion for tracking from monocular videos. Class-specific models can be more accurate than their generic counterparts, as they can take advantage of appearance priors learned from labeled datasets and learned or hand-coded motion constraints. Recent effort has aimed to leverage advances in class-specific modeling to improve generic tracking, particularly for appearance models. Bertinetto et al. [18] and Held et al. [91] show that convolutional networks, traditionally used for classification and class-specific detection, can be trained for generic object tracking. Our work extends this line of work, and shows strategies for evaluating (Chapter 5)

and improving (Chapter 7, Chapter 8) such generic appearance models. Chapter 8 is particularly inspired by Bertinetto et al. [18] and Held et al. [91], but re-purposes an existing object detection model (Mask R-CNN) to build a unified approach for detecting, segmenting and tracking objects. Recent work, including this thesis, focuses primarily on improving generic appearance models, but one could similarly build more accurate, generic motion models. We hope that our new dataset (Chapter 5) will also encourage improved generic motion modeling, which can reason about trajectories of common objects (such as people or cars) as well as objects that have been ignored in the past (hand-held objects or accessories).

Part I

Generalizing to changes in appearance and context

Chapter 3

Model robustness in the wild

3.1 Introduction

Applying state-of-the-art image recognition systems to videos reveals a troubling phenomenon: models correctly recognize objects in one frame, but fail to do so in the very next frame (Figure 3.1). In practice, this *flickering* of predictions is treated as an unfortunate but unavoidable property of image-based models. This issue can be mitigated in offline settings by smoothing predictions over time. However, online smoothing isn't nearly as effective and incurs a delay, resulting in catastrophic mistakes in downstream applications: *e.g.*, flickering object classifications have reportedly led to fatal autonomous vehicle collisions [25].

At its root, prediction flicker is a manifestation of a broader issue: current models lack *robustness* to small input perturbations. In the machine learning community, model robustness has typically been analyzed on images perturbed by an adversary [24, 81], or by hand-designed strategies, such as rotations or blurs [56, 65, 92, 95]. However, these benchmarks rely on synthetically modifying the input image, serving at best as *proxies* for evaluating robustness to *natural* perturbations, which are common in videos.

In this work, we systematically analyze the prevalence of flicker across vision models. Taking inspiration from the robustness literature, we evaluate models on *perceptually similar* images, which we sample from nearby video frames. However, nearby frames can still exhibit drastic changes (*e.g.*, significant occlusions), which

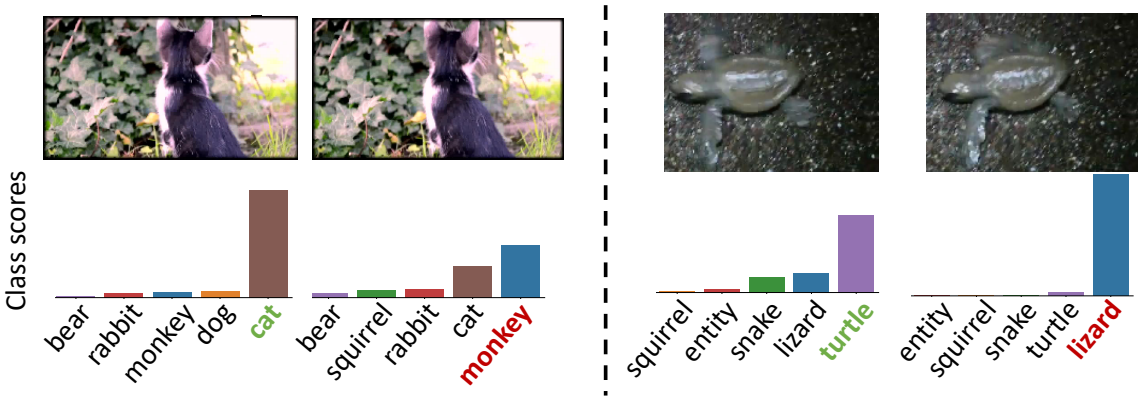


Figure 3.1: Examples of natural perturbations from nearby video frames and resulting classifier predictions from a ResNet-152 model fine-tuned on ImageNet-Vid. While the images appear almost identical to the human eye, the classifier confidence changes substantially.

may cause even robust models to fail. We discard such frame pairs by employing human expert labelers to evaluate model robustness only on perceptually similar images, unlike prior work [85]. As a cornerstone of our investigation, we introduce two test sets for evaluating model robustness: `ImageNet-Vid-Robust` and `YTBB-Robust`, carefully curated from the ImageNet-Vid and Youtube-BB datasets [185, 194]. To the best of our knowledge these are the first datasets of their kind, containing tens of thousands of images that are *human reviewed* and grouped into thousands of perceptually similar sets. In total, our datasets contain 3,139 sets of temporally adjacent and visually similar images (57,897 images total).

We use these datasets to measure the robustness of current models to small, naturally occurring perturbations. Our testbed contains over 47 different models, varying model types (CNNs, transformers), architectures (*e.g.*, AlexNet, ResNet) and training methods (*e.g.*, adversarial training, augmentation). To systematically characterize flicker, we also introduce a stringent robustness metric.

Our experiments show that all models in our testbed degrade significantly in the presence of small, natural perturbations in video frames. Under our metric, we find such perturbations in `ImageNet-Vid-Robust` and `YTBB-Robust` induce median accuracy drops of 16% and 10% respectively for classification, and a median 14 point

AP drop for detection[1]. Even for the best-performing classification models trained on public datasets, we observe an accuracy drop of 14% for `ImageNet-Vid-Robust` and 8% for `YTBB-Robust`. Recently introduced, contrastive models trained on weakly supervised web images [182] can reduce this gap, but require over 400 million images, and still exhibit noticeable gaps of 6.1% and 6.7%, respectively

Our results show that robustness to natural perturbations in videos is problematic for a wide variety of models. Practical deployment of models, especially in safety-critical environments like autonomous driving, requires predictions that are not only accurate, but also robust over time. Our analysis indicates that ensuring reliable predictions on *every frame* of a video is an important direction for future work.

3.2 Background

Adversarial examples. While various forms of adversarial examples have been studied, the majority of research focuses on ℓ_p robustness [24, 81, 261]. However, it is unclear whether adversarial examples pose a problem for robustness outside of a truly worst case context. It is an open question whether perfect robustness against a ℓ_p adversary will induce robustness to realistic image distortions such as those studied in this paper. Recent work has proposed less adversarial image modifications such as small rotations & translations [8, 56, 65, 115], hue and color changes [95], image stylization [77] and synthetic image corruptions such as Gaussian blur and JPEG compression [76, 92]. Even though the above examples are more realistic than the ℓ_p model, they still synthetically modify the input images to generate perturbed versions. In contrast, our work performs no synthetic modification and instead uses unmodified video frames.

Studying robustness in videos. In recent work, Gu et al. [85] exploit the temporal structure in videos to study robustness. However, their experiments suggest a substantially smaller drop in accuracy. The primary reason for this is a less stringent metric used in [85]. By contrast, our PM-k metric is inspired by the "worst-of-k" metric used in prior work [56], highlighting the sensitivity of models to natural perturbations. In the appendix, we study the differences between the two metrics in

[1]We only evaluated detection on `ImageNet-Vid-Robust` as bounding-box labels in Youtube-BB are not temporally dense enough for our evaluation.

Figure 3.2: Temporally adjacent frames may not be visually similar. We show three randomly sampled frame pairs where the nearby frame was marked as "dissimilar" to the anchor frame during human review and then discarded from our dataset.

more detail. Furthermore, the lack of human review and the high label error-rate we discovered in Youtube-BB (Table 3.1) presents a potentially troubling confounding factor that we resolve in our work.

Distribution shift. Small, benign changes in the test distribution are often referred to as *distribution shift*. Recht et al. [186] explore this phenomenon by constructing new test sets for CIFAR-10 and ImageNet and observe substantial performance drops for a large suite of models on the newly constructed test sets. Similar to our Figure 3.3, the relationship between their original and new test set accuracies is also approximately linear. However, the images in their test set bear little visual similarity to images in the original test set, while all of our failure cases are on perceptually similar images. In a similar vein of study, [217] studies distribution shift *across* different computer vision data sets such as Caltech-101, PASCAL, and ImageNet.

Temporal consistency in computer vision. Authors of Jin et al. [112] explicitly identify flickering failures and use a technique reminiscent of adversarially robust training to improve image-based models. A similar line of work focuses on improving object detection in videos as objects become occluded or move quickly [66, 117, 245, 272]. The focus in this work has generally been on improving object detection when objects transform in a way that makes recognition difficult from a single frame, such as fast motion or occlusion. In this work, we document a broader set of failure cases for image-based classifiers and detectors and show that failures occur when the neighboring frames are imperceptibly different.

Table 3.1: Dataset statistics of `ImageNet-Vid-Robust` and `YTBB-Robust`. For `YTBB-Robust`, we updated the labels from for 41% (834) of the accepted anchors due to incomplete labels in Youtube-BB.

		`ImageNet-Vid Robust`	`YTBB Robust`
Anchor frames	Reviewed	1,314	2,467
	Accepted	1,109 (84%)	2,030 (82%)
	Labels updated	-	834 (41%)
Frame pairs	Reviewed	26,029	45,631
	Accepted	21,070 (81%)	36,827 (81%)

3.3 Evaluating temporal robustness

`ImageNet-Vid-Robust` and `YTBB-Robust` are sourced from videos in the ImageNet-Vid and Youtube-BB datasets [185, 194]. All but one [2] of the object classes in ImageNet-Vid and Youtube-BB are from the WordNet hierarchy [160] and direct ancestors of ILSVRC-2012 classes. Using the WordNet hierarchy, we construct a canonical mapping from ILSVRC-2012 classes to ImageNet-Vid and Youtube-BB classes, which allows us to evaluate off-the-shelf ILSVRC-2012 models on `ImageNet-Vid-Robust` and `YTBB-Robust`. We provide more background on the source datasets in the appendix.

3.3.1 Dataset construction

Next, we describe how we extracted sets of naturally perturbed frames from ImageNet-Vid and Youtube-BB to create `ImageNet-Vid-Robust` and `YTBB-Robust`. A straightforward approach would be to select a set of anchor frames and use temporally adjacent frames in the video with the assumption that such frames contain only small perturbations from the anchor. However, as Figure 3.2 illustrates, this assumption is frequently violated, especially due to fast camera or object motion.

Instead, we first collect *preliminary* datasets of natural perturbations following the same approach, and then manually review each of the frame sets. For each video, we randomly sample an anchor frame and take $k = 10$ frames before and after the anchor

[2] the class “skateboard" in Youtube-BB is not present in ILSVRC-2012

frame as candidate perturbation images[3]. This results in two datasets containing one anchor frame each from 3,139 videos, with approximately 20 candidate perturbation per anchor frame[4].

Next, we curate the dataset with the help of four expert human annotators. The goal of the curation step is to ensure that each anchor frame and its nearby frames are correctly labeled with the same ground truth class, and that the anchor frame and the nearby frames are visually similar.

Denser labels for Youtube-BB. As Youtube-BB contains only a single category label per frame at 1 frame per second, annotators first inspected each anchor frame individually and added any missing labels. In total, annotators corrected the labels for 834 frames, adding an average of 0.5 labels per anchor frame. These labels are then propagated to nearby, unlabeled frames at the native frame rate and verified in the next step. ImageNet-Vid densely labels all classes per frame, so we skipped this step for this dataset.

Frame pairs review. Next, for each pair of anchor and nearby frames, a human annotates (i) whether the pair is correctly labeled in the dataset, and (ii) whether the pair is similar. We took several steps to mitigate the subjectivity of this task and ensure high annotation quality. First, we trained reviewers to mark frames as dissimilar if the scene undergoes any of the following transformations: significant motion, significant background change, or significant blur change. We asked reviewers to mark each dissimilar frame with one of these transformations, or "other", and to mark a pair of images as dissimilar if a distinctive feature of the object is only visible in one of the two frames (such as the face of a dog). If an annotator was unsure about the correct label, she could mark the pair as "unsure". Second, we present only a single pair of frames at a time to reviewers because presenting videos or groups of frames could cause them to miss large changes due to the phenomenon of change blindness [170].

Verification. In the previous stage, all annotators were given identical labeling instructions and individually reviewed a total of 71,660 image pairs. To increase consistency in annotation, annotators jointly reviewed all frames marked as dissimilar, incorrectly labeled, or "unsure". A frame was only considered similar to its anchor if

[3]For `YTBB-Robust` we use a subset of the anchor frames used by Gu et al. [85].
[4]Anchor frames near the start or end of the video may have less than 20 candidate frames.

a strict majority of the annotators marked the pair as such.

After the reviewing was complete, we discarded all anchor frames and candidate perturbations that annotators marked as dissimilar or incorrectly labeled. The final datasets contain a combined total of 3,139 anchor frames with a median of 20 similar frames each.

3.3.2 The `pm-k` evaluation metric

Given the datasets introduced above, we propose a metric to measure a model's robustness to natural perturbations. In particular, let $A = \{a_1, ..., a_n\}$ be the set of valid anchor frames in our dataset. Let $Y = \{y_1, ..., y_n\}$ be the set of labels for A. We let $\mathcal{N}_k(a_i)$ be the set of frames marked as similar to anchor frame a_i. In our setting, $\mathcal{N}_k$ is a subset of the $2k$ temporally adjacent frames (plus/minus k frames from the anchor).

Classification. The standard classification accuracy on the anchor frame is $\text{acc}_{\text{orig}} = 1 - \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{0/1}(f(a_i), y_i)$, where $\mathcal{L}_{0/1}$ is the standard 0-1 loss function. We define the `pm-k` analog of accuracy as

$$\text{acc}_{\text{pmk}} \;=\; 1 - \frac{1}{N}\sum_{i=1}^{N} \max_{b \in \mathcal{N}_k(a_i)} \mathcal{L}_{0/1}(f(b), y_i)\;, \tag{3.1}$$

which corresponds to picking the worst frame from each set $\mathcal{N}_k(a_i)$ before computing accuracy. We note the similarity of the `pm-k` metric to standard ℓ_p-robustness. If we let $\mathcal{N}_k(a_i)$ be the set of *all* images within an ℓ_p ball of radius ϵ around a_i, then the notions of robustness are identical.

Detection. The standard metric for detection is mean average precision (mAP) of the predictions at a fixed intersection-over-union (IoU) threshold [142]. We define the `pm-k` metric analogous to that for classification: We replace each anchor frame with the nearest frame that minimizes the average precision (AP, averaged over recall thresholds) of the predictions, and compute `pm-k` as the mAP on these worst-case neighboring frames.

3.4 Main results

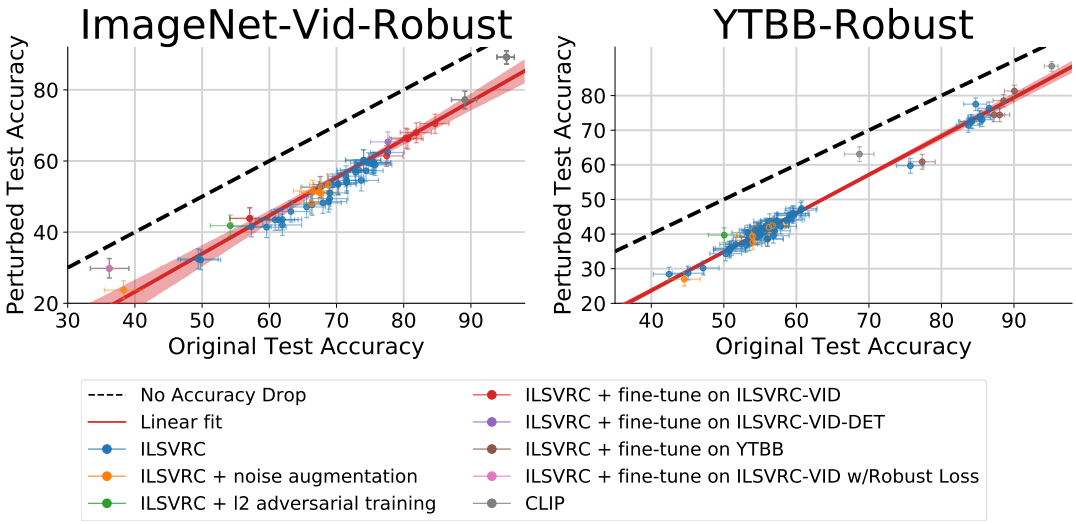


Figure 3.3: Model accuracy on original vs. perturbed images. Each data point corresponds to one model in our testbed (shown with 95% Clopper-Pearson confidence intervals). If models were robust to perturbations, we would expect them to fall on the dashed line ($y = x$). Instead, we find they all lie significantly below this ideal line, consistently exhibiting a significant accuracy drop to perturbed frames. Each perturbed frame was taken from a ten frame neighborhood (approximately 0.3 seconds) of the original frame, and reviewed by experts to confirm visual similarity to the original frame.

We evaluate a testbed of 47 classification models and three detection models on `ImageNet-Vid-Robust` and `YTBB-Robust`. We first discuss the various types of classification models evaluated with the `pm-k` classification metric. Second, we evaluate the performance of detection models on `ImageNet-Vid-Robust` using use the bounding box annotations inherited from ImageNet-Vid and using a variant of the `pm-k` metric for detection. We then analyze the errors made on the detection adversarial examples to isolate the effects of *localization* errors vs. *classification* errors. Finally, we analyze the impact of dataset review, video compression, and video frame rate on the accuracy drop.

Figure 3.4: Naturally perturbed examples for detection. Red boxes indicate false positives; green boxes indicate true positives; white boxes are ground truth. Classification errors are common failures, such as the fox on the left, which is classified correctly in the anchor frame, and misclassified as a sheep in a nearby frame. However, detection models also have *localization* errors, where the object of interest is not correctly localized in addition to being misclassified, such as the airplane (middle) and the motorcycle (right). All visualizations show predictions with confidence over 0.5.

3.4.1 Classification

The classification robustness metric is acc_{pmk} defined in Equation (3.1). For frames with multiple labels, we count a prediction as correct if the model predicts *any* of the correct classes for a frame. In Figure 3.3, we plot the benign accuracy, acc_{orig}, versus the robust accuracy, acc_{pmk}, for all classification models in our test bed and find a consistent drop from acc_{orig} to acc_{pmk}. Further, we note that the relationship between acc_{orig} and acc_{pmk} is approximately linear, indicating that while improvements in the benign accuracy do result in improvements in the worst-case accuracy, they do not suffice to resolve the accuracy drop due to natural perturbations. We provide implementation details and hyperparameters for all models in the supplementary.

Our test bed consists of six model types with increasing levels of supervision. We present results for representative models from each model type in Section 3.4.1.

ILSVRC Trained The WordNet hierarchy enables us to repurpose models trained for the 1,000 class ILSVRC-2012 dataset on `ImageNet-Vid-Robust` and `YTBB-Robust` We evaluate a wide array of ILSVRC-2012 models (available from [31]) against our

Table 3.2: Accuracies of six model types and the best performing model (shown with 95% Clopper-Pearson confidence intervals). Δ denotes accuracy drop between evaluation on anchor frame (acc_{orig}) and worst frame in similarity set (acc_{pmk}). The model architecture is ResNet-50 unless noted otherwise. 'FT' denotes 'fine-tuning.' See Section 3.4.1 for details.

Model Type	Accuracy Original	Accuracy Perturbed	Δ
`ImageNet-Vid-Robust`			
Trained on ILSVRC	67.5 [64.7, 70.3]	52.5 [49.5, 55.5]	15.0
+ Noise Augmentation	68.8 [66.0, 71.5]	53.2 [50.2, 56.2]	15.6
+ ℓ_∞ robustness (ResNext-101)	54.3 [51.3, 57.2]	40.8 [39.0, 43.7]	12.4
+ FT on ImageNet-Vid	80.8 [78.3, 83.1]	65.7 [62.9, 68.5]	15.1
+ FT PM-k loss on ImageNet-Vid	36.2 [33.3, 39.1]	29.8 [27.1, 32.5]	6.4
+ FT on ImageNet-Vid (ResNet-152)	84.8 [82.5, 86.8]	70.2 [67.4, 72.8]	14.6
+ FT on ImageNet-Vid-Det	77.6 [75.1, 80.0]	65.4 [62.5, 68.1]	12.3
CLIP Zero-Shot	95.3 [93.8, 96.4]	89.2 [87.2, 91.0]	6.1
`YTBB-Robust`			
Trained on ILSVRC	57.0 [54.9, 59.2]	43.8 [41.7, 46.0]	13.2
+ Noise Augmentation	62.3 [60.2, 64.4]	45.7 [43.5, 47.9]	16.6
+ ℓ_∞ robustness (ResNext-101)	53.6 [51.4, 55.8]	43.2 [41.0, 45.3]	10.4
+ FT on Youtube-BB	91.4 [90.1, 92.6]	82.0 [80.3, 83.7]	9.4
+ FT on Youtube-BB (ResNet-152)	92.9 [91.6, 93.9]	84.7 [83.0, 86.2]	8.2
CLIP Zero-Shot	95.2 [93.9, 95.8]	88.5 [87.0, 89.8]	6.7

Table 3.3: Detection and localization mAP for Faster R-CNN and R-FCN models. Both detection and localization suffer from significant mAP drops due to perturbations. (R-FCN was trained on ILSVRC Det and VID 2015, and evaluated on the 2015 subset of ILSVRC-VID 2017, indicated by *.)

Task	Model	mAP Original	mAP Perturbed	mAP Δ
Detection	FRCNN, ResNet 50	62.8	48.8	14.0
	FRCNN, ResNet 101	63.1	50.6	12.5
	R-FCN, ResNet 101 [245]*	79.4*	63.7*	15.7*
Localization	FRCNN, ResNet 50	76.6	64.2	12.4
	FRCNN, ResNet 101	77.8	66.3	11.5
	R-FCN, ResNet 101*	80.9*	70.3*	10.6*

Table 3.4: Impact of human review on `ImageNet-Vid-Robust` and `YTBB-Robust` on original and perturbed accuracy, using ResNet-152 fine-tuned on ImageNet-Vid and Youtube-BB, respectively.

		Accuracy		
	Reviewed	Original	Perturbed	Δ
`ImageNet-Vid-Robust`	✗	80.3	64.1	16.2
	✓	84.8	70.2	14.4
`YTBB-Robust`	✗	88.1	78.1	10.0
	✓	92.9	84.7	8.9

natural perturbations. Since these datasets present a substantial distribution shift from the original ILSVRC-2012 validation set, we expect the *benign* accuracy acc_{orig} to be lower than the comparable accuracy on the ILSVRC-2012 validation set. However, our main interest here is in the *difference* between the original and perturbed accuracies acc_{orig} - acc_{pmk}. A small drop in accuracy would indicate that the model is robust to small changes that occur naturally in videos. Instead, we find significant median drops of 15.0% and 13.2% in accuracy on our two datasets, indicating sensitivity to such changes.

Noise augmentation One hypothesis for the accuracy drop from original to perturbed accuracy is that subtle artifacts and corruptions introduced by video compression schemes could degrade performance when evaluating on these corrupted frames. The worst-case nature of the `pm-k` metric could then be focusing on these

corrupted frames. One model for these corruptions are the perturbations introduced in [92]. To test this hypothesis, we evaluate models augmented with a subset of the perturbations (exactly one of: Gaussian noise, Gaussian blur, shot noise, contrast change, impulse noise, or JPEG compression). We found that these augmentation schemes did not improve robustness against our perturbations substantially, and still result in a median accuracy drop of 15.6% and 16.6% on the two datasets.

ℓ_∞-robustness. We evaluate the model from [247], which currently performs best against ℓ_∞-attacks on ImageNet. We find that this model has a smaller accuracy drop than the two aforementioned model types on both datasets. However, the robust model achieves substantially lower original and perturbed accuracy than either of the two model types above, and the robustness gain is modest (3% compared to models of similar benign accuracy). In section 4.3 of Taori et al. [211], the authors further analyze the performance of ℓ_∞-robust models on `ImageNet-Vid-Robust` and `YTBB-Robust`.

Fine-tuning on video frames. To adapt to the new class vocabulary and the video domain, we fine-tune several network architectures on the ImageNet-Vid and Youtube-BB training sets. For Youtube-BB, we train on the anchor frames used for training in [85], and for ImageNet-Vid we use all frames in the training set. The resulting models significantly improve in accuracy over their ILSVRC pre-trained counterparts (e.g., 13% on `ImageNet-Vid-Robust` and 34% on `YTBB-Robust` for ResNet-50). This improvement in accuracy results in a modest improvement in robustness for `YTBB-Robust`, but still suffers from a substantial 9.4% drop. On `ImageNet-Vid-Robust`, there is almost no change in the drop from 15.0% to 15.1%.

Fine-tuning with a robust loss. Training on videos optimizes for the *average* accuracy on video frames. However, our goal at test-time is to improve the worst-case, PM-k accuracy. We adopt a strategy inspired by work in adversarial robustness [153], which uses the PM-k metric as the training loss. Specifically, for each frame x_t, let the standard training loss be for a model f be $L(x_t, y_t; f)$. We instead train the model using

$$\hat{L}(f(x_t), y_t) = \max_{\hat{x} \in \mathcal{N}_k(x_t)} L(f(\hat{x}), y_t),$$

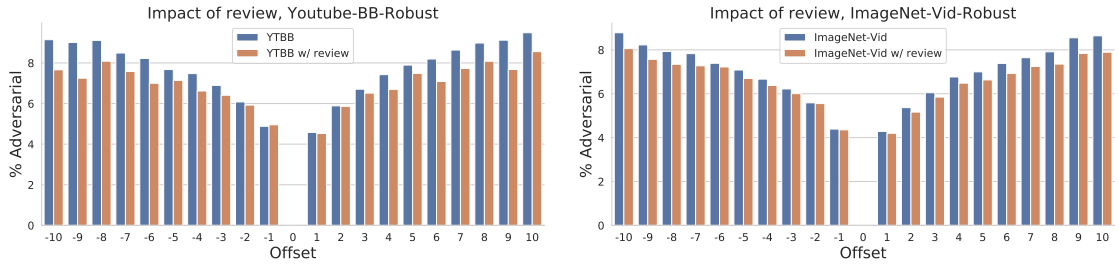


Figure 3.5: We plot how often each frame offset resulted in error, across all models, before and after review. Frames further away more frequently cause errors. Our review reduces errors by removing dissimilar frames, especially ones further away.

where $\mathcal{N}_k(x_t)$ contains all images within k frames of x_t with labels that match y_t. Unfortunately, this results in a drastic drop in both the original and perturbed accuracies by 31.3% and 22.7% respectively. However, the strategy does reduce the robustness gap from 15.1% to 6.4%, suggesting this loss may be a promising avenue for future improvements in robustness. We provide implementation details and further analysis of this model in the supplementary.

Fine-tuning for detection on video frames. We further analyze whether additional supervision in the form of bounding box annotations improves robustness. To this end, we train the Faster R-CNN *detection* model [190] with a ResNet-50 backbone on ImageNet-Vid. Following standard practice, the detection backbone is pre-trained on ILSVRC-2012. To evaluate this detector for classification, we assign the class with the most confident bounding box as label to the image. We find that this transformation reduces accuracy compared to the model trained for classification (77.6% vs. 80.8%). While there is a slight reduction in the accuracy drop caused by natural perturbations, the reduction is well within the error bars for this test set.

Contrastive Language-Image Pre-training (CLIP) Recent advancements in large scale contrastive learning has leveraged supervision from text to achieve high zero shot performance on down stream tasks [110, 182]. We evaluate the performance of the largest CLIP model[5] trained on 400 million image, text pairs from the internet.

[5]The underlying model was a large visual transformer evaluated on 336 x 336 images (ViT-L/14@336px)

We evaluate two versions of this model, a "zero-shot" variant trained solely on 400 million images, text pairs and a "linear-probe" variant where the last linear layer was fine-tuned on ILSVRC-2012. We find that the zero shot variant while still suffering from a 6% accuracy drop is significantly more robust and accurate than any of the other models in our test bed. We note that due to the sheer amount of training data and the size of the model, these models are *incredibly* expensive to train and are out of reach to the computational resources of most researchers. Thus we leave further investigation of the robustness of these models to future work.

3.4.2 Detection

We further study the impact of natural perturbations on object detection. Specifically, we report results for two related tasks: object localization and detection. Object detection is the standard computer vision task of correctly classifying an object and finding the coordinates of a tight bounding box containing the object. "Object localization", meanwhile, refers to only the subtask of finding the bounding box, *without* attempting to correctly classify the object.

We provide our results on `ImageNet-Vid-Robust`, which contains dense bounding box labels unlike Youtube-BB, which only labels boxes at 1 frame per second. We use the popular Faster R-CNN [190] and R-FCN [41, 245] architectures for object detection and localization and report results in Table 3.3. For the R-FCN architecture, we use the model from [245][6]. We first note the significant drop in mAP of 12 to 15 points for object detection due to perturbed frames for both the Faster R-CNN and R-FCN architectures. Next, we show that localization is indeed easier than detection, as the mAP is higher for localization than for detection (e.g., 76.6 vs 62.8 for Faster R-CNN with a ResNet-50 backbone). Perhaps surprisingly, however, switching to the localization task does *not* improve the drop between original and perturbed frames, indicating that natural perturbations induce both classification and localization errors. We show examples of detection failures in Figure 3.4.

[6]This model was originally trained on the 2015 subset of ImageNet-Vid. We evaluated this model on the 2015 validation set because the method requires access to pre-computed bounding box proposals which are available only for the 2015 subset of ImageNet-Vid.

3.4.3 Impact of Dataset Review

We analyze the impact of our human review, described in Section 3.3.1, on the classifiers in our testbed. First, we compare the original and perturbed accuracies of a representative classifier (ResNet-152 finetuned) on frames with and without review in Section 3.4.1. We find that before review, the gap between the two accuracies is 16.2 and 10.0 on `ImageNet-Vid-Robust` and `YTBB-Robust` respectively. Our review improves the original accuracy by 3 to 4% (by discarding mislabeled or blurry anchor frames), and improves perturbed accuracy by 5 to 6% (by discarding dissimilar frame pairs). As a result, our review reduces the accuracy drop by 1.8% on `ImageNet-Vid-Robust` and 1.1% on `YTBB-Robust`. These results indicate that the changes in model predictions are indeed due to a lack of robustness, rather than due to significant differences between adjacent frames.

To further analyze the impact of our review on model errors, we plot how frequently each offset distance from the anchor frame results in a model error across all model types in Figure 3.5. Larger offsets indicate pairs of frames further apart in time. For both datasets, we find that such larger offsets lead to more frequent model errors. Our review reduces the fraction of errors across offsets, especially for large offsets, which are more likely to display large changes from the anchor frame.

3.4.4 Video compression analysis

One concern with analyzing performance on video frames is the impact of video compression on model robustness. In particular, the 'mp4' videos in `ImageNet-Vid-Robust` contain 3 frame types: 'i-', 'p-', and 'b-' frames. 'p-frames' are compressed by referencing pixel content from previous frames, while 'b-frames' are compressed via references to previous and future frames. 'i-frames' are stored without references to other frames.

We compute the original and perturbed accuracies, as well as the accuracy drop for a subset without each frame type in Table 3.5. While there are modest differences in accuracy due to compression, our analysis suggests that the sensitivity of models is not significantly due to the differences in quality of frames due to video compression.

Table 3.5: Analyzing results based on compressed frame type (See Section 3.4.4).

	Accuracy			
	Original	Perturbed	Δ	# anchors
All frames	84.8	70.2	14.6	1109
w/o 'i-frames'	84.7	70.3	14.4	1104
w/o 'p-frames'	83.9	73.7	10.2	415
w/o 'b-frames'	85.4	73.2	12.2	699

Table 3.6: `ImageNet-Vid-Robust` subsets with fixed FPS.

	Accuracy			
FPS	Original	Perturbed	Δ	# Videos
25	87.3	73.3	14.0	292
29	87.7	74.9	12.8	383
30	78.3	61.7	16.6	313

3.4.5 FPS analysis

Next, we analyze how video frame rate impacts model accuracy. At low frame rates, nearby frames may be more likely to be dissimilar, or exhibit artifacts such as motion blur. We show in Table 3.6 that videos in `ImageNet-Vid-Robust` range from 25 to 30 FPS. We evaluate a fine-tuned ResNet-152 model on subsets of the dataset corresponding to different frame rates, and find that the *gap* between original and perturbed accuracy is similar across these subsets, and similar to the gap for the entire dataset. This suggests that low frame rates do not account for the drop in accuracy, and different frame rates do not significantly impact the results.

3.5 Discussion

We analyze and quantify a common phenomenon in image models: flicker in predictions over time, which is caused by a lack of model robustness to natural perturbations. We show this results in significant accuracy drops for a wide range of classification and detection models. We highlight two key avenues for future research:

Building more robust models. Our benchmarks provide a standard robustness measure for classification and detection models. In Section 3.4.1, we found that several

models suffer from substantial accuracy drops due to natural perturbations. Further, improvements with respect to artificial perturbations (like image corruptions or ℓ_∞ adversaries) induce only modest robustness improvements. One exception to this bleak overview are recent contrastive learning approaches trained on large-scale web data [182], which confer partial robustness to natural perturbations. We hope our standardized benchmarks will enable progress in improving the robustness of such models, and in generalizing their improvements to models trained on more limited datasets.

Further natural perturbations. Videos provide a straightforward method for collecting natural perturbations of images, enabling the study of realistic forms of robustness. Other methods for generating such natural perturbations are likely to provide additional insights into robustness. As an example, photo sharing websites contain many near-duplicate images: image pairs of the same scene captured at different times, viewpoints, or from a different camera [186]. More generally, devising similar, domain-specific strategies to collect, verify, and measure robustness to natural perturbations in domains such as natural language processing or speech recognition is a promising direction for future work.

Acknowledgements. We thank Rohan Taori for providing models trained for robustness to image corruptions, and Pavel Tokmakov for his help with training detection models on ImageNet-Vid. This research was generously supported in part by ONR awards N00014-17-1-2191, N00014-17-1-2401, and N00014-18-1-2833, the DARPA Assured Autonomy (FA8750-18-C-0101) and Lagrange (W911NF-16-1-0552) programs, an Amazon AWS AI Research Award, and a gift from Microsoft Research.

Chapter 4

Detecting Invisible People

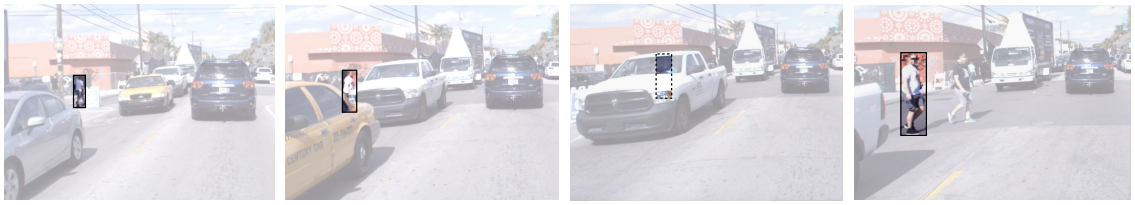


Figure 4.1: We visualize an online tracking scenario from Argoverse [36] that requires tracking a pedestrian through a complete occlusion. Such applications cannot wait for objects to re-appear (*e.g.*, as re-identification approaches do): autonomous agents must properly react *during* the occlusion. We treat online detection of occluded people as a *short-term forecasting* challenge.

4.1 Introduction

In the previous chapter, we explored the robustness of models to small, nearly imperceptible changes that occur between video frames, such as partial occlusions. In this chapter, we will explore solutions for addressing a more extreme version of this problem: what happens when objects become fully occluded, or *invisible* to the camera? Standard object detection methods have seen immense progress over the last few years, but require objects to be *visible to the camera* in the image. However,

objects that are fully occluded (and thus, invisible) continue to exist and move in the world. Indeed, object permanence is a fundamental visual cue exhibited by infants in as early as 3 months [10, 100]. Practical autonomous systems must similarly reason about objects under such occlusions to ensure safe operation (Figure 4.1). Interestingly, existing work on object detection and tracking tends to de-emphasize this capability, either choosing to completely ignore highly-occluded instances for evaluation [61, 142, 194, 249], or simply downweighting them because they occur so rarely that they fail to materially affect overall performance [158]. One reason that invisible-object detection may have been under-emphasized in the tracking community is that for *offline* analysis, one can post-hoc reason about the presence of an occluded object by relinking detections *after* it reappears. This approach has spawned the large subfield of reidentification (ReID). However, in an *online* setting (such as an autonomous vehicle that must make decisions given the available sensor information), intelligent agents must be able to instantaneously reason about occluded objects *before* they re-appear.

Problem formulation: We begin by introducing benchmarks and metrics for evaluating the task of detecting and tracking invisible people. To do so, we repurpose existing tracking benchmarks and introduce metrics for evaluating this task that appropriately reward detection of occluded people. To ensure benchmarks are online, we forbid algorithms from accessing future frames when reporting object states for the current frame. Although this task requires reasoning about object trajectories, it can be evaluated as both a *detection* and a *tracking* problem. For the latter, we introduce extensions to tracking metrics in the supplement. When analyzing our metrics, it becomes readily apparent that human annotation of ground-truth occluded objects is challenging. We provide pilot human vision experiments in Section 4.4 that show annotators are still consistent, but exhibit larger variation in labeling the pixel position of occluded instances. This suggests that algorithms for occluded object detection should report *distributions* over object locations rather than precise discrete (bounding box) locations. Inspired by metrics for evaluating multimodal distributions in the forecasting literature [36], we explore probabilistic algorithms that make k predictions which are evaluated by Top-k accuracy.

Analysis: Perhaps not surprisingly, our first observation is that performance of state-of-the-art detectors and trackers plummets on occluded people, from 68.5% to

28.4%; it is far easier to detect visible objects than invisible ones! This underscores the need for the community to focus on this underexplored problem. We introduce two simple but key innovations for addressing this task, which improve performance from 28.4% to 39.8%. (a) We recast the problem of online tracking of occluded objects as a *short-term forecasting* challenge. We explore state-of-the-art deep forecasting networks, but find that classic linear dynamics models (Kalman filters) perform quite well. (b) Because modeling occlusions is of central importance, we cast the problem as one of 3D tracking given 2D image measurements.

Novelty: While there exists considerable classic work on 3D tracking from 2D [28, 40, 192, 206], much focuses on 3D modeling of tracked objects. Instead, we find that the 3D structure of scene occluders is important for understanding where tracked objects can "hide". Typically such dense 3D understanding requires calibrated multiview sensors [59, 220]. Instead, we show that recent advances in uncalibrated *monocular depth estimation* provide "good enough" estimates of relative depth that still enable dense freespace reasoning. This is crucial because monocular depth has the potential to be far more scalable [233]. To our knowledge, ours is the first work to use uncalibrated depth estimates for multi-object tracking and detection of occluded objects.

Overview: After reviewing related work, we present our core algorithmic contributions, including straightforward but crucial extensions to classic linear dynamics models to (a) incorporate putative depth observations from a monocular network and (b) forecast object state even during occlusions. We conclude with extensive evaluations on three datasets [50, 158, 232] repurposed for detecting occluded objects.

4.2 Background

Amodal object detection aims to segment the full extent of objects that may be partially (but not *fully*) occluded. [273] introduces this task with a dataset labeled by multiple annotators, which is later expanded by [273]. More recently, [181] introduces a larger dataset of amodal annotations on the KITTI [75] dataset. Approaches in this setting largely rely on training variants of standard detectors (*e.g.* [90]) on amodal annotations generated synthetically from modal datasets [54, 139, 251, 266]. As this line of work addresses detection from a single image, it requires objects to be at

least *partially visible*. By contrast, we target fully occluded people, which cannot be recovered from a single frame.

Multi-object tracking requires tracking across partial and full occlusions. Approaches for this task address occlusions post-hoc in an *offline* manner, using appearance-based re-identification models to identify occluded objects after they become visible. These appearance-based models can be incorporated into tracking approaches, as part of a graph optimization problem [15, 178, 258] or online linking [16, 238]. In this work, we point out that some approaches *internally* maintain online estimates of the position of occluded people [16, 19, 238], but explicitly choose not to report these internal predictions, as they tend to be noisy and, thus, are penalized heavily by current benchmarks. We provide two simple extensions to these internal predictions that significantly improve detection of occluded people while preserving accuracy on visible people. [82] tracks occluded objects using contextual 'supporters', but requires a user to initialize a single object to track in uncluttered scenes; by contrast, we simultaneously detect and track people in large crowds.

Other work shares our motivation of tracking in 3D but relies on additional depth sensors [73] or stereo setups [35, 105]. Finally, many surveillance-based tracking systems explicity reason about object occupancy and occlusion, but require calibrated cameras to compute ground plane coordinates [1, 71, 104, 121, 124]. By contrast, our work emphasizes detection of *occluded* people in *uncalibrated, monocular* videos. To do so, we use monocular depth estimators via technical innovations that address noise in predicted depth estimates. Our method generalizes to arbitrary videos, since estimating monocular depth is far more scalable than retrieving additional sensor information for any video.

Forecasting approaches predict pedestrian trajectories in future, unobserved frames. These approaches leverage social cues from nearby pedestrians or semantic scene information to better model person trajectories [125, 133, 152, 172, 198, 250]. Recently, data-driven approaches have also been proposed for learning social cues [3, 193]. We note that detection of fully occluded people can be formulated as forecasting the trajectory of a visible person in future frames, where the positions of the occluded person are unobserved, but the rest of the frame *can* be observed. Our approach uses a constant-velocity model to forecast trajectories, equipped with depth cues from the observed frames, to improve detection of occluded people. In Section 4.4.3,

we show that while this approach can use a more powerful forecasting model, the constant-velocity approximation is sufficient in our setting.

4.3 Method

We build an online approach for detecting invisible people starting with a simple tracker, using estimated trajectories of visible people to forecast their location during occlusions. We describe our tracking mechanism, building upon [239]. While such trackers *internally* forecast the location of occluded people for improved tracking, these forecasts tend to be noisy and cannot directly localize occluded people. To address this, we incorporate depth cues from a monocular depth estimator to reason about occlusions in 3D.

4.3.1 Background

To detect people during occlusions, we build on a simple online tracker [239] that estimates the trajectories of visible people. We briefly describe aspects relevant to our approach, but refer the reader to [239] for a more detailed explanation. In the first frame, this tracker instantiates a track for each detected person. The tracker adds each track to its "active" set, representing people that have been seen so far. Each track maintains a Kalman Filter whose state space encodes the position (x, y), aspect ratio (a), height (h), and corresponding velocities ($\dot{x}, \dot{y}, \dot{a}, \dot{h}$) of the person.The filter's process model assumes a constant velocity model with gaussian noise (i.e., $x_t = x_{t-1} + \dot{x_{t-1}} + \epsilon_x$). At each successive frame, the tracker first runs the *predict* step of the filter, using the process model to forecast the location of the track in the new frame. Next, each detection in the current frame is matched to this set of active tracks based on appearance features, and distance to the tracks' forecasted location (as estimated by the filter). A new track is created for all detections that are unmatched. If a track is matched to a detection, the detection is used as a new observation to update the track's filter, and the detection is reported as part of the track. Importantly, if a track does not match to any detection, its forecasted box is *not* reported. When a track is not matched to a detection for more than N_{age} frames, it is deleted.

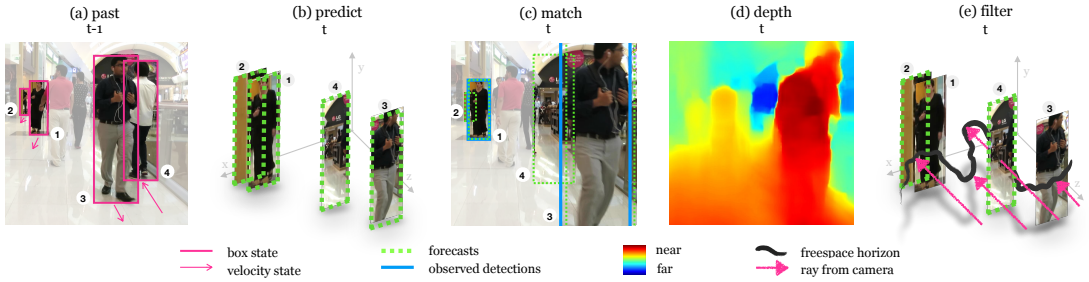


Figure 4.2: (a) Frame $t-1$ has active tracks $\{1, 2, 3, 4\}$, each with an internal state of its 2D position, size, velocity, and *depth* (see text). (b) We forecast tracks in 3D for frame t. (c) Tracks are matched to observed detections at t using spatial and appearance cues. Matched tracks are considered visible (*e.g.* 1, 3). Tracks which don't match to a visible detection (*e.g.* 2, 4) may be occluded, or simply incorrectly forecasted. (d) To resolve this ambiguity, we leverage depth cues from a monocular depth estimator, to compute (e) the *freespace horizon*. The region between the camera and the horizon must be freespace, while the area beyond it is unobserved, and so may contain *occluded* objects. Tracks lying beyond the freespace horizon are reported as occluded (*e.g.* 2). Tracks *within* freespace (*e.g.* 4) should have been visible, but did not match to any visible detections. Hence, we assume these tracks are incorrectly forecasted, and we delete them.

4.3.2 Short-term forecasting across occlusions

Although this tracker *internally* forecasts the positions of all tracks at each step, its estimates are used only to improve the association of tracks to detections, and are not reported externally. However, these internally forecasted track locations are crucial as they may correspond to an occluded person. We show that naively reporting these track locations leads to significant *recall* of occluded people, but the noise in these estimates results in poor precision. Further, these noisy estimates lead to a small decrease in *overall* accuracy, as standard benchmarks largely focus on visible people. We improve these estimates by augmenting them with 3D information. Specifically, we use a monocular depth estimator [141] to get per pixel depth estimates of the scene. We then augment our Kalman Filter state space with the *inverse* depth. Inverse depth is a commonly used representation predicted by depth estimators [132, 141] due to important benefits, including the ability to represent points at infinity and ability to model uncertainty in pixel disparity space (commonly used for stereo-based depth estimation [166]). Our state space thus additionally includes $1/z$ variable.

4.3.3 Tracking in 3D camera coordinates using 2D image coordinates

Equipped with depth estimates, we formulate tracking with a constant velocity model in 3D using 2D measurements. Unlike prior work which assumes linear dynamics in (projected) 2D image measurements, our dynamics model operates in 3D using depth cues, resulting in far more realistic person trajectories. We derive our uncalibrated tracker by demonstrating that the unknown camera focal length f can be folded into a motion noise parameter that can be easily tuned on a training set. Hence our final method runs without calibration on arbitrary videos.

Let us model objects as cylinders with centroids (X_t, Y_t, Z_t), height H and aspect ratio A_t. We model object height as constant, but allow for varying aspect ratios because people are non-rigid. We can then compute image-measured bounding boxes with centroid (x_t, y_t) and dimensions (h_t, a_t) as follows:

$$x_t = f\frac{X_t}{Z_t}, \quad y_t = f\frac{Y_t}{Z_t}, \quad h_t = f\frac{H}{Z_t}, \quad a_t = A_t \tag{4.1}$$

We extend the commonly used constant velocity model with Gaussian noise from 2D [19, 238] to 3D:

$$X_t = X_{t-1} + \dot{X}_{t-1} + \epsilon_X, \quad \epsilon_X \sim \mathcal{N}(0, \sigma_X), \tag{4.2}$$

where similar equations hold for Y_t, Z_t and A_t. Let the observed (inverse) depth from a depth estimator associated with an object be $1/z_t$. Since image measurements are given by perspective projection of real world coordinates, we have the following equations (assuming Gaussian image noise):

$$x_t = f\frac{X_t}{Z_t} + \epsilon_x, \quad \epsilon_x \sim \mathcal{N}(0, \sigma_x) \tag{4.3}$$

$$\frac{1}{z_t} = \frac{1}{Z_t} + \epsilon_z, \quad \epsilon_z \sim \mathcal{N}(0, \sigma_z) \tag{4.4}$$

with similar equations for y_t, h_t, and a_t. Note that inverse depth naturally assumes a large uncertainty in far away regions, and a small uncertainty in nearby regions.

Defining a 3D state space leads us to a modified formulation, written as

$$\left(f\frac{X_t}{Z_t}, f\frac{Y_t}{Z_t}, \frac{1}{Z_t}, A_t, f\frac{H}{Z_t}, f\frac{\dot{X}_t}{Z_t}, f\frac{\dot{Y}_t}{Z_t}, \dot{A}_t\right) \tag{4.5}$$

We can therefore rewrite Equation (4.2) as:

$$f\frac{X_t}{Z_t} \approx f\frac{X_t}{Z_{t-1}} = f\frac{X_{t-1}}{Z_{t-1}} + f\frac{\dot{X}_{t-1}}{Z_{t-1}} + f\frac{\epsilon_X}{Z_{t-1}} \tag{4.6}$$

$$x_t \approx x_{t-1} + \dot{x}_{t-1} + f\frac{\epsilon_X}{Z_{t-1}} \tag{4.7}$$

where the approximation holds if depths are smooth over time ($Z_t \approx Z_{t-1}$). Technically, the above is no longer a linear dynamics model since the noise depends on the state. But the equation suggests that *one can approximately apply a Kalman filter on 2D image measurements augmented with a temporal noise model that is scaled by the estimated inverse-depth of the object.* Intuitively, this suggests that one should enforce smoother tracks for objects far away. Our approach thus scales the process noise (ϵ_X) for far away objects, leading to more accurate predictions. Algorithmically, [239] by default scales process and observation noise covariances according to the person's height; our approach instead multiplies the process covariance by the person's estimated depth, computed by aggregating past monocular depth observations and state estimates over time.

Assumptions. Because we do not assume calibrated cameras, we do not know f. Rather, we make use of training videos provided in standard tracking benchmarks and simply tune scaled variances $\sigma'_X = f\sigma_X$ directly on the training set. We make two additional assumptions: that people move with constant velocity in 3D, and that depth estimates are smooth over time. Although these do not always hold in real world scenarios, we empirically find that our method generalizes to diverse scenarios.

Filtering estimates lying in freespace. Equipping our state space with depth information allows us to forecast 3D trajectories. Meanwhile, applying a monocular depth estimator allows us to determine regions in 3D space that are occluded to the camera without requiring calibration. Specifically, if our approach forecasts a person at a point $P_f = (x_f, y_f, z_f)$, we can determine whether P_f should be visible to the camera by estimating whether P_f lies in the freespace [59] between the camera and

its nearest occluder. In the filter stage in Figure 4.2, we visualize one slice of the "freespace horizon": points beyond this horizon are occluded, while points between the camera and the horizon should be visible.

Concretely, let z_o be the (observed) depth of the horizon at (x_f, y_f). If the forecasted depth (z_f) lies closer to the camera than the horizon depth (z_o), as with person "4" in Figure 4.2 (e), then the person must be in the *freespace* between the camera and its closest object, and therefore visible. If we *do not* detect this person, then we assume the forecast is an error, and either suppress the forecasted box for the current frame (in the case of small errors, when $z_f < \alpha_{\text{supp}} z_o$) or delete the track entirely (for large errors, when $z_f < \alpha_{\text{delete}} z_o$). A key advantage of this approach is the ability to reason about occlusions arising not only from interactions between tracked people, but also from natural occluders such as trees or cars. Section 4.4.3 shows that this modification is critical for improving the precision of our trajectory forecasts.

Camera motion. Camera motion is challenging, as our approach assumes linear dynamics for trajectories. To address this, we follow prior work (e.g., [16]) in estimating a non-linear pixel warp W between neighboring frames which maps pixel coordinates (x_{t-1}, y_{t-1}) in one frame to the next (x_t, y_t). This warp is then used to align boxes forecasted using frames up to $t-1$ with frame t. Note that this alignment assumes the motion of dynamic objects is small relative to the scene motion, allowing for the use of an image registration algorithm [60]. Despite the simplicity of this modification, we show in the supplement that it helps considerably for the moving camera sequences. We also detail our algorithm with pseudo-code in the supplement. We proceed to an empirical analysis of the task and prior methods, showing the benefits of each component of our proposed approach.

4.4 Experimental Results

We first describe our proposed benchmarks, including the datasets and our proposed metrics for evaluating the task of detecting occluded people. Next, we conduct an oracle study in Section 4.4.1 to analyze how well existing approaches can detect occluded people. We then compare our proposed approach to these state-of-the-art approaches in multiple settings in Section 4.4.2. Finally, we analyze each component

Figure 4.3: We visualize bounding boxes labeled by multiple (4) in-house annotators (**left**). During small occlusions, annotators strongly agree. During large occlusions (less than 10% visible, last frame), annotators still agree to a fair extent (average IoU overlap of 60%, **right**), but require temporal video context. We use these to justify our Top-k evaluation and motivate our probabilistic tracking approach.

of our approach with a detailed ablation study in Section 4.4.3.

Dataset. Evaluating our approach is challenging, as most datasets do not annotate occluded objects. The MOT-17 [158], MOT-20 [50] and PANDA [232] datasets are key exceptions which label both visible and occluded people, along with a *visibility* field indicating what portion of the person is visible to the camera. We find that a majority of the annotations in these datasets (over 85% in each dataset) are people that are at least partially visible, leading standard evaluations on these datasets to underemphasize occluded people. To address this, we separately evaluate accuracy on the subset of fully *occluded* people (indicated by $< 10\%$ visibility). MOT-17 contains 7 sequences with publicly available groundtruth, and 7 test sequences with held-out groundtruth. We evaluate on these 14 sequences. MOT-20 contains 8 sequences, of which 4 have held-out groundtruth. PANDA officially releases a high-resolution 2FPS groundtruth for its 10 train and 5 test sequences. Because tracking and forecasting is challenging at such low frame rates, we reached out to the authors who provided a high-frame rate (30FPS), low-resolution groundtruth for 9 train videos. We report results on MOT-20 and PANDA train set without tuning our pipeline on any of the videos in these datasets. From visual inspection, we found that visibility labels in PANDA tend to be noisy (see the supplement), and so we define objects with up to 33% visibility as occluded. We carry out the analysis including oracle and ablation study on MOT-17 train and report the final results on MOT-17 test, MOT-20 and PANDA datasets. In all, these three datasets target a diverse set of application scenarios – static surveillance cameras, car-mounted cameras, and hand-held cameras.

Table 4.1: Oracle ablations on MOT-17 train reporting Top-5 F1, Top-1 F1 and IDF1 for occluded and all people, using Faster R-CNN detections. 'Occl strat' stands for Occlusion Strategy. We report the Top-5 mean and standard deviation for 3 runs.

Detections	Tracks	Occl Strat	Online?	Top-5				Top-1 F1	
				Occl F1	Occl Prec	Occl Rec	All F1	Occl	All
Groundtruth (vis.)	Groundtruth	Interpolate	✗	87.3 ±0.1	83.8 ±0.2	91.1 ±0.1	98.0 ±0.0	79.8	96.8
Faster R-CNN	Groundtruth	Interpolate	✗	46.4 ±0.1	65.5 ±0.1	35.9 ±0.1	70.5 ±0.0	34.4	68.1
Groundtruth (vis.)	DeepSORT	Interpolate	✗	53.3 ±0.2	86.7 ±0.1	38.5 ±0.2	92.3 ±0.0	44.4	92.0
Faster R-CNN	DeepSORT	Interpolate	✗	32.2 ±0.0	60.8 ±0.2	21.9 ±0.0	69.9 ±0.0	23.2	68.4
Faster R-CNN	DeepSORT	Forecast	✓	29.8 ±0.2	29.5 ±0.4	30.2 ±0.1	69.4 ±0.0	20.9	66.5

Metric. As most benchmarks consist primarily of visible people, existing metrics which measure performance across all people underemphasize the accuracy of detecting occluded people. We propose detection and tracking metrics (see supplement for latter) which evaluate accuracy on occluded people, as indicated by visibility $< 10\%$ and on all (visible and invisible) people. Since localizing fully-occluded people involves higher positional uncertainty than visible people, we allow algorithms to predict k potential locations for each person.

Top-k F1: We start by modifying the standard detection evaluation protocol [61, 142]. For every person, we allow methods to report k predictions, $P = \{p_1, p_2, \ldots, p_k\}$. We match these predictions to all groundtruth boxes based on intersection-over-union (IoU). We define the overlap between a groundtruth g and P as the maximum overlap with the predictions p_i in P — *i.e.*, $\text{IoU}(g, P) = \max_i \text{IoU}(g, p_i)$. We use this overlap definition and perform standard matching between predictions and groundtruth, with a minimum overlap threshold of α_{IoU}.

When evaluating accuracy across all people, matched groundtruth boxes are true positives (TP), all unmatched groundtruth are false negatives (FNs, or misses), and unmatched detections are false positives (FP). When evaluating accuracy on occluded people, only matched *occluded* groundtruth boxes count as TPs, only unmatched *occluded* groundtruth boxes count as FNs, and all unmatched detections count as FPs. Intuitively, when evaluating metrics for occluded people, we do not penalize a detector for correctly detecting a visible person, but we *do* penalize it for false positives that do not match any visible or occluded person.

We now describe how the k-vector of predictions is obtained: in addition to a state mean (first sample), our probabilistic method maintains covariances for x and z state

variables which result in a 2D gaussian. Since these gaussians may extend incorrectly into freespace, we perform rejection sampling to accumulate k-1 predictions which respect freespace constraints. This gives us P. For baseline methods that are not probabilistic or do not have access to a depth map, we artificially simulate this distribution by tuning two scale factors that control the size of gaussians as a function of a bounding box's height. We tune these scale factors on MOT-17 train and use them throughout experiments.

Top-1 F1: When $k = 1$, this metric is simply the standard F1 metric. We additionally report this Top-1 F1 for occluded and *all* people. We do not use the standard 'average precision' (AP) metric as most detectors and trackers on the MOT and PANDA datasets do not report confidences.

To guide evaluation, we conduct a human vision experiment with 10 in-house annotators who annotated 991 boxes in 59 tracks with occlusion phases. Figure 4.3 shows that annotators have lower consistency when labeling occluded people than visible people. To address this ambiguity in localizing occluded people, we choose a low $\alpha_{IoU} = 0.5$ and $k = 5$ in our experiments.

Implementation details. We empirically set parameters in our approach on MOT-17 train with Faster R-CNN [190] detections. The optimal thresholds for filtering forecasts on the train set are $\alpha_{\text{delete}} = 0.88, \alpha_{\text{supp}} = 1.06$[1]. During occlusion we treat a person as a point, freezing its aspect ratio and height. We fix N_{age} to 30. The supplement presents further details of our method, parameters and their tuning protocol, including improvements by tuning N_{age}. We tune on MOT-17 train and apply these tuned parameters on MOT-17 test, MOT-20, and PANDA. We find that our method and its hyperparameters tuned on the train set generalize well to the test set. We use [141] for monocular depth estimates, which has been shown to work well in the wild. While these estimates can be noisy, we qualitatively find that the *relative* depth orderings used in our approach are fairly robust.

[1]Note that $\alpha_{\text{supp}} > 1$ allows the forecasted depth to be closer to the camera than the observed depth, accounting for potential noise in the depth estimator to reduce the number of forecasts that are suppressed.

4.4.1 Oracle Study

What is the impact of *visible* detection on occluded detection? We first evaluate an offline approach which uses groundtruth detections and tracks for visible people to (linearly) interpolate detections for occluded people in Table 4.1. As this method perfectly localizes visible people, and most people in this benchmark are visible, it achieves a high overall Top-5 F1 of 98.0 (Table 4.1, row 1). Additionally, despite using simple linear interpolation, this oracle also achieves a high Top-5 F1 of 87.3 for *invisible* people. This result indicates that although long-term forecasting of pedestrian trajectories may require higher-level reasoning [133, 152, 198], short-term occlusions may be modeled with simple linear models.

Next, we evaluate the same approach with detections from a Faster R-CNN [190] model in place of groundtruth (Table 4.1, row 2). This leads to a significant drop in both overall and occluded accuracy, indicating that improvements in *visible* person detection can improve detection for invisible people. Finally, although Occluded Top-5 F1 drops, it is significantly above chance, suggesting that current detectors equipped with appropriate trackers can detect invisible people.

What is the impact of *tracking* on occluded detection? So far, we have assumed oracle linking of detections, allowing for linear interpolation of bounding boxes to detect people through occlusion. We now evaluate the impact of using an online tracker, equipped with re-identification, on detecting occluded people. Removing the oracle results in a drastic drop in accuracy: the Top-5 F1 score for occluded people drops by over 30 points (87.3 to 53.3, Table 4.1 row 3) using groundtruth detections, and 14 points with Faster R-CNN detections (46.4 to 32.2, Table 4.1 row 4). Despite this significant drop in Occluded Top-5 F1, the overall Top-5 F1 is significantly more stable (from 98.0 to 92.3 for groundtruth detections and 70.5 to 69.9 for Faster R-CNN), showing that *overall* person detection underemphasizes the importance of detecting occluded people.

Can online approaches work? These results indicate that in the offline setting, existing visible-person detection and tracking approaches can detect invisible people via interpolation. We now evaluate a simple *online* approach, which uses an off-the-shelf visible person detector (Faster R-CNN), equipped with a tracker (DeepSORT) and linear (constant velocity) forecasting for detecting invisible people (Table 4.1, row

Table 4.2: Detection and tracking results on MOT-17 [158], MOT-20 [50] and PANDA [232] train. We evaluate on public detections provided with MOT-17 (DPM [68], FRCNN [190], SDP [252]), two trackers that operate on public detections (Tracktor++ [16], MIFT [99]), and CenterTrack [270] which does not use public detections. We use (public FRCNN, *visible* groundtruth) detections for (MOT-20, PANDA). Our method improves on occluded people across all trackers.

		Top-5 F1		Top-1 F1	
		Occl	All	Occl	All
MOT-17	DPM	17.2	46.7	13.2	46.5
	+ Ours	24.6 (+7.4)	49.3 (+2.6)	17.4	48.4
	FRCNN	28.4	68.5	20.1	67.4
	+ Ours	39.8 (+11.4)	70.5 (+2.0)	26.7	68.5
	SDP	45.2	80.5	35.8	79.8
	+ Ours	51.2 (+6.0)	80.8 (+0.3)	38.5	79.4
	Tracktor++	32.4	77.0	22.7	76.8
	+ Ours	45.4 (+13.0)	77.2 (+0.2)	33.2	76.5
	MIFT	37.8	75.9	29.9	75.1
	+ Ours	44.9 (+7.1)	75.6 (-0.3)	33.8	74.3
	CTrack	38.7	84.8	29.4	84.2
	+ Ours	47.9 (+9.2)	84.4 (-0.4)	36.4	83.4
MOT-20	FRCNN	42.5	71.2	27.5	70.7
	+ Ours	46.1 (+3.6)	71.5 (+0.3)	28.6	70.9
PANDA	GT (visible)	45.5	90.6	30.5	90.5
	+ Ours	49.5 (+4.0)	90.5 (-0.1)	34.1	90.3

5). Moving to an online setting results in a similar Top-5 F1 score but significantly reduces the precision for occluded persons, from 60.8 to 29.5. This is expected as even though linear forecasting recalls slightly more number of boxes than offline interpolation (recall from 21.9 to 30.2), its naive nature results in many more false positives resulting in a much lower precision and therefore, a similar F1 score. In Section 4.4.3, we present simple modifications to this approach that recover much of this performance gap.

4.4.2 Comparison to Prior Work

Next, we apply our approach to the output of existing methods to evaluate its improvement over prior work. Table 4.2 shows results on the MOT-17 train set,

Table 4.3: Results on MOT-17 and MOT-20 test set. The best, second-best and third-best methods are highlighted.

		Top-5 F1		Top-1 F1	
		Occl	All	Occl	All
MOT-17	Ours	43.4	76.8	31.4	75.6
	MIFT [99]	38.4	77.3	29.7	76.7
	UnsupTrack [118]	35.9	78.1	26.6	77.4
	GNNMatch [168]	35.2	74.3	26.3	73.7
	GSM_Tracktor [144]	35.4	73.8	26.2	73.2
	Tracktor++ [16]	33.3	73.3	24.8	73.0
MOT-20	Ours	46.9	76.7	33.3	75.2
	Tracktor++ [16]	44.2	76.0	34.2	75.3
	UnsupTrack [118]	41.7	71.4	30.9	70.8
	SORT20 [239]	38.5	65.2	27.3	63.6

showing our approach improves significantly in Occluded Top-5 F1 ranging from 6.0 to 13.0 points, while maintaining the overall F1. Detecting invisible people requires reliable amodal detectors for visible people (ref. Section 4.4.1). For this reason, we use *visible* groundtruth detections from PANDA, similar to the oracle experiments in Section 4.4.1, as no public set of amodal detections come with PANDA (unlike MOT-17 or MOT-20). Table 4.2 shows that our method improves the detection of occluded people by 4.0% on PANDA using groundtruth visible detections and by 3.6% on MOT-20 using the Faster-RCNN public detections. We explicitly do not tune our hyperparameters for these two datasets, showing that our method is robust to changes in video data distribution. MOT-20 and PANDA contain a few sequences with top-down views, where occlusions are rare. We disable our depth and occlusion reasoning on such sequences; please see supplement.

As MOT-17 and MOT-20 test labels are held out, we worked with the MOTChallenge authors to implement our metrics on the test server. Table 4.3 shows that MIFT[2][99] and Tracktor++ [16] achieve the highest Occluded Top-5 F1 amongst prior online approaches on MOT-17 and MOT-20 test respectively. Applying our approach on top of these methods improves results significantly by 5.0% to 43.4 F1 and by 2.7% to 46.9 F1, leading to a new state-of-the-art for occluded person detection on

[2]MIFT is referred to as ISE_MOT17R on the MOT leaderboards

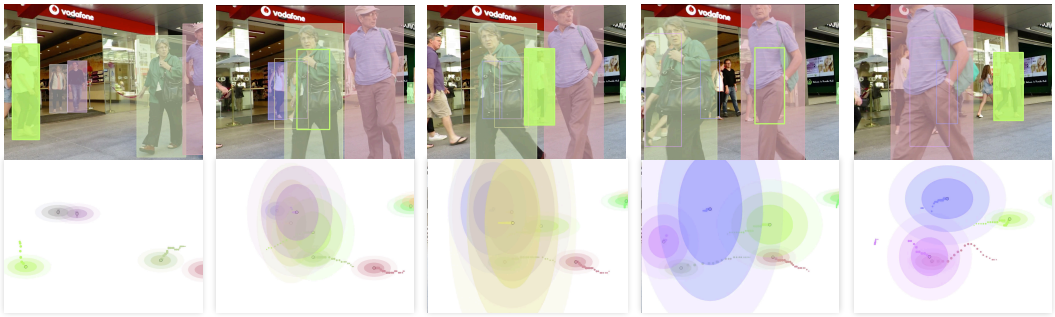


Figure 4.4: Our probabilistic model reports a *distribution* over 3D location during occlusions. We visualize (occluded, visible) detection with (outlined, filled-in) bounding boxes (**top**). We provide "birds-eye-view" top-down visualizations of Gaussian distributions over 3D object centroids with covariance ellipses (**bottom**). During occlusion, variance grows roughly linearly with the number of consecutively-occluded frames. We are also able to correctly predict depth of occluded people in the top down view, e.g. in the second last frame, which would not be possible with single-frame monocular depth estimates. During evaluation, we truncate the uncertainty using our freespace estimates (not visualized). Please refer to the supplement video.

MOT-17 and MOT-20 test.

Table 4.2 shows that our method consistently improves occluded F1. However, it sometimes results in a drop in overall accuracy. We attribute this to the increased number of false positives introduced while tackling the challenging task of detecting invisible people. These false positives for invisible people are counted as false positives for *all* people, whether visible or invisible. This causes existing metrics to penalize methods for even *trying* to detect invisible people. In safety critical applications, where worst-case accuracy may be more appropriate, our approach significantly improves during complete occlusions by up to 13.0% on MOT-17, while mildly decreasing average accuracy by 0.4%.

4.4.3 Ablation Study

We now study the impact of each component of our approach in Table 4.4, focusing on the Occluded Top-5 F1 metric using Faster R-CNN detections on the MOT-17 train set. First, we show that the DeepSORT tracker, upon which our approach is built, results in a 28.4 Occluded Top-5 F1. Reporting the internal, linear forecasts from the tracker increases the score to 29.8, driven primarily by a 12.5% improvement in

Table 4.4: MOT-17 train ablations. Each row adds a component to the row above. 'Dep. noise' is depth-aware noise.

	Top-5				Top-1 F1	
	Occl F1	Occl Prec	Occl Rec	All F1	Occl	All
DeepSORT	28.4 ±0.1	71.9 ±0.2	17.7 ±0.1	68.5 ±0.0	20.1	67.4
+ Forecast	29.8 ±0.2	29.5 ±0.4	30.2 ±0.1	69.4 ±0.0	20.9	66.5
+ Egomotion	32.2 ±0.2	33.1 ±0.3	31.3 ±0.1	70.4 ±0.0	23.2	67.9
+ Freespace	35.7 ±0.0	47.7 ±0.1	28.6 ±0.0	70.4 ±0.0	25.7	68.4
+ Dep. noise	39.8 ±0.2	52.6 ±0.6	32.0 ±0.0	70.5 ±0.1	26.7	68.5

recall. Compensating for camera motion provides another 2.4% improvement. Next, leveraging depth cues to incorporate freespace constraints, as detailed in Section 4.3.3, improves accuracy by 3.5%, driven primarily by a 14.6% jump in precision, indicating that this component drastically reduces false positives. Finally, we add depth-aware process noise to handle perspective transformations between 2D and 3D coordinates, which leads to an improvement of 4.1%, resulting in a final score of 39.8. Only a 1.0% improvement in F1 as compared to 4.1% with Top-5 F1 suggests that our uncertainty estimates are significantly improved by the depth-aware process noise scaling. In all, our approach leads to an improvement of 11.4% over the baseline. Figure 4.4 presents a sample result from our approach, where the person in the green bounding box is detected throughout two full occlusion phases, marked with an unfilled box.

One concern with our approach might be that the average depth inside a person's bounding box may contain pixels from the background or an occluder. To verify the impact of this, we evaluate a variant where we use segmentation masks for all the bounding boxes in MOT-17's FRCNN public detections using MaskRCNN [90]. We initialize the z state variable in the model with the average depth inside this mask. On doing so, the Top-1 occluded F1 increases from 26.7 to 27.3, indicating that masks can help with estimating the person's depth, but boxes are a reasonable approximation. We kindly refer the reader to our supplement for further ablative analysis, including an analysis of more recent depth estimators, ablations on moving *vs*. stationary sequences, and failure cases (in supplementary video).

Forecasting: We evaluate replacing our linear forecaster with state-of-the-art forecasters. We supply these forecasters with a birds-eye-view representation of visible person trajectories. As these forecasters forecast only the birds-eye-view

(x, z) coordinates, we rely on our approach's estimates of the height, width, and y coordinate. We evaluate two trajectory forecasting approaches for crowded scenes, Social GAN (SGAN) [87] and STGAT [101]. SGAN and STGAT result in Occluded Top-5 F1 scores of 36.0 and 36.4 respectively. While this improves over the baseline at 28.4, it underperforms our linear forecaster at 39.8. This suggests that simple linear models suffice for short, frequent occlusions. We refer the reader to the supplement for more details and analysis.

4.5 Discussion

We propose the task of detecting fully-occluded objects from uncalibrated monocular cameras in an online manner. Our experiments show that current detection and tracking approaches struggle to find occluded people, dropping in accuracy from 68% to 28% F1. Our oracle experiments reveal that interpolating across tracklets in an offline setting noticeably improves F1, but the task remains difficult because underlying object detectors do not perform well during large occlusions. We propose an online approach that forecasts the trajectories of occluded people, exploiting depth estimates from a monocular depth estimator to better reason about potential occlusions. Our approach can be applied to the output of existing detectors and trackers, leading to significant accuracy gains of 11% over the baseline, and 5% over state-of-the-art. We hope our problem definition and initial exploration of this safety-critical task encourages others to do so as well.

Acknowledgements. We thank Gengshan Yang for his help with generating 3D visuals, Patrick Dendorfer for incorporating our metrics with the MOT challenge server, and Xueyang Wang for sharing the low-resolution version of the PANDA dataset. We thank Laura Leal-Taixé and Simon Lucey for insightful discussions, internal reviewers at the Robotics Institute, CMU for reviewing early drafts, and participants of the human vision experiment. This work was supported by the CMU Argo AI Center for Autonomous Vehicle Research, the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051, and the National Science Foundation (NSF) under grant number IIS-1618903.

Part II

Generalizing to large vocabularies

Chapter 5

TAO: A Large-Scale Dataset for Tracking Any Object

5.1 Introduction

In the previous part, we analyzed how computer vision models generalize to in-the-wild settings, where the appearance of objects and scenes may differ from that of the training distribution. This work largely focused on models for a few objects, such as people or cars. Now, we explore how well models can generalize to *large* vocabularies of hundreds or thousands of object classes. Scaling to such large vocabularies leads to new challenges for dataset collection and annotation, method development, and evaluation. Building a dataset which covers a large vocabulary of classes can require a significant annotation budget, and evaluation strategies for small vocabulary tasks do not readily transfer to large vocabulary settings. We address these challenges in the following two chapters. We start by describing a new dataset we have built for large vocabulary object tracking in Chapter 5, and then highlight issues (and solutions) in current evaluations of large vocabulary methods in Chapter 6.

A key component in the success of modern object detection methods was the introduction of large-scale, diverse benchmarks, such as MS COCO [142] and LVIS [88]. By contrast, multi-object tracking datasets tend to be small [158, 225], biased towards short videos [254], and, most importantly, focused on a very small vocabulary of

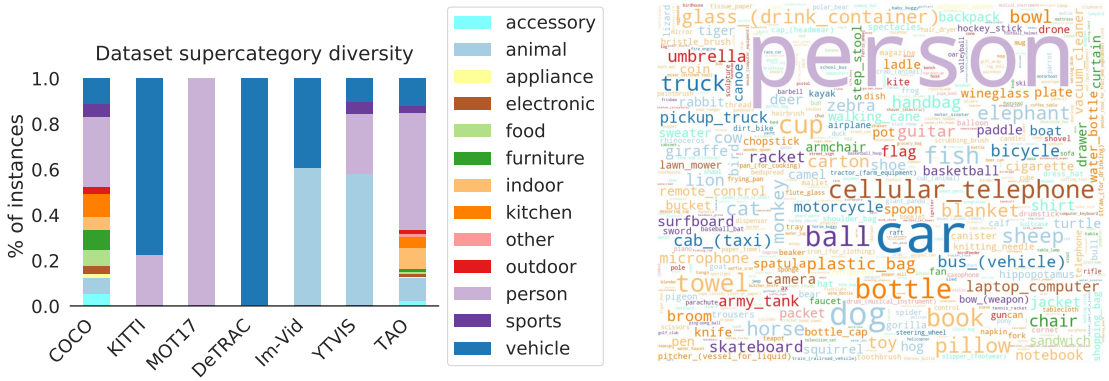


Figure 5.1: (left) Super-category distribution in existing multi-object tracking datasets compared to TAO and COCO [142]. Previous work focused on people, vehicles and animals. By contrast, our bottom-up category discovery results in a more diverse distribution, covering many small, hand-held objects that are especially challenging from the tracking perspective. (right) Wordcloud of TAO categories, weighted by number of instances, and colored according to their supercategory.

categories [158, 225, 235] (see Table 5.1). As can be seen from Figure 5.1, they predominantly target people and vehicles. Due to the lack of proper benchmarks, the community has shifted towards solutions tailored to the few videos used for evaluation. Indeed, Bergmann et al. [16] have recently and convincingly demonstrated that simple baselines perform on par with state-of-the-art (SOTA) multi-object trackers.

In this work we introduce a large-scale benchmark for Tracking Any Object (TAO). Our dataset features 2,907 high resolution videos captured in diverse environments, which are 30 seconds long on average, and has tracks labeled for 833 object categories. We compare the statistics of TAO to existing multi-object tracking benchmarks in Table 5.1 and Figure 5.1, and demonstrate that it improves upon them both in terms of complexity and in terms of diversity (see Figure 5.2 for representative frames from TAO). Collecting such a dataset presents three main challenges: (1) how to select a large number of diverse, long, high-quality videos; (2) how to define a set of categories covering all the objects that might be of interest for tracking; and (3) how to label tracks for these categories at a realistic cost. Below we summarize our approach for addressing these challenges. A detailed description of dataset collection is provided in Section 5.4.

Existing datasets tend to focus on one or just a few domains when selecting the videos, such as outdoor scenes in MOT [158], or road scenes in KITTI [74]. This

Table 5.1: Statistics of major multi-object tracking datasets. TAO is by far the largest dataset in terms of the number of categories, and the total duration of videos used for evaluation. In addition, we ensure that each video is challenging (long, containing several moving objects) and of high quality.

Dataset	Classes	Videos		Avg length (s)	Tracks / video	Min resolution	Ann. fps	Total Eval length (s)
		Eval.	Train					
MOT17 [158]	1	7	7	35.4	112	640x480	30	248
KITTI [74]	2	29	21	12.6	52	1242x375	10	365
UA-DETRAC [235]	4	40	60	56	57.6	960x540	5	2,240
ImageNet-Vid [194]	30	1,314	4,000	10.6	2.4	480x270	~25	13,928
YTVIS [254]	40	645	2,238	4.6	1.7	320x240	5	2,967
TAO (Ours)	833	2,407	500	36.8	5.9	640x480	1	88,605

results in methods that fail when applied in the wild. To avoid this bias, we construct TAO with videos from as many environments as possible. We include indoor videos from Charades [204], movie scenes from AVA [84], outdoor videos from LaSOT [63], road-scenes from ArgoVerse [36], and a diverse sample of videos from HACS [267] and YFCC100M [213]. We ensure all videos are of high quality, with the smallest dimension larger or equal to 480px, and contain at least 2 moving objects. Table 5.1 reports the full statistics of the collected videos, showing that TAO provides an evaluation suite that is significantly larger, longer, and more diverse than prior work. Note that TAO contains fewer training videos than recent tracking datasets, as we intentionally dedicate the majority of videos for in-the-wild *benchmark* evaluation, the focus of our effort.

Given the selected videos, we must choose *what* to annotate. Most datasets are constructed with a top-down approach, where categories of interest are pre-defined by benchmark curators. That is, curators first select the subset of categories deemed relevant for the task, and then collect images or videos expressly for these categories [51, 142, 222]. This approach naturally introduces curator bias. An alternative strategy is bottom-up, open-world *discovery* of what objects are present in the data. Here, the vocabulary emerges post factum [84, 88, 268], an approach that dates back to LabelMe [196]. Inspired by this line of work, we devise the following strategy to discover an ontology of objects relevant for tracking: first annotators are asked to label *all* objects that either move by themselves or are moved by people. They then give names to the labeled objects, resulting in a vocabulary that is not only significantly larger, but is also qualitatively different from that of any existing

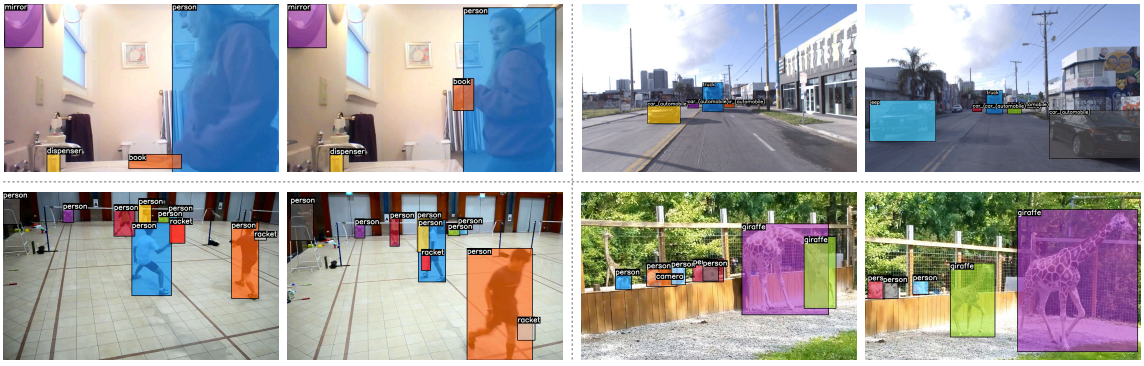


Figure 5.2: Representative frames from TAO, showing videos sourced from multiple domains with annotations at two different timesteps.

tracking dataset (see Figure 5.1). To facilitate training of object detectors, that can be later used by multi-object trackers on our dataset, we encourage annotators to choose categories that exists in the LVIS dataset [88]. If no appropriate category can be found in the LVIS vocabulary, annotators can provide free-form names (see Section 5.4.2 for details).

Exhaustively labeling tracks for such a large collection of objects in 2,907 long videos is prohibitively expensive. Instead, we extend the federated annotation approach proposed in [88] to the tracking domain. In particular, we ask the annotators to label tracks for up to 10 objects in every video. We then separately collect exhaustive labels for every category for a subset of videos, indicating whether all the instances of the category have been labeled in the video. During evaluation of a particular category, we use only videos with exhaustive labels for computing precision and all videos for computing recall. This allows us to reliably measure methods' performance at a fraction of the cost of exhaustively annotating the videos. We use the LVIS federated mAP metric [88] for evaluation, replacing 2D IoU with 3D IoU [254]. For detailed comparisons, we further report the standard MOT challenge [158] metrics in our paper appendix [47].

Equipped with TAO, we set out to answer several questions about the state of the tracking community. In particular, in Section 5.5 we report the following discoveries: (1) SOTA trackers struggle to generalize to a large vocabulary of objects, particularly for infrequent object categories in the tail; (2) while trackers work significantly better for the most-explored category of people, tracking people in diverse scenarios (e.g.,

frequent occlusions or camera motion) remains challenging; (3) when scaled to a large object vocabulary, multi-object trackers become competitive with user-initialized trackers, despite the latter being provided with a ground truth initializations. We hope that these insights will help to define the most promising directions for future research.

5.2 Related work

The domain of object tracking is subdivided based on the way the tracks are initialized. Our work falls into the multi-object tracking category, where all the objects out of a fixed vocabulary of classes have to be detected and tracked. Other formulations include user-initialized tracking, and saliency-based tracking. In the remainder of this section we will first review the most relevant benchmarks datasets in each of these areas, and then discuss SOTA methods for multi-object and user-initialized tracking.

5.2.1 Benchmarks

Multi-object tracking (MOT) is the task of tracking an unknown number of objects from a known set of categories. Most MOT benchmarks [70, 74, 158, 235] focus on either people or vehicles (see Figure 5.1), motivated by surveillance and self-driving applications. Moreover, they tend to include only a few dozen videos, captured in outdoor or road environments, encouraging methods that are overly adapted to the benchmark and do not generalize to different scenarios (see Table 5.1). In contrast, TAO focuses on diversity both in the category and visual domain distribution, resulting in a realistic benchmark for tracking *any* object.

Several works have attempted to extend the MOT task to a wider vocabulary of categories. In particular, the ImageNet-Vid [194] benchmark provides exhaustive trajectories annotations for objects of 30 categories in 1314 videos. While this dataset is both larger and more diverse that standard MOT benchmarks, videos tend to be relatively short and the categories cover only animals and vehicles. The recent YTVIS dataset [254] has the most broad vocabulary to date, covering 40 classes, but the majority of the categories still correspond to people, vehicles and animals. Moreover, the videos are 5 seconds long on average, making the tracking problem considerably

easier in many cases. Unlike previous work, we take a bottom-up approach for defining the vocabulary. This results in not only the largest set of categories among MOT datasets to date, but also in a qualitatively different category distribution. In addition, our dataset is over 7 times larger than YTVIS in the number of frames. The recent VidOR dataset [200] explores Video Object Relations, including tracks for a large vocabulary of objects. But, since ViDOR focuses on relations rather than tracks, object trajectories tend to be missing or incomplete, making it hard to repurpose for tracker benchmarking. In contrast, we ensure TAO maintains high quality for both accuracy and completeness of labels (see our paper appendix [47] for a quantitative analysis).

Finally, several recent works have proposed to label masks instead of bounding boxes for benchmarking multi-object tracking [225, 254]. In collecting TAO we made a conscious choice to prioritize scale and diversity of the benchmark over pixel-accurate labeling. Instance mask annotations are significantly more expensive to collect than bounding boxes, and we show empirically that tracking at the box level is already a challenging task that current methods fail to solve.

User-initialized tracking forgoes a fixed vocabulary of categories altogether and instead relies on the user to provide bounding box annotations for the objects that need to be tracked at test time [63, 98, 126, 222, 241]. The benchmarks in this category tend to be larger and more diverse than their MOT counterparts, but most of them still offer a tradeoff between the number of videos in the benchmarks and the average length of the videos (see our paper appendix [47]). Moreover, even if the task itself is category-agnostic, empirical distribution of categories in the benchmarks tends to be heavily skewed towards a few common objects. We study whether this bias in category selection results in methods failing to generalize to more challenging objects by evaluating state-of-the-art user-initialized trackers on TAO in Section 5.5.2.

Semi-supervised video object segmentation differs from user-initialized tracking in that both the input to the tracker and the output are object masks, not boxes [174, 249]. As a result, such datasets are a lot more expensive to collect, and videos tend to be extremely short. The main focus of the works in this domain [32, 122, 223] is on accurate mask propagation, not solving challenging identity association problems, thus their effort is complementary to ours.

Saliency-based tracking is an intriguing direction towards open-world tracking, where the objects of interest are defined not with a fixed vocabulary of categories, or manual annotations, but with bottom-up, motion- [165, 174] or appearance-based [33, 231] saliency cues. Our work similarly uses motion-based saliency to define a comprehensive vocabulary of categories, but presents a significantly larger benchmark with class labels for each object, enabling the use and evaluation of large-vocabulary object recognition approaches.

5.2.2 Algorithms

Multi-object trackers can be categorized into people and multi-category trackers. The former have been mainly developed on the MOT benchmark [158] and follow the tracking-by-detection paradigm, linking outputs of person detectors in an offline, graph-based framework [14, 15, 27, 58]. These methods mainly differ in the way they define the edge cost in the graph. Classical approaches use overlap between detections in consecutive frames [111, 178, 262]. More recent methods define edge costs based on appearance similarity [159, 191], or motion-based models [3, 37, 39, 134, 189, 198]. Very recently, Bergmann et al. [16] proposed a simple baseline approach that performs on par with SOTA people trackers, which repurposes an object detector's bounding box regression capability to predict the position of an object in the next frame. Notice that all these methods have been developed and evaluated on the relatively small MOT dataset, which consists of 14 videos captured in very similar environments. By contrast, TAO provides a much richer, more diverse set of videos, encouraging trackers more robust to tracking challenges such as occlusion and camera motion.

The more general multi-object tracking scenario is usually studied using ImageNet-Vid [194]. Methods in this group also use offline, graph-based optimization to link frame-level detections into tracks. To define the edge potentials, in addition to bounding box overlap, Feichtenhofer et al. [66] propose to use a similarity embedding, which is learned jointly with the detector. Alternatively, Kang et al. [117] directly predict short tubelets, and Xiao et al. [245] incorporate a spatio-temporal memory module inside a detector. Inspired by [16], we show that a simple baseline approach, relying on the Viterbi algorithm for linking detections across frames [66, 80], performs on par with the methods mentioned above on ImageNet-Vid. We then use this

baseline for evaluating generic multi-object tracking on TAO in Section 5.5.2, and demonstrate that it struggles when faced with a large vocabulary and a diverse data distribution.

User-initialized trackers tend to rely on a Siamese network architecture that was first introduced for signature verification [29], and later adapted for tracking [18, 43, 91, 210]. They learn a patch-level distance embedding and find the closest patch to the one annotated in the first frame in the following frames. To simplify the matching problem, state-of-the-art approaches limit the search space to the region in which the object was localized in the previous frame. Recently there have been several attempts to introduce some ideas from CNN architectures for object detection into Siamese trackers. In particular, Li et al. [137] use the similarity map obtained by matching the object template to the test frame as input to an RPN-like module adapted from Faster-RCNN [190]. Later this architecture was extended by introducing hard negative mining and template updating [274], as well as mask prediction [228]. In another line of work, Siamese-based trackers have been augmented with a target discrimination module to improve their robustness to distractors [20, 44]. We evaluate several state-of-the-art methods in this paradigm for which public implementation is available [20, 43, 44, 138, 228] on TAO, and demonstrate that they achieve only a moderate improvement over our multi-object tracking baseline, despite being provided with a ground truth initialization for each track (see Section 5.5.2 for details).

5.3 Dataset design

Our primary goal in this work is collecting a large-scale dataset of videos with a diverse vocabulary of labeled object tracks for evaluating trackers in the wild. This requires designing a strategy for (1) video collection, (2) vocabulary discovery, (3) scalable annotation, and (4) evaluation. We detail our strategies for (2-4) in this section, and defer the discussion of video collection to Section 5.4.1.

Category discovery. Rather than manually defining a set of categories, we discover an object vocabulary from unlabeled videos which span diverse operating domains. Our goal is to focus on *dynamic* objects in the world. Towards this end, we ask annotators to mark all objects that *move* in our collection of videos, without any

object vocabulary in mind. We then construct a vocabulary by giving names for all the discovered objects, following the recent trend for open-world dataset collection [88, 268]. In particular, annotators are asked to provide a free-form name for every object, but are encouraged to select a category from the LVIS [88] vocabulary whenever possible. We detail this process further in Section 5.4.2.

Federation. Given this vocabulary, one option might be exhaustively labelling all instances of each category in all videos. Unfortunately, exhaustive annotation of a large vocabulary is expensive, even for images, as noted in [88]. We choose to use our labeling budget instead on collecting a large-scale, diverse dataset, by extending the federated annotation protocol of [88] from image datasets to videos. Rather than labeling every video v with every category c, we define three subsets of our dataset for each category: P_c, which contains videos where all instances of c are labeled, N_c, videos with no instance of c present in the video, and U_c, videos where *some* instances of c are annotated. Videos not belonging to any of these subsets are ignored when evaluating category c. For each category c, we only use videos in P_c and N_c to measure the *precision* of trackers, and videos in P_c and U_c to measure recall. We describe how to define P_c, N_c, and U_c in Section 5.4.2.

Granularity of annotations. To collect TAO, we choose to prioritize scale and diversity of the data at the cost of annotation granularity. In particular, we label tracks at 1 frame per second with bounding box labels but don't annotated segmentation masks. This allows us to label 833 categories in 2,907 videos at a relatively modest cost. Our decision is motivated by the observation of [222] that dense frame labeling does not change the relative performance of the methods.

Evaluation and metric. Traditionally, multi-object tracking datasets use either the CLEAR MOT metrics [17, 74, 158] or a 3D intersection-over-union (IoU) based metric [194, 254]. We report the former in our paper appendix [47] (introducing modifications for large-vocabularies of classes, including multi-class aggregation and federation), but focus our experiments on the latter. To formally define 3D IoU, let $G = \{g_1, \ldots, g_T\}$ and $D = \{d_1, \ldots, d_T\}$ be a groundtruth and predicted track for a video with T frames. 3D IoU is defined as: $\text{IoU}_{3\text{d}}(D, G) = \frac{\sum_{t=1}^{T} g_t \cap d_t}{\sum_{t=1}^{T} g_t \cup d_t}$. If an object is not present at time t, we assign g_t to an empty bounding box, and similarly for a missing detection. We choose 3D IoU (with a threshold of 0.5) as the default metric

for TAO, and provide further analysis in our paper appendix [47].

Similar to standard object detection metrics, (3D) IoU together with (track) confidence can be used to compute mean average precision across categories. For methods that provide a score for each frame in a track, we use the average frame score as the track score. Following [88], we measure precision for a category c in video v only if all instances of the category are verified to be labeled in it.

5.4 Dataset collection

5.4.1 Video selection

Most video datasets focus on one or a few operating domains. For instance, MOT benchmarks [158] correspond to urban, outdoor scenes featuring crowds of people, whereas AVA [84] is sourced from produced films, typically capturing actors with close shots in carefully staged scenes. As a result, methods developed on any single dataset (and hence domain) fail to generalize in the wild. To avoid this bias, we constructed TAO by selecting videos from a variety of existing video benchmarks to ensure diversity of scenes and objects.

Diversity. In particular, we used datasets for action recognition, self-driving cars, user-initialized tracking, as well as in-the-wild Flickr videos. In the action recognition domain we selected 3 datasets: Charades [204], AVA [84], and HACS [267]. Charades features complex human-human and human-object interactions, but all videos are indoor with limited camera motion. In contrast, AVA has a much wider variety of scenes and cinematographic styles but is scripted. HACS provides unscripted, in-the-wild videos. These action datasets are naturally focused on people and objects with which people interact. To include other animals and vehicles, we also source clips from LaSOT [63] (a benchmark for user-initialized tracking), BDD [257] and ArgoVerse [36] (benchmarks for self-driving cars). LaSOT is a diverse collection whereas BDD and ArgoVerse consist entirely of outdoor, urban scenes. Finally we sample in-the-wild videos from the YFCC100M [213] Flickr collection.

Quality. The videos are automatically filtered to remove short videos and videos with a resolution below 480p. For longer videos, as in AVA, we use [148] to extract scenes without shot changes. In addition, we manually reviewed each sampled video

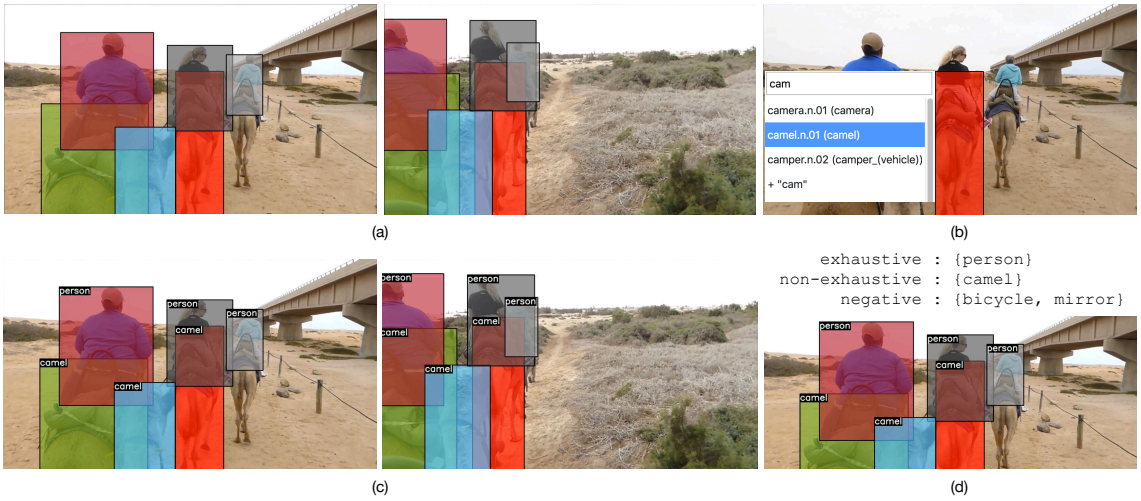


Figure 5.3: Our federated video annotation pipeline. First (a), annotators mine and track moving objects. Second (b), annotators categorize tracks using categories from the LVIS vocabulary or free-form text, producing the labeled tracks (c). Finally, annotators identify categories that are exhaustively annotated or verified to be absent. In this example (d), 'person's are identified as being exhaustively annotated, 'camel's are present but not exhaustively annotated and 'bicycle's and 'mirror's are verified as absent. Such federated labels allow one to accurately penalize false-positives and missed detections for exhaustively annotated and verified categories.

to ensure it is high quality: i.e., we removed grainy videos as well as videos with excessive camera motion or shot changes. Finally, to focus on the most challenging tracking scenarios, we only kept videos that contain at least 2 moving objects. The full statistics of the collected videos are provided in Table 5.1. We point out that many prior video datasets tend to limit one or more quality dimensions (in terms of resolution, length, or number of videos) in order to keep evaluation and processing times manageable. In contrast, we believe that in order to truly enable tracking in the open-world, we need to appropriately scale benchmarks.

5.4.2 Annotation pipeline

Our annotation pipeline is illustrated in Figure 5.3. We designed it to separate low-level tracking from high-level semantic labeling. As pointed out by others [12], semantic labeling can be subtle and error-prone because of ambiguities and corner-cases that arise in category boundaries. By separating tasks into low vs high-level, we

are able to take advantage of unskilled annotators for the former and highly-vetted workers for the latter.

Object mining and tracking. We combine object mining and track labeling into a single stage of annotation. Given the set of videos described above, we ask annotators to mark *objects that move at any point in the video*. To avoid overspending our annotation budget on a few crowded videos, we limited the number of labeled object per video to 10. Note that this stage is *category-agnostic*: annotators are not instructed to look for objects from any specific vocabulary, but instead use motion as a *saliency* cue for mining relevant objects. They are then asked to track these objects throughout the video, and label them with bounding boxes at 1 frame-per-second. Finally, the tracks are verified by one independent annotator. This process is illustrated in Figure 5.3, where we can see that 6 objects are discovered and tracked.

Object categorization. Next, we collected category labels for objects discovered in the previous stage and simultaneously constructed the dataset vocabulary. We focus on the large vocabulary from the LVIS [88] object detection dataset, which contains 1,230 synsets discovered in a bottom-up manner similar to ours. Doing so also allows us to make use of LVIS as a training set of relevant object detectors (which we later use within a tracking pipeline to produce strong baselines - Section 5.5.1). Because maintaining a mental list of 1,230 categories is challenging even for expert annotators, we use an auto-complete annotation interface to suggests categories from the LVIS vocabulary (Figure 5.3 (b)). The autocomplete interface displays classes with a matching synset (e.g., "person.n.01"), name, synonym, and finally those with a matching definition. Interestingly, we find that some objects discovered in TAO, such as "door" or "marker cap", do not exist in LVIS. To accommodate such important exceptions, we allow annotators to label objects with free-form text if they do not fit in the LVIS vocabulary.

Overall, annotators labeled 16,144 objects (95%) with 488 LVIS categories, and 894 objects (5%) with 345 free-form categories. We use the 488 LVIS categories for MOT experiments (because detectors can be trained on LVIS), but use all categories for user-initialized tracking experiments in our paper appendix [47].

Federated "exhaustive" labeling. Finally, we ask annotators to verify which categories are exhaustively labeled for each video. Specifically, for each category

c labeled in video v, we ask annotators whether all instances of c are labeled. In Figure 5.3, after this stage, annotators marked that 'person' is exhaustively labeled, while 'camel' is not. Next, we show annotators a sampled subset of categories that are not labeled in the video, and ask them to indicate categories which are absent in the video. In Figure 5.3, annotators indicated that 'bicycle' and 'mirror' are absent.

5.4.3 Dataset splits

We intend for TAO to be used primarily as an *evaluation* benchmark. We split TAO into three subsets: train, validation and test, containing 500, 988 and 1,419 videos respectively. Typically, 'train' splits tend to be larger than 'val' and 'test'. We choose to make TAO's training set small for several reasons. Firstly, the primary goal of TAO is to reliably benchmark trackers in-the-wild. Secondly, most MOT systems are modularly trained using image-based detectors with hyper-parameter tuning of the overall tracking system. In our case, we ensure the train set is sufficiently large for hyper-parameter tuning, and ensure that our large-vocabulary is aligned with large-vocabulary image datasets (e.g., LVIS). This allows us to devote most of our annotation budget for large-scale 'val' and held-out 'test' sets." We ensure that the videos in train, validation and test are well-separated. As an example, we ensure that each subject in the Charades dataset appears in only one of the train, validation or test sets. We provide further details on split construction in our paper appendix [47].

5.5 Analysis of state-of-the-art trackers

We now use TAO to analyze how well existing multi- and single-object trackers perform in the wild and when they fail. We tune the hyperparameters of each tracking approach on the 'train' set, and report results on the 'val' set. To capitalize on existing object detectors, we evaluate using the 488 LVIS categories in TAO. We begin by shortly describing the methods used in our analysis.

5.5.1 Methods

Detection. We analyze how well state-of-the-art object detectors perform on our dataset. To this end, we present results using a standard Mask R-CNN [190] detector trained using [243] in Section 5.5.2.

Multi-Object Tracking. We analyze SOTA multi-object tracking methods on ImageNet-Vid, the largest vocabulary dataset prior to TAO. We first clarify whether such approaches improve detection or tracking. Table 5.2 reports the standard ImageNet-Vid Detection mAP and Track mAP. The 'Detection' row corresponds to a detection-only baseline widely reported by prior work [66, 245, 272]. D&T [66] and STMN [245] are spatiotemporal architectures that produce SOTA improvements of 6-7% in detection mAP over a per-frame detector. However, both D&T and STMN post-process their per-frame outputs using the Viterbi algorithm, which iteratively links and re-weights the confidences of per-frame detections (see [80] for details). *When the same post-processing is applied to a single-frame detector, one achieves nearly the same performance gain (Table 5.2, last row).*

Table 5.2: ImageNet-Vid detection and track mAP; see text (left) for details.

	Viterbi	Det mAP	Track mAP
Detection		73.4 [245]	-
D&T [66]	✓	79.8	-
STMN [245]	✓	79.0	60.4
Detection	✓	79.2	60.3

Our analysis reinforces the bleak view of multi-object tracking progress suggested by [16]: while ever-more complex approaches have been proposed for the task, their improvements are often attributable to simple, baseline strategies. To foster meaningful progress on TAO, we evaluate a number of strong baselines in this work. We evaluate a powerful single-frame detector trained on LVIS [88] and COCO [142], followed by two linking methods: SORT [19], a simple, online linker initially proposed for tracking people, and the Viterbi post-processing step used by [66, 245], in Section 5.5.2.

Person detection and tracking. Detecting and tracking people have been a distinct focus in the multi-object tracking community. Section 5.5.2 compares the above baselines to a recent SOTA people-tracker [16].

User-initialized tracking. We additionally present results using user-initialized trackers. We evaluate several recent methods for which public implementation is available [20, 43, 44, 138, 228]. Unfortunately, these trackers do not provide a class

label for the objects they are tracking, and cannot directly be compared to multi-object trackers. However, these trackers *can* be evaluated with an oracle classifier, allowing us to directly compare their accuracy with the methods that simultaneously detect and track objects.

Oracles. Finally, to disentangle the complexity of object classification and tracking, we use two oracles. The first, a class oracle, computes the best matching between predicted and groundtruth tracks in each video. Predicted tracks that match to a groundtruth track with 3D IoU > 0.5 are assigned the category of their matched groundtruth track. Tracks that do not match to a groundtruth track are not modified, and are treated as false positives. This allows us to evaluate the performance of trackers assuming the semantic *classification* task is solved.

The second oracle computes the best possible assignment of per-frame detections to tracks, by comparing them with groundtruth. When doing so, class predictions for each detection are held constant. Any detections that are not matched are discarded. This oracle allows us to analyze the best performance we could expect given a fixed set of detections.

5.5.2 Results

How hard is object detection on TAO? We start by assessing the difficulty of the detection task on TAO. To this end we evaluate the SOTA object detector [90] using detection mAP. We train this model on a combination of LVIS and COCO, finding that training on LVIS alone led to a model that struggles to detect people. The final model achieves an mAP of 27.1 on TAO val at IoU 0.5, suggesting that single-frame detection is challenging on TAO.

Do multi-object trackers generalize to TAO? Table 5.3 reports results using tracking mAP on TAO. As a sanity check, we first evaluate a per-frame detector by assigning each detection to its own track. As expected, this achieves an mAP of nearly 0 (which isn't quite 0 due to the presence of short tracks).

Next, we evaluate two multi-object tracking approaches. We compare the SOTA Viterbi linking method to an online SORT tracker [19]. We tune SORT hyperparameters on our diverse 'train' set. Our paper appendix [47] shows that this tuning is key for good accuracy. The offline Viterbi algorithm takes over a month of processing

Method	Oracle Class	Oracle Track	Track mAP
Detection			0.6
Viterbi [66, 80]			6.3
SORT [19]			13.2
Detection		✓	31.5
Viterbi [66, 80]	✓		15.7
SORT [19]	✓		30.2
Detection	✓	✓	83.6

Table 5.3: SORT [19] and Viterbi linking [66, 80] provide strong baselines on TAO, but detection and tracking remain challenging. Relabeling and linking detections from current detectors using the class and track oracles is sufficient to achieve high performance, suggesting a pathway for progress on TAO.

Figure 5.4: SORT qualitative results, showing (left) a successful tracking result, and (right) a failure case due to semantic flicker between similar classes, suggesting that large-vocabulary tracking on TAO requires additional machinery.

time to run on our 'train' set, prohibiting thorough parameter tuning. Instead, we tune a post-processing parameter for Viterbi: the score threshold for reporting a detection at each frame. We detail our tuning procedure in our paper appendix [47].

Surprisingly, we find that the simpler, online approach of SORT outperforms Viterbi, perhaps because the latter has been heavily tuned for ImageNet-Vid. Because of its scalablity (to many categories and long videos) and relatively better performance, we focus on SORT for the majority of our experiments. However, the performance of both methods remains low, suggesting TAO presents a major challenge for the tracking community, requiring principled novel approaches.

To better understand the nature of the complexity of TAO, we separately measure the challenges of tracking and classification. To this end, we first evaluate the "track" oracle that perfectly links per-frame detections. It achieves a stronger mAP of 31.5, compared to 13.2 for SORT. Interestingly, providing SORT tracks with an oracle class label provides a similar improvement, boosting mAP to 30.2. We posit that these improvements are orthogonal, and verify this by combining them; we link detections with oracle tracks and assign these tracks oracle class labels. This provides the largest delta, dramatically improving mAP to 83.6%. This suggests that *large-vocabulary*

tracking requires jointly improving tracking and classification accuracy (e.g., reducing semantic flicker as shown in Fig. 5.4).

How well can we track people? We now evaluate tracking on one particularly important category: people. Measuring AP for individual categories in a federated dataset can be noisy [88], so we emphasize *relative* performance of trackers rather than their absolute AP. We evaluate Tracktor++ [16], the state-of-the-art method designed specifically for people tracking on our dataset, and compare it to the SORT and Viterbi baselines in Table 5.4. For fairness, we update Tracktor++ to use the same detector used by our SORT and Viterbi baselines, but only use the 'person' predictions from this detector. Additionally, we tune the score threshold for Tracktor++ on our 'train' set, but find the method is largely robust to this parameter (see our paper appendix [47]). We find that Tracktor++ strongly performs other approaches (36.7 AP), while SORT comes in second, modestly outperforming Viterbi (18.6 vs 16.5 AP). It is interesting to note that SORT, which can scale to all object categories, performs noticeably worse on all categories on average (13.2 mAP). Our paper appendix [47] shows that this delta between 'person' and other classes is even more dramatic using the MOTA metric (6.7 overall vs 54.8 for 'person'). We attribute the higher accuracy for the 'person' category to two factors: (1) a rich history of focused research on this one category, which has led to more accurate detectors and trackers, and (2) more complex categories present significant challenges, such as hand-held objects which undergo repeated occlusions during interactions.

Table 5.4: Person-tracking results on TAO. See text (left) for details.

Method	Person AP
Viterbi [66, 80]	16.5
SORT [19]	18.5
Tracktor++ [16]	36.7

To further investigate Tracktor++'s performance, we evaluate a simpler variant of the method from [16], which does not use appearance-based re-identification nor pixel-level frame alignment. We evaluate this variant on TAO, and find that removing these components reduces AP by over 8 points (from 36.7 to 25.9), suggesting that a majority of improvements over our baselines come from these two components. Our results contrast those of [16], which suggest that re-id and frame alignment are not particularly helpful. *Compared to prior benchmarks, we posit the diversity of TAO results in a challenging testbed for person tracking which encourages trackers robust to occlusion and camera jitter.*

Do user-initialized trackers generalize better? Next, we present results of recent user-initialized trackers in Table 5.5. For each object in TAO, we provide the user-initialized tracker with a groundtruth box. We consider two strategies for initialization. The standard approach (denoted 'Init first') initializes trackers using the first frame an object appears in, and runs trackers for the rest of the video. As the object may be partially occluded in this first frame, we additionally report a variant which initializes trackers using the frame with the largest bounding box (denoted 'Init biggest'), and runs trackers forwards and backwards in time.

Unlike multi-object trackers, most user-initialized trackers report a bounding box and confidence for objects at each frame, and do not explicitly report when an object is *absent* [222]. To resolve this, we modify each method to report an object as absent when the confidence drops below a threshold. We tune this threshold on the 'train' set in our paper appendix [47] and find that user-initialized trackers are particularly sensitive to this threshold.

We compare these trackers to SORT, supplying both with a class oracle. As expected, the use of a ground-truth initialization allows the best user-initialized methods to outperform the multi-object tracker. However, even with an oracle box initialization and an oracle classifier, tracking remains challenging on TAO. Indeed, most user-initialized trackers provide at most modest improvements over

Table 5.5: SOTA user-initialized tracking results on 'val'. Surprisingly, despite using an oracle initial bounding box, these methods provide only modest improvements over a multi-object tracker. Because some user-initialized trackers are trained on videos in TAO, we re-train them on their original train set with TAO videos removed, denoting this with *.

	Oracle		Track mAP	
Method	Box Init	Class	Init first	Init biggest
SORT		✓	30.2	
ECO [43]	✓	✓	23.7	30.4
SiamMask [228]	✓	✓	30.8	37.0
SiamRPN++ LT [138]	✓	✓	27.2	30.4
SiamRPN++ [138]	✓	✓	29.7	35.9
ATOM* [44]	✓	✓	30.9	**38.6**
DIMP* [20]	✓	✓	**33.2**	38.5

SORT, despite using an oracle box initialization. The 'Init biggest' strategy provides stronger improvements by initializing with easier frames, but this strategy cannot be used in *online* applications, as it requires access to the entire video. Our paper appendix [47] notes that user-initialized trackers can accurately track for a few frames after initialization, leading to improvements in MOTA, but provide little benefits in longer-term tracking. We hypothesize that the small improvement of user-initialized trackers over SORT is due to the fact that the former are trained on videos with a small vocabulary of objects with limited occlusions, leading to methods that do not generalize to the most challenging cases in TAO. One goal of user-initialized trackers is open-world tracking of objects without good detectors. TAO's large vocabulary allows us to analyze progress towards this goal, indicating that *large-vocabulary multi-object trackers may now address the open-world of objects as well as category-agnostic, user-initialized trackers.*

5.6 Discussion

Developing tracking approaches that can be deployed in-the-wild requires being able to reliably measure their performance. With nearly 3,000 videos, TAO provides such a robust evaluation benchmark. Our analysis provides new conclusions about the state of tracking, while further raising a number of important questions to be explored in future work.

The role of user-initialized tracking. User-initialized trackers aim to track *any* object, without requiring category-specific detectors. In this work, we raise a provocative question: with the advent of large vocabulary object detectors [88], to what extent can (detection-based) multi-object trackers perform generic tracking *without* user initialization? Table 5.5, for example, shows that large-vocabulary datasets (such as TAO and LVIS) now allow multi-object trackers to match or outperform user-initialization for a number of categories.

Specialized tracking approaches. Our hope in collecting TAO is to measure progress in tracking in-the-wild. A valid question is whether progress may be better achieved by building trackers for *application-specific* scenarios. An indoor robot, for example, has little need for tracking elephants. However, success in many computer

vision fields has been driven by the pursuit of *generic* approaches, that can then be tailored for specific applications. We do not build one class of object detectors for indoor scenes, and another for outdoor scenes, and yet another for surveillance videos. We believe that tracking will similarly benefit from targeting diverse scenarios. Of course, due to its size, TAO also lends itself to use for evaluating trackers for specific scenarios or categories, as in Section 5.5.2 for 'person.'

Video object detection. Although image-based object detectors have shown significant improvements in recent years, our analysis in Section 5.5.1 suggests that simple post-processing of detection outputs remains a strong baseline for detection in videos. While we do not emphasize it in this work, we note that TAO can also be used to measure progress in video object *detection*, where the goal is not to maintain the identity of objects, but simply to reliably detect them in each frame of a video. The large vocabulary in TAO particularly provides avenues for incorporating temporal information to resolve classification errors, which remain challenging (see Figure 5.4).

Acknowledgements. We thank Jonathon Luiten and Ross Girshick for detailed feedback on the dataset and drafts, and Nadine Chang and Kenneth Marino for reviewing early drafts. Annotations for this dataset were provided by Scale.ai. This work was supported in part by the CMU Argo AI Center for Autonomous Vehicle Research, the Inria associate team GAYA, and by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00345. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright annotation theron. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of IARPA, DOI/IBC or the U.S. Government.

Chapter 6

Evaluating Large-Vocabulary Detectors: The Devil is in the Details

6.1 Introduction

In the previous chapter, we presented a new dataset, TAO, for tracking large vocabularies of objects. A qualitative issue we found when applying detectors on TAO was that they often output many overlapping detections for varying classes. This arises from the fact that object detectors today do not have confidence estimates that can be compared across classes: they are not well-calibrated. In this chapter, we show that current evaluation metrics may encourage such miscalibration, and propose new evaluations and calibration techniques for addressing this issue.

The task of object detection is commonly benchmarked by the mean of a per-category performance metric, usually average precision (AP) [61, 142]. This evaluation methodology is designed to treat all categories *independently*: the AP for each category is determined by its confidence-ranked detections and is not influenced by the other categories. On one-hand, this is a desirable property as it treats all classes equally. On the other hand, it ignores cross-category score calibration, a key property in real-world use cases.

Surprisingly, in practice object detection benchmarking diverges from the goal of category-independent evaluation. Cross-category interactions enter into evaluation

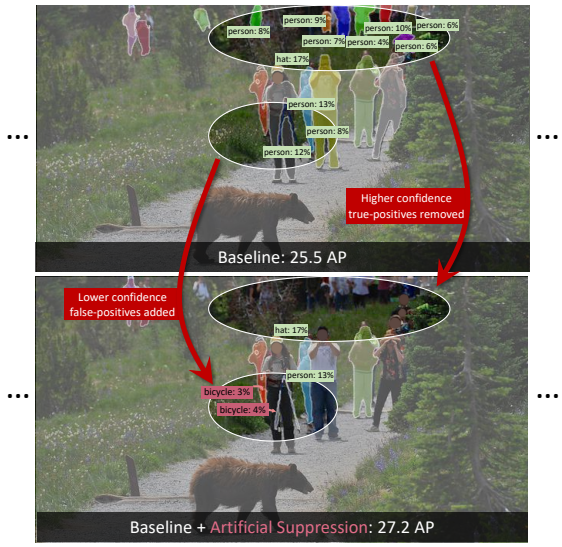


Figure 6.1: **The standard object detection average precision (AP) implementation can be gamed by an unintuitive re-ranking strategy.** Top: A detector normally outputs its top-k most confident detections per image. Bottom: We discover an unintuitive re-ranking strategy that can increase AP substantially by reducing the number of detections output for frequent classes (*e.g.*, 'person') and increasing the number output for rarer classes (*e.g.*, 'bicycle'). This re-ranking balances AP better across categories, but counterintuitively removes some higher confidence true positives while also adding some lower confidence false positives, as shown above. We analyze why this happens in practice, how to fix it, and explore the consequences of the proposed solution.

due to a seemingly innocuous implementation detail: the number of detections per image, across all categories, is limited to make evaluation tractable [88, 142]. If a detector would exceed this limit, then a policy must be chosen to reduce its output. The commonly used policy ranks all detections in an image by confidence and retains the top-scoring ones, up to the limit. This policy naturally outputs the detections that are most likely to be correct according to the model.

However, this natural policy is not necessarily the best policy given the objective of maximizing AP. We will demonstrate a counterintuitive result: there exists a policy, which can achieve higher AP, that discards a well-chosen set of higher-confidence detections in favor of promoting lower-confidence detections; see Figure 6.1. We first derive this result using a simple toy example with a perfectly calibrated detector. Then, we show that given a real-world detection model, we can employ this new

ranking policy to improve AP on the LVIS dataset [88] by a non-trivial margin. This policy is unnatural because it directly contradicts the model's confidence estimates—*even when they are perfectly calibrated*—and shows that AP, as implemented in practice, can be vulnerable to gaming-by-re-ranking.

This analysis reveals that the default AP implementation neither achieves the goal of being independent per class nor, to the extent that it involves cross-category interactions, does it measure cross-category score calibration with a principled methodology. Further, the metric can be gamed. To address these limitations, first we fix AP to make it truly independent per class, and second, given the practical importance of calibration, we consider a complementary metric, AP^{Pool}, that directly measures cross-category ranking.

Our fix to the standard AP implementation removes the detections-per-image limit and replaces it with a per-class limit over the entire evaluation set. This simple modification leads to tractable, class-independent evaluation. We examine how recent advances on LVIS fare under the new evaluation by benchmarking recently proposed loss functions, classifier head modifications, data sampling strategies, network backbones, and classifier retraining schemes. Surprisingly, we find that many gains in AP stemming from these advances do not translate into improvements for the proposed category-independent AP evaluation. This finding shows that the standard AP is sensitive to changes in cross-category ranking. However, this sensitivity is an unintentional side-effect of the detection-per-image limit, not a principled measure of how well a model ranks detections across categories.

To enable more reliable benchmarking, we propose to *directly* measure improvements to cross-category ranking with a complementary metric, AP^{Pool}. AP^{Pool} pools detections from all classes and computes a single precision-recall curve — the detection equivalent of micro-averaging from the information retrieval community [154]. To optimize AP^{Pool}, true positives for *all* classes must rank ahead of false positives for *any* class, making it a principled measure of cross-category ranking. We extend simple score calibration approaches to work for large-vocabulary object detection and demonstrate significant AP^{Pool} improvements that result in state-of-the art performance.

6.2 Related work

Large-vocabulary detection. Object detection research has largely focused on small-to-medium vocabularies (*e.g.*, 20 [61] to 80 [142] classes), though notable exceptions exist [49, 96]. Recent detection benchmarks with hundreds [131, 269] to over one-thousand classes [88] have renewed interest in large-vocabulary detection. Most approaches re-purpose models originally designed for small vocabularies, with modifications aimed at class imbalance. Over-sampling images with rare classes to mimic a balanced dataset [88] is simple and effective. Another strategy leverages advances from the long-tail *classification* literature, including classifier retraining [116, 264] and using a normalized classifier [147, 227]. Finally, recent work proposes several new loss functions to reduce the penalty for predicting rare classes, *e.g.*, equalization loss (EQL) [147], balanced group softmax (BaGS) [140] or the CenterNet2 Federated loss [271]. We analyze these advances in large-vocabulary detection, finding that a number of them do not show improvements under our fixed, independent per-class AP evaluation, indicating that they improve existing AP by modifying cross-category rankings.

Detection evaluation. Average precision (AP) is the most common object detection metric, used by PASCAL [61], COCO [142], OpenImages [131], and LVIS [88]. Conceptually, AP evaluates detectors independently for each class. We show that common implementations deviate from this conceptual goal in important ways, and propose potential fixes and alternatives. Prior work analyzing AP focuses on comparisons across classes, *e.g.* Hoiem *et al.* [93] present a normalized average precision (AP_N) and Zhang *et al.* [264] propose 'sampled AP', but does not expose the issues covered in this paper. Our procedural fix for AP computation removes the impact of cross-category scores on evaluation, and thus we propose a variant, AP^{Pool}, which explicitly rewards better cross-category rankings. From an information retrieval perspective, AP^{Pool} is the micro-averaging counterpart to AP [154], which evaluates macro-averaged performance, and has been used as a diagnostic in prior work [52].

Model calibration. A well-calibrated model is one that provides accurate probabilistic confidence estimates. Calibration has been explored extensively in the classification setting, including parametric approaches, such as Platt scaling [179] and beta calibra-

tion [128], and non-parametric approaches, such as histogram binning [259], isotonic regression [260], bayesian binning into quantiles (BBQ) [161]. While small neural networks tend to be well-calibrated [164], Guo *et al.* [86] show that deep networks are heavily uncalibrated. Kuppers *et al.* [130] extend this analysis to deep network based object detectors and show that size and position of predicted boxes helps reduce calibration error. We also apply calibration strategies to object detectors, but find that *per-class* calibration is crucial for improving AP^{Pool}.

6.3 Pitfalls of AP on large-vocabulary detection

Through both toy and real-world examples, we show that cross-category scores impact AP in counterintuitive ways.

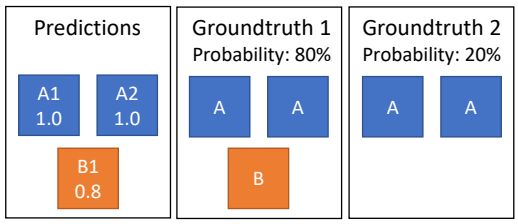
6.3.1 Background

The standard object detection evaluation aims to evaluate each class independently. In practice, however, this independence is broken due to an apparently harmless implementation detail: to evaluate efficiently, benchmarks limit the number of detections a model can output per image (*e.g.* to 100). In practice, this limit is set (hopefully) to be high enough that detections beyond it are unlikely to be correct. Importantly, this limit is shared across all classes, implicitly requiring models to rank predictions *across* classes.

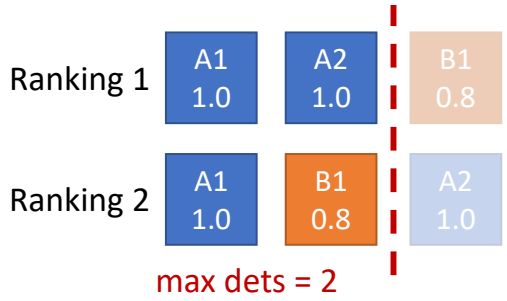
Our analysis shows that this detections-per-image limit, when used with a class-balanced evaluation like AP, can enable an unintuitive ranking policy to perform better than the natural policy of ranking detections by their estimated confidence. The effect size is correlated with increasing the number of categories or the average instances per category.

6.3.2 Analysis

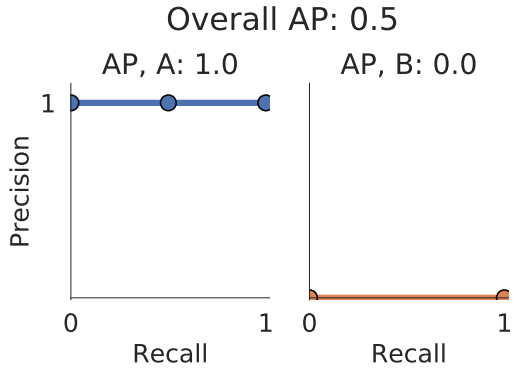
A toy example. Consider a toy evaluation on a dataset with two classes, as shown in Figure 6.2. For simplicity, suppose we have access to a detector that is perfectly calibrated: when the model outputs a prediction with confidence s (*e.g.* 0.3), the prediction is a true positive $100 \cdot s\%$ (*e.g.* 30%) of the time. We consider evaluating



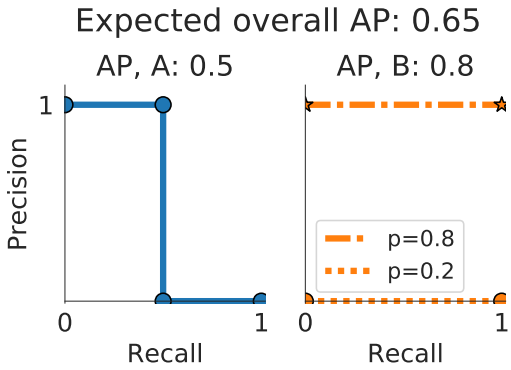
(a) Left: Predictions from a *perfectly calibrated* model: a prediction with confidence s is correct $100 \cdot s\%$ of the time. Middle, Right: the two possible groundtruth scenarios and their probabilities.



(b) Two potential rankings of the predictions. With detections-per-image limited to 2, the two rankings report different predictions (*i.e.* only those left of the dashed line).



(c) Ranking 1 precision and recall. Since A1, A2 have a precision of 1.0, AP for class A is 1.0. Class B has no predictions, so the AP is 0.0, leading to an overall AP of 0.5.



(d) Ranking 2 precision and recall. AP for A is 0.5. For B: B1 is either a true positive (AP 1.0) or not (AP 0.0). On average, this results in AP 0.8 for B, and overall AP of 0.65.

Figure 6.2: **Limiting detections-per-image rewards unintuitive rankings.** A toy scenario showing the interplay between a class-balanced AP evaluation and a limit on the number of detections per image. A perfectly calibrated model should output 'Ranking 1' in (b) since it ranks detections that are more likely correct first. However, given a detections-per-image limit of 2, 'Ranking 2' yields a higher AP even though it ranks a detection that is more likely incorrect (B1) ahead of one that is more likely correct (A2). Note that by removing the limit, the rankings across categories become fully independent and both rankings would result in an equal overall AP for the two rankings (0.75 in expectation; not visualized).

this model's outputs under two different rankings, using the standard class-balanced AP evaluation with a limit of two detections per image.

Under this setting, consider the predictions w.r.t. two possible groundtruth scenarios in Section 6.3.1. The model predicts two instances for class A (A1, A2) with confidence 1.0, and one instance for class B (B1), with confidence 0.8. Since the model is perfectly calibrated (by assumption), we know A1 and A2 are true positives 100% of the time, while B1 is a true positive 80% of the time.

With these predictions, consider the two potential rankings depicted in Section 6.3.1. Ranking 1 appears ideal: it ranks more confident detections before lower confident ones, as is standard practice. By contrast, Ranking 2 is arbitrary: B1 is ranked above A2, despite having lower confidence.

Surprisingly, Ranking 2 *outperforms* Ranking 1 under the AP metric with a limit of two detections per image, as shown in Section 6.3.1 and Section 6.3.1. While Ranking 1 gets a perfect AP of 1.0 for class A, it gets 0 AP for class B, leading to an overall AP of 0.5. By contrast, while Ranking 2 leads to a lower AP for class A (0.5), it scores an expected AP of 0.8 for class B, yielding an overall AP of 0.65!

Of course, this is a toy scenario, concocted to highlight an evaluation pitfall using an artificially low detections-per-image limit of only two predictions. We now show that a similar effect exists for a real-world detection benchmark.

A real-world example. The LVIS [88] dataset uses the evaluation described above, with a limit of 300 detections per image. We investigate whether an artificial ranking policy, as in Figure 6.2, can lead to improved AP on this dataset. Concretely, we evaluate a simple policy: we first discard all but the top k scoring detections per class across the entire evaluation dataset. Given the predictions in Section 6.3.1, applying this policy with $k = 1$ leads to Ranking 2 from Section 6.3.1: an arbitrary ranking which, nevertheless, leads to a higher AP than the baseline Ranking 1.

This ranking policy, combined with the detections-per-image limit, is unintuitive: it explicitly discards high-scoring predictions for many classes in order to fit low-scoring predictions from other classes into the detections-per-image limit, as shown by our toy example in Section 6.3.1 and with real-world detections in Figure 6.1. Using a baseline Mask R-CNN model [90] (see supp. for details), we find that this strategy, with $k = 10,000$, improves LVIS AP by 1.2 points, and AP_r by 2.9 points, as shown in Table 6.1. Note that this results purely from a modified *ranking policy*,

dets/class	dets/im	AP	AP_r	AP_c	AP_f
∞ (Ranking 1)	300	22.6	12.6	21.1	28.6
10,000 (Ranking 2)	300	**23.8** (+1.2)	15.5 (+2.9)	22.7	28.5

Table 6.1: **Unintuitive Ranking 2 (Section 6.3.1) improves LVIS AP.** Artificially limiting the number of detections per class across the entire validation set leads to *higher* LVIS AP when using the standard limit of 300 detections per image, perhaps paradoxically. In Figure 6.1 we show how this ranking policy (which, again, improves AP) suppresses some higher-confidence detections in favor detections that the model estimates are more likely incorrect.

	LVIS				COCO
dets/**im**	AP	AP_r	AP_c	AP_f	AP
100	18.2	6.5	15.8	26.1	37.4
300	22.6	12.6	21.1	28.6	37.5 (+0.1)
1,000	25.0 (+2.4)	16.8 (+4.2)	24.1	29.7	37.5 (+0.1)
2,000	25.6 (+3.0)	18.1 (+5.5)	24.6	29.9	37.5 (+0.1)
5,000	26.0 (+3.4)	19.7 (+7.1)	24.9	30.0	37.5 (+0.1)
10,000	26.1 (+3.5)	19.8 (+7.2)	25.0	30.1	37.5 (+0.1)

Table 6.2: **Increasing the limit on detections per image *significantly* improves LVIS AP.** AP_r improves by over 7 points (over 50% relative), indicating many accurate rare class predictions are ignored due to the default limit of 300 detections per image. By contrast, this limit does not significantly impact COCO, which contains a significantly smaller vocabulary.

without any changes to the evaluation or model. This non-trivial improvement is roughly the magnitude achieved by a typical new method published at CVPR (*e.g.* [140, 209]). The relatively larger improvement to AP_r suggests that under the standard confidence-based ranking, accurate predictions for rare classes are crowded out by frequent class predictions due to the detections-per-image limit.

Although this limit appears high (at 300 detections-per-image), LVIS contains over a thousand object classes: even outputting a single prediction for each class is impossible under the limit. The assumption is that detections beyond the first 300 are likely to be false positives. Table 6.2 verifies that this assumption is incorrect: increasing the limit on detections per image leads to significantly higher results on the LVIS dataset. In particular, the AP for rare categories improves *drastically* from 12.6 to 19.5 with a higher limit.

When is gameability an issue? Given the impact of the detections-per-image

Subset	# classes	# instances per class	AP@dets/im 300	AP@dets/im 5,000	Δ
R	337	3.6	18.5	19.0	+0.5
C	461	28.4	24.6	25.0	+0.4
F	405	569.0	28.7	30.0	+1.3
R C	798	17.9	22.2	23.3	+1.1
C F	866	281.2	24.7	27.4	+2.7
R F	742	312.2	24.1	26.6	+2.5
R C F	1,203	203.4	22.6	26.0	+3.4

Table 6.3: **Analyzing dets/image limit on LVIS subsets.** We restrict a baseline model to a subset of classes and evaluate on the subset. 'R', 'C', and 'F' indicate rare, common and frequent. We compare the AP change (Δ) at the default 300 dets/im limit *vs*. a high limit of 5,000. The change is more prominent for subsets with more classes and more instances per class, suggesting it is driven by both large vocabularies and the number of labeled objects.

limit on LVIS, a natural question is whether this also affects the widely used COCO dataset. Table 6.2 shows that increasing this limit does not significantly change COCO AP, suggesting the limit has not negatively impacted COCO evaluation. We hypothesize that this is due to the significantly smaller vocabulary in COCO relative to the detections limit: with only 80 classes, detections beyond the top 100 per image are unlikely to impact AP.

To analyze this hypothesis, we evaluate on subsets of LVIS. Given a baseline model trained on LVIS, we restrict its predictions to a subset of classes, and report AP with a low and a high detections-per-img limit in Table 6.3. We find that on subsets which have small vocabularies and few labeled instances per class, the gap between AP in the two settings is small (0.4-0.5 points). However, when there are many labeled instances in the evaluation set (as with the 'F' subset), or the vocabulary is large (as with the second and third blocks of the table), the gap is much higher. This suggests that AP is sensitive to the detections limit on large vocabulary datasets, particularly if they contain many labeled instances per image.

dets/**class**	dets/**im**	AP	AP_r	AP_c	AP_f
1,000	∞	21.9	17.7	22.2	23.5
5,000	∞	25.0 (+3.1)	19.5 (+1.8)	24.4	28.2
10,000	∞	25.6 (+3.7)	19.7 (+2.0)	24.7	29.1
30,000	∞	26.0 (+4.1)	19.8 (+2.1)	24.9	29.8
50,000	∞	26.0 (+4.1)	19.9 (+2.2)	25.0	30.0

Table 6.4: **LVIS AP evaluation with varying limits on the number of detection/class, with no limit on detections/image.** A limit of 10,000 balances evaluation speed, memory, and AP well.

6.4 AP without cross-category dependence

We now address this undesirable interaction between AP and cross-category scores. We have already diagnosed that this interaction is caused by the detections-per-image limit. In theory, then, the solution is simple: don't limit the number of detections per image. Of course, this is impossible in practice, as we cannot evaluate infinite detections. How, then, can we approximate this hypothetical evaluation?

Higher detections-per-image limit. A natural option is to have a large, but finite, detections-per-image limit. Predictions beyond a very high limit are exceedingly unlikely to be correct, and thus may not affect the evaluation. Indeed, Table 6.2 shows that increasing the limit beyond 5,000 does not significantly affect AP. Unfortunately, this results in prohibitively slower evaluation: on LVIS validation, a baseline model's outputs are 15× larger using a limit of 5,000 detections than at the default limit of 300 (37GB *vs*. 2.4GB). Moreover, submitting such results to an evaluation server, as required for the LVIS test sets, is impractical.

Limit detections-per-class. We now present an alternative, tractable implementation. Rather than discarding low-scoring detections per image, we discard low-scoring detections per class across the dataset. That is, given a model's output on the evaluation set, the benchmark would only evaluate the top k predictions per class, discarding the rest.

We find that this strategy significantly reduces the storage and time requirements for evaluation. Table 6.4 shows that limiting detections to 10,000 per class across the dataset achieves a good balance. This limit yields 98.5% of full AP while increasing file size and evaluation time only by a factor of 2× (compared to 15× for the previous

strategy), making evaluation tractable. In principle, this limit depends on the size of the evaluation set, similar to how the standard per-image limit depends on the vocabulary size and labeling density. In practice, the LVIS validation and test sets all contain 20,000 images and thus a single limit suffices.

This evaluation may appear similar to the undesirable Ranking 2 in Figure 6.2. However, Ranking 2 is an undesirable *strategy* for resolving competition across classes, while our evaluation *removes* this competition altogether by providing an independent detection budget per class. This evaluation has a natural appeal when viewing detection as an information retrieval task, the field from which AP originates: the detector is allowed to 'retrieve' up to k detections (or 'documents') per class from the entire evaluation set (or 'corpus'). In practice, various strategies exist for efficiently selecting the top k detections over a large set of images.

We recommend this latter strategy of limiting detections per class, with no limit per image. In the remainder of the paper, we refer to the default evaluation (with a detections-per-image limit) as 'AP^{Old}' and our new, recommended version that limits detections per class as 'AP^{Fixed}'.

6.5 Impact on long-tailed detector advances

We have shown that the current AP evaluation introduces subtle, undesirable interactions with cross-category rankings due to the detections-per-image limit. However, it remains unclear to what extent this issue meaningfully affects prior conclusions drawn on LVIS. To analyze this, we evaluate the importance of different design choices in LVIS detectors with the original evaluation ('AP^{Old}'), with a limit of 300 predictions per image, and our modified evaluation ('AP^{Fixed}'), with a limit of 10,000 detections per class across the whole evaluation set (with no per-image limit).

Experimental setup. The following experiments use Mask R-CNN [90]. Unless noted differently: we use a ResNet-50 [89] backbone with FPN [143] pre-trained on ImageNet [194] and fine-tuned on LVIS v1 [88] for 180k iterations with repeat factor sampling, minibatch of 16 images, learning rate of 0.02 decayed by 0.1× at 120k and 160k iterations, and weight decay of 1e−4. Batch norm [103] parameters are frozen. Results are reported on LVIS v1 validation using the mean of three runs with different

Loss	AP^{Old}	AP^{Fixed}
Softmax CE	22.3	25.5
Sigmoid BCE	22.5 (+0.2)	25.6 (+0.1)
EQL [209]	24.0 (+1.7)	26.1 (+0.6)
Federated [271]	**24.7** (+2.4)	**26.3** (+0.8)
BaGS [140]	24.5 (+2.2)	25.8 (+0.3)

(a) **Loss functions.** Choosing the right loss is more important under AP^{Old}, providing an improvement of up to +2.4 AP. Under our proposed AP^{Fixed}, the impact of losses is reduced, to at most +0.8 AP. This result indicates that these loss functions may primarily modify cross-category rankings (also see Figure 6.3).

Loss	Obj	Norm	AP^{Old}	AP^{Fixed}
Softmax CE	✗	✗	22.3	25.5
	✓	✗	23.2 (+0.9)	25.3 (−0.2)
	✗	✓	23.2 (+0.9)	**26.3** (+0.8)
	✓	✓	24.4 (+2.1)	**26.3** (+0.8)
Sigmoid BCE	✓	✓	24.2 (−0.2)	**26.3** (+0.0)
EQL [209]	✓	✓	24.7 (+0.3)	26.1 (−0.2)
Federated [271]	✓	✓	**25.1** (+0.7)	**26.3** (+0.0)
BaGS [140]	✓	✓	**25.1** (+0.7)	26.2 (−0.1)

(b) **Classifier modifications.** We evaluate two ideas commonly used for improving long-tail detection: an objectness predictor ('Obj') [140], and L2-normalizing both the linear classifier weights and input features ('Norm'). Once again, we find that these components improve the baseline significantly under the AP^{Old}, but provide minor improvements under our AP^{Fixed}. Nevertheless, our results indicate these components provide a strong, simple baseline that erases the impact of the training loss choice.

Sampler	AP^{Old}	AP^{Fixed}
Uniform	18.4	22.8
CAS	19.2 (+0.8)	21.5 (−1.3)
RFS	**22.3** (+3.9)	**25.5** (+2.7)

(c) **Samplers.** Category Aware Sampling (CAS) and Repeat Factor Sampling (RFS) are common sampling strategies for addressing class imbalance. While both strategies outperform the uniform sampling baseline under AP^{Old}, only RFS provides significant improvements under AP^{Fixed}.

Phase 1	Phase 2	AP^{Old}	AP^{Fixed}
RFS	-	22.3	**25.5**
Uniform	RFS	21.6 (−0.7)	24.9 (−0.6)
Uniform	CAS	23.1 (+0.8)	24.9 (−0.6)
RFS	CAS	**23.6** (+1.3)	**25.6** (+0.1)

(d) **Classifier retraining.** We evaluate the efficacy of training detectors in two phases, a common technique [116, 227]. Phase 1: the model is trained end-to-end with one sampler. Phase 2: only the final classification layer is trained, using a different sampler. This strategy improves AP^{Old}, but not AP^{Fixed}, suggesting that classifier retraining may primarily modify cross-category rankings.

Backbone	AP^{Old}	AP^{Fixed}
ResNet-50	22.3	25.5
ResNet-101	24.6 (+2.3)	27.7 (+2.2)
ResNeXt-101	**26.2** (+3.9)	**28.7** (+3.2)

(e) **Stronger backbones.** Using larger backbones consistently improves the detector under both AP^{Old} and AP^{Fixed}, indicating, as one might expect, that larger backbones improve overall detection quality and not just cross-category rankings. ResNeXt-101 uses the 32x8d configuration.

Table 6.5: **Impact of various design choices on the LVIS v1 validation dataset, comparing AP^{Old} to AP^{Fixed}.** Unless specified otherwise, each experiment uses a ResNet-50 FPN Mask R-CNN model trained with Repeat Factor Sampling (RFS) for 180k iterations with 16 images per batch. All numbers are the average of three runs with different random seeds and initializations.

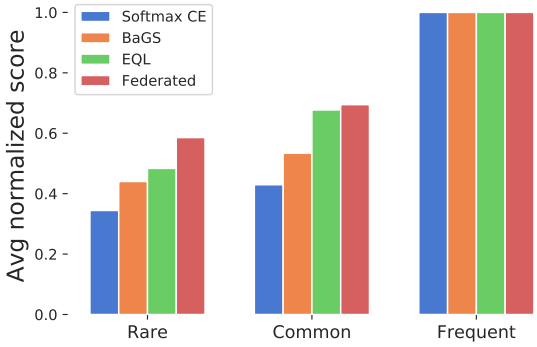


Figure 6.3: **Score distribution induced by different loss functions for LVIS rare, common, and frequent categories.** Compared to the baseline softmax CE loss, BaGS, EQL, and Federated losses tilt the distribution to be more uniform, modifying ranking of detections across categories.

random seeds.

6.5.1 Case studies

Loss functions. As discussed in Section 6.2, a number of new losses have been proposed in the past year. We analyze three in particular: EQL [209], BaGS [140], and a 'Federated' loss [271]. Section 6.4 (first column) shows that, under the original evaluation, the choice of loss function can robustly improve the AP of a baseline model by up to 2.4 points, from 22.3 using softmax cross-entropy (CE) to 24.7 using the Federated loss. These gains suggests the choice of loss function is important. However, under our 'AP^{Fixed}', the losses are more similar, differing by at most 0.8 points.

To gain insight into why the losses improve AP^{Old} more than AP^{Fixed}, we plot the score distribution for the LVIS rare, common, and frequent categories (normalized so the average score for frequent categories is 1.0). Figure 6.3 shows that the EQL, BaGS, and Federated losses tilt the distribution to be more uniform relative to softmax CE loss. This boosts the confidence of rare category detections, making them more likely to appear in the 300 detections-per-image limit. This suggests that these losses change cross-category rankings compared to softmax CE loss in a way that AP^{Old} rewards. Because AP^{Fixed} is category independent, it does not reward cross-category ranking modifications.

Classifier heads. Next, we evaluate two common modifications to the linear classifier in detectors in Section 6.4. The first modification trains a linear *objectness* binary classifier in parallel to the K-way classifier [140, 188, 227], denoted 'Obj'. The second L2-normalizes the input features and classifier weights during training and inference [147, 226, 227], denoted 'Norm.' We share implementation details in supplementary.

The first block in Section 6.4 shows that while adding an objectness predictor modestly improves AP^{Old} (+0.9), it results in a slightly lower AP^{Fixed} (−0.2). This discrepancy suggests the objectness predictor optimizes the ranking of predictions across classes, but doesn't meaningfully improve the quality of the detections. On the other hand, using a normalized classifier consistently leads to higher accuracy under both AP^{Old} (+0.9) and AP^{Fixed} (+0.8). Finally, we find that applying both these modifications to the classifier results in a strong baseline under both AP^{Old} and AP^{Fixed}. The second block in Section 6.4 further shows that under AP^{Fixed}, the choice of loss function is largely irrelevant when both of these classifier modifications are used.

Sampling strategies. Modifying the image sampling strategy is a common approach for addressing class imbalance in LVIS. Section 6.4 analyzes three strategies: Uniform, which samples images uniformly at random; Class Aware Sampling (CAS), which first samples a category and then an image containing that category; and Repeat Factor Sampling (RFS) [88], which oversamples images containing rare classes. RFS consistently and significantly outperforms the others under both AP^{Old} and AP^{Fixed}. Surprisingly, while CAS outperforms uniform sampling under AP^{Old}, it hurts accuracy under AP^{Fixed}, suggesting that CAS improves primarily due to how it ranks predictions across classes.

Classifier retraining. A common alternative to training with a single sampler is to train the model end-to-end using one sampler, and fine-tune the linear classifier with a different sampler [116, 264]. Under AP^{Old}, carefully choosing the samplers for these phases appears important, improving by +1.3 AP. However, under AP^{Fixed}, this improvement disappears, indicating that on LVIS, classifier retraining primarily improves by aligning scores across classes.

Stronger backbones. Finally, we evaluate the improvements due to stronger

	AP^{Pool}			
dets/im	AP	AP_r	AP_c	AP_f
300	26.2	8.0	16.7	27.0
1,000	26.8 (+0.6)	10.6 (+2.6)	19.8	27.6
2,000	27.0 (+0.8)	11.0 (+3.0)	20.5	27.7
5,000	27.0 (+0.8)	11.3 (+3.3)	20.8	27.7
10,000	27.0 (+0.8)	11.3 (+3.3)	20.8	27.7

Table 6.6: **Impact of limiting detections-per-image on AP^{Pool}.** As expected, AP^{Pool} is less sensitive to this limit than AP^{Old} because each instance, rather than each class, is weighted equally.

backbone architectures. We evaluate four progressively stronger models: ResNet-50, ResNet-101 [89], and ResNeXt-101 32x8d [248]. Unlike many other LVIS-specific design choices, we find that the choice of a larger backbone consistently improves accuracy for both AP^{Old} and AP^{Fixed}.

6.5.2 Discussion: something gained, something lost

AP^{Fixed} makes AP evaluation category independent by design. As a result, it is no longer vulnerable to gaming-by-re-ranking, as we demonstrate is possible with AP^{Old} in Section 6.3. However, by benchmarking several recent advances in long-tailed object detection we observe evidence that several of the improvements may be due to better cross-category rankings, because the improvements that were observed with AP^{Old} largely disappear when evaluated with AP^{Fixed}. While AP^{Old} improperly evaluated calibration, AP^{Fixed} is *invariant* to calibration: *i.e.*, per-category, monotonic score transformations do not change AP^{Fixed}.

Neither AP^{Old} nor AP^{Fixed} appropriately specifies how detectors should be deployed in the real world, a task which *requires* score calibration. In the simplest example, one may want to produce a demo that visualizes all detections above a global score threshold (*e.g.* 0.5) and expect to see consistent results across all categories. Given this practical demand, we consider in the next section a variant of AP, called AP^{Pool}, that directly rewards cross-category rankings, without the vulnerability to gaming displayed by AP^{Old}. Furthermore, we develop a simple detector score calibration method and show that it improves AP^{Pool}.

6.6 Evaluating cross-category rankings

An independent, per-class evaluation is appealing in its simplicity. Most practical applications, however, require comparing the confidence of predictions across classes to form a unified understanding of the objects in an image. As an extreme example, note that a detector can output arbitrary range of scores for each class for a truly independent evaluation: that is, all detections for one class (say, 'banana') may have confidences above 0.5, while all detections for another class (say, 'person') have confidences below 0.5. Using such a detector in practice requires carefully calibrating scores across classes—an open challenge that is not evaluated by current detection evaluations.

6.6.1 AP^{Pool}: A cross-category rank sensitive AP

To address this, we consider a complementary metric, AP^{Pool}, which explicitly evaluates detections across all classes together [52]. To do this, we first match predictions to groundtruth per-class, following the standard evaluation. Next, instead of computing a precision-recall (PR) curve for each class, we pool detections across all classes to generate a single PR curve across all classes, and compute the Average Precision on this curve to get AP^{Pool}.

This evaluation has two key properties. First, it ranks detections across all classes to generate a single precision-recall curve, incentivizing detectors to rank confident predictions above lower confidence ones. Second, it weights all groundtruth instances, rather than classes, equally. This removes a counterintuitive effect, illustrated in Figure 6.2, that can occur with class averaging. Further, it reduces the impact of the detections-per-image limit, as low-confidence predictions for some rare classes do not significantly impact the evaluation. Because of this, however, the evaluation is influenced more by frequent classes than rare ones. To analyze performance for rare classes, we further report three diagnostic evaluations which evaluate predictions only for classes within a specified frequency: AP_r^{Pool} (for rare classes), AP_c^{Pool} (common), and AP_f^{Pool} (frequent).

	AP^{Fixed}				AP^{Pool}			
Loss	AP	AP_r	AP_c	AP_f	AP	AP_r	AP_c	AP_f
Softmax CE	25.5	18.9	24.9	29.1	25.6	11.5	20.5	26.2
Sigmoid BCE	25.6 (+0.1)	19.4	24.9	28.9	25.6 (+0.0)	10.8	20.1	26.1
EQL [209]	26.1 (+0.6)	19.9	26.1	28.9	25.9 (+0.3)	11.3	22.9	26.3
Federated [271]	**26.3** (+0.8)	20.7	24.9	30.2	**27.8** (+2.2)	16.1	22.0	28.2
BaGS [140]	25.8 (+0.3)	17.9	25.6	29.5	26.0 (+0.4)	9.1	20.8	26.4

Table 6.7: **AP^{Fixed} and AP^{Pool} for models trained with varying losses.** Federated significantly outperforms others under AP^{Pool}.

6.6.2 Analysis

How does the dets/im limit affect AP^{Pool}? Table 6.6 analyzes how the detections-per-image limit impacts AP^{Pool}. As expected, increasing this limit does not significantly affect AP^{Pool}: while AP can change drastically due to a few additional true positives for rare classes, AP^{Pool} treats true positives for all classes equally. Increasing the limit beyond 300 detections improves the diagnostic AP^{Pool}_r metric, but only mildly improves AP^{Pool} by 0.8 points. Nonetheless, for consistency, we evaluate models with the same detections as AP^{Fixed}: the top 10,000 per class, with no per-image limit.

Do losses impact AP^{Pool}? Next, we analyze various losses under AP^{Pool}, though we also analyze other detector components in supp. Table 6.7 compares losses under AP^{Fixed} and AP^{Pool}. Perhaps surprisingly, while EQL and BaGS do not meaningfully impact AP^{Pool}, the Federated loss improves by 2.2 points over the baseline softmax CE loss. This provides a new perspective for the Federated loss: Although it does not explicitly calibrate models, it improves *cross-category* ranking of predictions compared to other losses.

6.6.3 Calibration

We now propose a simple and effective strategy for improving AP^{Pool}. We re-purpose classic techniques for calibrating model uncertainty for the task of large-vocabulary object detection. Calibration aims to ensure that the model's confidence for a prediction corresponds to the probability that the prediction is correct. In the detection setting, if a model detects a box with confidence s, it should correctly

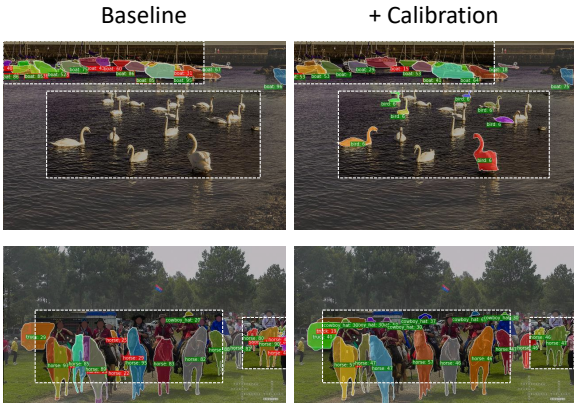


Figure 6.4: Examples illustrating the effect of calibration. Each row shows the 20 highest-scoring predictions from the baseline, uncalibrated model (left) and its calibrated version (right). True-positives and false-positives (at IoU 0.5) are indicated with a green and red label, respectively. The calibrated model increases the rank of low-confidence but accurate predictions, such as the 'bird's (top row) and 'cowboy hat's (bottom), over incorrect predictions with artificially high scores, such as some 'boat's (top), and 'horse's (bottom).

localize a groundtruth box of the same category s% of the time [130]. While this property is not necessary for AP^{Pool}, it provides a sufficient condition for improving cross-category rankings (AP^{Pool} only requires that true positives are ranked higher than false positives across all classes, without requiring the scores to be *probabilistically* calibrated).

Following [130], we analyze various calibration strategies: histogram binning [259], Bayesian Binning into Quantiles (BBQ) [161], beta calibration [128], isotonic regression [260], and Platt scaling [179]. Prior work on calibrating detectors applies calibration strategies to predictions across all classes [130]. However, this approach does not account for class frequency: rare classes may, for example, have lower-scoring predictions than frequent classes. Instead, we propose to calibrate each class individually, allowing the method to boost scores of under-confident classes and diminish scores of over-confident classes.

Standard calibration strategies require a held-out dataset for calibration. However, in the large-vocabulary setting, many classes have only a handful of examples in the entire dataset. We instead calibrate directly on the *training* set. To understand the impact of this choice, we also report an upper-bound by calibrating on the *validation*

Calibration	AP^{Pool}	AP_r^{Pool}	AP_c^{Pool}	AP_f^{Pool}
Uncalibrated	27.8	**16.1**	22.0	28.2
Histogram Bin	28.6 (+0.8)	12.4	20.6	29.2
BBQ (AIC)	28.8 (+1.0)	13.6	21.6	29.3
Beta calibration	**29.5** (+1.7)	12.8	22.7	**30.0**
Isotonic reg.	28.3 (+0.5)	14.4	22.2	28.7
Platt scaling	**29.5** (+1.7)	13.1	**22.8**	**30.0**
Calibrate on validation (upper-bound oracle)				
HistBin	30.1 (+2.3)	24.4	**27.8**	30.2
BBQ (AIC)	30.0 (+2.2)	22.9	26.9	30.2
Beta calibration	29.8 (+2.0)	22.4	25.2	30.1
Isotonic reg.	**30.3** (+2.5)	**24.6**	27.2	**30.4**
Platt scaling	29.8 (+2.0)	22.2	24.9	30.1

Table 6.8: **Calibrating detection outputs on the train set significantly improves AP pooled.** The gains are due to improved rankings across categories. Calibrating on validation significantly improves AP_r^{Pool}, indicating calibration remains challenging in the tail. All models trained with the Federated loss.

set.

Table 6.8 reports AP^{Pool} using various calibration approaches applied to a model trained with the Federated loss. The results show that calibrating per class improves AP^{Pool} by 1.7 points, from 27.8 to 29.5, and the choice of calibration strategy is not critical. Surprisingly, calibrating on the *validation* set, as in the second block, outperforms training set calibration by only 0.8 points, suggesting that calibrating on the training set is a viable strategy. However, calibrating on the validation set significantly improves AP_r^{Pool} while calibrating on the training set *harms* AP_r^{Pool}, indicating that calibrating rare classes remains an open challenge. Figure 6.4 presents qualitative examples of this improvement: calibration increases the scores of underconfident, accurate predictions from some classes (*e.g.* 'cowboy hat') and suppresses overconfident predictions from others (*e.g.* 'horse').

6.7 Discussion

Robust, reliable evaluations are critical for advances in large-vocabulary detection. Our analysis reveals that current evaluations fail to properly handle cross-category interactions by neither eliminating them (as intended) nor evaluating them in a

principled fashion (as potentially desired). We show that, as a result, the current AP implementation (AP^{Old}) is vulnerable to gaming. We propose AP^{Fixed}, which addresses this gameability by removing the effect of cross-category score calibration, and recommend it as a *replacement* for AP^{Old} moving forward. AP^{Fixed} provides new conclusions about the importance of different LVIS advances. Finally, we recommend a complementary *diagnostic* metric, AP^{Pool}, for applications requiring cross-category score calibration, and show that a simple calibration strategy offers off-the-self detectors solid improvements to AP^{Pool}.

Part III

Generalizing to any object

Chapter 7

Towards Segmenting Anything That Moves

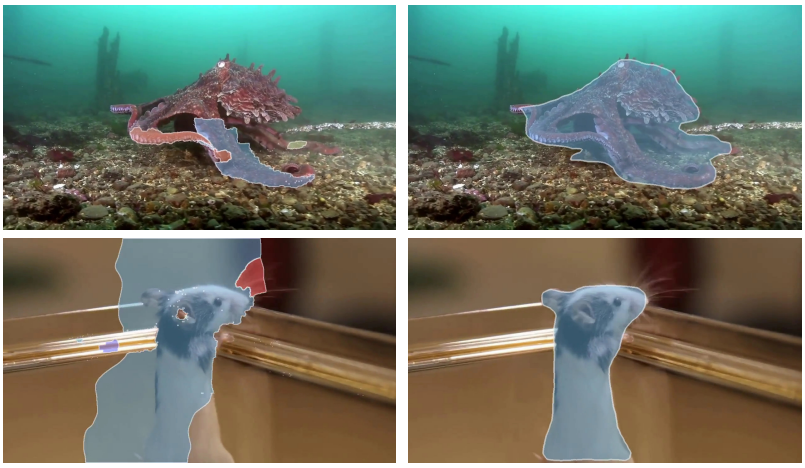


Figure 7.1: Detecting and segmenting all objects, regardless of category, is key for many perception and robotics tasks. Bottom-up grouping approaches, e.g. [119] (left), aim to tackle this task, but lag behind the quality of closed-world methods that detect a fixed set of N categories. Our work (right) bridges this gap, accurately segmenting generic moving objects, even ones unseen in training.

7.1 Introduction

People have the remarkable ability to thrive while frequently encountering things they have never seen before. Our approaches for machine perception, meanwhile, often remain trapped in a *closed world*, as in the case of object recognition, where approaches are designed to recognize and name one of N pre-defined classes. In the previous part, we scaled such approaches to a large number of classes, covering a wider range of objects in the real world. But practical robot autonomy requires robust perception in the open-world: even a self-driving car must be able to detect never-before-seen obstacles and debris, regardless of what particular semantic *name* it happens to associate with. In this part, we tackle how to segment and track such novel objects through diverse scenes.

In the computer vision community, open-world recognition is typically addressed from a machine-learning perspective such as zero-shot learning [205] or open-set classification [197]. We advocate a different approach that has its roots in classic vision: perceptual grouping. Specifically, we wish to segment out *all* moving object instances in a video stream, including never-before-seen object categories. Defining the notion of a generic, never-before-seen object is notoriously challenging [4]. We intentionally focus on *moving* objects so as to take advantage of the "common fate" principle of grouping: pixels that move together should tend to be grouped together into objects [167].

Indeed, the problem of spatio-temporal grouping is a classic "mid-level" visual understanding task, dating back to the iconic work of Marr [155, 237]. Pre-deep learning solutions tend to follow bottom-up computational strategies for self-organization and clustering, often of long-term pixel trajectories [119, 165]. In the static image case, pixels can grouped by relying on Gestaltian notions of appearance similarity and curvilinear edge continuity [167]. One long-standing challenge in perceptual organization has been operationalizing these cues into an accurate algorithm for spatio-temporal grouping. *Our key observation is that many of the recent advances in closed-world instance segmentation can be repurposed for open-world spatio-temporal grouping.*

We first validate the performance of our proposed approach on the Freiburg Berkeley Motion Segmentation benchmark (FBMS). Because the standard measure

used in FBMS does not penalize false positives, we find that trivial solutions can score well. We analyze the official metric in detail and propose a new, more informative evaluation. We achieve state-of-the-art results on both measures, and specifically outperform the next-best method of Keuper et al. [119] by 11.4% on our proposed measure.

To further study our method, we introduce the DAVIS-Moving and YTVOS-Moving benchmarks for motion-based grouping. We create these by selecting videos from the DAVIS 2017 [180] and YTVOS [249] datasets where *all* moving objects are labeled. On these new benchmarks, we strongly outperform top-down, closed world methods such as Mask R-CNN, as well as traditional bottom-up grouping methods. In particular, our approach is competitive with a top-down method for categories seen during training, but *outperforms both top-down and bottom-up approaches for unseen categories by 27%*.

To sum up, our contributions are three-fold: (1) we propose the first deep learning-based method for spatio-temporal grouping; (2) we propose a more informative metric and larger, more diverse benchmarks to enable further progress; (3) we report state-of-the-art results on the FBMS dataset and our larger, proposed benchmarks. The code and trained models will be made publicly available.

7.2 Related Work

Spatio-temporal grouping: Segmenting and tracking objects based on their motion has a rich history. An early work [202] proposed treating this task as a spatio-temporal grouping problem, a philosophy espoused by a number of more recent approaches, including [30, 83, 119], as well as [165], which introduced FBMS. In particular, these methods track each pixel individually with optical flow, encode the motion information of a pixel in a compact descriptor and then obtain an instance segmentation by clustering the pixels based on motion similarity. Unlike these works, our approach is driven primarily by a top-down learning algorithm followed by a simple linking step to generate spatio-temporal segmentations. The most relevant approach in this respect is [72], which trains a CNN to detect (but not segment) moving objects, and combines these detections with clustered pixel trajectories to derive segmentations. By contrast, our approach directly outputs segmentations at each frame, which we

link together with an efficient tracker. Very recently, Bideau *et al*. [23] proposed to combine a heuristic-based motion segmentation method [22, 163] with a CNN trained for semantic segmentation for the task of moving object segmentation. Their method, however, does not handle discontinuous motion. In addition, the fact that they rely strongly on heuristic motion estimates allows our learning-based approach to outperform their method on FBMS by a wide margin. In very recent work, Xie *et al*. [246] introduced a deep learning approach for motion segmentation that segments and tracks moving objects using a recurrent neural network. By comparison, our method uses a simple, overlap-based tracker that performs competitively with the learned tracker from [246] while producing significantly fewer false positive segmentations (see Supplementary).

Foreground/Background Video Segmentation: Several works have focused on the binary version of the video segmentation task, separating all the moving objects from the background. Early approaches [62, 135, 169, 230] relied on heuristics in the optical flow field, such as closed motion boundaries in [169] to identified moving objects. These initial estimates were then refined with appearance, utilizing external cues, such as saliency maps [230], or object shape estimates [135]. Another line of work focused on building probabilistic models of moving objects using optical flow orientations [22, 163]. None of these methods are based on a robust learning framework and struggle to generalize well to unseen videos. The recent introduction of a standard benchmark, DAVIS 2016 [174], has led to a renewed interest. More recent approaches propose deep models for directly estimating motion masks, as in [108, 214, 215]. These approaches are similar to ours in that they also use a two-stream architecture to separately process motion and appearance, but they are unable to segment *individual* object instances, one of our primary goals. Our method separately segments and tracks each individual moving object in a video.

Object Detection: The task of segmenting object instances from still images has seen immense success in recent years, bolstered by large, standard datasets such as COCO [142]. However, this standard task focuses on segmenting every instance of objects belonging to a fixed list of categories, leading to methods that are designed to be blind to objects that fall outside the categories in the training set.

Two recent works have focused on extending these models to detect generic objects. [96] aims to generalize segmentation models to new categories, but requires

bounding box annotations for each new category. More relevant to our approach, [109] aims to detect all "object"-like regions in an image, outputting a binary objectness mask. While we share their goal of segmenting unseen objects, our approach additionally provides instance masks for each object.

7.3 Approach

We propose a two-stream spatio-temporal grouping method that uses appearance and motion cues to segment all moving objects in a video. Our approach, illustrated in Figure 7.3, takes a frame together with a corresponding optical flow as input, and passes them through an "appearance stream" (top) and a "motion stream" (bottom) respectively. The resulting features are combined and passed to the joint region proposal network (RPN), which learn to detect and segment moving objects irrespective of their category.

Our approach shares inspiration with prior work that proposes two-stream approaches for object detection [72, 80, 84, 173], with two key differences. First, we design a novel region proposal module that learns to fuse both appearance and motion information to generate moving object detections. Second, to overcome the dearth of appropriate training data, we develop a stage-wise training strategy that allows us to leverage synthetic data to train our motion stream, image datasets to train our appearance stream, and a small amount of real video data to train the joint model.

We first discuss the architecture and training strategy for the motion and appearance streams individually, and then detail how to combine these streams into one coherent architecture. Finally, we describe a simple tracker that we use for linking detections across time, allowing us to produce spatio-temporal groupings that span across many frames.

7.3.1 Motion-based Segmentation

We start by training a motion-based instance segmentation model. As mentioned above, this requires videos with segmentation masks for all moving objects, which is difficult to obtain. Fortunately, prior work has shown that synthetic data can be used for some low-level tasks, such as flow estimation [53] and binary motion segmen-

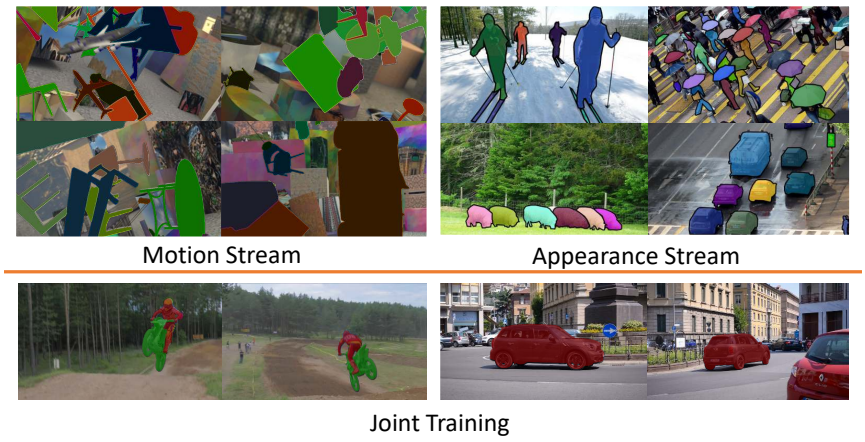


Figure 7.2: We train our motion stream on FlyingThings3D [157] (top left), our appearance stream on COCO [142] (top right), and our joint model on DAVIS'16 [174] and a YTVOS [249] subset (bottom).

tation [214]. Inspired by this, we train our motion stream on the FlyingThings3D dataset [157], which contains nearly 2,700 synthetically generated sequences of 3D objects traveling in randomized trajectories, captured with a camera also traveling along a random trajectory. The dataset provides groundtruth optical flow, as well as segmentations for both static and moving objects (See Figure 7.2). We train our motion-stream using the moving instance labels from [214], treating all moving objects as a single category, and all other pixels, including static objects, as background. The resulting model learns to segment moving objects irrespective of their category. In fact, this model is oblivious to the whole notion of an object and is capable of segmenting parts that exhibit independent motion (see Figure 7.5). We discuss more details and variants of this approach in Section 7.5.3.

7.3.2 Appearance-based Segmentation

In order to incorporate appearance information, we next train an image-based object segmentation model that aims to segment the full extent of generic objects. Fortunately, large datasets exist for training image-based instance segmentation models. Here, we train on the MS COCO dataset [142], which contains approximately 120,000 training images with instance segmentation masks for each object in 80 categories. We could train our appearance stream following the standard Mask R-CNN training

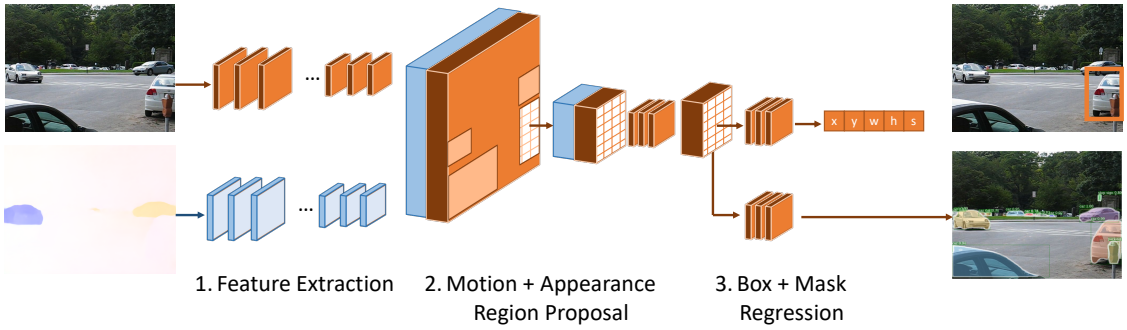


Figure 7.3: Our model uses an appearance stream (blue) and a motion stream (orange) to extract features from RGB and optical flow frames, respectively. Our region proposal network fuses features from both streams and passes them to the box and mask regression heads.

procedure, which jointly localizes and classifies each object in an image belonging to the 80 categories. However, this results in a model that, while proficient at segmenting 80 categories, is blind to objects from any other, novel category. Instead, we train an "objectness" Mask R-CNN by combining each of the 80 categories into a single "object" category. In Section 7.5.3, we will show that this "objectness" training (1) provides a significant improvement over standard training, and (2) leads to a model that generalizes surprisingly well to objects that are not labeled in MS COCO.

7.3.3 Two-Stream Model

Equipped with the individual appearance and motion streams, we now propose a two-stream architecture for fusing these information sources. In order to clearly describe our two-stream model, we take a brief detour to describe the Mask R-CNN architecture. Mask R-CNN contains three stages: (1) **Feature extraction:** a "backbone" network, such as ResNet [89], is used to extract features from an image. (2) **Region proposal:** A region proposal layer uses these features to selects regions likely to contain an object. Finally, (3) **Regression:** for each proposed region, the corresponding backbone features are pooled to a fixed size, and fed as input to bounding box and mask regression heads.

To build a two-stream instance segmentation model, we extract the backbone from our individual appearance-based and motion-based segmentation models. Next,

as depicted in Figure 7.3, we propose a "two-stream" RPN that uses these two backbones, instead of a single backbone, to predict proposals from *spatio-temporal* features, extracted from the optical flow (blue) and RGB (orange) backbones. These features are concatenated and fed to a short series of convolutional layers to reduce the dimensionality to match that of Mask R-CNN, allowing us to maintain the architecture of stages (2) and (3). Intuitively, we expect the appearance stream to behave as a generic object detector, and our motion stream to help detect novel objects that the appearance stream may miss and filter out static objects.

Although this may appear similar to prior approaches for building a two-stream detection model, it differs in a key detail: prior approaches obtain region proposals either only from appearance features [72, 80, 84], or from appearance and motion features individually [173]. By contrast, we propose a novel proposal module that *learns* to fuse motion and appearance features to find object-like regions.

We train our joint model on subsets of the DAVIS and YouTube Video Object Segmentation datasets (as detailed in Section 7.5.1). We experiment with various strategies for training this joint model in Section 7.5.3.

7.3.4 Tracking

So far, we have focused on segmenting moving objects in each frame of a video. To maintain object identities and to continue segmenting objects after they stop moving, we implement a simple, overlap-based tracker inspired by [19]. First, we remove all detections with score below α_{low}. On the first frame, all high scoring detections (score $> \alpha_{\text{high}}$) are used to initialize a track, which we define simply as a sequence of linked detections. At each successive frame, we compute the mask intersection over union between the most recent segmentation for each active track and predicted objects at $t+1$, and use Hungarian Matching to assign predicted objects to tracks. Unmatched predictions are discarded if their score is $< \alpha_{\text{high}}$; else, they are used to initialize a new track. Tracks that have not been assigned a new object for up to t_{inactive} frames are marked as inactive.

Tracking static objects: To continue tracking moving objects when they stop moving, we need to be able to detect static objects. A naïve way to do this is to run the objectness model trained in Section 7.3.2 in parallel with our two-stream model

at every frame. However, this would be computationally expensive. Fortunately, our appearance stream shares the backbone of the objectness model. Thus, we only need to apply the (inexpensive) stages (2) and (3) of the objectness model on the appearance features extracted by our two-stream network. Using this, we can efficiently output a set of moving and static object predictions for each frame in a video. We merge the two outputs by removing any predicted static object that overlaps with a predicted moving object. We use the same tracker described above, using only moving objects to initialize tracks.

7.4 Evaluation

To evaluate methods for spatio-temporal grouping, we desire a metric that rewards segmenting and tracking moving objects, but penalizes the detection of static objects or background. While there has been a rich line of prior work related to our goal, standard metrics surprisingly do not satisfy these criterion. We propose a novel metric that does.

The default metric in FBMS [165] was designed for grouping-based approaches, but does not penalize false positive predictions. Recently, Bideau *et al*. [21] tackled this issue by measuring the difference between the number of groundtruth moving objects and the number of predicted moving objects (Δ Obj). However, this complicates method comparisons by relying on two separate metrics; instead, we propose a single and intuitive F-measure that evaluates a method's ability to detect all and only moving objects.

Figure 7.4 (middle) visualizes the default FBMS metric which matches each predicted segment with a groundtruth segment so as to maximize IoU overlap, *ignoring* any unmatched predictions. This means the default F-measure does not penalize false positive segments, unfairly favoring methods that generate a large number of predictions. By contrast, our proposed F-measure, depicted in Figure 7.4 (right), counts unmatched predictions as false positives.

More precisely, we describe our metric roughly following the notation in [165]. For each video, let c_i be the pixels belonging to a predicted region i, and g_j be all the pixels belonging to a groundtruth non-background region j. While [165] omits unlabeled pixels from evaluation, we include all pixels in the groundtruth.

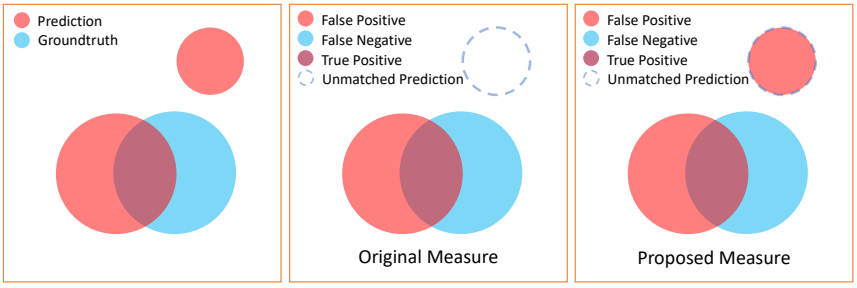


Figure 7.4: Left: we visualize a toy example with two predicted (red) segmentations and one groundtruth (blue) segmentation. While the original FBMS measure (middle) ignores predicted segments that do not match a groundtruth segment, such as the dashed circle, our proposed measure (right) penalizes all false-positives.

Let P_{ij} be the precision, R_{ij} be the recall, and F_{ij} be the F-measure corresponding to this pair of predicted and groundtruth regions, as follows:

$$P_{ij} = \frac{|c_i \cap g_j|}{|c_i|}, R_{ij} = \frac{|c_i \cap g_j|}{|g_j|}, F_{ij} = \frac{2P_{ij}R_{ij}}{P_{ij} + R_{ij}}$$

Following [165], we use the Hungarian algorithm to find a matching between predictions and groundtruth that maximizes the sum of the F-measure over all assignments. Let $g(c_i)$ be the groundtruth matched to each predicted region; for any c_i that is not matched to a groundtruth cluster, $g(c_i)$ is set to an empty region. We define our metric as follows:

$$P = \frac{\sum_i |c_i \cap g(c_i)|}{\sum_j |c_i|}, R = \frac{\sum_i |c_i \cap g(c_i)|}{\sum_i |g_j|}, F = \frac{2PR}{P + R}$$

Any unlabeled pixel in a predicted region c_i will reduce precision and F-measure, penalizing the segmentation of static or unlabeled objects. In our experiments, we report results with both the official and our proposed measure.

7.5 Experiments

We first analyze each component of our proposed model with experimental results. Next, we compare our approach to prior work in spatio-temporal grouping on three datasets.

7.5.1 Datasets

An ideal dataset for training our model would contain a large number of videos where every moving object has labeled instance masks, and static objects are not labeled. Three candidate datasets exist for this task: YouTube Video Object Segmentation (YTVOS) [249], DAVIS 2016 [174], and FBMS [165]. While YTVOS contains over 3,000 short videos with instance segmentation labels, not all objects in these videos are necessarily labeled, and both moving as well as static objects may be labeled. The DAVIS 2016 dataset contains instance segmentation masks (provided with DAVIS 2017) for only the moving objects, but only contains 30 training videos. Finally, although FBMS contains a total of 59 sequences with labeled instance segmentation masks for moving objects, prior work evaluates on the entire dataset, preventing us from training on any sequences in the dataset in order to provide a fair comparison.

To overcome this lack of data, we use heterogeneous data sources to train our model in a stagewise fashion. As described earlier, we train our appearance stream on COCO [142]. We train our motion stream on FlyingThings3D [157], a synthetic dataset of 2,700 videos of randomly moving 3D objects. Finally, we fine-tune our joint model on DAVIS2016 and the training subset of YTVOS-Moving. We use a held-out set of 100 YTVOS-Moving sequences for evaluation.

7.5.2 Implementation Details

Network Architecture: Our two-stream model is built off Mask R-CNN [90] with a ResNet-50 backbone. We will publicly release the code and exact configuration for training, highlight some important details here, and note further details in supplementary. All our models are trained using the publicly available PyTorch implementation of Detectron [219]. In general, we use the original hyper-parameters provided by the authors of Mask R-CNN. The backbone for every model is pre-trained on ImageNet [194]. When constructing our two-stream model, we initialize the bounding box and mask heads from the appearance-only model.

Tracking: We set the confidence threshold for initializing tracks, as described in Section 7.3.4, to $\alpha_{\text{high}} = 0.9$, and remove any detections with confidence lower than $\alpha_{\text{low}} = 0.7$. We allow tracks to stay alive for up to $t_{\text{inactive}} = 10$ frames (approximately $0.33s$ for most videos), although we found the final results are fairly insensitive to

this parameter. To detect objects before they move, we first run our tracker forwards, and then backwards in time.

7.5.3 Ablation analysis

Evaluation: We analyze our model by benchmarking various configurations on the DAVIS 2016 dataset [174]. For ablation, we found it helpful to use the standard detection mean average precision (mAP) metric [142] in place of video object segmentation metrics, which require tracking and obfuscate analysis of our architecture choices. We report both detection and segmentation mAP at an IoU threshold of 0.5

Motion stream

To begin, we explore training strategies for the motion stream of our model. We train our motion stream on the FlyingThings3D dataset, as described in Section 7.3.1. This dataset provides groundtruth flow, which we could use for training. However, at inference time, we only have access to noisy, estimated flow. In order to match flow in the real world, we estimate flow on FlyingThings3D using two optical flow estimation methods: FlowNet2 and LiteFlowNet. For both methods, we use the version of their model that is trained on synthetic data and fine-tuned on real data.

In Table 7.1, we compare three strategies for training on FlyingThings3D. We start by training using only FlowNet2 flow as input ("FlowNet2"). We hypothesize that training directly on noisy, estimated flow can lead to difficulties in early training. To overcome this, we train a variant starting with groundtruth flow, and fine-tune on FlowNet2 flow ("FlowNet2 ← Groundtruth" row). We find that this provides a significant improvement (2.7%). We also considered using a more recent flow estimation method, LiteFlowNet [102] ("LiteFlowNet ← Groundtruth" row). Surprisingly, we find that FlowNet2 provides significant improvements for detection, despite performing worse on standard flow estimation benchmarks. Qualitatively, we found that FlowNet2 provides sharper results along boundaries than LiteFlowNet, which may aid in localizing objects.

Figure 7.5 shows qualitative results of the motion stream. Despite never having seen real images with segmentation labels, this model is able to group together parts that move alike, while separating objects with disparate motion.

Table 7.1: Comparing training with different flow estimation methods on FlyingThings3D, reporting mAP on DAVIS '16 val. "← Groundtruth" means we first train with groundtruth (synthetic) flow. See Section 7.5.3 for details.

Flow type	Det @ 0.5	Seg @ 0.5
FlowNet2	40.5	23.9
FlowNet2 ← Groundtruth	43.2	24.1
LiteFlowNet ← Groundtruth	33.8	24.0

Figure 7.5: Despite being trained for segmentation only on synthetic data, our motion stream (visualized) is able to separately segment object parts in real objects. See Sec. 7.5.3 for details.

Appearance Stream

While our motion stream is proficient at grouping similarly-moving pixels, it lacks any priors for real world objects and will not hesitate to oversegment common objects, such as the man in Figure 7.5. To introduce these useful priors, we turn our attention to the appearance stream of our model.

As described in Section 7.3.2, we train our appearance stream on the COCO dataset [142]. We evaluate two variants of training. First, we train a standard, "class-specific" Mask R-CNN, that outputs a set of boxes and masks for each of the 80 categories in the COCO Dataset. At inference time, we combine the boxes and masks predicted for each category into a single "object" category. Second, we train an "objectness" Mask R-CNN, by collapsing all the categories in COCO to a single category *before* training.

We show results from these two variants in Table 7.2. Our "objectness" model significantly outperforms the standard "class-specific" model by nearly 8%. We further compare the two models qualitatively in Figure 7.6, noting that our objectness model

Table 7.2: Comparison of training our appearance stream with and without category labels on MS COCO (Class-specific and Objectness, respectively), reporting mAP on DAVIS '16 val. Training without category labels allows the model to generalize beyond the training categories. See also Figure 7.6 and Section 7.5.3

COCO Training	Det @ 0.5	Seg @ 0.5
Class-specific	42.0	40.2
Objectness	49.8	48.3

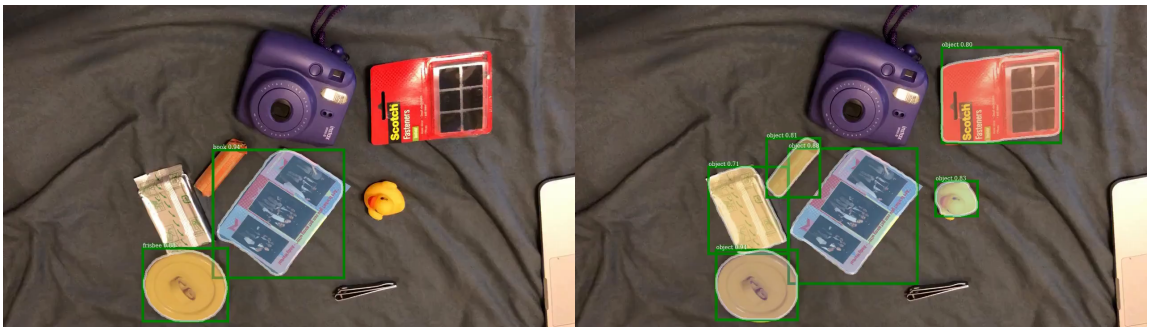


Figure 7.6: Unlike standard object detectors trained on COCO (left), our objectness model (right) detects objects from categories outside of COCO, such as the packet of film, a roll of quarters, a rubber duck, and a packet of fasteners. Both models visualized at confidence threshold of 0.7. See Section 7.5.3 for details.

better generalizes to non-COCO categories.

Joint training

Finally, we combine our appearance and flow streams in a single two-stream model, depicted in Figure 7.3 and described in detail in Section 7.3.3. We experiment with different strategies for training this joint model. Throughout these experiments, we initialize the flow stream with the "FlowNet2 ← Groundtruth" model from Section 7.5.3, and use the objectness model from Section 7.5.3 to initialize the appearance stream, the box and mask prediction heads, and the RPN. We show the results in Table 7.3.

We start by training this joint model directly on the DAVIS 2016 training set, which achieves 79.1% mAP. We note that even with joint-training, using the objectness model for initialization provides a significant boost over using a category-specific detector (73.8%). Next, to maintain the generalizability of the objectness model,

Table 7.3: Comparing two-stream training strategies, reporting mAP on DAVIS '16 val. Preserving knowledge from the individual streams is critical for good accuracy. See Section 7.5.3 for details.

Variant	Det @ 0.5	Seg @ 0.5
Joint Training, class-specific	73.8	70.3
Joint Training, objectness	79.1	73.3
+ Freeze appearance	81.9	76.7
+ Freeze motion	83.7	76.4
+ Freeze mask	83.9	77.4

Table 7.4: Comparing training sources, reporting mAP on DAVIS '16 val. The lack of static objects in 'YTVOS-moving' leads to worse performance, but fine-tuning on DAVIS provides the best model. See Section 7.5.3 for details.

Joint Training Data	Det @ 0.5	Seg @ 0.5
DAVIS	83.9	77.4
YTVOS-moving	79.9	75.8
DAVIS ← YTVOS-moving	85.1	77.9

we also train a variant where we freeze the weights of the appearance stream. This provides nearly a 3% improvement in accuracy. Similarly, to maintain the generic "grouping" nature of the synthetically-trained flow stream, we freeze the flow stream, providing us with an additional 2% improvement.

Finally, we hypothesize that while features from the flow stream are helpful for localizing generic moving objects, appearance information is sufficient for segmentation. We verify this hypothesis by training one last variant where the mask head uses only appearance stream features, and freeze its weights to those of the objectness model. Indeed, this provides a modest improvement of 1% in segmentation AP.

Training Data: Next, we train our joint model on YTVOS-Moving (Section 7.5.1) and show results in Table 7.4. Unfortunately, this dataset contains very few static objects, causing the model to detect both static and moving objects, leading to a significant (5%) drop in performance. However, fine-tuning this model on the DAVIS 16 training set leads to our best model (DAVIS ← YTVOS-moving).

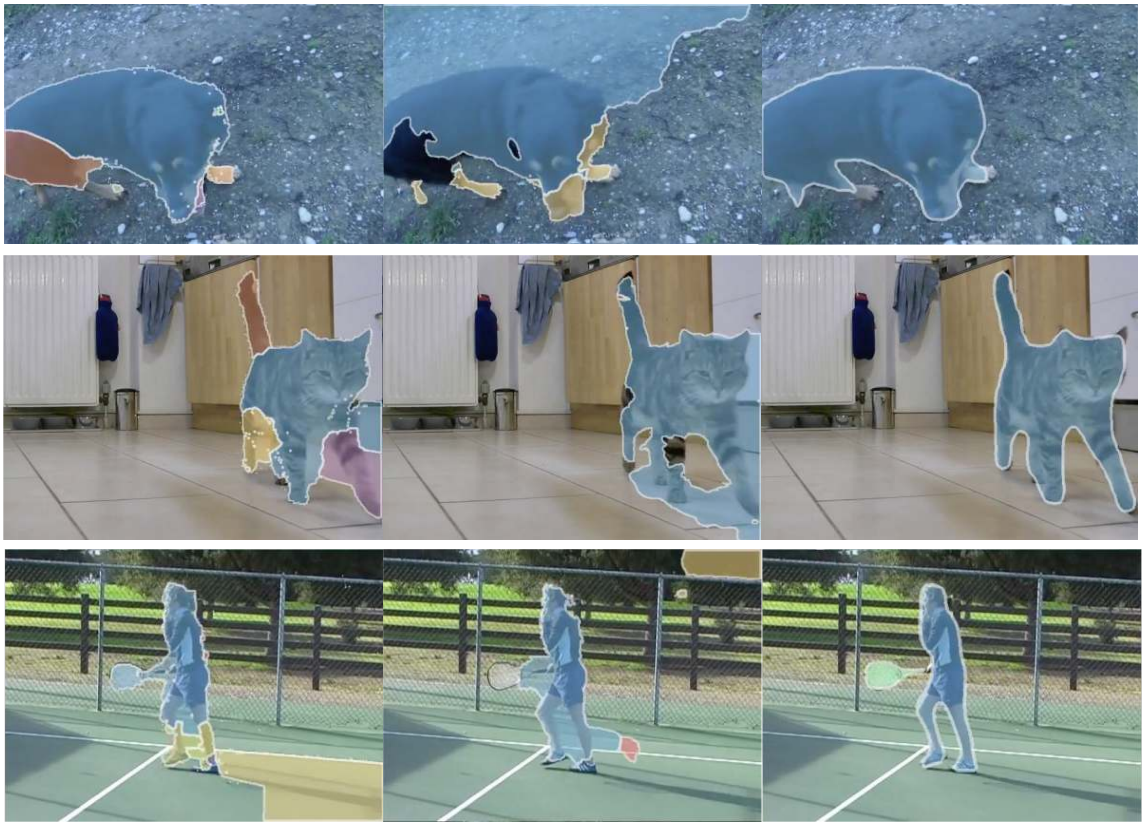


Figure 7.7: Qualitative results comparing our approach to two state-of-the-art methods. Prior work frequently exhibits over- or under-segmentation, such as the cat (middle row, [119]) and the dog (top row, [23]), respectively. Our method fuses motion and appearance information to segment the full extent of moving objects.

Table 7.5: FBMS 59 results using the official metric [165], which does not penalize detecting unlabeled objects. We report precision (P), recall (R), F-measure (F), and the number of objects for which the F-measure > 0.75 (N). Ours-A is our model's appearance stream only, and Ours-J is our joint model. Both Ours-A and Ours-J out-perform all prior work. As expected, since this metric does not penalize false positives, Ours-A outperforms Ours-J.

	Training set				Test set			
	P	R	F	$N/65$	P	R	F	$N/69$
[212]	83.0	70.1	76.0	23	77.9	59.1	67.3	15
[119]	86.9	71.3	78.4	25	87.6	70.2	77.9	25
[255]	89.5	70.7	79.0	26	91.5	64.8	75.8	27
[120]	**93.0**	72.7	81.6	29	**95.9**	65.5	77.9	28
Ours-A	89.2	**79.0**	**83.8**	**43**	88.6	**80.4**	**84.3**	**40**
Ours-J	85.1	78.5	81.7	39	80.8	75.8	78.2	39

7.5.4 Comparison to prior work

Official FBMS: We first evaluate our method against prior work on the standard FBMS benchmark in Table 7.5. As discussed in Section 7.5.1, this metric does not penalize false positive detections. As expected, our appearance stream alone, despite segmenting both static and moving objects, performs best on this metric ('Ours-A'), outperforming all prior work by 6.4% in F-measure on the TestSet, and 2.2% on the TrainingSet [1]. For completion, we also report the performance of our joint model ('Ours-J'), which compares favorably to state-of-the-art despite the flawed metric. Our improvements on this metric are likely driven by improvements in segmentation boundaries (see Figure 7.7).

Proposed FBMS: Finally, we report results on our proposed metric in Table 7.6. Recall that our proposed metric generally follows the official metric, but additionally penalizes detection of static objects. We compare to all methods from Table 7.5 whose final results on FBMS were accessible or provided by the authors through personal communication. On this proposed metric, we first note that, as expected, the performance of our appearance model baseline is significantly worse than our final, joint model, by 9.2% on TestSet and 6% on TrainingSet in F-measure. More importantly, our final model strongly out performs prior work in F-measure by 11.3%

[1]Note that despite the name, we do not use either set for training.

Table 7.6: FBMS 59 results on our proposed metric. Ours-A is our appearance stream, Ours-J is our joint model. We compare to prior methods for which we were able to obtain code or results.

	Training set			Test set		
	P	R	F	P	R	F
[212]	74.8	61.7	65.5	66.8	49.2	53.6
[119]	68.1	68.5	67.1	70.0	64.6	65.0
Ours-A	61.6	**80.4**	64.0	66.8	**84.7**	70.3
Ours-J	**75.0**	77.8	**73.2**	**77.0**	83.0	**76.3**

on the TestSet, and 6.1% on the TrainingSet. In addition to improving segmentation boundaries, our approach effectively removes spurious segmentations of background regions and object parts (Figure 7.7).

Qualitative results: We qualitatively compare our approach with Keuper *et al*. [119] and Bideau *et al*. [23] in Figure 7.7[2] In the top row of Figure 7.7, [119] oversegments the dog into multiple parts, and [23] merges the dog with the background, whereas our approach fully segments the dog. Similarly, the cat in the middle row is over-segmented by [119] and under-segmented by [23], but well-segmented by our approach. In the final row, both [119] and [23] exhibit segmentation and tracking errors; the region corresponding to the man's foot (colored yellow for Keuper *et al*. and red for Bideau *et al*.) are mistakenly tracked into a background region thus segmenting part of the background as a moving object. Meanwhile, our object-based tracker fully segments the person and the tennis racket with high precision. We show further qualitative results in supplementary material.

DAVIS-Moving: We further evaluate our method on a subset of the DAVIS 17 dataset. Unlike DAVIS 2016, the 2017 version provides instance-level masks for objects, but contains sequences with labeled static or unlabeled moving objects. For evaluation, we manually select 22 of 30 validation videos without these issues, and refer to this subset as DAVIS-Moving. We compare to [119], the best FBMS method we can obtain code for, with our proposed metric in Table 7.7. Surprisingly, we find a much larger gap in performance on this dataset; while [119] achieves 42.3% on F-measure with our proposed metric, our approach improves significantly to 77.9%. We

[2] [23] only segments objects while they move. We provide an evaluation using an alternative FBMS labeling they propose in our supplementary.

Table 7.7: DAVIS-Moving results on our proposed metric. We compare to the best FBMS method for which we could obtain code.

	P	R	F
[119]	39.4	53.8	42.3
Mask R-CNN	70.8	75.6	71.6
Ours	**78.3**	**78.8**	**78.1**

Table 7.8: YTVOS-Moving results on our proposed metric. For fairness, we evaluate our method *without* YTVOS training. We compare to the best FBMS method for which we could obtain code.

	P	R	F
[119][3]	35.3	28.7	26.6
Mask R-CNN	70.4	49.5	53.6
Ours w/o YTVOS	**74.5**	**66.4**	**68.3**

believe this gap may be due to faster, more articulated motion and higher resolution videos in DAVIS 17, which severely affect [119] but not our method.

YTVOS-Moving: Finally, we evaluate on sequences from YTVOS-Moving (selected from YTVOS, as described in Section 7.5.1). Unlike FBMS and DAVIS, YTVOS contains diverse objects, such as octopuses and snakes. For fairness, we evaluate a version of our final model that was never trained on YTVOS, and show results in Table 7.8. We show that Mask R-CNN struggles to detect such objects, while our approach strongly improves performance from 53.6% to 67.7% in F-measure. We further break down these results by splitting the YTVOS-Moving dataset into two subsets: videos which contain COCO-category objects, which our model has seen

[3][119] errored on some sequences, so we report numbers on a subset. By comparison, Ours w/o YTVOS achieves 71.9% F-measure on this subset.

Table 7.9: YTVOS-Moving results on seen (COCO) vs. novel objects using our proposed metric.

	COCO Objects			Novel objects		
	P	R	F	P	R	F
[119]	28.2	25.4	20.6	41.8	31.6	31.9
Mask R-CNN	77.6	60.9	65.1	61.9	37.1	40.6
Ours w/o YTVOS	74.4	66.8	**66.8**	74.6	66.2	**67.6**

during training, and videos which contain novel objects not from COCO categories in Table 7.9. While Mask R-CNN is competitive with our approach on COCO categories (underperforming our model by 1.7% F-measure), it significantly underperforms compared to our approach on novel objects, by 27% F-measure. We show qualitative results in supplementary material.

7.6 Conclusion

We proposed a simple learning-based approach for spatio-temporal grouping. Our method provides two key insights. First, learning based approaches are able to generalize to never-before-seen objects (Section 7.5.3). Second, synthetic data can be used to train a truly generic grouping method with little priors on real world objects. As a result, our approach achieves state-of-the-art results on the FBMS benchmark dataset. Finally, to enable further research in this direction, we introduced a new metric as well as two new benchmarks (DAVIS-Moving, YTVOS-Moving).

Acknowledgements: We thank Pia Bideau for providing evaluation code, and Nadine Chang, Kenneth Marino and Senthil Purushwalkam for reviewing drafts and discussions. Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00345. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes not withstanding any copyright annotation theron. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of IARPA, DOI/IBC or the U.S. Government.

Chapter 8

Learning to Track Any Object

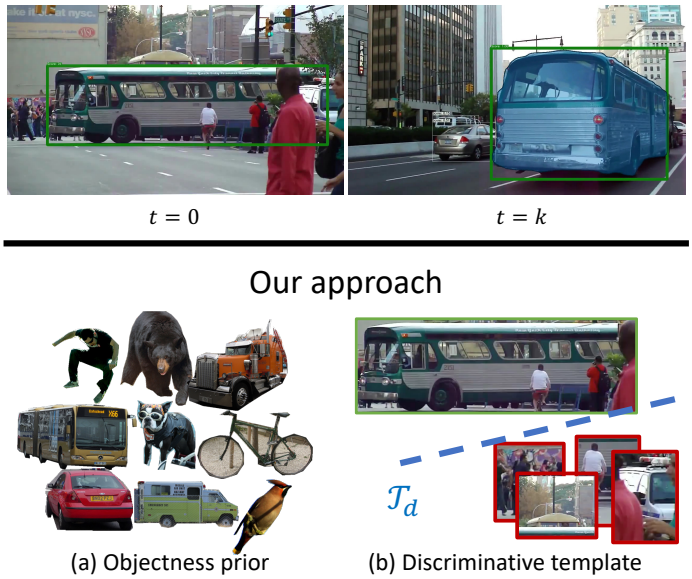


Figure 8.1: Objects of interest in generic, user-initialized tracking share a common set of *objectness* traits. Our approach (a) learns a generic *objectness* prior from image-based datasets, and (b) adapts it to a specific object of interest (e.g. the bus in the top left) by computing a linear discriminator between the object and its background in closed form. This allows tracking objects through significant deformations without latching onto distractors.

8.1 Introduction

Tracking is an essential element of video analysis. Extracting spatio-temporal regions corresponding to objects from a video is not only the end goal for surveillance and video labeling [107], but also an important intermediate representation for tasks such as action recognition [234, 263]. The previous chapter focuses primarily on *segmenting* never-before-seen objects, and tracking them for short periods of time with a simple linking approach. In this chapter, we tackle long-term tracking through challenging, diverse scenes.

Unfortunately, tracking in general is notoriously difficult and potentially ambiguous. Consider the example in Figure 8.1 of tracking a bus with only one side visible initially. Without prior knowledge, it is unclear whether the back side of the bus (visible in future frames) is a different viewpoint of the same bus, or a new object itself. In practice, many tracking approaches struggle to resolve such ambiguities, tending to diverge to an object part which is most similar to the initial template (e.g., the back window of the bus). Successful tracking in these scenarios necessitates *object priors*. Indeed, approaches for category-specific tracking, where the tracked object categories are known before hand, heavily rely on priors in the form of category-specific object detectors [6, 16, 239, 244]. By contrast, approaches for user-initialized tracking have largely eschewed such priors [18, 26, 274] in the pursuit of tracking generic objects, sometimes known as model-free tracking [126, 242]. However, generic objects still share a common set of *objectness* traits [5]. How can we operationalize this implicit constraint into a useful prior?

In this work, we repurpose category-*specific* appearance models into a generic *objectness* prior that can be used for category-*agnostic* tracking. In essence, we show that model-free tracking is far easier with better models! Doing so requires tackling two key challenges, shown in Figure 8.1: (1) How do we best adapt a category specific prior into a generic objectness prior? (2) How do we further adapt this generic prior to the particular instance of interest?

To address (1), we build a joint model for category-specific object detection and category-agnostic tracking (Figure 8.2). It is based on the Mask R-CNN [90] object detection architecture. For tracking, it takes as an additional input an object template in the first frame and computes its feature embedding. This template is then used to

compute the similarity between the object of interest and a new frame. The similarity map is in turn applied to reweight spatial features from the new frame to detect only the object of interest. Importantly, training the network jointly on image and video datasets, allows us to both capture a generic object appearance model from the diverse image data and learn to use it in a category-agnostic way for tracking.

To address (2) – e.g., better separating the bus in Figure 8.1 from other vehicles, such as the van on the right – we propose a lightweight on-the-fly adaptation strategy. We compute a linear separator ($\mathcal{T}_d$ in Figure 8.1) between the object of interest and other objects in the first frame. This separator is computed in closed form in a fully differentiable manner, and applied in future frames to compute similarities.

An intriguing property of our proposed architecture is that it can be used both as a single-object tracker and an object detector. Moreover, by capitalizing on the mask prediction branch of [90], we are able to train and test the same network for instance and video object segmentation. To sum up, we present a single unified approach for object detection, tracking, instance and video object segmentation.

We evaluate our model on two very recent, large scale datasets for object tracking: OxUvA [222] and GOT [98]. The former is focused on long-term object tracking, with objects undergoing a lot of appearance variation and occlusion. In contrast, the videos in GOT are shorter, but contain diverse object categories, covering more than 560 object classes. On both datasets our method outperforms the state-of-the-art by a large margin. Next, we show results competitive with the state-of-the-art on the LTB-35 dataset from the VOT 2018 Long Term challenge [127]. Finally, we validate the quality of our masks on DAVIS'17 dataset for video segmentation [180], demonstrating that our unified approach performs on par with specialized video segmentation methods that don't finetune on the test videos.

Our contributions are three-fold: (1) we incorporate an objectness prior in a generic tracker with a joint model for object detection, tracking, instance and video object segmentation; (2) we propose a lightweight strategy for computing discriminative object templates in an end-to-end fashion for efficiently handling distractors; (3) our method demonstrates state-of-the-art results on three benchmark datasets for object tracking and video object segmentation.

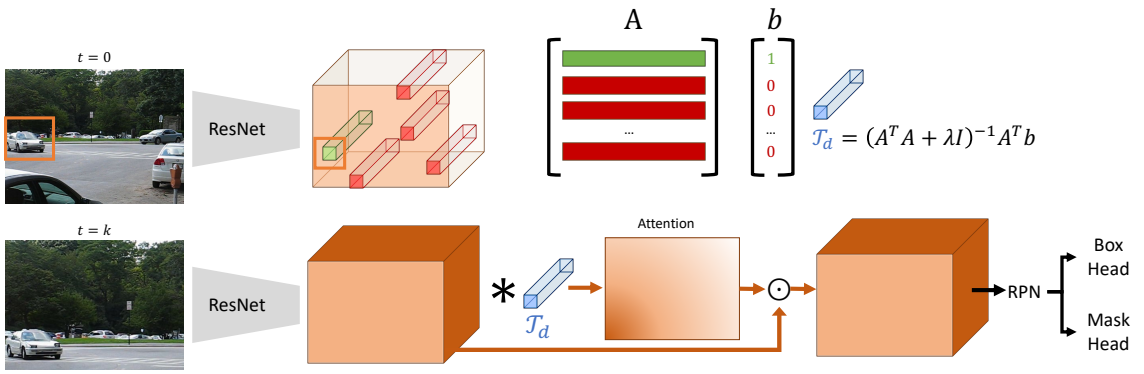


Figure 8.2: Overview of our approach. First, we use a state-of-the-art object detector [90] to extract features for the template image containing the object to be tracked (top left). Next, we compute a discriminative template that separates the features corresponding to the tracked object from the distractors in the first frame using linear regression (top right). Finally, attention masks computed with this template are used to reweight the feature maps of the detector to focus on the object of interest (bottom). Note that unlike standard, category-specific detectors, our box-head and mask-head output a single, category-agnostic prediction for the tracked object.

8.2 Related work

Single object tracking. Classical approaches for single object tracking, which requires tracking an object given a bounding box annotation in the first frame, were based on the tracking-by-detection paradigm: in many cases the detector is used to first to localize all the objects in a frame. The box corresponding to the object of interest was then selected by a discriminative classifier trained on the first frame annotation [9, 97, 114]. Correlation filters were commonly used for classification due to their efficiency [26]. To address appearance variation, some models updated the object template over time [97, 208]. Recent approaches learn correlation filters on top of deep features [43, 221].

Current methods for tracking largely ignore the objectness prior provided by detectors. Instead, they rely on a Siamese network architecture (initially introduced for signature verification [29]) adapted for tracking [18, 91, 210].

Recently, there have been several attempts to introduce ideas from CNN-based detection architectures into Siamese trackers. In particular, Li et al. [137] use the

similarity map obtained by matching the object template to the test frame as input to an RPN-like module adapted from Faster R-CNN [190]. Later this architecture was extended by introducing hard negative mining and template updating [274], adding a mask prediction branch [228], and using deeper models [138]. Our approach differs in that instead of integrating components of object detectors into a tracking pipeline in a heuristic way, we turn a state-of-the-art object detection framework into a tracker. This allows our model to fully utilize the objectness prior learned on COCO, outperforming the heuristic-based approaches significantly.

Video object segmentation. Methods for video object segmentation take a precise object mask as input in the first frame and output pixel-level segmentations for the object in each frame. Early methods for this task were based on mask propagation through a graph connecting superpixels in the neighboring frames [218, 236]. More recently, these methods have been outperformed by deep-learning based approaches, which capitalize on the success of image segmentation architectures [32, 175]. In particular, they fine-tune a model trained for foreground-background segmentation using the annotation in the first frame and evaluate it on the remaining frames of the video. Some approaches also update the model using its own predictions to handle appearance variation [223]. While these methods demonstrate impressive accuracy, they remain slow due to the need to update the model during evaluation. Alternative approaches, that do not require network fine-tuning have been proposed recently [38, 224, 253], but remain inferior in performance.

These approaches treat video object segmentation as a problem *independent* from object tracking, with the recent exception of [228]. In contrast, we adapt the intuition from [90] that instance masks can be computed as a by-product of object detection. Our tracker with a mask prediction branch achieves competitive performance on DAVIS'17 video object segmentation benchmark without requiring mask-level supervision on the first frame.

Object detection. CNN architectures for detection have brought significant progress, replacing classical methods for object detection that relied on hand-crafted features and part-based models [68]. Early approaches [78, 79] trained CNNs to classify pre-computed object proposals. More recent approaches solve the detection problem

in an end-to-end way [145, 188, 190]. In particular, RCNN-like architectures [90, 190] operate in a two-stage fashion: first an RPN proposes a set of boxes, and pools features from each box region. Next, separate branches classify the object and refine the box coordinates. [143] introduced feature pyramid network (FPN) to aggregate features from several network layers. Finally, Mask R-CNN [90] extended this model to instance segmentation by adding a mask prediction branch. In this work, we convert this architecture into an object tracker by introducing a lightweight discriminative template matching block before the RPN. The resulting attention map guides the RPN to propose only boxes corresponding to the object of interest. Disabling the matching component turns the model into a standard object detector.

8.3 Method

An ideal model for tracking by detection can be described as a generic object detector that can be efficiently adapted to detect a specific object in a specific scene. In this section, we propose such an approach, shown in Figure 8.2. Our model leverages advances in standard object *detection* architectures by progressively incorporating modifications to build a state-of-the-art *tracker*, while maintaining the model's detection capabilities.

We begin by briefly describing the Mask R-CNN architecture in Section 8.3.1. We then discuss our strategy of incorporating Siamese-like template matching into this model in a principled way in Section 8.3.2. Next, we propose our discriminative templates that efficiently integrate information about the distractors in Section 8.3.4. Finally, we discuss our strategy for training the unified model on object detection, tracking, and video segmentation datasets in Section 8.3.5.

8.3.1 Preliminaries

A Mask R-CNN detector, shown in Figure 8.2, consists of a backbone network (often a ResNet), a Region Proposal Network, and bounding box classification, regression and mask prediction heads. The former takes a frame as input and outputs a set of feature maps $\{C_1, C_2, C_3, C_4, C_5\}$, extracted from the respective blocks of the backbone and encoding the image with different degrees of spatial and semantic

granularity. In practice, the output of the first block is discarded, due to memory constraints. The remaining feature maps are then updated via top-down lateral connections to propagate the information for the coarse but semantically rich top layers to the more spatially precise bottom layers, resulting in the final set of feature maps $\{P_2, P_3, P_4, P_5\}$ (see [143] for details). The feature dimensionality of these maps is fixed to 256, but their spatial dimensions decrease from fine to coarse, thus the resulting architecture is referred to as Feature Pyramid Network (FPN).

An RPN is implemented as a 3×3 convolutional layer that is applied to each FPN level in a sliding window fashion, outputting an objectness score for each of the anchor boxes centered at the corresponding location. Crucially, the anchor boxes only capture various aspect ratios of the boxes, wheres scale variation is handled by the FPN. That is, a $1 \times 1 \times D$ dimensional feature at location (x, y) in P_5 represents the largest possible object centered at that location, whereas a feature of the same dimension at the corresponding location in P_2 represents the smallest possible object in centered in the same region. We use this observation to derive our scale-invariant object template in Section 8.3.2.

Finally, the top k boxes according to the RPN score are selected, and an ROI-Pool operation is used to convert their feature representations to a fixed size. The resulting features are passed to separate bounding box classification, regression, and mask prediction branches (see [90] for details). We now describe our approach to efficiently adapting this architecture to the task of object tracking.

8.3.2 Tracking as generalized object detection

Given a bounding box around the object of interest in the first frame, how can we adapt the Mask R-CNN detector to only track that specific instance? We take inspiration from Siamese-based approaches for tracking that store an object template from the first frame and compute a similarity between the template and the test frame representation in a sliding-window fashion. Differently from those methods, instead of localizing the objects directly via template matching, we use the resulting similarity map to reweight the feature representation of an object detector. This allows us to reuse the rest of the detection architecture and train the model jointly on images and videos.

Our key observation is that every object can be represented by a $1 \times 1 \times D$ feature $\mathcal{T}$ in one FPN layer, corresponding to its scale and center location. Thus, we begin by extracting the corresponding representation for the template box in the first frame. In the standard detection training setup, Mask R-CNN assigns each groundtruth box in an image to a specific level in the feature pyramid, adding a loss that enforces that features at that scale generate a proposal around the groundtruth box. We use this same mapping to map our template box to the corresponding FPN level, and use the feature in that level that corresponds to the center of the template box. At test time, however, the scale of the object might have changed. Conveniently, we do not need to update the template to account for this, since scale variation is already handled by the FPN. Thus, we simply compute the similarity maps at all the levels of the feature pyramid via $S_i = P_i \star \mathcal{T}$, where $\star$ stands for cross-correlation.

Next, instead of directly using the resulting similarities to localize the object, we propose to instead treat them as attention maps to guide the detector. To this end, we update the original FPN representations via $P_i \leftarrow P_i \cdot S_i$, where $\cdot$ stands for the dot product. Notice that this operation simply reweights the original representation, preserving the information used by the RPN in the next stage. Thus, we can naturally capitalize on the strong objectness prior learned by detectors on COCO, as well as learn to produce objects masks for free. This re-weighted feature representation is used to generate and pool features for region proposals. The pooled, re-weighted features are finally passed through class-agnostic bounding box and mask regression heads.

At test time, our model produces multiple detections with confidence scores at every video frame. By default, we select the highest-scoring detection to construct the track, but we can make use of multiple detections by re-ranking them with external cues, such as predictions of an object dynamics model, or temporal smoothness cues (Section 8.4.2).

8.3.3 Joint Detection and Tracking

The modifications described above convert a standard Mask R-CNN detector into a tracker, which can not directly be used as a standard detector. We present two modifications that allow the tracker to be trained and evaluated as a standard, image-

based detection model. First, when applied to a single image, we disable the attention module. Equivalenty, this can be thought of as setting the attention to a uniform value of 1 at all pixel locations. Second, in order to output a class-specific bounding box and mask, as in standard Mask R-CNN, we instantiate a separate final layer for the box and mask regression heads for detection. Note that our model shares all parameters for detection and tracking *except* this final fully connected layer. We show in Section 8.4.3 that training jointly for detection with tracking improves tracking accuracy, while allowing our model to operate as a powerful single-frame detector, which can be useful for identifying distractor objects during tracking.

8.3.4 Discriminative Templates

Consider the frames from one of the videos in GOT shown in Figure 8.3 together with the corresponding similarity map S_i from the appropriate FPN level of our model. The model is supposed to track the cart in this video, however, the similarity map for the test frame shown in the bottom left is not localized on the object. We now propose a simple and efficient way of learning a discriminative template, increasing the robustness of the tracker.

Recall that in the FPN a feature vector at each location encodes an object centered at that region at the corresponding scale. Thus, sampling a large enough pool of features from all the levels outside of the ground truth bounding box naturally provides us with a training set for learning a linear discriminator for the object of interest in a given video. Moreover, such a discriminator can be found efficiently in a closed form via least squares. In particular, given a template $\mathcal{T}$ and a set of negatives $N = \{\mathbf{n}_1, \mathbf{n}_2, \ldots, \mathbf{n}_q\}$, we define the data matrix A, and the label vector $\mathbf{y}$ as follows:

$$A = \left[\mathcal{T}; \mathbf{n}_1; \mathbf{n}_2; \ldots; \mathbf{n}_q\right], \mathbf{y} = \left[1; 0; 0; \ldots; 0\right] \tag{8.1}$$

We then want to find a vector $\mathcal{T}_d$, which we call a discriminative template, that minimizes $\|A\mathcal{T}_d - \mathbf{y}\|_2^2$ holds. A closed form solution is available via:

$$\mathcal{T}_d = (A^T A + \lambda I)^{-1} A^T \mathbf{y}, \tag{8.2}$$

where I is the identity matrix and λ is a regularization hyper-parameter. We then

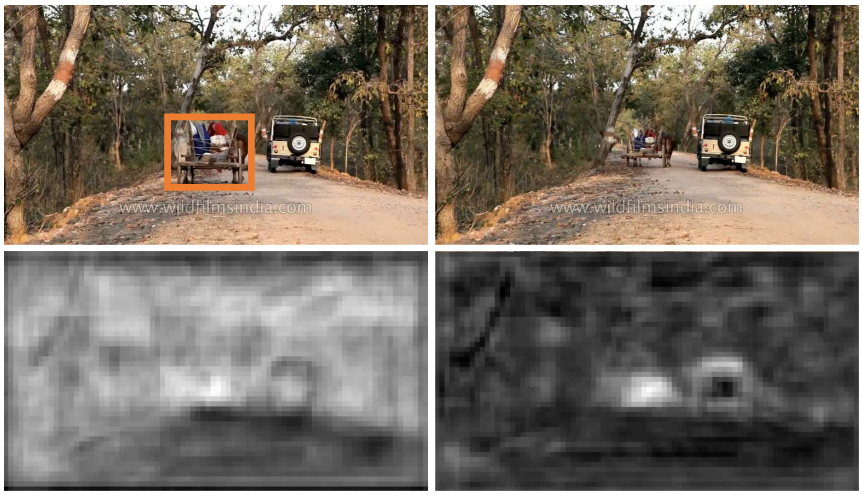


Figure 8.3: The effect of our proposed discriminative template on an example of a video from GOT-10k dataset [98]. By simply using the center feature of the bounding box around the cart (top left), the resulting attention map (bottom left) for the test frame (top right) is not focused on the object. In contrast, our discriminative template (bottom right) results in a much better attention map.

use $\mathcal{T}_d$ to compute the similarity maps in the same way: $S_i = P_i \star \mathcal{T}_d$.

Note that computing $\mathcal{T}_d$ requires only a matrix inverse and matrix multiplications, operations which are fully differentiable in the elements of A, and can be implemented in standard deep learning frameworks. Thus, we can backpropagate though this computation. This guides the backbone to learn a feature space where objects can be separated via a linear classifier in an end-to-end manner.

Figure 8.3 (bottom right) shows that our discriminative template indeed significantly increases the precision of the similarity maps by incorporating the information about the distractors in a principled way. We now describe how we train our unified framework on dataset for object detection, tracking, and video segmentation.

8.3.5 Training

We first train our model on COCO for object detection [142] following Mask R-CNN training [90]. We then transfer the learned objectness prior to the tracking task.

To this end we add the discriminative template computation, and attention reweighting components described above, and fine-tune it on the ImageNet VID [194]

and YTVOS [249] datasets. As ImageNet Vid does not provide segmentation groundtruth, we do not use it to update the mask branch. When fine-tuning for tracking, we make three simple modifications: (1) Our training batches consist of *pairs* of frames: for a batch of size K, we sample K videos, and then sample a *template* frame and a *search* frame at random from the video[1]; (2) We re-weight FPN features of the search frame using the feature corresponding to the template frame's bounding box; (3) Only a single, class-agnostic groundtruth box is used for training, which is the one corresponding to the tracked object in the search frame. These minor modifications allow us to maximally preserve the objectness priors learned on COCO.

8.4 Experiments

We begin with introducing the datasets used to train and evaluate our model, and providing the implementation details. Next we analyze the various choices made while designing our approach in Section 8.4.3. Finally, we compare our method to the state-of-the-art in Section 8.4.4.

8.4.1 Datasets and evaluation

We use the COCO [142] dataset to train our model for object detection, and ImageNet VID and YTVOS [249] to train the tracking module. We evaluate on two very recent, large scale tracking benchmarks: OxUvA [222] for long term tracking and GOT [98] for tracking of diverse objects. In addition, we use the DAVIS'17 [180] dataset for video object segmentation to evaluate the quality of the masks produced by our tracker, and the LTB35 videos from the VOT 2018 long term challenge to benchmark [127] against prior submissions to the challenge. We describe each of these datasets in more detail in the supplementary material.

[1]While we could limit the template frame to be the first frame of the video (as at test time), this would drastically reduce the diversity of our frame pairs.

8.4.2 Implementation details

Network architecture and training We use the Mask R-CNN detection framework throughout our experiments. In particular, we use the ResNet-50 FPN backbone, which achieves a useful balance between accuracy and efficiency. Our final model is trained for detection on MS COCO, as described in Sec. 8.4.3 and for bounding-box tracking on ImageNet VID and YTVOS. We will release training and evaluation code along with trained models upon acceptance.

Temporal heuristics Prior tracking approaches rely heavily on temporal information to simplify the tracking problem. As these heuristics can obscure the improvement of the underlying matching approach, we show results using *no temporal information* in Sec. 8.4.3 and 8.4.4, and show state-of-the-art results without heuristics on OxUvA. For completeness, we implement one simple heuristic which we ablate in Sec. 8.4.3. At every frame t, our detector outputs a set of candidate detections $D_t = \{d_{1,t}, \ldots, d_{k,t}\}$, along with a confidence score $c_{i,t}$ for each detection. In our standard implementation, we select the detection d_t^* with the highest confidence. To incorporate temporal smoothness, we implement a simple heuristic: for each candidate box $d_{i,t}$, compute the mask intersection-over-union $j_{i,t}$ with the predicted detection d_{t-1}^* at the previous frame, and update the confidence as $c_{i,t} \leftarrow \alpha c_{i,t} + (1-\alpha) j_{i,t}$. Then, we select d_t^* as the detection that maximizes this reweighted confidence. We set $\alpha = 0.6$ for all of our experiments. In order to avoid latching onto distractors, we temporarily disable this smoothness component if the track is broken, i.e. the IoU between the object locations at time t and $t+1$ is small ($< \alpha_{\text{low}}$); we re-enable the smoothness component if we maintain a smooth track for n frames, i.e. a track with consecutive object locations that have IoU $> \alpha_{\text{recover}}$. We set $\alpha_{\text{low}} = 0.1$, $\alpha_{\text{recover}} = 0.3$, and $n = 30$. We always show results both with and without this component for clarity.

8.4.3 Ablation study

In this section we analyze the influence of different components of our approach on the final performance. We use the *dev* sets of OxUvA and GOT-10k datasets for analysis, due to their complexity and diversity. Note that OxUvA requires explicitly thresholding confidence scores in order to detect when an object is not present in

Table 8.1: Evaluating the influence of different components of our approach on the OxUvA *dev* and GOT-10k *val* sets.See Section 8.4.3 for details.

Det Init?	Joint Det Train?	$\mathcal{T}_d$	Smooth?	OxUvA AUC	GOT AO
✗	✗	center	✗	63.2	64.7
✓	✗	center	✗	64.9	68.4
✓	✓	center	✗	65.8	68.6
✓	✓	mean diff	✗	67.6	68.8
✓	✓	mean pos	✗	69.1	69.1
✓	✓	lin. reg.	✗	71.1	69.5
✓	✓	lin. reg.	✓	72.1	73.0

the video. For ablation, we report the area under the ROC curve (i.e., TPR vs. FPR curve) to better understand the performance of ablated components across score thresholds. GOT-10k does not require setting such a threshold, so we use the standard Average Overlap metric described in Section 8.4.1.

We start with a baseline variant of our approach, which is trained only on videos labeled for tracking and achieves 63.2 OxUvA AUC and 64.7 GOT AO. Next, we evaluate the importance of object priors by pretraining our model on COCO as a generic object detector. This variant, shown in row 2, results in an 1.7% improvement in OxUvA AUC and a 3.7% improvement in GOT AO. Next, we train our model for detection and tracking jointly. As expected, this multi-task training strategy provides a modest bump on both OxUvA and GOT, leading to a model that improves in tracking while additionally being able to perform single-image detection. These improvements confirm our intuition that object priors are critical for tracking, and that the universal nature of our model is indeed helpful in transferring information from object detection datasets.

As described in Sec. 8.3.4, our framework is flexible, admitting various strategies for computing a discriminative template, $\mathcal{T}_d$. We analyze a few strategies for computing this template. In particular we compare our proposed linear regression framework (denoted as $\mathcal{T}_d$ ='lin. reg.') to two simple baselines: a non-discriminative one, that simply averages several features vectors sampled from the ground truth bounding box (denoted with $\mathcal{T}_d$ = 'mean pos'), and a discriminative one that uses the difference between the means of positive and negative samples as a template (denoted with

Table 8.2: In the top half, we show the current reported state-of-the-art results on OxUvA, from [222]. As these methods perform poorly, we first run recent state-of-the-art trackers (DaSiam [274] and SiamMask [228]). We show that our approach significantly improves over both prior state-of-the-art, as well as these recent works.

Approach	TPR	TNR	GM
LCT [151]	22.7	43.2	31.3
MDNet [162]	42.1	0	32.4
TLD [114]	14.1	94.9	36.6
SiamFC + R [18]	35.4	43.8	39.7
DaSiam [274]	40.0	84.2	58.0
SiamMask	50.4	88.7	66.9
Ours w/o temporal	63.2	79.1	70.8
Ours	65.5	78.2	71.6

$\mathcal{T}_d$ ='mean diff'). Note that these can be seen as special cases of linear regression. First, we observe that all these variants increase the model's performance, but the linear regression approach results in the largest improvement of 5.3% OxUvA AUC and 0.9% GOT AO. Second, the 'mean diff' baseline actually shows the worst performance, which is counterintuitive. We attribute this result to the fact that simply subtracting the mean of the negative examples from the template leads to unstable behavior during training. In contrast, our principled approach to computing the template simplifies optimization. Finally, we show that incorporating the temporal smoothness (Section 8.4.2) provides significant improvements, particularly for short-term tracking as in GOT.

Discussion. Performing ablations on two diverse tracking datasets allows understanding the impact of ablated components for different challenges. For example, the use of detection priors seems to be significantly more pronounced in GOT, which requires tracking diverse objects, than for OxUvA. This is to be expected: while video datasets are large enough to learn priors for common objects, image-based datasets like COCO provide priors for more diverse categories. Meanwhile, our discriminative templates provide a significant improvement on OxUvA, but a more modest improvement on GOT. We attribute this to the fact that our discriminative template is able to avoid latching onto distractors when the object of interest disappears, a phenomenon that is far more common in the long-term OxUvA dataset than on GOT.

Table 8.3: We present results on the val set of GOT-10k. Prior methods on GOT-10k train only on the GOT-10k training set. For a fair comparison, we compare to DaSiam and SiamMask, which are trained on external data. By leveraging objectness priors and detection mechanisms, our method significantly improves, likely due to the diversity of objects in GOT.

Approach	AO	SR
DaSiam [274]	46.0	54.3
SiamMask [228]	66.8	78.3
Ours w/o temporal	69.5	79.1
Ours	**73.0**	**82.8**

8.4.4 Comparison to the state-of-the-art

We now compare our full approach to state-of-the art methods in object tracking and video object segmentation.

OxUvA evaluation We begin by presenting comparisons on the *dev* set of the OxUvA long term tracking benchmark in Table 8.2. We compare to the state-of-the-art approaches reported in [222]. As the approaches reported in [222] perform poorly qualitatively and quantitatively, we further evaluate two more recent trackers: DaSiam [274] and SiamMask [228] on this dataset. We use their publicly available code.

As shown in Table 8.2, our evaluation of DaSiam [274] and SiamMask [228] outperform the methods reported in [222]. Next, we evaluate our approach without using *any* temporal information in the "w/o temporal" row. This variant is completely stateless, and individually performs matching on each frame of the video. By contrast, almost all prior tracking approaches use heuristic temporal smoothing to improve the performance of their models. Despite this lack of temporal information, our approach strongly outperforms all prior work, including the recent work of [228, 274], by 3.9% in GM. By adding the simple temporal heuristic described in Sec. 8.4.2, we further improve our results by 0.8%. We report results on the held out *test* set in supplementary.

GOT evaluation To validate our conclusions above, we further evaluate our approach on GOT-10k. As prior methods evaluated on GOT-10k use only the GOT-

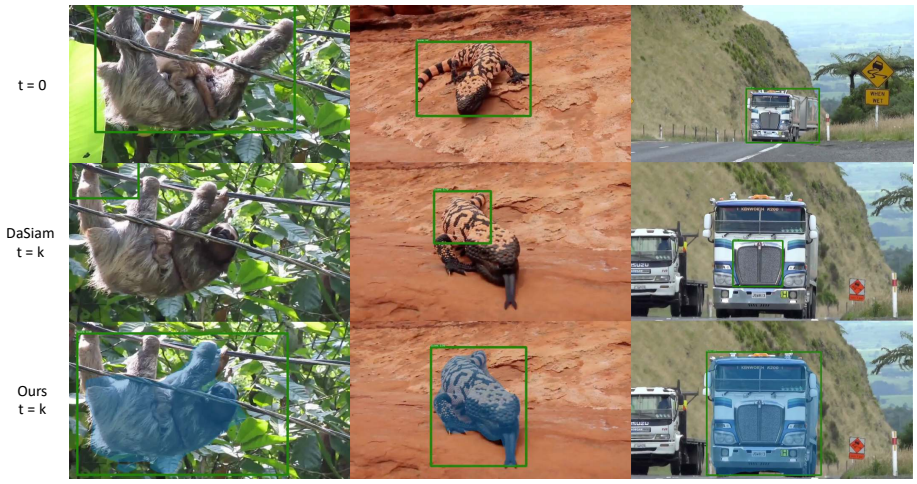


Figure 8.4: We compare our method to DaSiam [274]. While DaSiam struggles to accurately localize objects, leveraging objectness priors allows us to detect, track, and segment the full extent of objects.

10k training set for training, we can not fairly compare our approach to them. Instead, we use [228, 274] as baselines, which are the best method prior to ours on OxUvA. We report results on the validation set in Table 8.3, and additionally show results on the test set in the supplementary material. As can be see from the table, we outperform both of these works by over 4%, which we attribute to the ability of our method to generalize to diverse object categories.

LTB-35 Evaluation We compare to state-of-the-art methods on the LTB-35 benchmark, which was used to evaluate long term tracking in the VOT 2018-LT challenge. This dataset focuses on tracking in videos over 2 minutes long on average, where the object of interest can frequently disappear and reappear in the video. We compare to state-of-the-art results in Table 8.4, as well as SiamMask[228]. Note that prior approaches, other than [228], provide dataset-specific hyperparameters that are tuned for this dataset. By contrast, we use a single model, a single set of hyperparameters, and a simple temporal heuristic across all datasets. Despite this, our method obtains a competitive F-measure of 61.2 on this dataset while outperforming on other datasets.

Mask evaluation on DAVIS'17 Finally, we evaluate our unified approach on the task of video object segmentation. To this end we use the validation set of

Table 8.4: F-measure on LTB35 (VOT 2018-LT challenge). While prior methods use dataset-specific hyperparameters, we present a *single* model for all experiments that is competitive on VOT while outperforming on other datasets.

Approach	P	R	F
SiamMask [228]	64.8	38.5	48.3
DaSiam-LT [274]	62.7	58.8	60.7
MBMD [265]	63.4	58.9	61.0
SiamRPN++ [138]	65.0	61.0	62.9
Ours w/o temporal	61.0	56.9	58.9
Ours	61.2	61.2	61.2

Table 8.5: DAVIS '17 validation results with intersection-over-union ($\mathcal{J}$) and F-measure ($\mathcal{F}$). Most prior methods require a labeled mask in the first frame ('Mask sup') or perform computationally expensive end-to-end fine-tuning per video ('Deep FT' row), our method efficiently and accurately segments objects without mask supervision.

	Measure	PReMVOS [150]	CINM [11]	FeelVOS [224]	SiamMask [228]	Ours
	Mask sup?	✓	✓	✓	✗	✗
	Deep FT?	✓	✓	✗	✗	✗
$\mathcal{J}$	Mean	73.9	67.2	69.1	54.3	59.2
	Recall	73.1	74.5	79.1	62.8	68.6
	Decay	16.2	24.6	17.5	19.3	8.4
$\mathcal{F}$	Mean	81.8	74.0	74.0	58.5	67.8
	Recall	88.9	81.6	83.8	67.5	76.1
	Decay	19.5	26.2	20.1	20.9	12.0

DAVIS'17, and compare to the state-of-the-art approaches, including the ones that require finetuning the model on the test sequences. The results are presented in Table 8.5. We show qualitative results of our method in the supplementary material.

All methods in Table 8.5, with the exception of SiamMask and our method, require pixel-perfect segmentation in the first video frame and operate at a speed of less than 2 frames-per-second. By contrast, our method adds only a small overhead to the underlying detection model used. For our experiments, we used a ResNet-50 FPN backbone for Mask R-CNN, which led to a speed of approximately 7FPS. Despite using less supervision and computational time, our approach is competitive with dedicated video segmentation methods that use pixel-level masks in the first frame.

Detection evaluation on COCO As discussed in Section 8.3.3, our model can be used as a detector at test time. Although our focus is on tracking, we find that our model outputs high quality detections, providing a COCO instance segmentation mAP of 30.5, compared to 34.4 for an equivalent standalone detector that cannot track objects.

Qualitative results. We show qualitative results in Fig. 8.4, comparing our results with DaSiamRPN [274]. Note that [274] struggles to localize objects as they undergo scale and appearance changes, such as the truck in the third column. By contrast, leveraging objectness information allows our approach to localize the full extent of objects, while simultaneously providing *instance segmentations* for the object of interest.

8.5 Conclusion

This paper introduces a novel generic object tracking approach built on top of a state-of-the-art object-detection framework. The resulting model can be trained jointly for the two tasks, effectively incorporating objectness priors into tracking. Additionally, we propose learning discriminative templates in a fully differentiable manner that encode information both about the object of interest and about the distractors, increasing the tracker's robustness. Finally, we extend our method to the related task of video object segmentation by simply adding a mask prediction branch.

Our resulting framework for tracking and video segmentation demonstrates state-of-the-art results on two recent tracking datasets (OxUvA and GOT10k), and also shows competitive performance on the DAVIS'17 benchmark for video object segmentation. We empirically show that these improvements are largely due to the generic objectness prior learned from the COCO dataset.

Chapter 9

Discussion

The past decade of progress has significantly advanced computer vision models in the closed world of benchmark datasets, which usually contain only a few classes. In this thesis, we explored how to extend vision approaches and benchmarks to the open world, which can contain arbitrary scenes and objects. In Part I, we evaluated and addressed the robustness of models to changes in object and scene appearances, including partial and full occlusions. Part II presents datasets, evaluations, and approaches for scaling detectors and trackers to large (albeit finite) vocabularies of classes. Finally, Part III introduces approaches for detecting and tracking *arbitrary* objects, without any fixed vocabulary.

Approaches. In the long run, building accurate methods for open-world recognition will require going beyond supervised, image datasets. The interaction between different modalities, such as video, text, audio, and depth, in unlabeled data will improve the generalization ability of our methods. In the community, recent models trained on weakly supervised, multi-modal data have shown surprisingly useful properties, as in Radford et al. [182]: they can generalize to rare classes, and appear to be more robust on the robustness benchmarks introduced in our work (Chapter 3, Taori et al. [211]). Analyzing whether these models generalize to entirely unseen objects, reducing their dependency on billions of images, and incorporating additional modalities will be an important pathway for open-world recognition moving forward.

Tasks. A running challenge for open-world recognition is developing appropriate intermediate tasks and evaluations. Ideally, one could evaluate methods in an end

application: can an autonomous vehicle navigate around unknown objects, or can a video understanding system recognize actions involving unknown objects? This may suffice for certain applications, but progress in computer vision has excelled when we define generic, atomic tasks that are useful for many downstream applications. What might such tasks look like? In this thesis, we focused on some concrete tasks, which include scaling up to large object vocabularies, and building methods to segment and track novel objects. The former extends methods to a large (albeit fixed) set of class names, while the latter provides segmentation masks around any object of interest. Are there alternatives to these task definitions that can provide richer information about any object or image region? For example, rather than naming pixels with object classes, perhaps we should re-visit *attribute*-based taxonomies [64]. Attributes can describe pixels (or objects) with traits that generalize across object classes, such as 'movable,' 'living,' or 'heavy.' Further, current tasks focus primarily on objects, largely ignoring the hierarchical and compositional nature of visual recognition. Current segmentation methods (including ours in Chapter 7), for example, can accurately recognize the pixels on a car, but cannot provide finer segmentations of the objects' parts (such as the door handle, the car doors, or wheels), unless the parts are explicitly enumerated in the vocabulary. Can we instead design tasks that encourage methods to learn a hierarchical grouping of pixels in images and videos? Such a mid-level task, as explored in [7, 156], could generalize to more diverse applications, which require going beyond object-level understanding.

Bibliography

[1] Vitaly Ablavsky and Stan Sclaroff. Layered graphical models for tracking partially occluded objects. *TPAMI*, 33(9):1758–1775, 2011. 36

[2] Ankur Agarwal and Bill Triggs. Tracking articulated motion with piecewise learned dynamical models. In *ECCV*, volume 3, 2004. 10

[3] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *CVPR*, 2016. 36, 59

[4] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *CVPR*. IEEE, 2010. 9, 96

[5] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2189–2202, 2012. 116

[6] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. People-tracking-by-detection and people-detection-by-tracking. In *CVPR*, 2008. 10, 10, 116

[7] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 33(5):898–916, May 2011. 134

[8] Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv preprint arXiv:1805.12177*, 2018. 17

[9] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *CVPR*, 2009. 118

[10] Renée Baillargeon and Julie DeVos. Object permanence in young infants: Further evidence. *Child development*, 62(6):1227–1246, 1991. 34

[11] Linchao Bao, Baoyuan Wu, and Wei Liu. Cnn in mrf: Video object segmentation via inference in a cnn-based higher-order spatio-temporal mrf. In *CVPR*, 2018. 131

[12] Adela Barriuso and Antonio Torralba. Notes on image annotation. *arXiv*

preprint arXiv:1210.3448, 2012. 63

[13] Abhijit Bendale and T. Boult. Towards open world recognition. *CVPR*, 2015. 9

[14] Jerome Berclaz, Francois Fleuret, and Pascal Fua. Robust people tracking with global trajectory optimization. In *CVPR*, 2006. 59

[15] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011. 36, 59

[16] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019. 36, 36, 41, 46, 47, 47, 47, 54, 59, 59, 66, 66, 69, 69, 69, 69, 116

[17] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *Journal on Image and Video Processing*, 2008:1, 2008. 61

[18] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional Siamese networks for object tracking. In *ECCV*, 2016. 10, 11, 60, 116, 118, 128

[19] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 36, 39, 66, 67, 68, 68, 68, 69, 102

[20] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *CVPR*, 2019. 60, 60, 66, 70

[21] Pia Bideau and Erik Learned-Miller. A detailed rubric for motion segmentation. *arXiv preprint arXiv:1610.10033*, 2016. 103

[22] Pia Bideau and Erik Learned-Miller. It's moving! a probabilistic model for causal motion segmentation in moving camera videos. In *ECCV*, 2016. 8, 98, 98

[23] Pia Bideau, Aruni RoyChowdhury, Rakesh R Menon, and Erik Learned-Miller. The best of both worlds: Combining CNNs and geometric constraints for hierarchical motion segmentation. In *CVPR*, 2018. 8, 98, 110, 112, 112, 112, 112, 112

[24] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 2018. https://arxiv.org/abs/1712.03141. 3, 15, 17

[25] NTS Board. Collision between vehicle controlled by developmental automated driving system and pedestrian. *Nat. Transpot. Saf. Board, Washington, DC, USA, Tech. Rep. HAR-19-03*, 2019. 15

[26] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 116, 118

[27] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *ICCV*, 2009. 59

[28] Ted J Broida, S Chandrashekhar, and Rama Chellappa. Recursive 3-d motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, 1990. 35

[29] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "Siamese" time delay neural network. In *NIPS*, 1994. 60, 118

[30] Thomas Brox and Jitendra Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010. 8, 97

[31] Remi Cadene. Pretrained models for pytorch. `https://github.com/Cadene/pretrained-models.pytorch`. Accessed: 2019-05-20. 23

[32] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *CVPR*, 2017. 58, 119

[33] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 DAVIS challenge on VOS: Unsupervised multi-object segmentation. *arXiv preprint arXiv:1905.00737*, 2019. 59

[34] Joao Carreira and Cristian Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010. 9

[35] Michael Chan, Dimitri Metaxas, and Sven Dickinson. Physics-based tracking of 3d objects in 2d image sequences. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 432–436. IEEE, 1994. 36

[36] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *CVPR*, 2019. 33, 34, 55, 62

[37] Boyu Chen, Dong Wang, Peixia Li, Shuang Wang, and Huchuan Lu. Real-time 'Actor-Critic' tracking. In *ECCV*, 2018. 59

[38] Jingchun Cheng, Yi-Hsuan Tsai, Wei-Chih Hung, Shengjin Wang, and Ming-Hsuan Yang. Fast and accurate online video object segmentation via tracking parts. In *CVPR*, 2018. 119

[39] Wongun Choi and Silvio Savarese. Multiple target tracking in world coordinate with single, minimally calibrated camera. In *ECCV*, 2010. 59

[40] Javier Civera, Andrew J Davison, and JM Martinez Montiel. Inverse depth parametrization for monocular slam. *IEEE transactions on robotics*, 24(5):

932–945, 2008. 35

[41] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NeurIPS*, pages 379–387, 2016. 28

[42] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. https://ieeexplore.ieee.org/document/1467360. 10

[43] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017. 60, 60, 66, 70, 118

[44] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, 2019. 60, 60, 66, 70

[45] Achal Dave, Pavel Tokmakov, and Deva Ramanan. Towards segmenting everything that moves. In *ICCV Workshop on Holistic Video Understanding*, 2019. 2

[46] Achal Dave, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. Learning to track any object. In *ICCV Workshop on Holistic Video Understanding*, 2019. 2

[47] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. TAO: A large-scale benchmark for tracking any object. In *ECCV*, 2020. 2, 56, 58, 58, 61, 62, 64, 65, 67, 68, 69, 69, 70, 71

[48] Achal Dave, Piotr Dollár, Deva Ramanan, Alexander Kirillov, and Ross Girshick. Evaluating large-vocabulary object detectors: The devil is in the details. *arXiv preprint arXiv:2102.01066*, 2021. 2

[49] Thomas Dean, Mark A Ruzon, Mark Segal, Jonathon Shlens, Sudheendra Vijayanarasimhan, and Jay Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013. 76

[50] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020. 35, 42, 46

[51] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 55

[52] Chaitanya Desai, Deva Ramanan, and Charless C Fowlkes. Discriminative models for multi-class object layout. *IJCV*, 95(1):1–12, 2011. 76, 88

[53] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox.

Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 99

[54] Kiana Ehsani, Roozbeh Mottaghi, and Ali Farhadi. Segan: Segmenting and generating the invisible. In *CVPR*, 2018. 35

[55] Ian Endres and Derek Hoiem. Category independent object proposals. In *ECCV*, 2010. 9

[56] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017. 3, 15, 17, 17

[57] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014. 9

[58] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*, 2008. 59

[59] Andreas Ess, Konrad Schindler, Bastian Leibe, and Luc Van Gool. Improved multi-person tracking with active occlusion handling. In *ICRA Workshop on People Detection and Tracking*, 2009. 35, 40

[60] Georgios D Evangelidis and Emmanouil Z Psarakis. Parametric image alignment using enhanced correlation coefficient maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1858–1865, 2008. 41

[61] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 2010. 34, 43, 73, 76, 76

[62] Alon Faktor and Michal Irani. Video segmentation by non-local consensus voting. In *BMVC*, 2014. 98

[63] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. LaSOT: A high-quality benchmark for large-scale single object tracking. In *CVPR*, 2019. 55, 58, 62

[64] Ali Farhadi, Ian Endres, Derek Hoiem, and D. Forsyth. Describing objects by their attributes. *CVPR*, pages 1778–1785, 2009. 9, 134

[65] Alhussein Fawzi and Pascal Frossard. Manitest: Are classifiers really invariant? In *BMVC*, 2015. https://arxiv.org/abs/1507.06535. 3, 15, 17

[66] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, 2017. 18, 59, 59, 66, 66, 66, 66, 68, 68, 68, 69

[67] Pedro F. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61:55–79, 2004. 10

[68] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE trans-*

actions on pattern analysis and machine intelligence, 32(9), 2009. 10, 46, 119

[69] M. Fischler and Robert A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22:67–92, 1973. 10

[70] Robert Fisher, J Santos-Victor, and J Crowley. Context aware vision using image-based active recognition. *EC's Information Society Technology's Programme Project IST2001-3754*, 2001. 57

[71] Francois Fleuret, Jerome Berclaz, Richard Lengagne, and Pascal Fua. Multicamera people tracking with a probabilistic occupancy map. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):267–282, 2007. 36

[72] Katerina Fragkiadaki, Pablo Arbelaez, Panna Felsen, and Jitendra Malik. Learning to segment moving objects in videos. In *CVPR*, 2015. 8, 8, 8, 97, 99, 102

[73] Shan Gao, Zhenjun Han, Ce Li, Qixiang Ye, and Jianbin Jiao. Real-time multipedestrian tracking in traffic scenes via an rgb-d-based layered graph model. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2814–2825, 2015. 36

[74] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012. 54, 55, 57, 61

[75] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 35

[76] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *NeurIPS*, pages 7538–7550, 2018. 17

[77] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. ImageNet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019. 17

[78] Ross Girshick. Fast R-CNN. In *ICCV*, 2015. 119

[79] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 119

[80] Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *CVPR*, 2015. 59, 66, 68, 68, 68, 69, 99, 102

[81] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 3, 15, 17

[82] Helmut Grabner, Jiri Matas, Luc Van Gool, and Philippe Cattin. Tracking the invisible: Learning where the object might be. In *CVPR*, 2010. 36

[83] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010. 8, 97

[84] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. AVA: A video dataset of spatio-temporally localized atomic visual actions. In *CVPR*, 2018. 55, 55, 62, 62, 99, 102

[85] Keren Gu, Brandon Yang, Jiquan Ngiam, Quoc Le, and Jonathon Shlens. Using videos to evaluate image model robustness. In *ICML Workshop on SafeML*, 2019. https://arxiv.org/abs/1904.10076. 16, 17, 17, 20, 26

[86] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017. 77

[87] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *CVPR*, 2018. 50

[88] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 9, 53, 55, 56, 56, 56, 61, 61, 61, 61, 62, 64, 66, 69, 71, 74, 75, 76, 76, 76, 79, 83, 86

[89] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 83, 87, 101

[90] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 35, 49, 67, 79, 83, 105, 116, 117, 118, 119, 120, 120, 121, 124

[91] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 FPS with deep regression networks. In *ECCV*, 2016. 10, 11, 60, 118

[92] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019. https://arxiv.org/abs/1903.12261. 3, 15, 17, 26

[93] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *ECCV*, 2012. 76

[94] J. Hosang, Rodrigo Benenson, Piotr Dollár, and B. Schiele. What makes for effective detection proposals? *TPAMI*, 38:814–830, 2016. 9

[95] Hossein Hosseini and Radha Poovendran. Semantic adversarial examples. In *CVPR Workshop*, 2018. https://arxiv.org/abs/1804.00499. 3, 15, 17

[96] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. In *CVPR*, 2018. 9, 76, 98

[97] Yang Hua, Karteek Alahari, and Cordelia Schmid. Online object tracking with proposal selection. In *ICCV*, 2015. 118, 118

[98] Lianghua Huang, Xin Zhao, and Kaiqi Huang. GOT-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*, 2018. 58, 117, 124, 125

[99] Piao Huang, Shoudong Han, Jun Zhao, Donghaisheng Liu, Hongwei Wang, En Yu, and Alex ChiChung Kot. Refinements in motion and appearance for online multi-object tracking. *arXiv preprint arXiv:2003.07177*, 2020. 46, 47, 47

[100] Yan Huang and Irfan Essa. Tracking multiple objects through occlusions. In *CVPR*, 2005. 34

[101] Yingfan Huang, HuiKun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *ICCV*, 2019. 50

[102] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 2018. 106

[103] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 2015. 83

[104] Michael Isard and John MacCormick. Bramble: A bayesian multiple-blob tracker. In *ICCV*, 2001. 36

[105] Omid Hosseini Jafari, Dennis Mitzel, and Bastian Leibe. Real-time rgb-d based people detection and tracking for mobile robots and head-worn cameras. In *ICRA*, 2014. 36

[106] Ramesh Jain, WN Martin, and JK Aggarwal. Segmentation through the detection of changes due to motion. *Computer Graphics and Image Processing*, 11(1):13–34, 1979. 7

[107] Suyog Dutt Jain and Kristen Grauman. Click carving: Segmenting objects in video with point clicks. In *Fourth AAAI Conference on Human Computation and Crowdsourcing*, 2016. 116

[108] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmention of generic objects in videos. In *CVPR*, 2017. 98

[109] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Pixel objectness. *arXiv preprint arXiv:1701.05349*, 2017. 99

[110] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision, 2021. 27

[111] Hao Jiang, Sidney Fels, and James J Little. A linear programming approach

for multiple object tracking. In *CVPR*, 2007. 59

[112] SouYoung Jin, Aruni RoyChowdhury, Huaizu Jiang, Ashish Singh, Aditya Prasad, Deep Chakraborty, and Erik Learned-Miller. Unsupervised hard example mining from videos for improved object detection. In *ECCV*, 2018. 18

[113] K J Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *CVPR*, 2021. 9

[114] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1409–1422, 2012. 118, 128

[115] Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: analysis and improvement. *arXiv preprint arXiv:1711.09115*, 2017. 17

[116] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *ICLR*, 2020. 76, 84, 86

[117] Kai Kang, Hongsheng Li, Tong Xiao, Wanli Ouyang, Junjie Yan, Xihui Liu, and Xiaogang Wang. Object detection in videos with tubelet proposal networks. In *CVPR*, 2017. 18, 59

[118] Shyamgopal Karthik, Ameya Prabhu, and Vineet Gandhi. Simple unsupervised multi-object tracking. *arXiv preprint arXiv:2006.02609*, 2020. 47, 47

[119] Margret Keuper, Bjoern Andres, and Thomas Brox. Motion trajectory segmentation via minimum cost multicuts. In *ICCV*, 2015. 8, 95, 96, 97, 97, 110, 111, 112, 112, 112, 112, 112, 112, 112, 113, 113, 113, 113, 113

[120] Naeemullah Khan, Byung-Woo Hong, Anthony Yezzi, and Ganesh Sundaramoorthi. Coarse-to-fine segmentation with shape-tailored continuum scale spaces. In *CVPR*, 2017. 111

[121] Saad M Khan and Mubarak Shah. A multiview approach to tracking people in crowded scenes using a planar homography constraint. In *ECCV*, 2006. 36

[122] Anna Khoreva, Rodrigo Benenson, Eddy Ilg, Thomas Brox, and Bernt Schiele. Lucid data dreaming for video object segmentation. *International Journal of Computer Vision*, 127(9):1175–1197, 2019. 58

[123] Tarasha Khurana, Achal Dave, and Deva Ramanan. Detecting invisible people. *arXiv preprint arXiv:2012.08419*, 2020. 2

[124] Kyungnam Kim and Larry S Davis. Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In *ECCV*, 2006. 36

[125] Kris M Kitani, Brian D Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *ECCV*. Springer, 2012. 36

[126] Matej Kristan, Jiri Matas, Aleš Leonardis, Tomáš Vojíř, Roman Pflugfelder, Gustavo Fernandez, Georg Nebehay, Fatih Porikli, and Luka Čehovin. A novel performance evaluation methodology for single-target trackers. *TPAMI*, 38(11): 2137–2155, 2016. 58, 116

[127] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pfugfelder, Luka Cehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, Gustavo Fernandez, and et al. The sixth visual object tracking vot2018 challenge results, 2018. 117, 125

[128] Meelis Kull, Telmo Silva Filho, and Peter Flach. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Artificial Intelligence and Statistics*, 2017. 77, 90

[129] Weicheng Kuo, Bharath Hariharan, and Jitendra Malik. Deepbox: Learning objectness with convolutional networks. *ICCV*, 2015. 9

[130] Fabian Kuppers, Jan Kronenberger, Amirhossein Shantia, and Anselm Haselhoff. Multivariate confidence calibration for object detection. In *CVPR Workshops*, 2020. 77, 90, 90, 90

[131] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018. 76, 76

[132] Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv preprint arXiv:1907.01341*, 2019. 38

[133] Laura Leal-Taixé, Gerard Pons-Moll, and Bodo Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *ICCV Workshops*, 2011. 36, 45

[134] Laura Leal-Taixé, Michele Fenzi, Alina Kuznetsova, Bodo Rosenhahn, and Silvio Savarese. Learning an image-based motion context for multiple people tracking. In *CVPR*, 2014. 59

[135] Yong Jae Lee, Jaechul Kim, and Kristen Grauman. Key-segments for video object segmentation. In *ICCV*. IEEE, 2011. 98, 98

[136] B. Leibe, Nico Cornelis, K. Cornelis, and L. Gool. Dynamic 3d scene analysis from a moving vehicle. *CVPR*, 2007. 10

[137] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance

visual tracking with Siamese region proposal network. In *CVPR*, 2018. 60, 118

[138] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 60, 66, 70, 70, 119, 131

[139] Ke Li and Jitendra Malik. Amodal instance segmentation. In *ECCV*. Springer, 2016. 35

[140] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *CVPR*, 2020. 76, 80, 84, 84, 84, 85, 86, 89

[141] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 38, 38, 44

[142] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 9, 21, 34, 43, 53, 54, 55, 66, 73, 74, 76, 76, 98, 100, 100, 105, 106, 107, 124, 125

[143] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 83, 120, 121

[144] Qiankun Liu, Qi Chu, Bin Liu, and Nenghai Yu. Gsm: Graph similarity model for multi-object tracking. International Joint Conferences on Artificial Intelligence Organization, 2020. 47

[145] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 120

[146] Yang Liu, Idil Esen Zulfikar, Jonathon Luiten, Achal Dave, Aljosa Osep, D. Ramanan, Bastian Leibe, and L. Leal-Taixé. Opening up open-world tracking. *ArXiv*, abs/2104.11221, 2021. 9

[147] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 76, 76, 86

[148] Jakub Lokoč, Gregor Kovalčík, Tomás Souček, Jaroslav Moravec, and Premysl Čech. A framework for effective known-item search in video. In *ACMM*, 2019. doi: 10.1145/3343031.3351046. URL https://doi.org/10.1145/3343031.3351046. 62

[149] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981. 10

[150] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-

generation, refinement and merging for video object segmentation. In *ACCV*, 2018. 131

[151] Chao Ma, Xiaokang Yang, Chongyang Zhang, and Ming-Hsuan Yang. Long-term correlation tracking. In *CVPR*, 2015. 128

[152] Wei-Chiu Ma, De-An Huang, Namhoon Lee, and Kris M Kitani. Forecasting interactive dynamics of pedestrians with fictitious play. In *CVPR*, 2017. 36, 45

[153] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018. https://arxiv.org/abs/1706.06083. 26

[154] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *Introduction to information retrieval*. Cambridge university press, 2008. 75, 76

[155] David Marr. Vision: A computational investigation into the human representation and processing of visual information. mit press. *Cambridge, Massachusetts*, 1982. 96

[156] 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 *ICCV*, 2001. 134

[157] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. URL http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16. 100, 100, 105

[158] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 34, 35, 42, 46, 53, 54, 54, 55, 56, 57, 59, 61, 62

[159] Anton Milan, S Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 59

[160] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. 19

[161] Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *AAAI*, 2015. 77, 90

[162] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 128

[163] Manjunath Narayana, Allen Hanson, and Erik Learned-Miller. Coherent motion segmentation in moving camera videos using optical flow orientations. In *ICCV*, 2013. 8, 98, 98

[164] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities

with supervised learning. In *ICML*, 2005. 77

[165] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *TPAMI*, 36(6):1187–1200, 2014. 8, 59, 96, 97, 103, 103, 103, 104, 105, 111

[166] Masatoshi Okutomi and Takeo Kanade. A multiple-baseline stereo. In *CVPR*, volume 93, pages 63–69, 1991. 38

[167] Stephen E Palmer. *Vision science: Photons to phenomenology*. MIT press, 1999. 96, 96

[168] Ioannis Papakis, Abhijit Sarkar, and Anuj Karpatne. Gcnnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. *arXiv preprint arXiv:2010.00067*, 2020. 47

[169] Anestis Papazoglou and Vittorio Ferrari. Fast object segmentation in unconstrained video. In *ICCV*, 2013. 98, 98

[170] Harold Pashler. Familiarity and visual change detection. *Perception & psychophysics*, 44(4):369–378, 1988. 20

[171] V. Pavlovic, James M. Rehg, Tat-Jen Cham, and K. Murphy. A dynamic bayesian network approach to figure tracking using learned dynamic models. *ICCV*, 1999. 10

[172] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*. IEEE, 2009. 36

[173] Xiaojiang Peng and Cordelia Schmid. Multi-region two-stream R-CNN for action detection. In *ECCV*. Springer, 2016. 99, 102

[174] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 58, 59, 98, 100, 105, 106

[175] Federico Perazzi, Anna Khoreva, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images. In *CVPR*, 2017. 119

[176] Trung Pham, Vijay B. G. Kumar, Thanh-Toan Do, Gustavo Carneiro, and Ian Reid. Bayesian semantic instance segmentation in open set world. In *ECCV*, 2018. 9

[177] Pedro H. O. Pinheiro, Ronan Collobert, and Piotr Dollár. Learning to segment object candidates. In *NIPS*, 2015. 9

[178] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011.

36, 59

[179] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, pages 61–74, 1999. 76, 90

[180] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. *arXiv:1704.00675*, 2017. 97, 117, 125

[181] Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with KINS dataset. In *CVPR*, 2019. 35

[182] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 4, 17, 27, 31, 133

[183] D. Ramanan, D. Forsyth, and Andrew Zisserman. Tracking people by learning their appearance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:65–81, 2007. 10

[184] K. Rangarajan and M. Shah. Establishing motion correspondence. *CVGIP Image Underst.*, 54:56–73, 1991. 10

[185] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *CVPR*, 2017. 16, 19

[186] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *ICML*, 2019. https://arxiv.org/abs/1902.10811. 18, 31

[187] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017. 9

[188] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 86, 120

[189] Liangliang Ren, Jiwen Lu, Zifeng Wang, Qi Tian, and Jie Zhou. Collaborative deep reinforcement learning for multi-object tracking. In *ECCV*, 2018. 59

[190] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 9, 27, 28, 44, 45, 46, 60, 66, 119, 120, 120

[191] Ergys Ristani and Carlo Tomasi. Features for multi-target multi-camera tracking and re-identification. In *CVPR*, 2018. 59

[192] John W Roach and JK Aggarwal. Determining the movement of objects from a sequence of images. *IEEE Transactions on Pattern Analysis and Machine*

Intelligence, (6):554–562, 1980. 35

[193] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *ECCV*. Springer, 2016. 36

[194] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 16, 19, 34, 55, 57, 59, 61, 83, 105, 124

[195] Bryan C Russell, William T Freeman, Alexei A Efros, Josef Sivic, and Andrew Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006. 8

[196] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3):157–173, 2008. 55

[197] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *IEEE TPAMI*, 36(11):2317–2324, 2014. 96

[198] Paul Scovanner and Marshall F Tappen. Learning pedestrian dynamics from the real world. In *ICCV*, 2009. 36, 45, 59

[199] I. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *TPAMI*, PAMI-9:56–73, 1987. 10

[200] Xindi Shang, Donglin Di, Junbin Xiao, Yu Cao, Xun Yang, and Tat-Seng Chua. Annotating objects and relations in user-generated videos. In *ICMR*, 2019. 58

[201] Vaishaal Shankar, Achal Dave, Rebecca Roelofs, Deva Ramanan, Benjamin Recht, and Ludwig Schmidt. Do image classifiers generalize across time?, 2019. https://arxiv.org/abs/1906.02168. 1

[202] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *ICCV*, 1998. 8, 97

[203] Jianbo Shi and Carlo Tomasi. Good features to track. *CVPR*, pages 593–600, 1994. 10

[204] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016. 55, 62

[205] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013. 96

[206] Davide Spinello and Daniel J Stilwell. Nonlinear estimation with state-dependent gaussian observation noise. *IEEE Transactions on Automatic Control*, 55(6): 1358–1366, 2010. 35

[207] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1999. 7

[208] James S Supancic and Deva Ramanan. Self-paced learning for long-term tracking. In *CVPR*, 2013. 118

[209] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, 2020. 80, 84, 84, 85, 89

[210] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *CVPR*, 2016. 60, 118

[211] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *NeurIPS*, 2020. 2, 4, 26, 133

[212] Brian Taylor, Vasiliy Karasev, and Stefano Soatto. Causal video object segmentation from persistence of occlusions. In *CVPR*, 2015. 111, 112

[213] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *arXiv preprint arXiv:1503.01817*, 2015. 55, 62

[214] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning motion patterns in videos. In *CVPR*, 2017. 98, 100, 100

[215] Pavel Tokmakov, Cordelia Schmid, and Karteek Alahari. Learning to segment moving objects. *IJCV*, Sep 2018. ISSN 1573-1405. doi: 10.1007/s11263-018-1122-2. URL https://doi.org/10.1007/s11263-018-1122-2. 98

[216] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical report, Technical Report CMU-CS-91-132, Carnegie, Mellon University, 1991. 10

[217] Antonio Torralba, Alexei A Efros, et al. Unbiased look at dataset bias. In *CVPR*, 2011. https://ieeexplore.ieee.org/document/5995347. 18

[218] Yi-Hsuan Tsai, Ming-Hsuan Yang, and Michael J Black. Video segmentation via object flow. In *CVPR*, 2016. 119

[219] Roy Tseng. Detectron.pytorch. https://github.com/roytseng-tw/Detectron.pytorch. 105

[220] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008. 35

[221] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking.

In *CVPR*, 2017. 118

[222] Jack Valmadre, Luca Bertinetto, Joao F Henriques, Ran Tao, Andrea Vedaldi, Arnold WM Smeulders, Philip HS Torr, and Efstratios Gavves. Long-term tracking in the wild: A benchmark. In *ECCV*, 2018. 55, 58, 61, 70, 117, 125, 128, 129, 129, 129

[223] Paul Voigtlaender and Bastian Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017. 58, 119

[224] Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. Feelvos: Fast end-to-end embedding learning for video object segmentation. In *CVPR*, 2019. 119, 131

[225] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. MOTS: Multi-object tracking and segmentation. In *CPVR*, 2019. 53, 54, 58

[226] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 86

[227] Jiaqi Wang, Wenwei Zhang, Yuhang Zang, Yuhang Cao, Jiangmiao Pang, Tao Gong, Kai Chen, Ziwei Liu, Chen Change Loy, and Dahua Lin. Seesaw loss for long-tailed instance segmentation. *arXiv preprint arXiv:2008.10032*, 2020. 76, 84, 86, 86

[228] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, pages 1328–1338, 2019. 60, 60, 66, 70, 119, 119, 128, 129, 129, 129, 129, 130, 130, 130, 131, 131

[229] Weiyao Wang, Matt Feiszli, Heng Wang, and Du Tran. Unidentified video objects: A benchmark for dense, open-world segmentation. *ArXiv*, abs/2104.04691, 2021. 9

[230] Wenguan Wang, Jianbing Shen, and Fatih Porikli. Saliency-aware geodesic video object segmentation. In *CVPR*, 2015. 98, 98

[231] Wenguan Wang, Hongmei Song, Shuyang Zhao, Jianbing Shen, Sanyuan Zhao, Steven CH Hoi, and Haibin Ling. Learning unsupervised video object segmentation through visual attention. In *CVPR*, 2019. 59

[232] Xueyang Wang, Xiya Zhang, Yinheng Zhu, Yuchen Guo, Xiaoyun Yuan, Liuyu Xiang, Zerun Wang, Guiguang Ding, David Brady, Qionghai Dai, et al. Panda: A gigapixel-level human-centric video dataset. In *CVPR*, 2020. 35, 42, 46

[233] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging

the gap in 3d object detection for autonomous driving. In *CVPR*, 2019. 35

[234] Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Learning to track for spatio-temporal action localization. In *ICCV*, 2015. 116

[235] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv preprint arXiv:1511.04136*, 2015. 54, 55, 57

[236] Longyin Wen, Dawei Du, Zhen Lei, Stan Z Li, and Ming-Hsuan Yang. Jots: Joint online tracking and segmentation. In *CVPR*, 2015. 119

[237] Max Wertheimer. Untersuchungen zur lehre von der gestalt. ii. *Psychologische forschung*, 4(1):301–350, 1923. 96

[238] Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *WACV*. IEEE, 2018. doi: 10.1109/WACV.2018.00087. 36, 36, 39

[239] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017. 37, 37, 37, 40, 47, 116

[240] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):780–785, 1997. 7

[241] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 58

[242] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 116

[243] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 66

[244] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *ICCV*, 2015. 116

[245] Fanyi Xiao and Yong Jae Lee. Video object detection with an aligned spatial-temporal memory. In *ECCV*, 2018. 18, 25, 28, 28, 59, 66, 66, 66, 66, 66

[246] Christopher Xie, Yu Xiang, Dieter Fox, and Zaid Harchaoui. Object discovery in videos as foreground motion clustering. 2019. 8, 98, 98

[247] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *arXiv preprint arXiv:1812.03411*, 2018. 26

[248] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Ag-

gregated residual transformations for deep neural networks. In *CVPR*, 2017. 87

[249] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. YouTube-VOS: A large-scale video object segmentation benchmark. In *ECCV*, 2018. 34, 58, 97, 100, 105, 125, 125

[250] Kota Yamaguchi, Alexander C Berg, Luis E Ortiz, and Tamara L Berg. Who are you with and where are you going? In *CVPR 2011*. IEEE, 2011. 36

[251] Xiaosheng Yan, Feigege Wang, Wenxi Liu, Yuanlong Yu, Shengfeng He, and Jia Pan. Visualizing the invisible: Occluded vehicle segmentation and recovery. In *ICCV*, 2019. 35

[252] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*, 2016. 46

[253] Linjie Yang, Yanran Wang, Xuehan Xiong, Jianchao Yang, and Aggelos K Katsaggelos. Efficient video object segmentation via network modulation. In *CVPR*, 2018. 119

[254] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 53, 55, 56, 57, 58, 61

[255] Yanchao Yang, Ganesh Sundaramoorthi, and Stefano Soatto. Self-occlusions and disocclusions in causal video object segmentation. In *ICCV*, 2015. 111

[256] A. Yilmaz, Omar A. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38:13, 2006. 10

[257] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, June 2020. 62

[258] Qian Yu, Gérard Medioni, and Isaac Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *ICCV*, 2007. 36

[259] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, 2001. 77, 90

[260] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *SIGKDD*, 2002. 77, 90

[261] Haichao Zhang and Jianyu Wang. Towards adversarially robust object detection. In *ICCV*, pages 421–430, 2019. 17

[262] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008. 59

[263] Yubo Zhang, Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. A

structured model for action detection. *CVPR*, 2019. 116

[264] Yubo Zhang, Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. A study on action detection in the wild. *arXiv preprint arXiv:1904.12993*, 2019. 76, 76, 86

[265] Yunhua Zhang, Dong Wang, Lijun Wang, Jinqing Qi, and Huchuan Lu. Learning regression and verification networks for long-term visual tracking. *arXiv preprint arXiv:1809.04320*, 2018. 131

[266] Ziheng Zhang, Anpei Chen, Ling Xie, Jingyi Yu, and Shenghua Gao. Learning semantics-aware distance map with semantics layering network for amodal instance segmentation. In *ACM Multimedia*, 2019. 35

[267] Hang Zhao, Antonio Torralba, Lorenzo Torresani, and Zhicheng Yan. HACS: Human action clips and segments dataset for recognition and temporal localization. In *ICCV*, 2019. 55, 62

[268] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20k dataset. In *CVPR*, 2017. 55, 61

[269] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *IJCV*, 2019. 76

[270] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *arXiv:2004.01177*, 2020. 46

[271] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Joint COCO and LVIS workshop at ECCV 2020: LVIS challenge track technical report: CenterNet2. 2020. URL https://www.lvisdataset.org/assets/challenge_reports/2020/CenterNet2.pdf. 76, 84, 84, 85, 89

[272] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. *ICCV*, 2017. 18, 66

[273] Yan Zhu, Yuandong Tian, Dimitris Metaxas, and Piotr Dollár. Semantic amodal segmentation. In *CVPR*, 2017. 35, 35

[274] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, 2018. 60, 116, 119, 128, 128, 129, 129, 129, 129, 130, 130, 131, 132, 132