Copyright
by
Bo Xiong
2019

The Dissertation Committee for Bo Xiong
certifies that this is the approved version of the following dissertation:

# Learning to Compose Photos and Videos from Passive Cameras

Committee:

Kristen Grauman, Supervisor

James Hays

Qixing Huang

Scott Niekum

# Learning to Compose Photos and Videos from Passive Cameras

**by**

**Bo Xiong**

**DISSERTATION**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**DOCTOR OF PHILOSOPHY**

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2019

Dedicated to my family.

# Acknowledgments

I am very fortunate and grateful to have had the opportunity to be a graduate student at the University of Texas at Austin. Over these years, I am extremely fortunate to have received the support and guidance from many incredible mentors, colleagues, friends and family.

First and foremost, I would like to thank my advisor Prof. Kristen Grauman. This thesis would not have been possible without her guidance and support. I am very fortunate to have had the opportunity to work with Kristen since first day in graduate school. Kristen is a remarkable scientist, an inspiring advisor and a reliable collaborator who have showed me how to perform impactful research, how to deliver effective presentations and how to always remain passionate and optimistic. She has been the best advisor that I could ever dream of.

I am also grateful to my other thesis committee members, Professors James Hays, Qixing Huang, and Scott Niekum for their insightful comments and constructive feedback that strengthened my thesis.

My labmates have made my graduate school experience more exciting, memorable and fun. I have also learned a lot from my labmates. I want to thank Adriana, Chao-Yeh, Suyog, Dinesh, Aron, Viktoriia, Yu-Chuan, Antonino, Ruohan, Ziad, Danna, Wei-Lin, Tushar and Santhosh. I will always remember our research discussion, ping pong games, lunch conversations and especially the company in the lab before deadlines.

Furthermore, I want to thank many other friends and fellow graduate students for good friendships: Jianyu Huang, Yinan Zhao, Xinyu Wang, Yuepeng Wang, Haoran Zhang, Xiaoxia Wu, and Chuanfeng Yang.

Last but not least, I would like to thank my parents for their constant support and unconditional love. My gratitude and appreciation are much more than what these few words can express. Mom and Dad, thank you for encouraging me to pursue a career I like and for everything else along the way.

# Learning to Compose Photos and Videos from Passive Cameras

Publication No. _____

Bo Xiong, Ph.D.
The University of Texas at Austin, 2019

Supervisor: Kristen Grauman

Photo and video overload is well-known to most computer users. With cameras on mobile devices, it is all too easy to snap images and videos spontaneously, yet it remains much less easy to organize or search through that content later. With increasingly portable wearable and 360° computing platforms, the overload problem is only intensifying. Wearable and 360° cameras passively record everything they observe, unlike traditional cameras that require active human attention to capture images or videos.

In my thesis, I explore the idea of automatically composing photos and videos from unedited videos captured by "passive" cameras. Passive cameras (e.g., wearable cameras, 360° cameras) offer a more relaxing experience to record our visual world but they do not always capture frames that look like intentional human-taken photos. In wearable cameras, many frames will be blurry, contain poorly composed shots, and/or simply have uninteresting content. In 360° cameras, a single omni-directional image captures the entire visual world, and the photographer's intention and attention in that moment are unknown.

To this end, I consider the following problems in the context of passive cameras: 1) what visual data to capture and store, 2) how to identify foreground objects, and 3) how to enhance the viewing experience.

First, I explore the problem of finding the best moments in unedited videos. Not everything observed in a wearable camera's video stream is worthy of being captured and stored. People can easily distinguish well-composed moments from accidental shots from a wearable camera. This prompts the question: can a vision system predict the best moments in unedited video? I first study how to find the best moments in terms of short video clips. My key insight is that video segments from shorter user-generated videos are more likely to be highlights than those from longer videos, since users tend to be more selective about the content when capturing shorter videos. Leveraging this insight, I introduce a novel ranking framework to learn video highlight detection from unlabeled videos. Next, I show how to predict *"snap points"* in unedited video—that is, those frames that look like intentionally taken photos. I propose a framework to detect snap points that requires no human annotations. The main idea is to construct a generative model of what human-taken photos look like by sampling images posted on the Web. Snapshots that people upload to share publicly online may vary vastly in their content, yet all share the key facet that they were intentional snap point moments. This makes them an ideal source of positive exemplars for our target learning problem. In both settings, despite learning without any explicit labels, my proposed models outperform discriminative baselines trained with labeled data.

Next, I introduce a novel approach to automatically segment foreground objects in images and videos. Identifying key objects is an important intermediate step for automatic photo composition. It is also a prerequisite in graphics applications like image retargeting,

production video editing, and rotoscoping. Given an image or video frame, the goal is to determine the likelihood that each pixel is part of a foreground object. I formulate the task as a structured prediction problem of assigning an object/background label to each pixel (pixel objectness), and I propose an end-to-end trainable model that draws on the respective strengths of generic object appearance and motion in a unified framework. Since large-scale video datasets with pixel level segmentations are problematic, I show how to bootstrap weakly annotated videos together with existing image recognition datasets for training. In addition, I demonstrate how the proposed approach benefits image retrieval and image retargeting. Through experiments on multiple challenging image and video segmentation benchmarks, our method offers consistently strong results and improves the state-of-the-art results for fully automatic segmentation of foreground objects.

Building on the proposed foreground segmentation method, I finally explore how to predict viewing angles to enhance photo composition after identifying those foreground objects. Specifically, I introduce snap angle prediction for 360° panoramas, which are a rich medium, yet notoriously difficult to visualize in the 2D image plane. I explore how intelligent rotations of a spherical image may enable content-aware projection with fewer perceptible distortions. Whereas existing approaches assume the viewpoint is fixed, intuitively some viewing angles within the sphere preserve high-level objects better than others. To discover the relationship between these optimal snap angles and the spherical panorama's content, I develop a reinforcement learning approach for the cubemap projection model. Implemented as a deep recurrent neural network, our method selects a sequence of rotation actions and receives reward for avoiding cube boundaries that overlap with important foreground objects. Our proposed method offers a 5x speedup compared

to exhaustive search.

Throughout, I validate the strength of the proposed frameworks on multiple challenging datasets against a variety of previously established state-of-the-art methods and other pertinent baselines. Our experiments demonstrate the following: 1) our method can automatically identify the best moments from unedited videos; 2) our segmentation method substantially improves the state-of-the-art on foreground segmentation in images and videos and also benefits automatic photo composition; 3) our viewing angle prediction for 360° imagery can enhance the viewing experience. Although my thesis mainly focuses on passive cameras, a portion of the proposed methods are also applicable to general user generated images and videos.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

Photo overload is well-known to most computer users. With cameras on mobile devices, it is all too easy to snap images and videos spontaneously, yet it remains much less easy to organize or search through that content later. This is already the case when the user actively decides which images are worth taking. What happens when that user's camera is always on, worn at eye-level, has the ability to capture the entire visual world from its optical center, and has the potential to capture everything he sees throughout the day? With increasingly portable wearable (like Google Glass, Looxcie, etc.) and 360° computing platforms, the photo overload problem is only intensifying. Wearable and 360° cameras passively record everything they observe, unlike traditional cameras that require active human attention to capture images or videos.

For both wearable videos and 360° content, not everything captured by cameras or observed in a video stream is worthy of being captured and stored. In the case of wearable cameras, even though the camera follows the wearer's activity and approximate gaze, relatively few moments actually result in snapshots the user would have intentionally decided to take, were he actively manipulating the camera. Many frames will be blurry, contain poorly composed shots, and/or simply have uninteresting content. In 360° cameras, a single omni-directional image captures the entire visual world, and the photographers in-

tention and attention in that moment are unknown. It is impractical for people to manually filter out irrelevant or badly-composed photos by re-watching the photo collections and the video streams. With "always-on" cameras and 360° content, it is also impractical to only rely on people to process days of videos. While the problem is particularly pronounced for passive cameras, it also affects typical unedited user-collected videos (e.g., videos captured with mobile phones), where good content is often mixed in with less interesting parts.

Although passive cameras offer a convenient way to record our daily activities, the quality of the captured photos or video clips is far from professional quality. For professional photographers, capturing and carefully selecting well-composed photos is only the beginning: they can also easily spend hours of hard work enhancing just one photograph. The amount of effort spent on enhancing photos is an important factor that distinguishes professional from amateur work. The enhancement operations include correcting exposure and contrast, applying filters, and altering important objects. Enhancement operations applied to foreground objects often differ from those applied to background. Therefore automatic and accurate separations between foreground objects and background can significantly improve the efficiency of photo enhancement. Object segmentation is already implemented in popular photo editing software like Photoshop. However, that function relies on low level image properties (e.g., color contrast) and therefore cannot always segment objects accurately, especially for objects that have low contrast against the background. There is clear need to develop more robust methods that can separate the foreground objects from the background to assist artists and photographers.

In my thesis, I explore the idea of automatically composing photos and videos

from large collections of unedited images and videos captured by "passive" cameras. Passive cameras (e.g., wearable cameras, 360° cameras) offer a more relaxing experience to record our visual world but they do not always capture frames that look like intentional human-taken photos. I propose data-driven methods that can automatically compose better photographs and videos for both amateur and professional photographers. My goal is to narrow the gap between the quality of visual data captured by "unintentional" photographers with passive cameras and by intentional human photographers. The automatic photo composition problem that I explore in this thesis can be further divided into the following three questions:

- **What to capture and store?** Photographers can easily use wearable cameras to record their daily lives. It remains challenging to organize or search through a large collection of visual data. A natural problem arising from this photo overload phenomenon is to determine what is worthy of being captured and stored. My goal is to design a framework that can automate the process of filtering out irrelevant or badly-composed photos and videos. Not all captured moments are equally important. I propose to study how to identify the best moments from unedited videos captured with passive or traditional cameras. I first present how to find the best moments in terms of short video clips from unedited user videos. Then I show how to find keyframes in a video that look like they could have been intentionally taken photos. The proposed framework can help photographers save effort in selecting well-composed photos or interesting video highlights. In both settings, despite learning without any explicit labels, my proposed models outperform discriminative baselines trained with labeled data.

- **How to identify important objects?** While the first question considers which moments in time constitute the best composed photos or clips, the second question explores which regions in space are most central to a photo or video. In particular, I next consider the foreground object segmentation problem for images and videos. Foreground objects naturally deserve more attention than the background. Post-processing for photos often treats foreground and background differently. Therefore automatic and accurate separations between foreground objects and background can significantly improve the efficiency of photo enhancement. Foreground segmentation is also a prerequisite in graphics applications like image retargeting, production video editing, and rotoscoping. Given a image or video frame, the goal is to determine the likelihood that each pixel is part of a foreground object. I formulate the task as a structured prediction problem of assigning an object/background label to each pixel, and I propose an end-to-end trainable model. Through experiments on multiple challenging image and video segmentation benchmarks, my method offers consistently strong results and improves the state-of-the-art results for fully automatic segmentation of foreground objects.

- **How to enhance the viewing experience?** Building on the second component of my thesis, the third major component of my thesis explores how can systems enhance the viewing experience by knowing the spatial extent of foreground objects. In particular, I consider 360° panoramas. 360° panoramas are notoriously difficult to visualize in the 2D image plane. I explore how intelligent rotations of a spherical image, together with foreground detection, enables content-aware projection with fewer perceptible distortions. I develop a reinforcement learning approach for the

cubemap projection model and my proposed method offers a 5x speedup compared to exhaustive search.

In the following sections, I will briefly introduce each of the four components towards my thesis idea of automatically composing photos from unedited images and videos. In Section 1.1 and 1.2, I address the problem of what visual data are highlights. Then I present my work on identifying important objects in Section 1.3. Finally, building on how to identify important objects, I present my work on enhancing the viewing experience for 360° panoramas in Section 1.4.

## 1.1 Learning Highlight Detection from Video Duration

As an attempt to mitigate the video overload problem, *video highlight detection* has attracted increasing attention in the research community. The video overload problem motivates the first component of my thesis: can a vision system detect video highlights in unedited user videos. The goal in highlight detection is to retrieve the moments—in the form of short video clips—that capture a user's primary attention or interest within an unedited video. See Figure 1.1.

A well-selected highlight can accelerate browsing many videos (since a user quickly previews the most important content), enhance social video sharing (since friends become encouraged to watch further), and facilitate video recommendation (since systems can relate unedited videos in a more focused way). Highlight detectors are typically *domain-specific* [179, 215, 213, 150, 142, 126], meaning they are tailored to a category of video or keywords/tags like skiing, surfing, etc. This accounts for the fact that the definition of

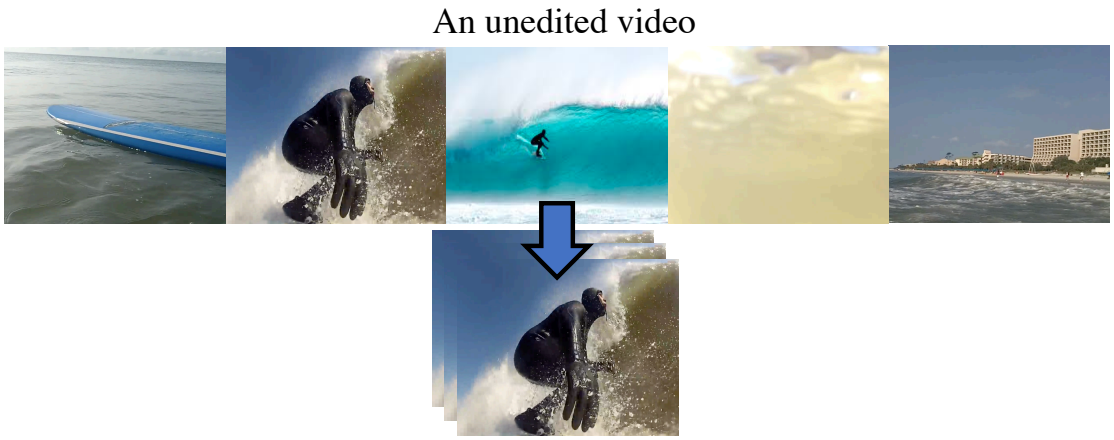


Figure 1.1: The goal in highlight detection is to retrieve the moments—in the form of short video clips—that capture a user's primary attention or interest within an unedited video. Please see Chapter 3 for the proposed approach for highlight detection and experimental results.

what constitutes a highlight often depends on the domain, e.g., a barking dog might be of interest in a dog show video, but not in a surfing video.

In the first major component of my thesis, I introduce a novel framework for domain-specific highlight detection. Our key insight is that user-generated videos, such as those uploaded to Instagram or YouTube, carry a latent supervision signal relevant for highlight detection: their duration. I hypothesize shorter user-uploaded videos tend to have a key focal point as the user is more selective about the content, whereas longer ones may not have every second be as crisp or engaging. More effort is required to film only the significant moments, or else manually edit them out later. Hence duration is an informative, though implicit, training signal about the value of the video content. I leverage duration as a new form of "weak" supervision to train highlight detectors with unedited videos.

Unlike existing supervised methods, our training data requirements are scalable, relying only on tagged video samples from the Web. Unlike existing weakly supervised methods, our approach can be trained discriminatively to isolate highlights from non-highlight time segments.

Given a category (domain) name, I first query Instagram to mine public videos which contain the given category name as hashtags. I use a total of 10M Instagram videos. Since the hashtag Instagram videos are very noisy, and since even longer videos will contain some highlights, I propose a novel ranking model that is robust to label noise in the training data. In particular, our model introduces a latent variable to indicate whether each training pair is valid or noisy. I model the latent variable with a neural network, and train it jointly with the ranking function for highlight detection. On two public challenging benchmark datasets (TVSum [171] and YouTube Highlights [179]), I demonstrate our approach improves the state of the art for domain-specific unsupervised highlight detection.

Chapter 3 gives more details on my proposed approach and results. This work originally was published in CVPR 2019 [208].

## 1.2 Detecting Snap Points in Egocentric Video with a Web Photo Prior

The first component of my thesis addresses the problem of finding video highlights—in the form of short video clips— from unedited user videos. The second component of my thesis considers the following problem: can a vision system predict *"snap points"* in unedited egocentric video—that is, those frames that look like intentionally taken photos? See Figure 1.2. While the goal in the first component is to find the best moments in terms of short video clips, the second component aims to find the best moments in terms

Detect “snap points” from unedited egocentric videos

Figure 1.2: My goal is to detect frames that look like intentionally taken photos from egocentric videos. The frame with the highest bar in the sequence would rate highest as a snap point. Please see Chapter 4 for the proposed approach for snap point detection and experimental results.

of keyframes in egocentric videos. Both the first and the second components address the problem of what visual data to capture and store.

Egocentric video contains a wide variety of scene types, activities, and actors. This is certainly true for human camera wearers going about daily life activities, and it will be increasingly true for mobile robots that freely explore novel environments. Accordingly, a snap point detector needs to be largely domain invariant and generalize across varied subject matter. An optimal snap point is likely to differ in subtle ways from its less-good temporal neighbors, i.e., two frames may be similar in content but distinct in terms of snap point quality. That means that cues beyond the standard texture/color favorites may be necessary. Finally, and most importantly, while it would be convenient to think of the problem in discriminative terms (e.g., training a snap point vs. non-snap point classifier), it is burdensome to obtain adequate and unbiased labeled data. Namely, we’d need people to manually mark frames that appear intentional, and to do so at a scale to accommodate arbitrary environments.

In the second major component of my thesis, I introduce an approach to detect

snap points from egocentric video that requires no human annotations. The main idea is to construct a generative model of what human-taken photos look like by sampling images posted on the Web. Snapshots that people upload to share publicly online may vary vastly in their content, yet all share the key facet that they were intentional snap point moments. This makes them an ideal source of positive exemplars for our target learning problem. Furthermore, with such a Web photo prior, I sidestep the issue of gathering negatively-labeled instances to train a discriminative model, which could be susceptible to bias and difficult to scale. In addition to this prior, my approach incorporates domain adaptation to account for the distribution mismatch between Web photos and egocentric video frames. Finally, I develop features suited to capturing the framing effects in snap points.

I propose two applications of snap point prediction. For the first, I show how snap points can improve object detection reliability for egocentric cameras. It is striking how today's object detectors fail when applied to arbitrary egocentric data. Unsurprisingly, their accuracy drops because detectors trained with human-taken photos (e.g., the Flickr images gathered for the PASCAL VOC benchmark) do not generalize well to the arbitrary views seen by an ego-camera. I show how snap point prediction can improve the precision of an off-the-shelf detector, essentially by predicting those frames where the detector is most trustworthy. For the second application, I use snap points to select keyframes for egocentric video summaries.

I apply my method to 17.5 hours of videos from both human-worn and robot-worn egocentric cameras. I demonstrate the absolute accuracy of snap point prediction compared to a number of viable baselines and existing metrics. Furthermore, I show its potential for object detection and keyframe selection applications. The results are a promising

step towards filtering the imminent deluge of wearable camera video streams.

Chapter 4 gives more details on my proposed approach and results.This work originally was published in ECCV 2014 [204] and a book chapter in MCVMC 2015 [205].

## 1.3 Pixel Objectness: Learning to Segment Generic Objects in Images and Videos

While the first two components of my thesis address the question of finding the best moments in time in terms of either short clips or keyframes from unedited user videos, the third component of my thesis explores which regions in space are most central to a photo or video. Finding the best moments in time can quickly filter out irrelevant video content while finding the most central regions in space can accelerate post-processing of editing photos or videos. Next, I consider the generic foreground object segmentation problem for images and videos.

While my focus is on foreground extraction for the sake of automatic photo composition, generic object segmentation in images and videos is also a fundamental vision problem with several applications. For example, a visual search system can use generic object segmentation to focus on the important objects in the query image, ignoring background clutter. It is also a prerequisite in graphics applications like image retargeting, production video editing, and rotoscoping. Knowing the spatial extent of objects can also benefit downstream vision tasks like scene understanding, caption generation, and summarization. In any such setting, it is crucial to segment "generic" objects in a *category-independent* manner. That is, the system must be able to identify object boundaries for objects it has never encountered during training. This differentiates the problem from tra-

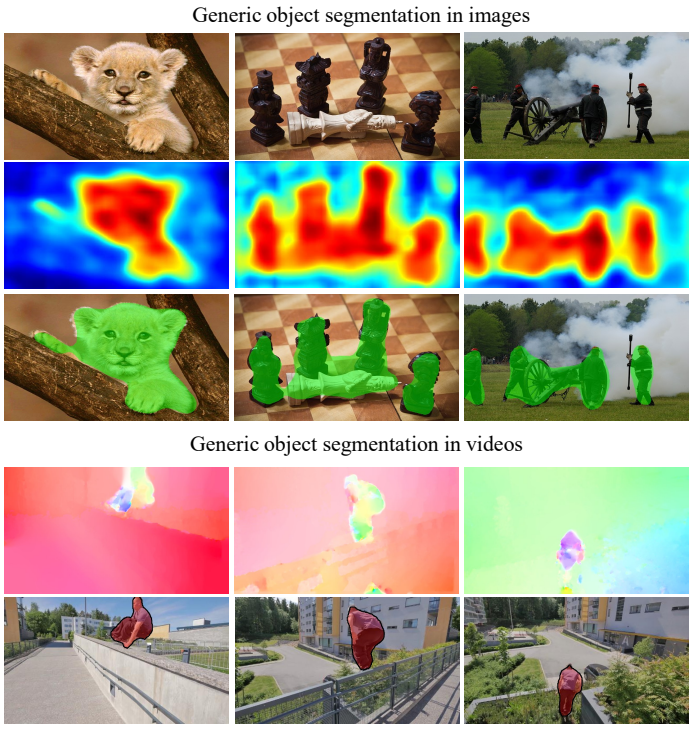


Figure 1.3: Given a novel image (top row), my method predicts an objectness map for each pixel (2nd row) and a single foreground segmentation (3rd row). Given a novel video, my end-to-end trainable model simultaneously draws on the strengths of generic object appearance and motion (4th row, color-coded optical flow images) to extract generic objects (last row). Please see Chapter 5 for the proposed approach for pixel objectness and experimental results.

ditional recognition or "semantic segmentation" [127, 24], where the system is trained specifically for predefined categories, and is not equipped to segment any others.

In the third main component of my thesis, I introduce *pixel objectness*, a new approach to generic object segmentation in images and video. Given a novel image or video frame, the goal is to determine the likelihood that each pixel is part of a foreground object (as opposed to background or "stuff" classes like grass, sky, sidewalks, etc.) Our

definition of a generic object follows that commonly used in the object proposal literature [4, 17, 6, 34, 224, 191]. Pixel objectness quantifies how likely a pixel belongs to an object of *any* class, and should be high even for objects unseen during training. See Fig. 1.3 (top). For the image case, pixel objectness can be seen as a pixel-level extension of window-level objectness [4], and hence the name for my method is a nod to that influential work.

I propose an end-to-end trainable model that draws on the respective strengths of generic (non-category-specific) object appearance and motion in a unified framework. Specifically, I develop a novel two-stream fully convolutional deep segmentation network where individual streams encode generic appearance and motion cues derived from a video frame and its corresponding optical flow. These individual cues are fused in the network to produce a final object versus background pixel-level binary segmentation for each video frame (or image). See Fig. 1.3 (bottom). The proposed network segments both static and moving objects without any human involvement. A second key contribution of my work is to explore how weaker annotations can be adopted to train the models. First, I show that, somewhat surprisingly, when training the appearance stream of our model with *explicit boundary-level* annotations for few categories pooled together into a single generic "object-like" class, pixel objectness generalizes well to *thousands* of unseen objects. This generalization ability is facilitated by an *implicit image-level* notion of objectness built into a pretrained classification network, which I transfer to my segmentation model during initialization. Second, to allow training with few densely labeled video examples, I show how to leverage readily available *image* segmentation annotations together with *weakly annotated video* data to train the motion stream of our model.

Through extensive experiments, I show that my model generalizes very well to unseen objects. For images, my method obtains state-of-the-art performance on the challenging ImageNet [30] and MIT Object Discovery [157] datasets. I also show how to leverage our segmentations to benefit object-centric image retrieval and content-aware image resizing. For video segmentation, my method advances the state-of-the-art for fully automatic video object segmentation on multiple challenging datasets, DAVIS [145], YouTube-Objects [151, 75, 183], and Segtrack-v2 [111]. My results show the reward of learning from both signals in a unified framework: a true synergy, often with substantially stronger results than what I can obtain from either one alone—even if they are treated with an equally sophisticated deep network.

Chapter 5 gives more details on my proposed approach and results. This work originally was published in CVPR 2017 [76] and TPAMI 2018 [207].

## 1.4 Snap Angle Prediction for 360° Panoramas

Above, I propose a method for foreground segmentation. By applying the proposed method to 360° panoramas, I now overview how to predict snap angles to enhance photo composition after identifying those foreground objects.

The recent emergence of inexpensive and lightweight 360° cameras enables exciting new ways to capture our visual surroundings. Unlike traditional cameras that capture only a limited field of view, 360° cameras capture the entire visual world from their optical center. Advances in virtual reality (VR) technology and promotion from social media platforms like Youtube and Facebook are further boosting the relevance of 360° images and videos.

However, viewing 360° content presents its own challenges. Currently three main directions are pursued: manual navigation, field-of-view (FOV) reduction, and content-based projection. In manual navigation scenarios, a human viewer chooses which normal field-of-view subwindow to observe, e.g., via continuous head movements in a VR headset, or mouse clicks on a screen viewing interface. In contrast, FOV reduction methods generate normal FOV videos by learning to render the most interesting or capture-worthy portions of the viewing sphere [175, 174, 69, 106]. While these methods relieve the decision-making burden of manual navigation, they severely limit the information conveyed by discarding all unselected portions. Projection methods render a wide-angle view, or the entire sphere, onto a single plane (e.g., equirectangular or Mercator) [170] or multiple planes [50]. While they avoid discarding content, any projection inevitably introduces distortions that can be unnatural for viewers. Content-based projection methods can help reduce perceived distortions by prioritizing preservation of straight lines, conformality, or other low-level cues [164, 97, 110], optionally using manual input to know what is worth preserving [21, 184, 20, 101, 196].

However, all prior automatic content-based projection methods implicitly assume that the *viewpoint* of the input 360° image is fixed. That is, the spherical image is processed in some default coordinate system, e.g., as the equirectangular projection provided by the camera manufacturer. This assumption limits the quality of the output image. Independent of the content-aware projection eventually used, a fixed viewpoint means some *arbitrary portions of the original sphere will be relegated to places where distortions are greatest*—or at least where they will require most attention by the content-aware algorithm to "undo".

In the last main component of my thesis, I propose to eliminate the fixed viewpoint

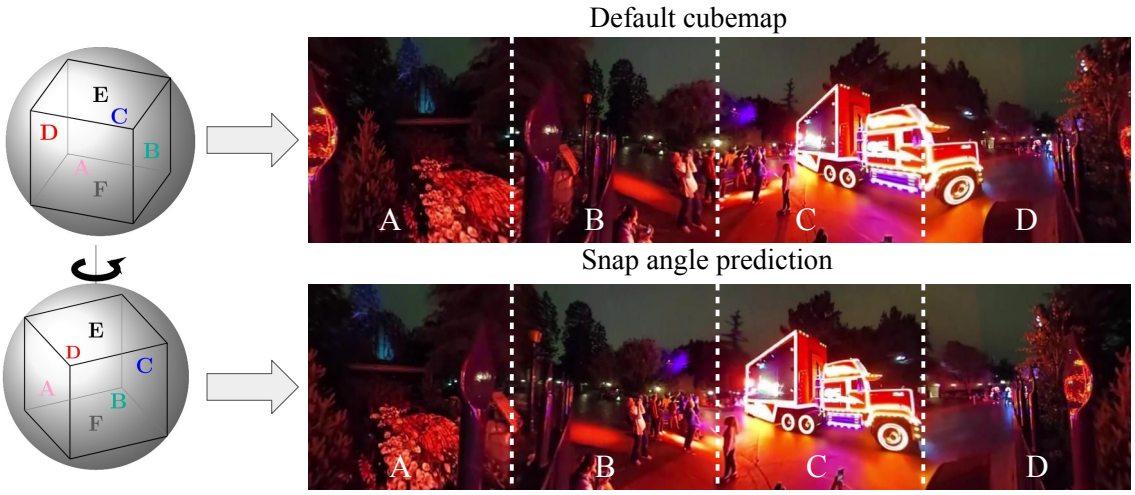


Figure 1.4: Comparison of a cubemap before and after snap angle prediction (dotted lines separate each face). Unlike prior work that assumes a fixed angle for projection, I propose to predict the cube rotation that will best preserve foreground objects in the output. For example, here my method better preserves the truck (third picture C in the second row). I show four (front, right, left, and back) out of the six faces for visualization purposes. Please see Chapter 6 for the proposed approach for snap angle prediction and experimental results. Best viewed in color or pdf.

assumption. My key insight is that an intelligently chosen viewing angle can immediately lessen distortions, even when followed by a conventional projection approach. In particular, I consider the widely used Cubemap projection [50, 1, 2]. A cubemap visualizes the entire sphere by first mapping the sphere to a cube with rectilinear projection (where each face captures a 90° FOV) and then unfolding the faces of the cube. Often, an important object can be projected across two cube faces, destroying object integrity. In addition, rectilinear projection distorts content near cube face boundaries more. See Figure 1.4, top. However, intuitively, some viewing angles—some cube orientations—are less damaging than others.

I introduce an approach to automatically predict *snap angles*: the rotation of the cube that will yield a set of cube faces that, among all possible rotations, most look like nicely composed human-taken photos originating from the given 360° panoramic image. While what comprises a "well-composed photo" is itself the subject of active research [100, 71, 204, 54, 92], we concentrate on a high-level measure of good composition, where the goal is to consolidate each (automatically detected) foreground object within the bounds of one cubemap face. See Figure 1.4, bottom.

Accordingly, I formalize the snap angle objective in terms of minimizing the spatial mass of foreground objects near cube edges. I develop a reinforcement learning (RL) approach to infer the optimal snap angle given a 360° panorama. I implement the approach with a deep recurrent neural network that is trained end-to-end. The sequence of rotation "actions" chosen by my RL network can be seen as a coarse-to-fine adjustment of the camera viewpoint, in the same spirit as how people refine their camera's orientation just before snapping a photo.

I validate the approach on a variety of 360° panorama images. Compared to several informative baselines, I demonstrate that 1) snap angles better preserve important objects, 2) my RL solution efficiently pinpoints the best snap angle, 3) cubemaps unwrapped after snap angle rotation suffer less perceptual distortion than the status quo cubemap, and 4) snap angles even have potential to impact recognition applications, by orienting 360° data in ways that better match the statistics of normal FOV photos used for today's pretrained recognition networks.

Chapter 6 gives more details on my proposed approach and results. This work originally was published in ECCV 2018 [206].

In conclusion, this chapter has provided a brief overview of my thesis work on automatically composing photos from unedited images and videos captured by passive cameras. My proposed methods can narrow the gap between the quality of visual data captured by "unintentional" photographers with passive cameras and by intentional human photographers. The proposed methods in this thesis are also cost-efficient in terms of human annotation requirement on training data by leveraging either weakly labeled data or unlabeled data.

## 1.5 Roadmap

In this chapter, I have provided a brief overview of my thesis on learning to compose photos from passive cameras and introduced each of the four components of my thesis. In next chapter, I review prior work surrounding the topics of my thesis. Afterwards, in Chapters 3, 4, 5, and 6, I present the proposed methods and experimental results for each of the four components of my thesis outlined above in Sections 1.1, 1.2, 1.3, and 1.4 respectively. Finally, I discuss possible directions for future work in Chapter 7 and summarize the key findings of my thesis in Chapter 8.

# Chapter 2

# Related Work

In this chapter, I review prior work relevant to the research that will be presented in Chapters 3, 4, 5, and 6. I review topics on video highlight detection and summarization, learning with noisy labels, egocentric videos, leveraging Web images, image segmentation, video segmentation, viewing wide-angle images and panoramas and recurrent networks for attention. In each topic, I discuss the problem statement and how prior techniques attempt to address different aspects of the problem. I also compare and contrast my proposed methods and prior work. The related work presented here serves to provide readers with useful background and to understand the existing techniques related to the research presented in this thesis.

## 2.1 Video Highlight Detection and Summarization

I first overview prior work in video highlight detection and summarization, and discuss how it relates to my proposed method for video highlight detection that will be presented in Chapter 3. In Section 2.2, I then discuss prior work on learning with noisy labels. My proposed framework for video highlight detection is motivated by prior work on learning with noisy labels.

### 2.1.1 Video Highlight Detection

Many prior approaches focus on highlight detection for sports video [158, 210, 181, 194]. Recently, supervised video highlight detection has been proposed for Internet videos [179] and first-person videos [215]. These methods all require human annotated ⟨highlight, source video⟩ pairs for each specific domain. The Video2GIF approach [57] learns from GIF-video pairs, which are also manually created. All supervised highlight detection methods require human edited/labeled ranking pairs. In contrast, the method I present in Chapter 3 does not use manually labeled highlights. My work on highlight detection offers a new way to take advantage of freely available videos from the Internet.

Unsupervised video highlight detection methods do not require video annotations to train. They can be further divided into methods that are domain-agnostic or domain-specific. Whereas a domain-agnostic approach like motion strength [136] operates uniformly on any video, domain-specific methods train on a collection of videos of the same topic. They leverage concepts like visual co-occurrence [27], category-aware reconstruction loss [222, 213], or collaborative sparse selection within a category [143]. Another approach is first train video category classifiers, then detect highlights based on the classifier scores [150] or spatial-temporal gradients from the classifier [142]. Like the domain-specific methods, my approach on highlight detection also tailors highlights to the topic domain; I gather the relevant training videos per topic automatically using keyword search on the Web. Unlike any existing methods, I leverage video duration as a weak supervision signal.

### 2.1.2 Video Summarization

Whereas highlight detection aims to score individual video segments for their worthiness as highlights, *video summarization* aims to provide a complete synopsis of the whole video, often in the form of a structured output, e.g., a storyline graph [96, 209], a sequence of selected keyframes [108] or clips [55, 221]. Video summarization is often formalized as a structured subset selection problem considering not just importance but also diversity [49, 129] and coherency [129]. Supervised summarization methods focus on learning a visual interestingness/importance score [108, 55], submodular mixtures of objectives [56, 212], or temporal dependencies [220, 221]. Unsupervised summarization methods often focus on low-level visual cues to locate important segments. Recent unsupervised and semi-supervised methods use recurrent auto-encoders to enforce that the summary sequence should be able to generate a sequence similar to the original video [213, 132, 221]. Many rely on Web image priors [91, 171, 93, 96] or semantic Web video priors [15]. While I also leverage Web data, my idea about duration is novel.

## 2.2 Learning with Noisy Labels

My work on learning to detect highlights from video duration is also related to learning from noisy data, a topic of broad interest in machine learning [138, 124]. The proportion SVM [217] handles noisy data for training SVMs where a fraction of the labels per group are expected to be incorrect, with applications to activity recognition [105]. Various methods explore how to train neural networks with noisy data [176, 153, 116].

Recent work on attention-based Multiple Instance Learning (MIL) helps focus

on reliable instances using a differentiable MIL pooling operation for bags of embeddings [70]. Inspired by this, I propose a novel attention-based loss to reliably identify valid samples from noisy training data in Chapter 3, but unlike [70], 1) I have "bags" defined in the space of ranking constraints, 2) the proposed attention is defined in the loss space, not in the feature space, 3) my model predicts scores at the instance level, not the "bag" level, and 4) my attention mechanism is extended with multiple heads to take into account a prior for the expected label noise level.

## 2.3 Egocentric Videos

Next, I overview prior work in egocentric videos, and discuss how it relates to my proposed method for snap point detection that will be presented in Chapter 4.

Egocentric video analysis, pioneered in the 90's [133, 173], is experiencing a surge of research activity thanks to today's portable devices. The primary focus is on object [155, 114] or activity recognition [172, 39, 98, 149, 41, 161, 114]. Compared with well-posed photographs, egocentric videos contain more uninformative frames, which are often poorly composed and illuminated [44]. Motions cues [155] in egocentric video are useful to segment foreground objects and therefore improve object recognition. Gaze information [114] can also improve both object and activity recognition. No prior work explores snap point detection, which I will introduce in detail in Chapter 4.

I consider object detection and keyframe selection as applications of snap points for unconstrained wearable camera data. In contrast, prior work for detection in egocentric video focuses on controlled environments (e.g., a kitchen) and handheld objects (e.g., the mixing bowl) [155, 114, 172, 39, 41]. Nearly all prior keyframe selection work

assumes third-person static cameras (e.g., [120, 122]), where all frames are already intentionally composed, and the goal is to determine which are representative for the entire video. In contrast, snap points aim to discover intentional-looking frames, not maximize diversity or representativeness. Some video summarization work tackles dynamic egocentric video [107, 128]. Such methods could exploit snap points as a filter to limit the frames they consider for summaries.

Methods in ubiquitous computing use manual intervention [133] or external non-visual sensors [64, 65] (e.g., skin conductivity or audio) to trigger the camera. My image-based approach for snap point detection in Chapter 4 is complementary; true snap points are likely a superset of those moments where abrupt physiological or audio changes occur.

## 2.4 Leveraging Web Images

My proposed method for snap point detection in Chapter 4 is also related to work on leveraging Web images. Photos that people upload to share publicly online may vary vastly in their content, yet all share the key facet that they were intentional snap point moments. This makes them an ideal source of positive exemplars for our snap point detection problem.

### 2.4.1 Predicting High-level Image Properties

A series of interesting work predicts properties from images like saliency [123], professional photo quality [88], memorability [72], aesthetics, interestingness [32, 54], or suitability as a candid portrait [43]. These methods train a discriminative model using various image descriptors, then apply it to label human-taken photos. In contrast, I develop a

generative approach with (unlabeled) Web photos, and apply it to *find* human-taken photos. Critically, a snap point need not be beautiful, memorable, etc., and it could even contain mundane content. Snap points are thus a broader class of photos. This is exactly what makes them relevant for the proposed object detection application; in contrast, an excellent aesthetics detector (for example) would fire on a narrower set of photos, eliminating non-aesthetic photos that could nonetheless be amenable to off-the-shelf object detectors.

### 2.4.2 Web Image Priors

The Web is a compelling resource for data-driven vision methods. Both the volume of images as well as the accompanying noisy meta-data open up many possibilities. Most relevant to my work are methods that exploit the biases of human photographers. This includes work on discovering iconic images of landmarks [165, 112, 197] (e.g., the Statue of Liberty) or other tourist favorites [60, 86, 23, 94] by exploiting the fact that people tend to take similar photos of popular sites. Web images can also serve as a useful prior for image super-resolution [178], scene completion [61] and image deblur [177]. Similarly, the photos users upload when trying to sell a particular object (e.g., a used car) reveal that object's canonical viewpoints, which can help select keyframes to summarize short videos of the same object [90]. Event video summarization [93] can also benefit from web image collections of the same event. My snap point method also learns about human framing or composition biases, but, critically, in a manner that transcends the specific content of the scene. That is, rather than learn when a popular landmark or object is in view, we want to know when a well-composed photo of *any* scene is in view. My proposed Web photo prior represents the photos humans intentionally take, independent of subject matter.

## 2.5 Image Segmentation

Next I overview prior work in image segmentation and explain the connections with my proposed pixel objectness approach that will be presented in Chapter 5.

### 2.5.1 Category-independent Image Segmentation

**Interactive image segmentation** algorithms such as the popular GrabCut [156] let a human guide the algorithm using bounding boxes or scribbles. These methods are most suitable when high precision segmentations are required such that some guidance from humans is worthwhile. While some methods try to minimize human involvement [73, 53], still typically a human is always in the loop to guide the algorithm. In contrast, my model for segmentation is fully automatic and segments foreground objects without any human guidance.

**Object proposal methods**, also discussed above, produce thousands of generic object proposals either in the form of bounding boxes [34, 224, 191] or regions [17, 6, 148, 68]. Generating thousands of hypotheses ensures high recall, but often results in low precision. Though effective for object detection, it is difficult to automatically filter out accurate proposals from this large hypothesis set without class-specific knowledge. My method for foreground extraction instead generates a *single* hypothesis of the foreground as my final segmentation.

**Saliency models** have also been widely studied in the literature. The goal is to identify regions that are likely to capture human attention. While some methods produce highly localized regions [121, 141, 12], others segment complete objects [26, 81, 123, 115, 223, 113]. While saliency focuses on objects that "stand out", my method is designed to seg-

ment all foreground objects, irrespective of whether they stand out in terms of low-level saliency. This is true even for the deep learning based saliency methods [141, 121, 223, 113] which like pixel objectness are end-to-end trained but prioritize objects that stand out.

### 2.5.2 Category-specific Image Segmentation

**Semantic segmentation** refers to the task of jointly *recognizing* and segmenting objects, classifying each pixel into one of $k$ fixed categories. Recent advances in deep learning have fostered increased attention to this task. Most deep semantic segmentation models include fully convolutional networks that apply successive convolutions and pooling layers followed by upsampling or deconvolution operations in the end to produce pixel-wise segmentation maps [127, 24]. However, these methods are trained for a fixed number of categories. My method for foreground segmentation is the first to show that a fully convolutional network can be trained to accurately segment *arbitrary* foreground objects. Though relatively few categories are seen in training, my model for foreground segmentation generalizes very well to unseen categories (as I demonstrate for 3,624 classes from ImageNet, only a fraction of which overlap with PASCAL, the source of my training masks).

**Weakly supervised joint segmentation** methods use weaker supervision than semantic segmentation methods. Given a batch of images known to contain the same object category, they segment the object in each one. The idea is to exploit the similarities within the collection to discover the common foreground. The output is either a pixel-level mask [192, 85, 95, 157, 25, 74] or bounding box [31, 182]. While joint segmentation is useful, its performance is limited by the shared structure within the collection; intra-class

viewpoint and shape variations pose a significant challenge. Moreover, in most practical scenarios, such weak supervision is not available. A stand alone single-image segmentation model like ours is more widely applicable.

**Propagation-based methods** transfer information from exemplars with human-labeled foreground masks [104, 52, 74]. They usually involve a matching stage between likely foreground regions and the exemplars. The downside is the need to store a large amount of exemplar data at test time and perform an expensive and potentially noisy matching process for each test image. In contrast, my segmentation model, once trained end-to-end, is very efficient to apply and does not need to retain any training data.

## 2.6 Video Segmentation

My proposed pixel objectness approach in Chapter 5 can also segment foreground objects in videos. I next discuss related work in automatic methods and human-guided methods for video segmentation and explain the connections with my work.

### 2.6.1 Automatic Video Segmentation Methods

Similar to image segmentation work, video segmentation has been explored under varying degrees of supervision or human interaction. Fully automatic or unsupervised video segmentation methods assume no human input on the video. However, unlike image segmentation that only relies on appearance cues, video segmentation can also utilize motion to segment generic objects. They can be grouped into two broad categories.

First we have the supervoxel methods [51, 211, 46] which oversegment the video into space-time blobs with cohesive appearance and motion. Their goal is to generate

mid-level video regions useful for downstream processing (e.g., action detection [214, 35]), whereas my goal is to produce space-time tubes which accurately delineate object boundaries.

Second we have the fully automatic methods that generate thousands of "object-like" space-time segments [199, 45, 140, 200]. While useful in accelerating object detection, it is not straightforward to automatically select the most accurate one when a single hypothesis is desired. Methods that do produce a single hypothesis [109, 144, 37, 190, 180, 67, 11, 99] strongly rely on motion to identify the objects, either by seeding appearance models with moving regions or directly reasoning about occlusion boundaries using optical flow. This limits their capability to segment static objects in video. In comparison, my method for video segmentation is fully automatic, produces a single hypothesis, and can segment both static and moving objects. Concurrent work [185] trains a deep network with synthetic data to predict moving objects from motion. My work for video segmentation differs in two ways: 1) I show how to bootstrap weakly annotated real videos together with existing image recognition datasets for training whereas their work is trained with synthetic data; 2) my framework for video segmentation learns from appearance and motion jointly whereas their work is only trained with motion.

Deep learning for combining motion and appearance in videos has proven to be useful in several other computer vision tasks such as video classification [139, 87], action recognition [167, 80], object tracking [195, 130] and even computation of optical flow [33]. While I take inspiration from these works, my work is the first deep framework for segmenting objects in videos in a fully automatic manner.

### 2.6.2 Human-guided Video Segmentation Methods

Related to the interactive methods for images discussed above, there are also approaches for semi-automatic video segmentation. Semi-supervised label propagation methods accept human input on a subset of frames, then propagate it to the remaining frames [154, 8, 38, 193, 75, 146, 134, 189, 14, 78, 89]. In a similar vein, interactive video segmentation methods leverage a human in the loop to provide guidance or correct errors, e.g., [9, 163, 152]. The deep learning-based human-guided video segmentation methods [14, 78, 89] typically focus more on learning object appearance from the manual annotations since the human pinpoints the object of interest. Motion is primarily used to propagate information or enforce temporal smoothness. In the proposed method for segmentation, both motion and appearance play an equally important role, and I show their synergistic combination results in a much better segmentation quality. Moreover, my method for video segmentation is fully automatic and uses no human involvement to segment a novel video.

## 2.7 Viewing Wide-angle Images and Panoramas

Finally, I overview prior work relevant to predicting snap angles, which will be presented in Chapter 6. I first discuss prior work on projection methods for viewing wide-angle images and panoramas in this section. In Section 2.8, I then discuss prior work for recurrent models, which motivate my proposed method for snap angle prediction.

### 2.7.1 Spherical Image Projection

Spherical image projection models project either a limited FOV [164, 22] or the entire panorama [170, 218, 50]. The former group includes rectilinear and Pannini [164] projection; the latter includes equirectangular, stereographic, and Mercator projections (see [170] for a review). Rectilinear and Pannini prioritize preservation of lines in various ways, but always independent of the specific input image. Since any projection of the full sphere must incur distortion, multi-view projections can be perceptually stronger than a single global projection [218]. Cubemap [50], the subject of my snap angle approach in Chapter 6, is a multi-view projection method; current approaches simply consider a cubemap in its default orientation.

### 2.7.2 Content-aware Projection

Built on spherical projection methods, content-based projections make image-specific choices to reduce distortion. Recent work [97] optimizes the parameters in the Pannini projection [164] to preserve regions with greater low-level saliency and straight lines. Interactive methods [21, 184, 20, 101] require a user to outline regions of interest that should be preserved or require input from a user to determine projection orientation [196]. My approach for snap angles is content-based and fully automatic. Whereas prior automatic methods assume a fixed viewpoint for projection, I propose to actively predict snap angles for rendering. Thus, my idea is orthogonal to 360° content-aware projection. Advances in the projection method could be applied in concert with my algorithm, e.g., as post-processing to enhance the rotated faces further. For example, when generating cubemaps, one could replace rectilinear projection with others [164, 97, 21] and keep the rest of the

learning framework unchanged. Furthermore, the proposed snap angles respect high-level image content—detected foreground objects—as opposed to typical lower-level cues like line straightness [20, 21] or low-level saliency metrics [97].

### 2.7.3 Viewing Panoramas

Since viewing 360° and wide-angle data is non-trivial, there are vision-based efforts to facilitate. The system of [102] helps efficient exploration of gigapixel panoramas. More recently, several systems automatically extract normal FOV videos from 360° video, "piloting" a virtual camera by selecting the viewing angle and/or zoom level most likely to interest a human viewer [175, 174, 69, 106].

## 2.8 Recurrent Networks for Attention

Though treating very different problems than ours, multiple recent methods incorporate deep recurrent neural networks (RNN) to make sequential decisions about where to focus attention. The influential work of [137] learns a policy for visual attention in image classification. Active perception systems use RNNs and reinforcement learning to select places to look in a novel image [16, 135], environment [79], or video [216, 5, 169] to detect certain objects or activities efficiently. Broadly construed, we share the general goal of efficiently converging on a desired target "view", but my problem domain of snap angle prediction is entirely different.

Having reviewed relevant prior work in this chapter, I now move on to present the technical details of the approach together with experimental results for each component in the upcoming chapters. In the next chapter, I consider the problem of video highlight detection from unedited user videos.

# Chapter 3

# Learning Highlight Detection from Video Duration

[1] In this chapter, I address the first component of my thesis: how to detect video highlights from unedited user videos. The goal in highlight detection is to retrieve the moments—in the form of short video clips—that capture a user's primary attention or interest within an unedited video. A well-selected highlight can accelerate browsing many videos (since a user quickly previews the most important content), enhance social video sharing (since friends become encouraged to watch further), and facilitate video recommendation (since systems can relate unedited videos in a more focused way).

Existing methods largely follow one of two strategies. The first strategy poses highlight detection as a supervised learning task [57, 179, 215]. Given unedited videos together with manual annotations for their highlights, a ranker is trained to score highlight segments higher than those elsewhere in the video [57, 179, 215]. While the resulting detector has the advantage of good discriminative power, the approach suffers from heavy, non-scalable supervision requirements. The second strategy instead considers highlight learning as a weakly supervised recognition task. Given domain-specific videos, the sys-

[1] The work in this chapter was supervised by Prof. Kristen Grauman and originally published in: "Less is More: Learning Highlight Detection from Video Duration ". Bo Xiong, Yannis Kalantidis, Deepti Ghadiyaram and Kristen Grauman. In Proceedings of the IEEE International Conference on Computer Vision, Long Beach, CA, June 2019.

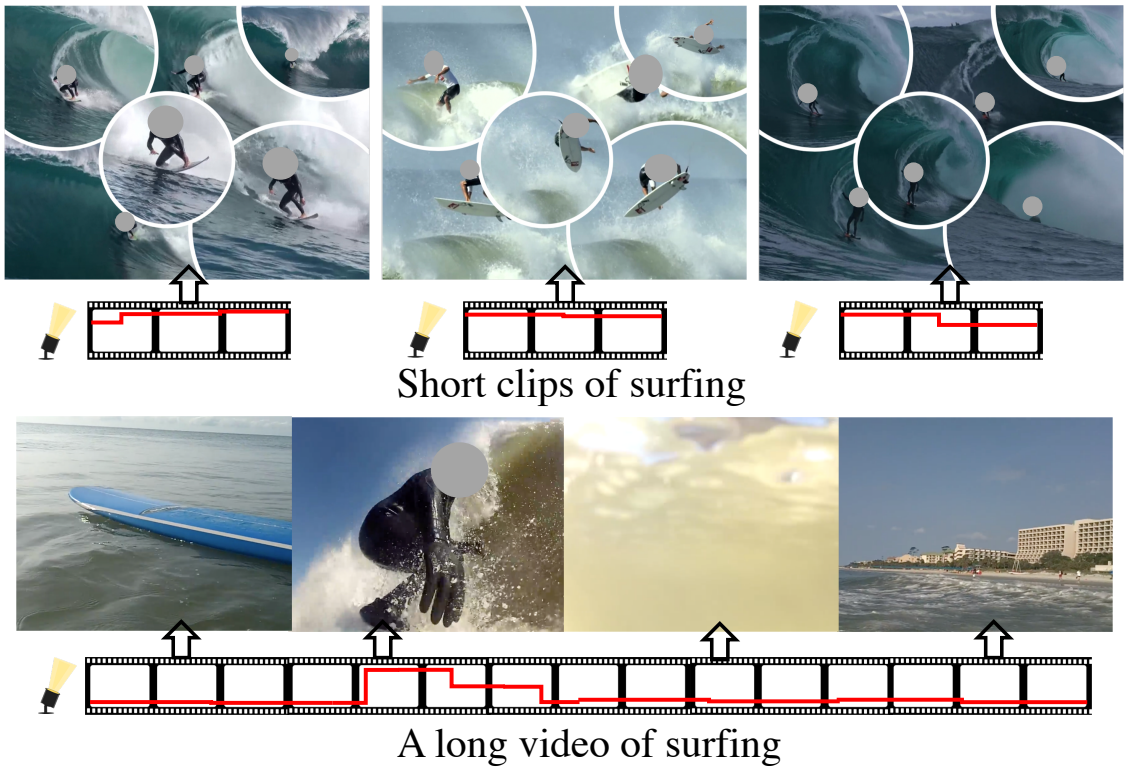


Figure 3.1: Video frames from three shorter user-generated video clips (top row) and one longer user-generated video (second row). Although all recordings capture the same event (surfing), video segments from shorter user-generated videos are more likely to be highlights than those from longer videos, since users tend to be more selective about their content. The height of the red curve indicates highlight score over time. I leverage this natural phenomenon as a free latent supervision signal in large-scale Web video.

tem discovers what appears commonly among the training samples, and learns to detect such segments as highlights in novel videos for the same domain [213, 150, 142, 126]. While more scalable in supervision, this approach suffers from a lack of discriminative power.

In the first major component of my thesis, I introduce a novel framework for

domain-specific highlight detection that addresses both these shortcomings. My key insight is that user-generated videos, such as those uploaded to Instagram or YouTube, carry a latent supervision signal relevant for highlight detection: their duration. I hypothesize shorter user-uploaded videos tend to have a key focal point as the user is more selective about the content, whereas longer ones may not have every second be as crisp or engaging. See Figure 3.1. I leverage duration as a new form of "weak" supervision to train highlight detectors with unedited videos. Unlike existing supervised methods, my training data requirements are scalable, relying only on tagged video samples from the Web. Unlike existing weakly supervised methods, my approach can be trained discriminatively to isolate highlights from non-highlight time segments. On two public challenging benchmark datasets (TVSum [171] and YouTube Highlights [179]), I demonstrate our approach improves the state of the art for domain-specific unsupervised highlight detection. Throughout, I use the term *unsupervised* to indicate the method does not have access to any manually created summaries for training. I use the term *domain-specific* to mean that there is a domain/category of interest specified by keyword(s) like "skiing", following [150, 213, 142, 126].

I first describe my approach for video highlight detection in Section 3.1, and then show results in Section 3.2. Please see Section 2.1 for prior work on detecting video highlights and Section 2.2 on learning with noisy labels, which motivates our proposed method for highlight detection.

## 3.1 Approach

We explore domain-specific highlight detection trained with unlabeled videos. We first describe how we automatically collect large-scale hashtag video data for a domain (Sec. 3.1.1). Then we present our novel framework for learning highlights aided by duration as a training signal (Sec. 3.1.2). The results will show the impact of our method to find highlights in standard public benchmarks (Sec. 3.2).

### 3.1.1 Large-scale Instagram Training Video

First we describe our data collection process. We choose Instagram as our source to collect videos because it contains a large amount of public videos associated with hashtags. In addition, because Instagram users tend to upload frequently via mobile for social sharing, there is a natural variety of duration and quality—some short and eye-catching videos, others less focused. The duration of a video from Instagram can vary from less than a second to 1 minute.

Our goal is to build domain-specific highlight detectors. Given a category name, we query Instagram to mine for videos that contain the given category name among their hashtags. For most categories, this returns at least $200,000$ videos. Since we validate our approach to detect highlights in the public TVSum and YouTube Highlights benchmarks [171, 179] (see Sec. 3.2), the full list of hashtags queried are *dog, gymnastics, parkour, skating, skiing, surfing, changing vehicle tire, getting vehicle unstuck, grooming an animal, making sandwich, parade, flash mob gathering, beekeeping, attempting bike tricks,* and *dog show*. Thus the data spans a range of domains frequently captured for sharing on social media or browsing for how-to's online. Altogether we acquire more than

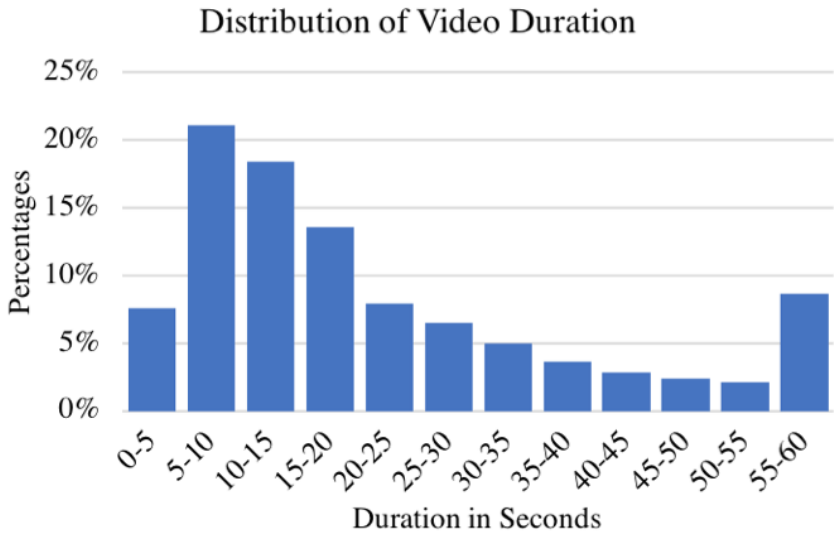


Figure 3.2: Durations for the 10M Instagram training videos.

10M training videos.

Figure 3.2 shows the distribution of their durations, which vary from less than a second to 1 minute. We see there is a nice variety of lengths, with two modes centered around short ($\sim$ 10 s) and "long" ($\sim$ 60 s) clips.

Postprocessing hashtags, injecting word similarity models, or chaining to related keywords could further refine the quality of the domain-specific data [131]. However, our experiments suggest that even our direct hashtag mining is sufficient to gather data relevant to the public video datasets we ultimately test on. Below we will present a method to cope with the inherent noise in both the Instagram tags as well as the long/short video hypothesis.

### 3.1.2 Learning Highlights from Video Duration

Next we introduce our ranking model that utilizes large-scale hashtagged video data and their durations for training video highlight detectors.

Recall that a video highlight is a short video segment within a longer video that would capture a user's attention and interest. Our goal is to learn a function $f(x)$ that infers the highlight score of a temporal video segment given its feature $x$ (to be specified below). Then, given a novel video, its highlights can be prioritized (ranked) based on each segment's predicted highlight score.

A supervised regression solution would attempt to learn $f(x)$ from a video dataset with manually annotated highlight scores. However, calibrating highlight scores collected from multiple human annotators is itself challenging. Instead, highlight detection can be formalized as a *ranking* problem by learning from human-labeled/edited video-highlight pairs [57, 179, 215]: segments in the manually annotated highlight ought to score more highly than those elsewhere in the original long video. However, such paired data is difficult and expensive to collect, especially for long and unconstrained videos at a large scale.

To circumvent the heavy supervision entailed by collecting video-highlight pairs, we propose a framework to learn highlight detection directly from a large collection of *unlabeled* video. As discussed above, we hypothesize that users tend to be more selective about the content in the shorter videos they upload, whereas their longer videos may be a mix of good and less interesting content. We therefore use the duration of videos as supervision signal. In particular, we propose to learn a scoring function that ranks video segments from shorter videos higher than video segments from longer videos. Since longer

videos could also contain highlight moments, we devise the ranking model to effectively handle noisy ranking data.

**Training data and loss:** Let $D$ denote a set of videos sharing a tag (e.g., *dog show*). We first partition $D$ into three non-overlapping subsets $D = \{D_S, D_L, D_R\}$, where $D_S$ contains shorter videos, $D_L$ contains longer videos, and $D_R$ contains the rest. For example, shorter videos may be less than 15 seconds, and longer ones more than 45 seconds (cf. Sec 3.2). Each video, whether long or short, is broken into uniform length temporal segments.[2]

Let $s_i$ refer to a unique video segment from the dataset, and let $v(s_i)$ denote the video where video segment $s_i$ comes from. The visual feature extracted from segment $s_i$ is $x_i$. Since our goal is to rank video segments from shorter videos higher than those from longer videos, we construct training pairs $(s_i, s_j)$ such that $v(s_i) \in D_s$ and $v(s_j) \in D_L$. We denote the collection of training pairs as $\mathcal{P}$. Since our dataset is large, we sample among all possible pairs, ensuring each video segment is included at least once in the training set. The learning objective consists of the following ranking loss:

$$L(D) = \sum_{(s_i, s_j) \in \mathcal{P}} \max\left(0, 1 - f(x_i) + f(x_j)\right), \tag{3.1}$$

which says we incur a loss every time the longer video's segment scores higher. The function $f$ is a deep convolutional network, detailed below. Note that whereas supervised highlight ranking methods [57, 179, 215] use rank constraints on segments from the *same* video—comparing those inside and outside the true highlight region—our constraints span segments from distinct short and long videos.

[2]We simply break them up uniformly into 2-second segments, though automated temporal segmentation could also be employed [150, 171].

**Learning from noisy pairs:** The formulation thus far assumes that no noise exists and that $D_s$ and $D_L$ only contain segments from highlights and non-highlights, respectively. However, this is not the case when learning from unedited videos: some video segments from long videos can also be highlights, and some short segments need not be highlights. Furthermore, some videos are irrelevant to the hashtags. Therefore, only a subset of our pairs in $\mathcal{P}$ have *valid* ranking constraints $(s_i, s_j)$, i.e., pairs where $s_i$ corresponds to a highlight and $s_j$ corresponds to a non-highlight. Ideally, a ranking model would only learn from valid ranking constraints and ignore the rest. To achieve this without requiring any annotation effort, we introduce binary latent variables $w_{ij}$, $\forall (s_i, s_j) \in \mathcal{P}$ to indicate whether a ranking constraint is valid. We rewrite the learning objective as follows:

$$\begin{aligned} L(D) = & \sum_{(s_i,s_j)\in\mathcal{P}} w_{ij} \;\; \max\left(0, 1 - f(x_i) + f(x_j)\right) \\ & \text{s.t.} \sum_{(s_i,s_j)\in\mathcal{P}} w_{ij} = p|\mathcal{P}|, \quad w_{ij} \in [0,1], \\ & \text{and} \;\; w_{ij} = h(x_i, x_j) \end{aligned} \tag{3.2}$$

where $h$ is a neural network, $|\mathcal{P}|$ is total number of ranking constraints, and $p$ is the anticipated proportion of ranking constraints that are valid. In the spirit of learning with a proportional loss [217], this cap on the total weights assigned to the rank constraints represents a prior for the noise level expected in the labels. For example, training with $p = 0.8$ tells the system that about 80% of the pairs are a priori expected to be valid. The summation of the binary latent variable $w_{ij}$ prevents the trivial solution of assigning $0$ to all the latent variables.

Rather than optimize binary latent selection variables with alternating minimization, we instead use real-valued selection variables, and the function $h(x_i, x_i)$ directly pre-

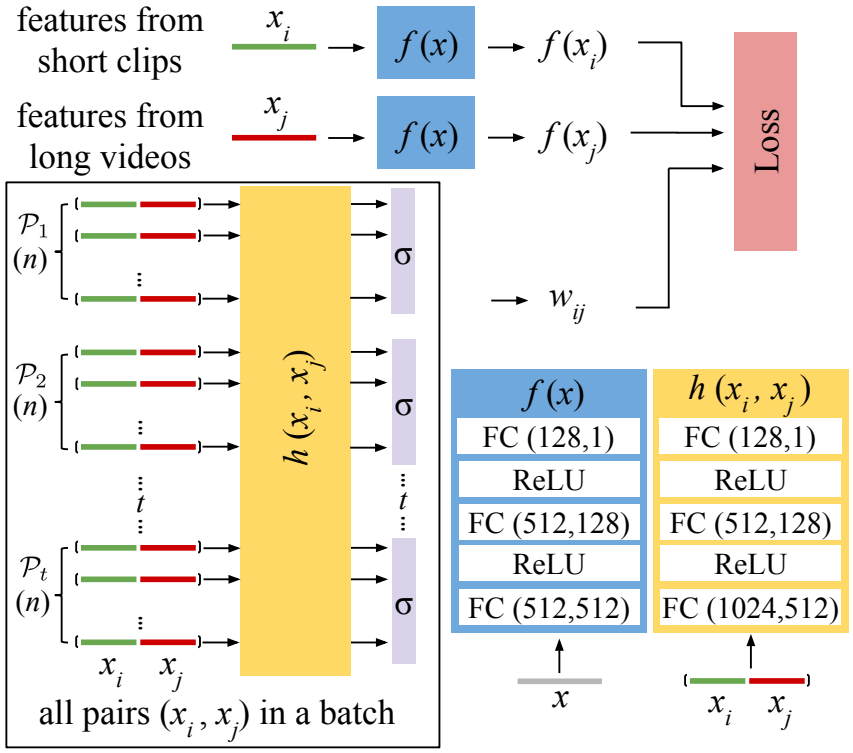


Figure 3.3: Network architecture details of our approach. The batch size is $b$. We group every $n$ instances of training pairs and feed them to a softmax function. Each batch has $t$ such groups ($b = nt$).

dicts those latent variables $w_{ij}$. The advantages are threefold. First, we can simultaneously optimize the ranking function $f$ and the selected training pairs. Second, the latent variable $w_{ij}$ is conditioned on the input features so it can learn whether a ranking constraint is valid as a function of the specific visual input. Third, by relaxing $w_{ij}$ to a continuous variable in the range from 0 to 1, we capture uncertainty about pair validity during training.

Finally, we parameterize the latent variables $w_{ij}$, which provide learned weights for the training samples, and refine our objective to train over batches while enforcing the noise level prior $p$. We split the training data into groups, each of which contains exactly

$n$ pairs. We then require that the latent variable $w_{ij}$ for instances within a group sum up to 1. In particular, let $\mathcal{P}_1, \ldots, \mathcal{P}_m$ be a random split of the set of pairs $\mathcal{P}$ into $m$ groups where each group contains exactly $n$ pairs, then the final loss becomes:

$$\begin{aligned} L(D) = & \sum_{g=1}^{m} \sum_{(s_i,s_j)\in\mathcal{P}_g} \tilde{w}_{ij} \ \max\left(0, 1 - f(x_i) + f(x_j)\right) \\ & \text{s.t.} \sum_{(s_i,s_j)\in\mathcal{P}_g} \tilde{w}_{ij} = \sum_{(s_i,s_j)\in\mathcal{P}_g} \sigma_g(h(x_i, x_j)) = 1, \\ & \tilde{w}_{ij} \in [0, 1], \end{aligned} \tag{3.3}$$

where $\sigma_g$ denotes the softmax function defined over the set of pairs in group $\mathcal{P}_g$. Note that now the group size $n$, together with the softmax, serves to uphold the label noise prior $p$, with $p = \frac{1}{n}$, while allowing a differentiable loss for the selection function $h$. Intuitively, smaller values of $n$ will speed up training at the cost of mistakenly promoting some invalid pairs, whereas larger values of $n$ will be more selective for valid pairs at the cost of slower training. In experiments, we fix $n$ to 8 for all results and datasets.

As $f$ learns from training data, the function $h$ helps $f$ to attend to training pairs that are consistent. Starting with the prior that there are more valid than invalid pairs, it learns to assign low (high) weights to training pairs that violate (satisfy) ranking constraints, respectively.

**Network structure:** We model both $f(x)$ and $h(x_i, x_j)$ with neural networks. We use a 3 hidden layer fully-connected model for $f(x)$. The function $h(x_i, x_j)$ consists of a 3 fully-connected layers, followed by a $n$-way softmax function, as shown in Eq.(3.3). See Fig. 3.3 for network architecture details.

**Video segment feature representation:** To generate features $x_i$ for a segment $s_i$ we use a 3D convolution network [58] with a ResNet-34 [62] backbone pretrained on Kinetics [19].

We use the feature after the pooling of the final convolution layer. Each video segment is thus represented by a feature of 512 dimensions.

**Implementation details:** We implement our model with PyTorch, and optimize with stochastic gradient with momentum for 30 epochs. We use a batch size of 2048 and set the base learning rate to 0.005. We use a weight decay of 0.00005 and a momentum of 0.9. With a single Quadro GP100 gpu, the total feature extraction time for a one-minute-long video is 0.50 s. After extracting video features, the total training time to train a model is one hour for a dataset of 20,000 video clips of total duration 1600 hours. At test time, it takes 0.0003 s to detect highlights in a new one-minute-long video after feature extraction.

## 3.2 Results

We validate our approach for highlight detection and compare to an array of previous methods, focusing especially on those that are unsupervised and domain-specific.

### 3.2.1 Experimental setup

**Datasets and metrics:** After training our model on the Instagram video, we evaluate it on two challenging public video highlight detection datasets: YouTube Highlights [179] and TVSum [171]. YouTube Highlights [179] contains six domain-specific categories: *surfing, skating, skiing, gymnastics, parkour*, and *dog*. Each domain consists of around 100 videos and the total accumulated time is 1430 minutes. TVSum [171] is collected from YouTube using 10 queries and consists of 50 videos in total from domains including *changing vehicle tire, grooming an animal, making sandwich, parade, flash mob gathering*, and others. Since the ground truth annotations in TVSum [171] provide frame-level importance

scores, we first average the frame-level importance scores to obtain the shot-level scores, and then select the top 50% shots (segments) for each video as the human-created summary, following [143, 142]. Finally, the highlights selected by our method are compared with 20 human-created summaries. We report mean average precision (mAP) for both datasets.

**Baselines:** We compare with nine state-of-the-art methods as reported in the literature. Here we organize them based on whether they require shot-level annotation (supervised) or not (unsupervised). Recall that our method is unsupervised and domain-specific, since we use no annotations and compose the pool of training video with tag-based queries.

- **Unsupervised baselines:** We compare with the following unsupervised methods: RRAE [213], MBF [27], KVS [150], CVS [143], SG [132], DeSumNet(DSN) [142], and VESD [15]. We also implement a baseline where we train classifiers (CLA) with our hashtagged Instagram videos. The classifiers use the same network structures (except the last layer is replaced with a $K$-way classification) and video features as our method. We then use the classifier score for highlight detection. CLA can be seen as a deep network variant of KVS [150]. We also implemented k-means and spectral clustering baselines, but found them inferior to the more advanced clustering method [27] reported below.
- **Supervised baselines:** We compare with the latent-SVM approach [179], which trains with human-edited video-highlight pairs, and the Video2GIF approach [57], a domain-agnostic method that trains with human-edited video-GIF pairs. Though these methods require annotations—and ours does not—they are of interest since they also use ranking formulations.

We present results for two variants of our method: **Ours-A**: Our method trained with Instagram data in a domain-*agnostic* way, where we pool training videos from all queried tags. We use a single model for all experiments; **Ours-S**: Our method trained with domain-*specific* Instagram data, where we train a separate highlight detector for each queried tag. For both variants, our method's training data pool is generated entirely automatically and uses no highlight annotations. A training video is in $D_S$ if its duration is between $8$ and $15$ s, and it is in $D_L$ if its duration is between $45$ and $60$ s. We discard all other videos. Performance is stable as long as we keep a large gap for the two cut off thresholds. Our networks typically converge after 20 epochs, and test performance is stable ($\pm 0.5\%$) when we train multiple times with random initializations.

### 3.2.2 Highlight Detection Results

**Results on YouTube Highlights dataset:** Table 3.1 presents the results on YouTube Highlights [179]. All the baseline results are as reported in the authors' original papers. Our domain specific method (Ours-S) performs the best—notably, it is even better than the *supervised* ranking-based methods. Compared to the unsupervised RRAE approach [213], our average gain in mAP is 18.1%. Our method benefits from discriminative training to isolate highlights from non-highlight video segments. Our method also outperforms the CLA approach that is trained on the same dataset as ours, indicating that our advantage is not due to the training data alone. CLA can identify the most discriminative video segments, which may not always be highlights. On average our method outperforms the LSVM approach [179], which is trained with domain-specific manually annotated data. While the supervised methods are good at leveraging high quality training data, they are

| | RRAE (unsup) [213] | GIFs (sup) [57] | LSVM (sup) [179] | CLA (unsup) | Ours-A (unsup) | Ours-S (unsup) |
|---|---|---|---|---|---|---|
| dog | 0.49 | 0.308 | **0.60** | 0.502 | 0.519 | 0.579 |
| gymnast. | 0.35 | 0.335 | 0.41 | 0.217 | **0.435** | 0.417 |
| parkour | 0.50 | 0.540 | 0.61 | 0.309 | 0.650 | **0.670** |
| skating | 0.25 | 0.554 | **0.62** | 0.505 | 0.484 | 0.578 |
| skiing | 0.22 | 0.328 | 0.36 | 0.379 | 0.410 | **0.486** |
| surfing | 0.49 | 0.541 | 0.61 | 0.584 | 0.531 | **0.651** |
| Average | 0.383 | 0.464 | 0.536 | 0.416 | 0.505 | **0.564** |

Table 3.1: Highlight detection results (mAP) on YouTube Highlights [179]. Our method outperforms all the baselines, including the supervised ranking-based methods [179, 57].

also limited by the practical difficulty of securing such data at scale. In contrast, our method leverages large-scale tagged Web video at scale, without manual highlight examples.

Our method trained with domain specific data (Ours-S) performs better than when it is trained in a domain-agnostic way (Ours-A). This is expected since highlights often depend on the domain of interest. Still, our domain-agnostic variant outperforms the domain-agnostic Video2GIF [57], again revealing the benefit of large-scale weakly supervised video for highlight learning.

Fig. 3.4 shows example highlights. Despite not having explicit supervision, our method is able to detect highlight-worthy moments for a range of video types.

**Results on TVSum dataset:** Table 3.2 presents the results on TVSum [171].[3] We focus the comparisons on unsupervised and domain-specific highlight methods. TVSum is a

[3]Results for CVS [143], DeSumNet [142] and VESD [15] are from original papers. All others (MBF [27], KVS [150] and SG [132]) are as reported in [15].

| | MBF [27] | KVS [150] | CVS [143] | SG [132] | DSN [142] | VESD [15] | CLA | Ours-A | Ours-S |
|---|---|---|---|---|---|---|---|---|---|
| Vehicle tire | 0.295 | 0.353 | 0.328 | 0.423 | - | - | 0.294 | 0.449 | **0.559** |
| Vehicle unstuck | 0.357 | 0.441 | 0.413 | 0.472 | - | - | 0.246 | **0.495** | 0.429 |
| Grooming animal | 0.325 | 0.402 | 0.379 | 0.475 | - | - | 0.590 | 0.454 | **0.612** |
| Making sandwich | 0.412 | 0.417 | 0.398 | 0.489 | - | - | 0.433 | 0.537 | **0.540** |
| Parkour | 0.318 | 0.382 | 0.354 | 0.456 | - | - | 0.505 | 0.602 | **0.604** |
| Parade | 0.334 | 0.403 | 0.381 | 0.473 | - | - | 0.491 | **0.530** | 0.475 |
| Flash mob | 0.365 | 0.397 | 0.365 | **0.464** | - | - | 0.430 | 0.384 | 0.432 |
| Beekeeping | 0.313 | 0.342 | 0.326 | 0.417 | - | - | 0.517 | 0.638 | **0.663** |
| Bike tricks | 0.365 | 0.419 | 0.402 | 0.483 | - | - | 0.578 | 0.672 | **0.691** |
| Dog show | 0.357 | 0.394 | 0.378 | 0.466 | - | - | 0.382 | 0.481 | **0.626** |
| Average | 0.345 | 0.398 | 0.372 | 0.462 | 0.424 | 0.423 | 0.447 | 0.524 | **0.563** |

Table 3.2: Highlight detection results (Top-5 mAP score) on TVSum [171]. All methods listed are unsupervised. Our method outperforms all the baselines by a large margin. Entries with "-" mean per-class results not available for that method.

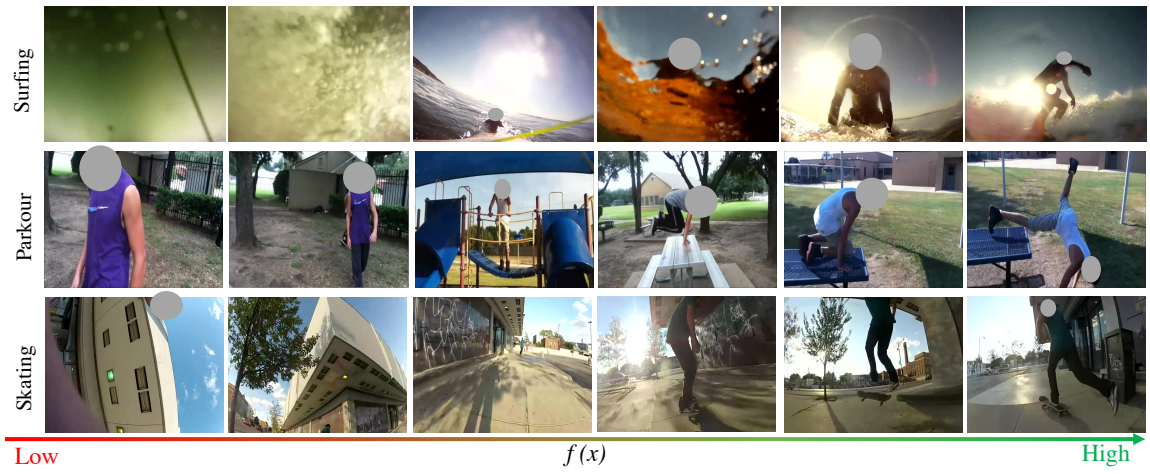


Figure 3.4: Example highlight detection results for the YouTube Highlights dataset [179]. We show our method's predicted ranking from low (left) to high (right) and present one frame for each video segment.

very challenging dataset with diverse videos. Our method outperforms all the baselines by a large margin. In particular, we outperform the next best method SG [132] by 10.1 points, a relative gain of 22%. SG learns to minimize the distance between original videos

and their summaries. The results reinforce the advantage of discriminatively selecting segments that are highlight-worthy versus those that are simply representative. For example, while a close up of a bored dog might be more *representative* in the feature space for dog show videos, a running dog is more likely to be a highlight. Our method trained with domain specific data (Ours-S) again outperforms our method trained in a domain-agnostic way (Ours-A).

**Instagram vs. YouTube for training:** Curious whether an existing large-scale collection of Web video might serve equally well as training data for our approach, we also trained our model on videos from YouTube8M [3]. Training on 6,000 to 26,000 videos per domain from YouTube8M, we found that results were inferior to those obtained with the Instagram data. We attribute this to two factors: 1) the YouTube-8M was explicitly curated to have fairly uniform-length "longer" (120-500 s) clips [3], which severely mutes our key duration signal, and 2) users sharing videos on Instagram may do so to share "moments" with family and friends, whereas YouTube seems to attract a wider variety of purposes (e.g., instructional videos, edited films, etc.) which may also weaken the duration signal.

### 3.2.3 Ablation Studies

Next we present an ablation study. All the methods are trained with domain-specific data. We compare our full method (Ours-S) with two variants: 1) **Ranking-D**, which treats all the ranking constraints as valid and trains the ranking function without the latent variables. This is similar to existing supervised highlight detection methods [57, 215]. 2) **Ranking-EM**, which introduces a binary latent variable and optimizes the ranking function and binary latent selection variable in an alternating manner with EM,

| Dataset | Ranking-D | Ranking-EM | Ours-S |
|---|---|---|---|
| YouTube | 0.425 | 0.458 | **0.564** |
| TVSum | 0.400 | 0.444 | **0.563** |

Table 3.3: Accuracy (mAP) in ablation study.

similar to [179]. Note that unlike our approach, here the binary latent variable is discrete and it is not conditioned on the input.

Table 3.3 shows the results. Our full method outperforms the alternative variants. In particular, our average gain in mAP over *Ranking-D* is 13.9% and 16.3% for Youtube and TVSum, respectively. This supports our hypothesis that ranking constraints obtained by sampling training pairs $(s_i, s_j)$ such that $v(s_i) \in D_s$ and $v(s_j) \in D_L$ are indeed noisy. By modeling the noise and introducing the latent selection variable, our proposed method improves performance significantly. Our method also significantly outperforms *Ranking-EM*, which also models noise in the training samples. In contrast to *Ranking-EM*, our method directly predicts the latent selection variable from input. In addition, we benefit from joint optimization and relaxation of the latent selection variable, which accounts for uncertainty.

Fig. 3.6 shows highlight detection accuracy as a function of training set size. We report this ablation for YouTube Highlights only, since the videos sharing tags with some TVSum categories max out at 24,000. As we increase the number of videos in each domain, accuracy also improves. The performance improves significantly ($6.5\%$ for Ours-S and $3.7\%$ for Ours-A) when the training data is increased from $1,000$ to $10,000$ in each domain, then starts to plateau.

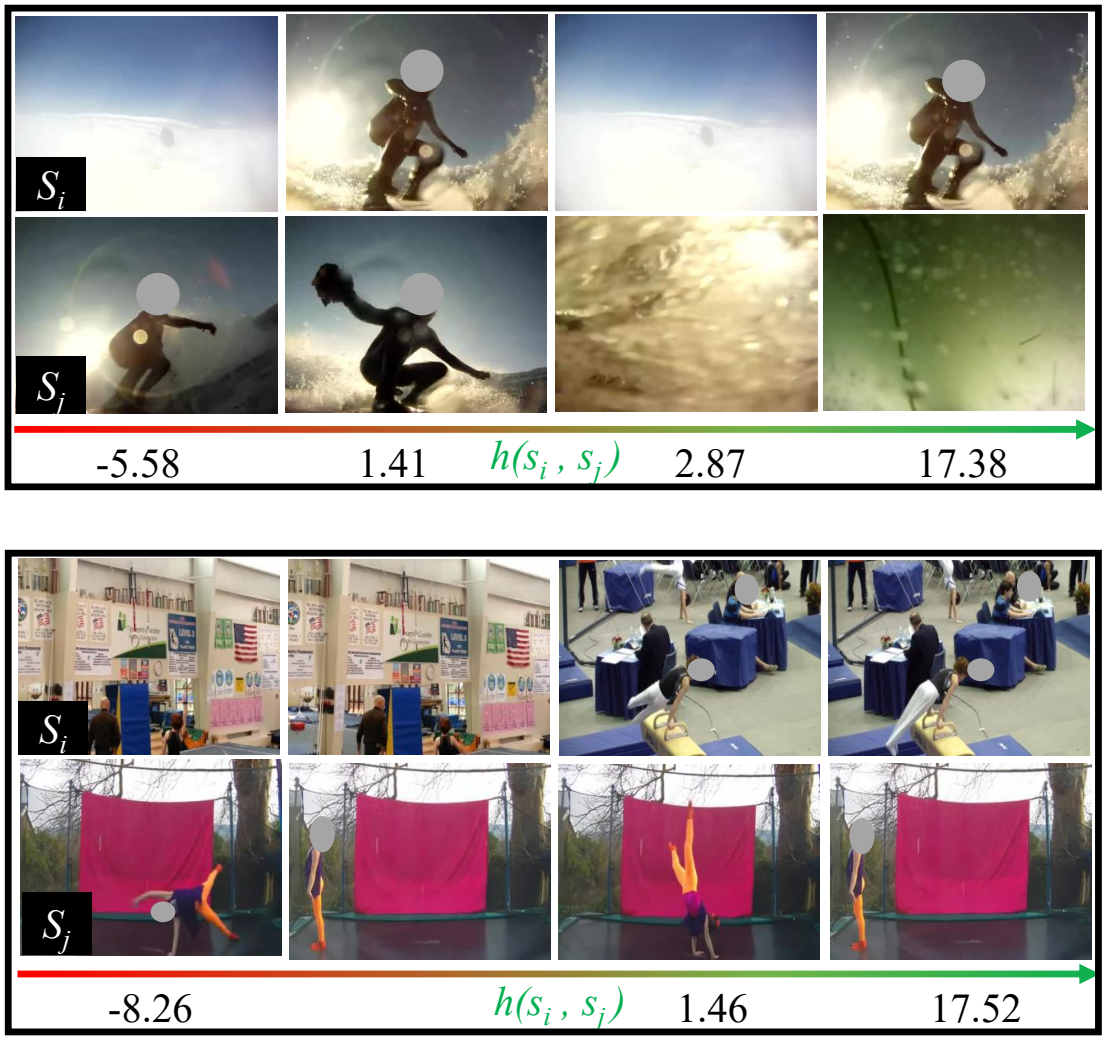


Figure 3.5: Predicted latent values (before softmax) for video segment pairs from YouTube Highlights. Higher latent value indicates higher likelihood to be a valid pair. The predicted latent value is high if $s_i$ (top row) is a highlight and $s_j$ (bottom row) is a non-highlight.

### 3.2.4 Understanding Learning from Duration

Finally, we investigate what each component of our model has learned from video duration. First, we test whether our model can distinguish segments from shorter videos

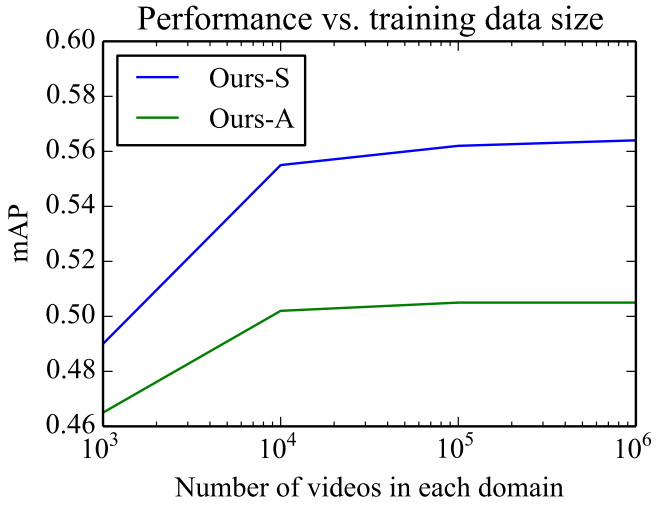


Figure 3.6: Accuracy vs. training set size on YouTube [179].

versus segments from longer videos. This is essentially a validation of the main training objective, without the additional layer of highlight accuracy. We train our model and reserve $20\%$ novel videos for testing. Each test pair consists of a randomly sampled video segment from a novel shorter video and one from a novel longer video. We use $f(x)$ to score each segment and report the percentage of successfully ranked pairs. Without the proposed latent weight prediction, our model achieves a $58.2\%$ successful ranking rate. Since it is higher than chance ($50\%$), this verifies our hypothesis that the distributions of the two video sources are different. However, the relatively low rate also indicates that the training data is very noisy. After we weight the test video pairs with $h(x_i, x_j)$, we achieve a $87.2\%$ success rate. The accuracy improves significantly because our latent value prediction function $h(x_i, x_j)$ identifies discriminative pairs.

Second, we examine video segment pairs constructed from the YouTube Highlights dataset alongside their predicted latent values (before softmax). See Fig. 3.5. Higher

latent values indicate higher likelihood to be a valid pair. Video segments ($s_i$) from the top row are supposed to be ranked higher than video segments ($s_j$) from the second row. When $s_i$ corresponds to a highlight segment and $s_j$ a non-highlight segment, the predicted latent value is high (last columns in each block). Conversely, the predicted latent value is extremely low when $s_i$ corresponds to a non-highlight segment and $s_j$ a highlight segment (first column in each block). Note if we group all the examples in each block into a softmax, all the training examples except the last will have negligible weights in the loss. This demonstrates that the learned $h(x_i, x_j)$ can indeed identify valid training pairs, and is essential to handle noise in training.

## 3.3 Summary

In this chapter, I presented a novel framework for video highlight detection. The key insight is that video segments from shorter user-generated videos are more likely to be highlights than those from longer videos, since users tend to be more selective about the content when capturing shorter videos. Leveraging this insight, I introduce a novel ranking framework that prefers segments from shorter videos, while properly accounting for the inherent noise in the (unlabeled) training data. In experiments on two challenging public video highlight detection benchmarks, the proposed method substantially improves the state-of-the-art for unsupervised highlight detection.

My proposed method learns to ignore invalid training samples from noisy training data. However, the proposed method cannot effectively distinguish invalid training samples from valid but hard training samples. One possible solution is to introduce curriculum learning to gradually recover hard training samples. Our current work also assumes train-

ing videos and test videos are from the same category domain. Future work will explore how to combine multiple pre-trained domain-specific highlight detectors for test videos in novel domains. Since the proposed method is robust to label noise and only requires weakly-labeled annotations like hashtags, it has the potential to scale to an unprecedented number of domains, possibly utilizing predefined or learned taxonomies for reusing parts of the model.

While the first component of my thesis addresses the question of finding video highlights in the form of short video clips, the second component of my thesis explores how to find the best moments in the form of keyframes. In the next chapter, I consider how to detect *"snap points"* in unedited egocentric videos.

# Chapter 4

# Detecting Snap Points in Egocentric Video with a Web Photo Prior

[1] In the previous chapter, I have presented how to find the best moments in unedited videos in terms of *short video clips*. In this chapter, I consider the problem of finding the best moments in videos in terms of *keyframes*. In particular, I address the following question: can a vision system predict *"snap points"* in unedited egocentric video—that is, those frames that look like intentionally taken photos?

To get some intuition for the task, consider the images in Figure 4.1. Can you guess which row of photos was sampled from a wearable camera, and which was sampled from photos posted on Flickr? Note that subject matter itself is not always the telling cue; in fact, there is some overlap in content between the top and the bottom rows. Nonetheless, we suspect it is easy for the reader to detect that a head-mounted camera grabbed the shots in the first row, whereas a human photographer purposefully composed the shots in the second row. These distinctions suggest that it may be possible to learn the generic properties of an image that indicate it is well-composed, independent of the literal content.

[1]The work in this chapter was supervised by Prof. Kristen Grauman and originally published in: "Detecting Snap Points in Egocentric Video with a Web Photo Prior ". Bo Xiong and Kristen Grauman. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, September 2014. An expanded article appeared in "Intentional Photos from an Unintentional Photographer: Detecting Snap Points in Egocentric Video with a Web Photo Prior". Bo Xiong and Kristen Grauman. In Mobile Cloud Visual

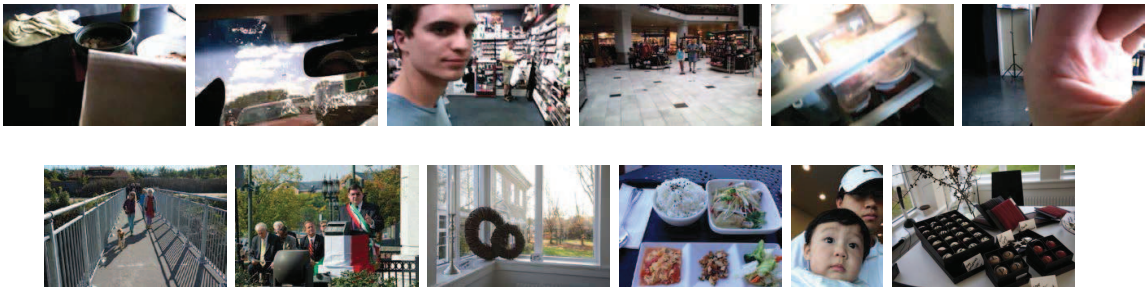

Figure 4.1: Can you tell which row of photos came from an egocentric camera?

While this anecdotal sample suggests detecting snap points may be feasible, there are several challenges. First, egocentric video contains a wide variety of scene types, activities, and actors. This is certainly true for human camera wearers going about daily life activities, and it will be increasingly true for mobile robots that freely explore novel environments. Accordingly, a snap point detector needs to be largely domain invariant and generalize across varied subject matter. Secondly, an optimal snap point is likely to differ in subtle ways from its less-good temporal neighbors, i.e., two frames may be similar in content but distinct in terms of snap point quality. That means that cues beyond the standard texture and color favorites may be necessary. Finally, and most importantly, while it would be convenient to think of the problem in discriminative terms (e.g., training a snap point vs. non-snap point classifier), it is burdensome to obtain adequate and unbiased labeled data.

To address the above challenges, I introduce an approach to detect snap points from egocentric video that requires no human annotations. The main idea is to construct a generative model of what human-taken photos look like by sampling images posted on

---

Media Computing, November 2015.

the Web. I also propose two applications of snap point prediction.

I first describe my approach for detecting snap points in Section 4.1, and then show results in Section 4.2. Please see Section 2.3 and 2.4 for prior work related to egocentric videos and leveraging Web images.

## 4.1 Approach

Our goal is to detect snap points, which are those frames within a continuous egocentric video that appear as if they were composed with intention, as opposed to merely observed by the person wearing the camera. In traditional camera-user relationships, this "trigger" is left entirely to the human user. In the wearable camera-user relationship, however, the beauty of being hands-free and always-on should be that the user no longer has to interrupt the flow of his activity to snap a photo. Notably, whether a moment in time is photoworthy is only partially driven by the subject matter in view. The way the photo is composed is similarly important, as is well-understood by professional photographers and intuitively known by everyday camera users.

We take a non-parametric, data-driven approach to learn what snap points look like. First, we gather unlabeled Web photos to build the prior (Sec. 4.1.1), and extract image descriptors that capture cues for composition and intention (Sec. 4.1.2). Then, we estimate a domain-invariant feature space connecting the Web and ego sources (Sec. 4.1.3). Finally, given a novel egocentric video frame, we predict how well it agrees with the prior in the adapted feature space (Sec. 4.1.4). To illustrate the utility of snap points, we also explore applications for object detection and keyframe selection (Sec. 4.1.5).

Figure 4.2: Example images from the SUN dataset [201].

Section 4.2.1 will discuss how we systematically gather ground truth labels for snap points using human judgments, which is necessary to evaluate our method, but, critically, is *not* used to train it.

### 4.1.1 Building the Web Photo Prior

Faced with the task of predicting whether a video frame is a snap point or not, an appealing solution might be to train a discriminative classifier using manually labeled exemplars. Such an approach has proven successful for learning other high-level image properties, like aesthetics and interestingness [32, 54], quality [88], canonical views [90], or memorability [72]. This is thanks in part to the availability of relevant meta-data for such problems: users on community photo albums manually score images for visual appeal [32, 88], and users uploading ads online manually tag the object of interest [90].

However, this familiar paradigm is problematic for snap points. Photos that appear human-taken exhibit vast variations in appearance, since they may have almost arbitrary content. This suggests that large scale annotations would be necessary to cover the space. Furthermore, snap points must be isolated within ongoing egocentric video. This means

that labeling *negatives* is tedious—each frame must be viewed and judged in order to obtain clean labels.

Instead, we devise an approach that leverages *unlabeled* images to learn snap points. The idea is to build a prior distribution using a large-scale repository of Web photos uploaded by human photographers. Such photos are by definition human-taken, span a variety of contexts, and (by virtue of being chosen for upload) have an enhanced element of *intention*. We use these photos as a generative model of snap points.

We select the SUN Database as our Web photo source [201], which originates from Internet search for hundreds of scene category names. Our choice is motivated by two main factors. First, the diversity of photos is high—899 categories in all drawn from 70K WordNet terms—and there are many of them ($130K$). Second, its scope is fairly well-matched with wearable camera data. Human- or robot-worn cameras observe a variety of daily life scenes and activities, as well as interactions with other people. SUN covers not just locations, but settings that satisfy "I am in a *place*, let's go to a *place*" [201], which includes many scene-specific interactions, such as shopping at a pawnshop, visiting an optician, driving in a car, etc. See Figure 4.2.

### 4.1.2 Image Descriptors for Intentional Cues

To represent each image, we designate descriptors to capture intentional composition effects.

**Motion:** Non-snap points will often occur when a camera wearer is moving quickly, or turning his head abruptly. We therefore extract a descriptor to summarize

*motion blur*, using the blurriness estimate of [29].[2]

**Composition:** Snap points also reflect intentional framing effects by the human photographer. This leads to spatial regularity in the main line structures in the image—e.g., the horizon in an outdoor photo, buildings in a city scene, the table surface in a restaurant—which will tend to align with the image axes. Thus, we extract a *line alignment* feature: we detect line segments using the method in [103], then record a histogram of their orientations with 32 uniformly spaced bins. To capture framing via the 3D structure layout, we employ the geometric class probability map [66]. We also extract GIST, HOG, self-similarity (SSIM), and dense SIFT, all of which capture alignment of interior textures, beyond the strong line segments. An accelerometer, when available, could also help gauge coarse alignment; however, these descriptors offer a fine-grained visual measure helpful for subtle snap point distinctions.

**Feature combination:** For all features but line alignment, we use code and default parameters provided by [201]. We reduce the dimensionality of each feature using PCA to compactly capture 90% of its total variance. We then standardize each dimension to $(\mu = 0, \sigma = 1)$ and concatenate the reduced descriptors to form a single vector feature space $X$, which we use in what follows.

Alternatively, we could use features from a deep neural network [168, 62] pre-trained for ImageNet classification [159].

[2]We also explored flow-based motion features, but found their information to be subsumed by blur features computable from individual frames.

### 4.1.3 Adapting from the Web to the Egocentric Domain

While we expect egocentric video snap points to agree with the Web photo prior along many of these factors, there is also an inherent mismatch between the statistics of the two domains. Egocentric video is typically captured at low-resolution with modest quality lenses, while online photos (e.g., on Flickr) are often uploaded at high resolution from high quality cameras.

Therefore, we establish a domain-invariant feature space connecting the two sources. Given unlabeled Web photos and egocentric frames, we first compute a subspace for each using PCA. Then, we recover a series of intermediate subspaces that gradually transition from the "source" Web subspace to the "target" egocentric subspace. We use the algorithm of [48] since it requires no labeled target data and is kernel-based.

Let $x_i$, $x_j \in X$ denote image descriptors for a Web image $i$ and egocentric frame $j$. The idea is to compute the projections of an input $x_i$ on a subspace $\phi(t)$, for all $t \in [0, 1]$ along the geodesic path connecting the source and target subspaces in a Grassmann manifold. Values of $t$ closer to 0 correspond to subspaces closer to the Web photo prior; values of $t$ closer to 1 correspond to those more similar to egocentric video frames. The infinite set of projections is achieved implicitly via the geodesic flow kernel [48] (GFK):

$$K_{GFK}(x_i, x_j) = \langle z_i^\infty, z_j^\infty \rangle = \int_0^1 (\phi(t)^T x_i)^T (\phi(t)^T x_j) dt, \tag{4.1}$$

where $z_i^\infty$ and $z_j^\infty$ denote the infinite-dimensional features concatenating all projections of $x_i$ and $x_j$ along the geodesic path.

Intuitively, this representation lets the two slightly mismatched domains (Web and ego) "meet in the middle" in a common feature space, letting us measure similarity

between both kinds of data without being overly influenced by their superficial resolution/sensor differences.

### 4.1.4 Predicting Snap Points

With the Web prior, image features, and similarity measure in hand, we can now estimate how well a novel egocentric video frame agrees with our prior. We take a simple data-driven approach. We treat the pool of Web photos as a non-parametric distribution, then estimate the likelihood of the novel ego frame under that distribution based on its nearest neighbors' distances.

Let $W = \{x_1^w, \ldots, x_N^w\}$ denote the $N$ Web photo descriptors, and let $x^e$ denote a novel egocentric video frame's descriptor. We retrieve the $k$ nearest examples $\{x_{n_1}^w, \ldots, x_{n_k}^w\} \subset W$, i.e., those $k$ photos that have the highest GFK kernel values when compared to $x^e$.[3] Then we predict the snap point confidence for $x^e$:

$$S(x^e) = \sum_{j=1}^{k} K_{GFK}(x^e, x_{n_j}^w), \tag{4.2}$$

where higher values of $S(x^e)$ indicate the test frame is more likely to be human-taken. For our dataset of $N = 130K$ images, similarity search is fairly speedy (0.01 seconds per test case in Matlab), and could easily be scaled for much larger $N$ using hashing or kd-tree techniques.

This model follows in the spirit of prior data-driven methods for alternative tasks, e.g., [162, 187, 60, 118], the premise being to keep the learning simple and let the data

[3] We use $k = 60$ based on preliminary visual inspection, and found results were similar for other $k$ values of similar order ($k \in [30, 120]$).

speak for itself. However, our approach is label-free, as all training examples are (implicitly) positives, whereas the past methods assume at least weak meta-data annotations.

While simple, our strategy is very effective in practice. In fact, we explored a number of more complex alternatives—one-class SVMs, Gaussian mixture models, non-linear manifold embeddings—but found them to be similar or inferior to the neighbor-based approach. The relatively lightweight computation is a virtue given our eventual goal to make snap point decisions onboard a wearable device.

### 4.1.5 Leveraging Snap Points for Egocentric Video Analysis

Filtering egocentric video down to a small number of probable snap points has many potential applications. We are especially interested in how they can bolster object detection and keyframe selection. We next devise strategies for each task that leverage the above predictions $S(x^e)$.

**Object detection:** In the object recognition literature, it is already disheartening how poorly detectors trained on one dataset tend to generalize to another [186]. Unfortunately, things are only worse if one attempts to apply those same detectors on egocentric video. Why is there such a gap? Precisely because today's object detectors are learned from human-taken photos, whereas egocentric data on wearable cameras—or mobile robots—consist of very few frames that match those statistics. For example, a person detector on PASCAL VOC trained with Flickr photos expects to see people in similarly composed photos, but only a fraction of egocentric video frames will be consistent and thus detectable.

Our idea is to use snap points to predict those frames where a standard object

detector (trained on human-taken images) will be most trustworthy. This way, we can improve precision; the detector will avoid being misled by incidental patterns in non-snap point frames. We implement the idea as follows, using the DPM as an off-the-shelf detector.[4] We score each test ego-frame by $S(x^e)$, then keep all object detections in those frames scoring above a threshold $\tau$. We set $\tau$ as 30% of the average distance between the Web prior images and egocentric snap points. For the remaining frames, we eliminate any detections (i.e., flatten the DPM confidence to 0) that fall below the confidence threshold in the standard DPM pipeline [42]. In effect, we turn the object detector "on" only when it has high chance of success.

**Keyframe selection:** As a second application, we use snap points to create keyframe summaries of egocentric video. The goal is to take hours of wearable data and automatically generate a visual storyboard that captures key events. We implement a simple selection strategy. First, we identify temporal event segments using the color- and time-based grouping method described in [107], which finds chunks of frames likely to belong to the same physical location or scene. Then, for each such event, we select the frame most confidently scored as a snap point.

Our intent is to see if snap points, by identifying frames that look intentional, can help distill the main events in hours of uncontrolled wearable camera data. Our implementation is a proof of concept to demonstrate snap points' utility. We are not claiming a new keyframe selection strategy, a problem studied in depth in prior work [120, 122, 107, 128].

[4] `http://www.cs.berkeley.edu/~rbg/latent/`

## 4.2 Results

### 4.2.1 Datasets and Collecting Ground Truth Snap Points

**Datasets:** We use two egocentric datasets. The first is the publicly available UT Egocentric Dataset (**Ego**)[5], which consists of four videos of 3-5 hours each, captured with a head-mounted camera by four people doing unscripted daily life activities (eating, working, shopping, driving, etc.). The second is a mobile robot dataset (**Robot**) newly collected for this project. In collaboration with our robotics colleagues, we used a wheeled robot to take a 25 minute video both indoors and outdoors on campus (coffee shops, buildings, streets, pedestrians, etc.). Its camera moves constantly from left to right, pauses, then rotates back in order to cover a wide range of viewpoints.

Both the human and robot datasets represent incidentally captured video from always-on, dynamic cameras and unscripted activity. We found other existing ego collections less suited to our goals, either due to their focus on a controlled environment with limited activity (e.g., making food in a kitchen [41, 114])) or their use of chest-mounted or fisheye lens cameras [149, 40], which do not share the point of view of intentional hand-held photos.

**Ground truth:** Our method requires no labeled data for learning: it needs only to populate the Web prior with human-taken photos. However, to *evaluate* our method, it is necessary to have ground truth human judgments about which ego-frames are snap points. The following describes our crowdsourced annotation strategy to get reliable ground truth.

We created a "magic camera" scenario to help MTurk annotators understand the

[5] `http://vision.cs.utexas.edu/projects/egocentric_data`

definition of snap points. Their instructions were as follows: *Suppose you are creating a visual diary out of photos. You have a portable camera that you carry all day long, in order to capture everyday moments of your daily life. ... Unfortunately, your magic camera can also trigger itself from time to time to take random pictures, even while you are holding the camera. At the end of the day, all pictures, both the ones you took intentionally and the ones accidentally taken by the camera, are mixed together.* ***Your task is to distinguish the pictures that you took intentionally from the rest of pictures that were accidentally taken by your camera.***

Workers were required to rate each image into one of four categories: (a) very confidently intentional, (b) somewhat confident intentional, (c) somewhat confident accidental, and (d) very confident accidental. Since the task can be ambiguous and subjective, we issued each image to 5 distinct workers. We obtained labels for 10,000 frames in the Ego data and 2,000 frames in the Robot data, sampled at random.

We establish confidence-rated ground truth as follows. Every time a frame receives a rating of category (a), (b), (c), or (d) from any of the 5 workers, it receives 5, 2, -1, -2 points, respectively. This lets us rank all ground truth examples by their true snap point strength. To alternatively map them to binary ground truth, we threshold a frame's total score: more than 10 points is deemed intentional, otherwise it is accidental. Annotators found 14% of the Ego frames and 23% of the Robot frames to be snap points, respectively. The total MTurk cost was about $500.

We experiment on the 2 datasets described above, Ego and Robot, which together comprise 17.5 hours of video. Since no existing methods perform snap point detection, we define several **baselines** for comparison:

- **Saliency [123]**: uses the CRF-based saliency method of [123] to score an image. This baseline reflects that people tend to compose images with a salient object in the center. We use the implementation of [32], and use the CRF's log probability output as the snap point confidence.

- **Blurriness [29]**: uses the blur estimates of [29] to score an image. It reflects that intentionally taken images tend to lack motion blur. Note, blur is also used as a feature by our method; here we isolate how much it would solve the task if used on its own, with no Web prior.

- **People likelihood**: uses a person detector to rank each frame by how likely it is to contain one or more people. We use the max output of the DPM [42] detector. The intuition is people tend to take images of their family and friends to capture meaningful moments, and as a result, many human-taken images contain people. In fact, this baseline also implicitly captures how well-composed the image is, since the DPM is biased to trigger when people are clear and unoccluded in a frame.

- **Discriminative SVM**: uses a RBF kernel SVM trained with the ground truth snap points/non-snap points in the Ego data. We run it with a leave-one-camera-wearer-out protocol, training on 3 of the Ego videos and testing on the 4th. This baseline lets us analyze the power of the unlabeled Web prior compared to a standard discriminative method. Note, it requires substantially more training effort than our approach.

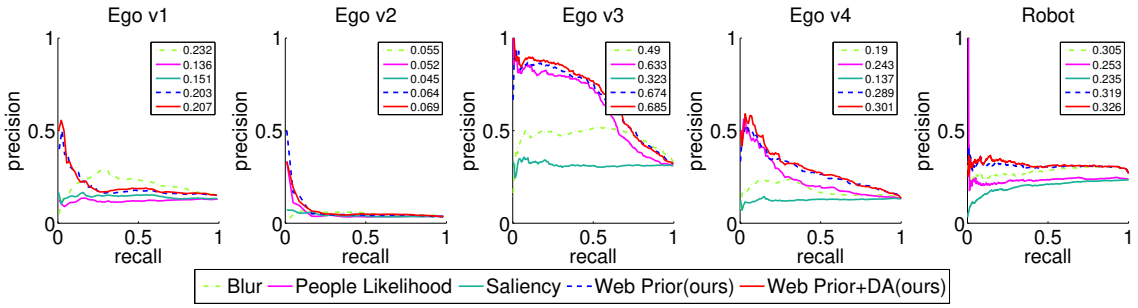


Figure 4.3: Snap point detection precision/recall on the four Ego videos (left) and the Robot video (right). Numbers in legend denote mAP. Best viewed in color.

### 4.2.2 Snap Point Accuracy

First, we quantify how accurately our method predicts snap points. Figure 4.3 shows the precision-recall curves for our method and the three unsupervised baselines (saliency, blurriness, people likelihood). Table 4.1 shows the accuracy in terms of two standard rank quality metrics, Spearman's correlation $\rho$ and Kendall's $\tau$. While the precision-recall plots compare predictions against the binarized ground truth, these metrics compare the full orderings of the confidence-valued predictions against the raw MTurk annotators' ground truth scores (cf. Sec. 4.2.1). They capture that even for two positive intentional images, one might look better than the other to human judges. We show results for our method with and without the domain adaptation (DA) step.

Overall, our method outperforms the baselines. Notably, the same prior succeeds for both the human-worn and robot-worn cameras. Using both the Web prior and DA gives best results, indicating the value of establishing a domain-invariant feature space to connect the Web and ego data.

On Ego video 4 (v4), our method is especially strong, about a factor of 2 better

| Methods | Ego v1 | | Ego v2 | | Ego v3 | | Ego v4 | | Robot | |
|---|---|---|---|---|---|---|---|---|---|---|
| rank coefficient | $\rho$ | $\tau$ | $\rho$ | $\tau$ | $\rho$ | $\tau$ | $\rho$ | $\tau$ | $\rho$ | $\tau$ |
| Blurriness | **0.347** | **0.249** | 0.136 | 0.094 | 0.479 | 0.334 | 0.2342 | 0.162 | 0.508 | 0.352 |
| People Likelihood | 0.002 | 0 | -0.015 | -0.011 | 0.409 | 0.289 | 0.190 | 0.131 | 0.198 | 0.134 |
| Saliency | 0.027 | 0.019 | 0.008 | 0.005 | 0.016 | 0.011 | -0.021 | -0.014 | -0.086 | -0.058 |
| Web Prior (Ours) | 0.321 | 0.223 | 0.144 | 0.100 | **0.504** | **0.355** | **0.452** | 0.317 | 0.530 | 0.373 |
| Web Prior+DA (Ours) | 0.343 | 0.239 | **0.179** | **0.124** | 0.501 | 0.353 | **0.452** | **0.318** | **0.537** | **0.379** |

Table 4.1: Snap point ranking accuracy (higher rank correlations are better).

than the nearest competing baseline (Blur). On v2, mAP is very low for all methods, since v2 has very few true positives (only 3% of its frames, compared to 14% on average for Ego). Still, we see stronger ranking accuracy with our Web prior and DA. On v3, People Likelihood fares much better than it does on all other videos, likely because v3 happens to contain many frames with nice portraits. On the Robot data, however, it breaks down, likely because of the increased viewpoint irregularity and infrequency of people.

While our method is nearly always better than the baselines, on v1 Blur is similar in ranking metrics and achieves higher precision for higher recall rates. This is likely due to v1's emphasis on scenes with one big object, like a bowl or tablet, as the camera wearer shops and cooks. The SUN Web prior has less close-up object-centric images; this suggests we could improve our prior by increasing the coverage of object-centric photos, e.g., with ImageNet-style photos.

Figure 4.4 shows examples of images among those our method ranks most confidently (top) and least confidently (bottom) as snap points, for both datasets. We see that its predictions capture the desired effects. Snap points, regardless of their content, do appear intentional, whereas non-snap points look accidental.

Figure 4.6 (left) examines the effectiveness of each feature we employ, were we to


Figure 4.4: Frames our method rates as likely (top) or unlikely (bottom) snap points.

take them individually. We see that each one has something to contribute, though they are best in combination (Fig. 4.3). HOG on Ego is exceptionally strong. This is in spite of the fact that the exact locations visited by the Ego camera wearers are almost certainly disjoint from those that happen to be in the Web prior. This indicates the prior is broad enough to capture the diversity in appearance of everyday environments.

All baselines so far required no labeled images, same as our approach. Next we compare to a discriminative approach that uses manually labeled frames to train a snap point classifier. Figure 4.5 shows the results, as a function of the amount of labeled data. We give the SVM labeled frames from the held-out Ego videos. (We do not run it for the Robot data, since the only available labels are scene-specific; it's not possible to run the leave-one-camera-wearer-out protocol.) *Despite learning without any explicit labels*, our method generally outperforms the discriminative SVM. The discriminative approach

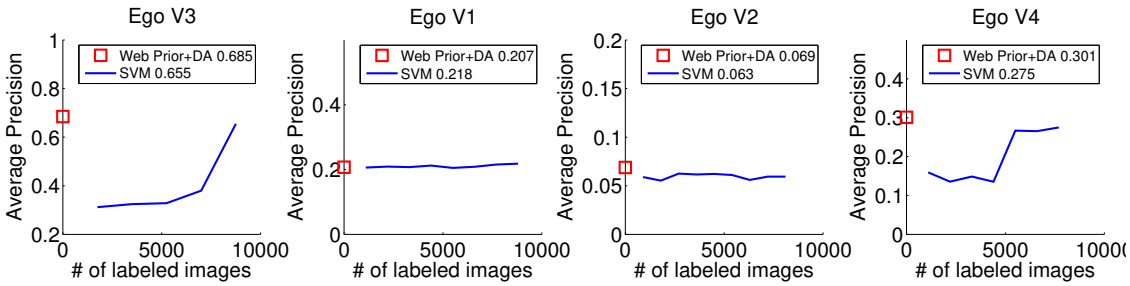


Figure 4.5: Comparison to supervised baseline. SVM's mAP (legend) uses *all* labeled data.

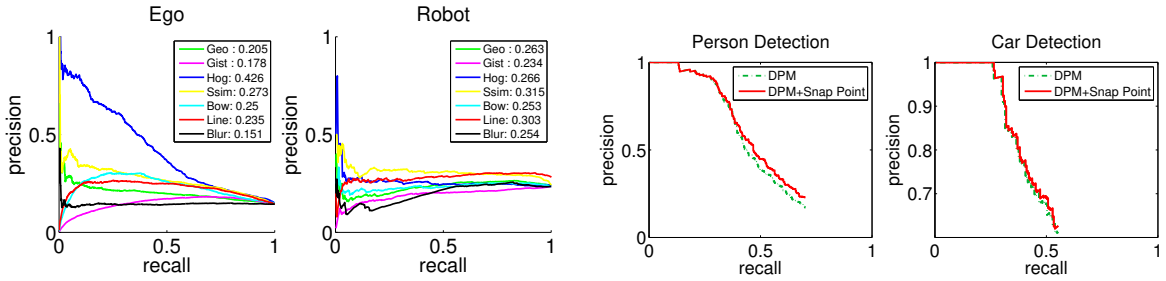


Figure 4.6: Left: Accuracy per feature if used in isolation. Right: Snap points boost precision for an off-the-shelf object detector by focusing on frames that look human-taken.

requires thousands of hand-labeled frames to come close to our method's accuracy in most cases. This is a good sign: while expanding the Web prior is nearly free, expanding the labeled data is expensive and tedious. In fact, if anything, Figure 4.5 is an optimistic portrayal of the SVM baseline. That's because both the training and testing data are captured on the very same camera; in general scenarios, one would not be able to count on this benefit.

The results above are essential to validate our main idea of snap point detection with a Web prior. Next we provide proof of concept results to illustrate the utility of snap points for practical applications.

### 4.2.3 Object Detection Application

Today's object detection systems are trained thoroughly on human-taken images—for example, using labeled data from PASCAL VOC or ImageNet. This naturally makes them best suited to run on human-taken images at test time. Our data statistics suggest only 10% to 15% of egocentric frames may fit this bill. Thus, using the method defined in Sec. 4.1.5, we aim to use snap points to boost object detection precision.

We collected ground truth person and car bounding boxes for the Ego data via DrawMe [202]. Since we could not afford to have all 17.5 hours of video labeled, we sampled the labeled set to cover 50%-50% snap points and non-snap points. We obtained labels for 1000 and 200 frames for people and cars, respectively (cars are more rare in the videos).

Figure 4.6 (right) shows the results, using the PASCAL detection criterion. We see that snap points improve the precision of the standard DPM detector, since they let us ignore frames where the detector is not trustworthy. Of course, this comes at the cost of some recall at the tails. This seems like a good trade-off for detection in video, particularly, since one could anchor object tracks using these confident predictions to make up the recall.

### 4.2.4 Keyframe Selection Application

Keyframe or "storyboard" summaries are an appealing way to peruse long egocentric video, to quickly get the gist of what was seen. Such summaries enable novel interfaces to let a user "zoom-in" on time intervals that appear most relevant. As a final proof-of-concept result, we apply snap points for keyframe selection, using the method

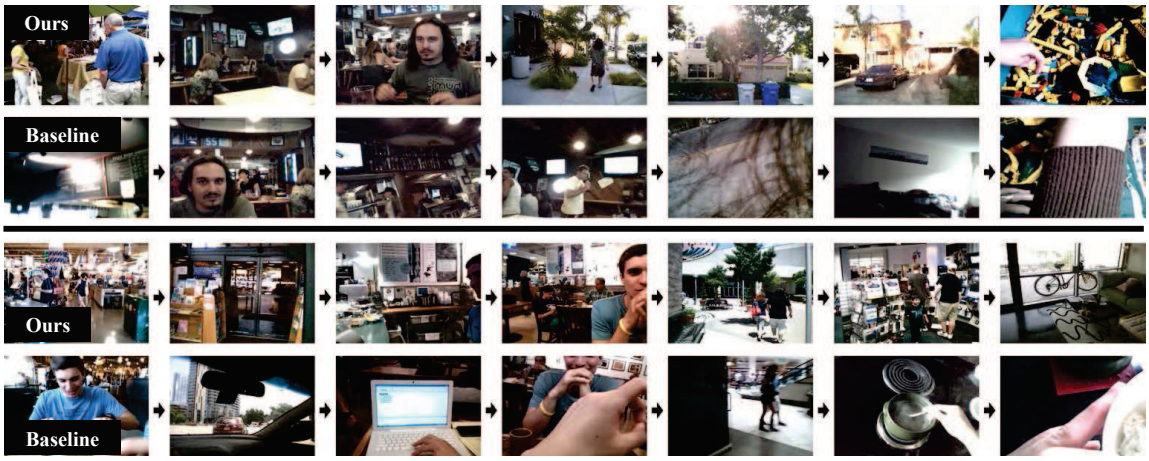


Figure 4.7: Example keyframe selections for two 4-hour Ego videos. In each example, top row shows snap point result, bottom shows result using only event segmentation.

defined in Sec. 4.1.5.

Figure 4.7 shows example results on Ego, where the average event length is 30 min. Keyframe selection requires subjective evaluation; we have no ground truth for quantitative evaluation. We present our results alongside a baseline that uses the exact same event segmentation as [107] (cf. Sec. 4.1.5), but selects each event's frame at random instead of prioritizing snap points. We see the snap point-based summaries contain well-composed images for each event. The baseline, while seeing the same events, uses haphazard shots that do not look intentionally taken.

## 4.3 Summary

In this chapter, I showed how to predict frames that look like intentionally taken photos from unedited egocentric videos. The main idea is to construct a generative model of what human-taken photos look like by sampling images posted on the Web. Despite

learning without any explicit labels, our proposed generative model outperforms discriminative baselines trained with labeled data.

My proposed method does not consider features from a deep neural network [168, 62] pre-trained for ImageNet classification [159]. Augmenting the proposed method with deep features could potential improve performance. In addition, feature extraction is currently a computation bottleneck for the proposed method.

While the second component of my thesis addresses the question of which moments in *time* constitute the best composed photos, the third component of my thesis explores which regions in *space* are most central to a photo or video. In the next chapter, I consider generic foreground object segmentation problem for images and videos.

# Chapter 5

# Pixel Objectness: Learning to Segment Generic Objects in Images and Videos

[1] In this chapter, I introduce a novel approach to automatically segment foreground objects in images and videos. Identifying key objects is an important intermediate step for automatic photo composition. It is also a prerequisite in graphics applications like image retargeting, production video editing, and rotoscoping.

Today there are two main strategies for generic object segmentation in images: saliency and object proposals. Saliency methods yield either highly localized attention maps [121, 141, 12] or a complete segmentation of the prominent object [219, 26, 81, 123, 115, 223, 113]. Saliency focuses on regions that stand out, which is not the case for all foreground objects. Alternatively, *object proposal* methods learn to localize all objects in an image, regardless of their category [17, 6, 34, 224, 191, 148, 68]. The aim is to obtain high recall at the cost of low precision, i.e., they must generate a large number of proposals (typically 1000s) to cover all objects in an image. This usually involves a

[1] The work in this chapter was supervised by Prof. Kristen Grauman and originally published in: "FusionSeg: Learning to Combine Motion and Appearance for Fully Automatic Segmentation of Generic Objects in Video". Suyog Dutt Jain*, Bo Xiong* and Kristen Grauman (*Both authors contributed equally to this work). In Proceedings of the IEEE International Conference on Computer Vision, Honolulu, Hawaii, July 2017. An expanded article appeared in "Pixel Objectness: Learning to Segment Generic Objects Automatically in Images and Videos". Bo Xiong*, Suyog Dutt Jain* and Kristen Grauman (*Both authors contributed equally to this work). In IEEE Transactions on Pattern Analysis and Machine Intelligence, August 2018.

multi-stage process: first bottom-up segments are extracted, then they are scored by their degree of "objectness". Relying on bottom-up segments can be limiting, since low-level cues may fail to pull out contiguous regions for complex objects. Furthermore, in practice, the accompanying scores are not so reliable such that one can rely exclusively on the top few proposals.

In video object segmentation, motion offers important additional cues for isolating foreground objects that may be difficult to find in an individual image. Yet existing methods fall short of leveraging both appearance and motion in a unified manner. On the one hand, interactive techniques *strongly rely on appearance* information stemming from human-drawn outlines on frames in the video, using motion primarily to propagate information or enforce temporal consistency [193, 75, 146]. On the other hand, fully automatic methods *strongly rely on motion* to seed the segmentation process by locating possible moving objects. Once a moving object is detected, appearance is primarily used to track it across frames [109, 144, 37]. Such methods can fail if the object(s) are static or when there is significant camera motion. In either paradigm, results can suffer because the two essential cues are treated only in a sequential or disconnected way.

Motivated by these shortcomings, I introduce *pixel objectness*, a new approach to generic foreground object segmentation in images and video. Given a novel image or video frame, the goal is to determine the likelihood that each pixel is part of a foreground object (as opposed to background or "stuff" classes like grass, sky, sidewalks, etc.)

I first describe my approach for pixel objectness in Section 5.1, and then show results in Section 5.2. Please see Section 2.5 and 2.6 for prior work related to image segmentation and video segmentation.

## 5.1 Approach

Our goal is to predict the likelihood of each pixel being a generic object as opposed to background. As defined in the influential work of [4], a generic object should have at least one of three properties: 1) a well-defined closed boundary; 2) a different appearance from their surroundings; 3) sometimes it is unique within the image and stands out as salient. Examples of generic objects include object classes defined in PASCAL categories and other object classes similar to PASCAL categories. Building on the terminology from [4], we refer to our task as *pixel objectness*. We use this name to distinguish our task from the related problems of salient object detection (which seeks only the most attention-grabbing foreground object) and region proposals (which seeks a ranked list of candidate object-like regions).

The proposed approach consists of a two-stream CNN architecture that infers pixel objectness from appearance and motion. Below we first present the appearance stream (Sec. 5.1.1), then the motion stream (Sec. 5.1.2), followed by a fusion layer that brings the two together (Sec. 5.1.3). Pixel objectness is applicable to either images and video. For images, we have only appearance to analyze, and the motion stream is bypassed.

### 5.1.1 Appearance Stream

Given an RGB image or video frame $\mathcal{I}$ of size $m \times n \times c$ as input, we formulate the task of generic object segmentation as densely labeling each pixel as either "object" or "background". Thus the output of pixel objectness is a binary map of size $m \times n$.

For an individual image, the main idea is to train the system to predict pixel objectness using a mix of *explicit* boundary-level annotations and *implicit* image-level object

category annotations. From the former, the system will obtain direct information about image cues indicative of generic foreground object boundaries. From the latter, it will learn object-like features across a wide spectrum of object types—but *without* being told where those objects' boundaries are.

To this end, for the appearance stream we train a fully convolutional deep neural network for the foreground-background object labeling task. We initialize the network using a powerful generic image representation learned from millions of images labeled by their object category, but lacking any foreground annotations. Then, we fine-tune the network to produce dense binary segmentation maps, using relatively few images with pixel-level annotations originating from a small number of object categories.

Since the pretrained network is trained to recognize thousands of objects, we hypothesize that its image representation has a strong notion of objectness built inside it, even though it never observes *any* segmentation annotations. Meanwhile, by subsequently training with explicit dense foreground labels, we can steer the method to fine-grained cues about boundaries that the standard object classification networks have no need to capture. This way, even if our model is trained with a limited number of object categories having pixel-level annotations, we expect it to learn generic representations helpful to pixel objectness.

Specifically, we adopt a deep network structure [24] originally designed for multi-class semantic segmentation (see Sec. 5.2 for more implementation details). We initialize it with weights pre-trained on ImageNet, which provides a representation equipped to perform image-level classification for some 1,000 object categories. Next, we take a modestly sized semantic segmentation dataset, and transform its dense semantic masks into binary

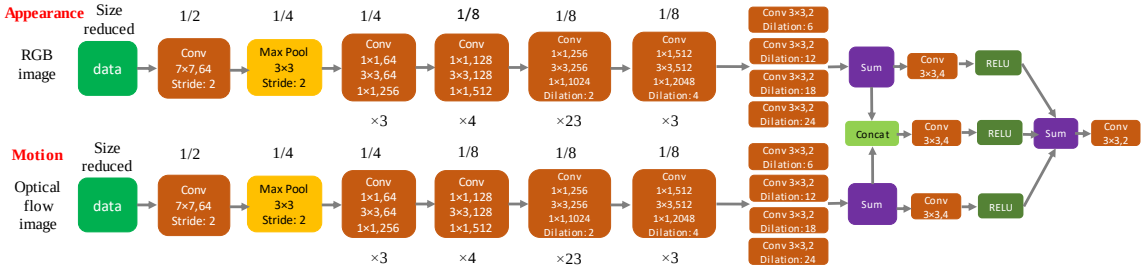


Figure 5.1: Network structure for our segmentation model. Each convolutional layer except the first 7× 7 convolutional layer and our fusion blocks is a residual block [63], adapted from ResNet-101. We show reduction in resolution at top of each box and the number of stacked convolutional layers in the bottom of each box. To apply our model to images, only the appearance stream is used.

object vs. background masks, by fusing together all its 20 categories into a single supercategory ("generic object"). We then train the deep network (initialized for ImageNet object classification) to perform well on the dense foreground pixel labeling task. The loss is the sum of cross-entropy terms over each pixel in the output layer. Our model supports end-to-end training.

### 5.1.2 Motion Stream

For the case of video segmentation, we have both the frame's appearance as well as its motion within the image sequence. Our complete video segmentation architecture consists of a two-stream network in which the appearance stream described thus far operates in parallel with a motion stream that processes the optical flow image, then joins the two in a fusion layer (see Fig. 5.1). We next discuss how to train a motion stream to densely predict pixel objectness from optical flow images only. Sec. 5.1.3 will explain how the two streams are merged.

The direct parallel to appearance-based pixel objectness discussed above would entail training the motion stream to map optical flow maps to video frame foreground maps. However, an important practical catch to that solution is training data availability. While ground truth foreground image segmentations are at least modestly available, datasets for video object segmentation masks are small-scale in deep learning terms, and primarily support evaluation. For example, Segtrack-v2 [111], a commonly used benchmark dataset for video segmentation, contains only 14 videos with 1066 labeled frames. DAVIS [145] contains only 50 sequences with 3455 labeled frames. None contain enough labeled frames to train a deep neural network. Semantic video segmentation datasets like CamVid [13] or Cityscapes [28] are somewhat larger, yet limited in object diversity due to a focus on street scenes and vehicles.

A good training source for our task would have ample frames with human-drawn segmentations on a wide variety of foreground objects, and would show a good mix of static and moving objects. No such large-scale dataset exists and creating one is non-trivial.

We propose a solution that leverages readily available *image* segmentation annotations together with *weakly annotated video* data to train our model. In brief, we temporarily decouple the two streams of our model, and allow the appearance stream (Sec. 5.1.1) to hypothesize likely foreground regions in frames of a large video dataset annotated only by bounding boxes. Since appearance alone need not produce perfect segmentations, we devise a series of filtering stages to generate high quality estimates of the true foreground. These instances bootstrap pre-training of the optical flow stream, then the two streams are joined to learn the best combination from minimal human labeled training videos.

More specifically, given a video dataset with bounding boxes labeled for each object,[2] we ignore the category labels and map the boxes alone to each frame. Then, we apply the appearance stream, thus far trained only from images labeled by their foreground masks, to compute a binary segmentation for each frame.

Next we deconflict the box and segmentation in each training frame. First, we refine the binary segmentation by setting all the pixels outside the bounding box(es) as background. Second, for each bounding box, we check if the the smallest rectangle that encloses all the foreground pixels overlaps with the bounding box by at least 75%. Otherwise we discard the segmentation. Third, we discard regions where the box contains more than 95% pixels labeled as foreground, based on the prior that good segmentations are rarely a rectangle, and thus probably the true foreground spills out beyond the box. Finally, we eliminate segments where object and background lack distinct optical flow, so our motion model can learn from the desired cues. Specifically, we compute the frame's optical flow using [119] and convert it to an RGB flow image [10]. If the 2-norm between a) the average value (each color channel is averaged separately) within the bounding box and b) the average value in a box (share the same center as the bounding box) whose height and width are twice the original size (ignore the part beyond the flow image) exceeds 30, the frame and filtered segmentation are added to the training set. See Fig. 5.2 for visual illustration of these steps.

To recap, bootstrapping from the preliminary appearance model, followed by bounding box pruning, bounding box tests, and the optical flow test, we can generate accurate

[2]We rely on ImageNet Video data, which contains 3862 videos and 30 diverse objects. See Sec. 5.2.

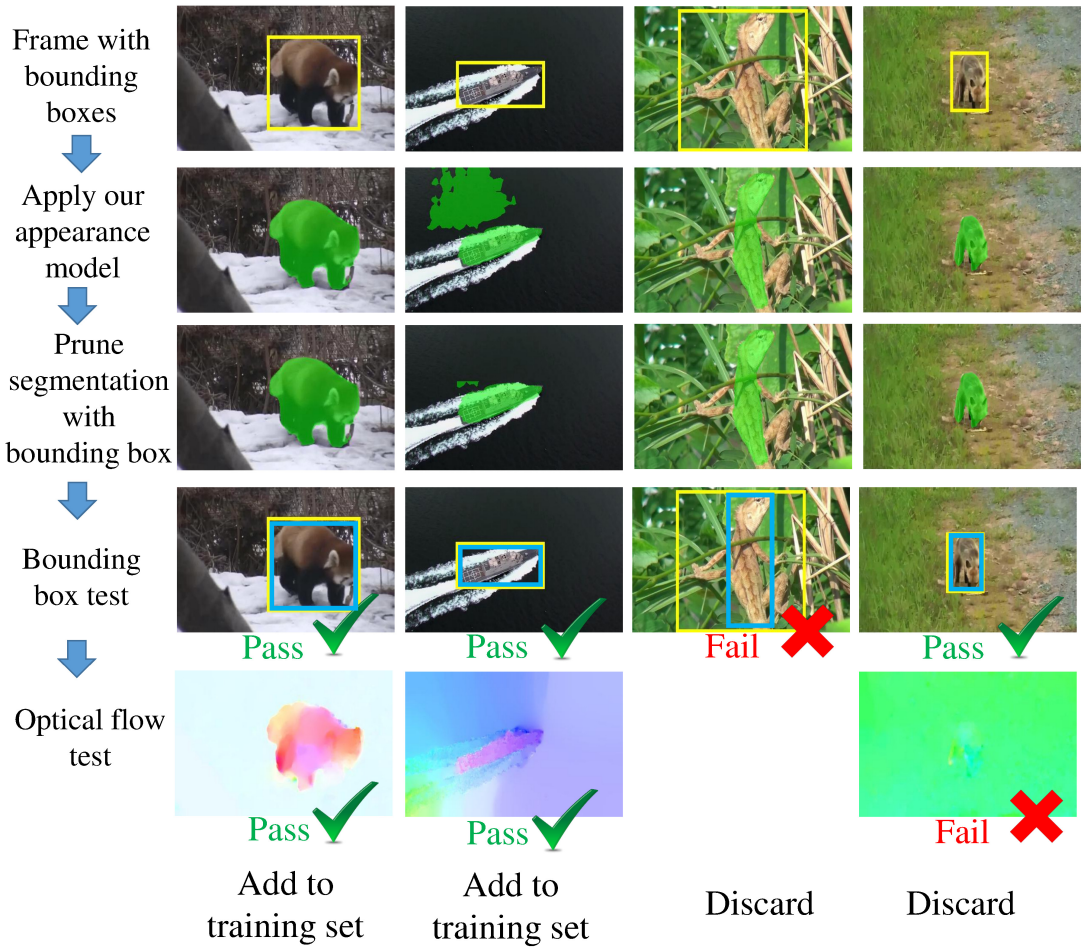


Figure 5.2: Procedure to generate (pseudo)-ground truth segmentations. We first apply the appearance model to obtain initial segmentations (second row, with object segment in green) and then prune by setting pixels outside bounding boxes as background (third row). Then we apply the bounding box test (fourth row, yellow bounding box is ground truth and blue bounding box is the smallest bounding box enclosing the foreground segment) and optical flow test (fifth row) to determine whether we add the segmentation to the motion stream's training set or discard it. Best viewed in color.

per-pixel foreground masks for thousands of diverse moving objects—for which no such datasets exist to date. Note that by eliminating training samples with these filters, we aim

to reduce label noise for training. However, at test time our system will be evaluated on standard benchmarks for which each frame is manually annotated (see Sec. 5.2).

With this data, we now turn to training the motion stream. Analogous to our strong generic appearance model, we also want to train a strong generic pixel objectness motion model that can segment foreground objects purely based on motion. Our motion model takes only optical flow as the input and is trained with automatically generated pixel level ground truth segmentations. In particular, we convert the raw optical flow to a 3-channel (RGB) color-coded optical flow image [10]. We use this color-coded optical flow image as the input to the motion network. We again initialize our network with pre-trained weights from ImageNet classification [159]. Representing optical flow using RGB flow images allows us to leverage the strong pre-trained initializations as well as maintain symmetry in the appearance and motion arms of the network.

An alternative solution might forgo handing the system optical flow, and instead input two raw consecutive RGB frames. However, doing so would likely demand more training instances in order to discover the necessary cues. Another alternative would directly train the joint model that combines both motion and appearance, whereas we first "pre-train" each stream to make it discover convolutional features that rely on appearance or motion alone, followed by a fusion layer (below). Our design choices are rooted in avoiding bias in training our model. Since the (pseudo) ground truth comes from the initial appearance network, training jointly from the onset is liable to bias the network to exploit appearance at the expense of motion. By feeding the motion model with only optical flow, we ensure our motion stream learns to segment objects from motion.

### 5.1.3 Fusion Model

The final processing in our pipeline joins the outputs of the appearance and motion streams, and aims to leverage a whole that is greater than the sum of its parts. We now describe how to train the joint model using both streams.

An object segmentation prediction is reliable if 1) either appearance or motion model alone predicts the object segmentation with very strong confidence or 2) their combination together predicts the segmentation with high confidence. This motivates the structure of our joint model.

We implement the idea by creating three independent parallel branches: 1) We apply a $3\times3$ convolution layer followed by a ReLU to the output of the appearance model. 2) We apply a $3\times3$ convolution layer followed by a ReLU to the output of the motion model. 3) We concatenate the outputs of the appearance and motion models, and apply a $3\times3$ convolution layer followed by a ReLU. We sum up the outputs from the three branches and apply a $3\times3$ convolution layer to obtain the final prediction. See Fig. 5.1.

As discussed above, we do not fuse the two streams in an early stage because we want them both to have strong independent predictions. We can then train the fusion model with very limited annotated video data, without overfitting. In the absence of large volumes of video segmentation training data, precluding a complete end-to-end training, our strategy of decoupling the individual streams and training works very well in practice.

## 5.2 Results

We first present pixel objectness results on image segmentation (Sec. 5.2.1) and two applications that benefit from predicting pixel objectness (Sec. 5.2.2). Then we show results on video segmentation (Sec. 5.2.3).

### 5.2.1 Results on Image Segmentation

We evaluate pixel objectness by comparing it to 16 recent methods in the literature, and also examine its utility for the two applications: image retrieval and image retargeting.

**Datasets:** We use three datasets which are commonly used to evaluate foreground object segmentation in images:

- **MIT Object Discovery:** This dataset consists of Airplanes, Cars, and Horses [157]. It is most commonly used to evaluate weakly supervised segmentation methods. The images were primarily collected using internet search and the dataset comes with per-pixel ground truth segmentation masks.
- **ImageNet-Localization:** We conduct a large-scale evaluation of our approach using ImageNet [159] ($\sim$1M images with bounding boxes, 3,624 classes). The diversity of this dataset lets us test the generalization abilities of our method.
- **ImageNet-Segmentation:** This dataset contains 4,276 images from 445 ImageNet classes with pixel-wise ground truth from [52].

**Baselines:** We compare to these state-of-the-art methods:

- **Saliency detection:** We compare to four salient object detection methods [219, 81, 223, 113], selected for their efficiency and state-of-the-art performance. All these methods are designed to produce a complete segmentation of the prominent object (vs. fixation maps; see Sec. 5 of [219]) and output continuous saliency maps, which are then thresholded by per image mean to obtain the segmentation.[3]

- **Object proposals:** We also compare with state-of-the-art region proposal algorithms, multiscale combinatorial grouping (MCG) [6] and DeepMask [148]. These methods output a ranked list of generic object segmentation proposals. The top ranked proposal in each image is taken as the final foreground segmentation for evaluation. We also compare with SalObj [115] which uses saliency to merge multiple object proposals from MCG into a single foreground.

- **Weakly supervised joint-segmentation methods:** These approaches rely on additional weak supervision in the form of prior knowledge that all images in a given collection share a common object category [157, 25, 84, 85, 95, 182, 74]. Note that our method lacks this additional supervision.

**Evaluation metrics:** Depending on the dataset, we use: 1) **Jaccard Score:** Standard intersection-over-union (IoU) metric between predicted and ground truth segmentation masks and 2) **BBox-CorLoc Score:** Percentage of objects correctly localized with a bounding box according to PASCAL criterion (i.e IoU $> 0.5$) used in [182, 31].

For MIT and ImageNet-Segmentation, we use the segmentation masks and evaluate using the Jaccard score. For ImageNet-Localization we evaluate with the BBox-

[3]This thresholding strategy was chosen because it gave the best results.

CorLoc metric, following the setup from [182, 31], which entails putting a tight bounding box around our method's output.

**Training details:** To generate the explicit boundary-level training data, we rely on the 1,464 PASCAL 2012 segmentation training images [36] and the additional annotations of [59], for 10,582 total training images. The 20 object labels are discarded and mapped instead to the single generic "object-like" (foreground) label for training. We train our model using the Caffe implementation of [24]. We optimize with stochastic gradient with a mini-batch size of 10 images. A simple data augmentation through mirroring the input images is also employed. A base learning rate of 0.001 with a $1/10$th slow-down every 2000 iterations is used. We train the network for a total of 10,000 iterations; total training time was about 8 hours on a modern GPU. We adopt the VGG [168] network structure for experiments on image segmentation in order to make fair comparison with DeepSaliency [113], which also adopts the VGG [168] network structure.

**MIT Object Discovery:** First we present results on the MIT dataset [157]. We do separate evaluation on the complete dataset and also a subset defined in [157]. We compare our method with 13 existing state-of-the-art methods including saliency detection [219, 81, 223, 113], object proposal generation [6, 148] plus merging [115] and joint-segmentation [157, 25, 84, 85, 95, 74]. We compare with author-reported results for the joint-segmentation baselines, and use software provided by the authors for the saliency and object proposal baselines.

Table 5.1 shows the results. Our proposed method outperforms several state-of-the-art saliency and object proposal methods—including recent deep learning techniques [223, 113, 148] in three out of six cases, and is competitive with the best performing method in

| Methods | MIT dataset (subset) | | | MIT dataset (full) | | |
|---|---|---|---|---|---|---|
| | **Airplane** | **Car** | **Horse** | **Airplane** | **Car** | **Horse** |
| **# Images** | 82 | 89 | 93 | 470 | 1208 | 810 |
| **Joint Segmentation** | | | | | | |
| Joulin et al. [84] | 15.36 | 37.15 | 30.16 | n/a | n/a | n/a |
| Joulin et al. [85] | 11.72 | 35.15 | 29.53 | n/a | n/a | n/a |
| Kim et al. [95] | 7.9 | 0.04 | 6.43 | n/a | n/a | n/a |
| Rubinstein et al. [157] | 55.81 | 64.42 | 51.65 | 55.62 | 63.35 | 53.88 |
| Chen et al. [25] | 54.62 | 69.2 | 44.46 | 60.87 | 62.74 | 60.23 |
| Jain et al. [74] | 58.65 | 66.47 | 53.57 | 62.27 | 65.3 | 55.41 |
| **Saliency** | | | | | | |
| Jiang et al. [81] | 37.22 | 55.22 | 47.02 | 41.52 | 54.34 | 49.67 |
| Zhang et al. [219] | 51.84 | 46.61 | 39.52 | 54.09 | 47.38 | 44.12 |
| DeepMC [223] | 41.75 | 59.16 | 39.34 | 42.84 | 58.13 | 41.85 |
| DeepSaliency [113] | 69.11 | 83.48 | 57.61 | **69.11** | 83.48 | **67.26** |
| **Object Proposals** | | | | | | |
| MCG [6] | 32.02 | 54.21 | 37.85 | 35.32 | 52.98 | 40.44 |
| DeepMask [148] | **71.81** | 67.01 | 58.80 | 68.89 | 65.4 | 62.61 |
| SalObj [115] | 53.91 | 58.03 | 47.42 | 55.31 | 55.83 | 49.13 |
| **Ours** | 66.43 | **85.07** | **60.85** | 66.18 | **84.80** | 64.90 |

Table 5.1: Quantitative results on MIT Object Discovery dataset. Our method outperforms several state-of-the-art methods for saliency detection, object proposals, and joint segmentation. (Metric: Jaccard score).

the others.

Our gains over the joint segmentation methods are arguably even more impressive because our model simply segments a single image at a time—no weak supervision!—and still substantially outperforms all weakly supervised techniques. We stress that in addition to the weak supervision in the form of segmenting common object, the previous best performing method [74] also makes use of a pre-trained deep network; we use strictly less total supervision than [74] yet still perform better. Furthermore, most joint segmen-

tation methods involve expensive steps such as dense correspondences [157] or region matching [74] which can take up to hours even for a modest collection of 100 images. In contrast, our method directly outputs the final segmentation in a single forward pass over the network and takes only 0.6 seconds per image for complete processing.

**ImageNet-Localization:** Next we present results on the ImageNet-Localization dataset. This involves testing our method on about 1 million images from 3,624 object categories. This also lets us test how generalizable our method is to unseen categories, i.e., those for which the method sees no foreground examples during training.

Table 5.2 (left) shows the results. When doing the evaluation over all categories, we compare our method with five methods which report results on this dataset [4, 182, 74] or are scalable enough to be run at this large scale [81, 6]. We see that our method significantly improves the state-of-the-art. The saliency and proposal methods [81, 4, 6] result in much poorer segmentations. Our method also significantly outperforms the joint segmentation approaches [182, 74], which are the current best performing methods on this dataset. In terms of the actual number of images, our gains translate into correctly segmenting 42,900 more images than [74] (which, like us, leverages ImageNet features) and 83,800 more images than [182]. This reflects the overall magnitude of our gains over state-of-the-art baselines.

Does our learned segmentation model only recognize foreground objects that it has seen during training, or can it generalize to unseen object categories? Intuitively, ImageNet has such a large number of diverse categories that this gain would not have been possible if our method was only over-fitting to the 20 seen PASCAL categories. To empirically

| ImageNet-Localization dataset | | |
|---|---|---|
| | All | Non-Pascal |
| # Classes | 3,624 | 3,149 |
| # Images | 939,516 | 810,219 |
| Alexe et al. [4] | 37.42 | n/a |
| Tang et al. [182] | 53.20 | n/a |
| Jain et al. [74] | 57.64 | n/a |
| Jiang et al. [81] | 41.28 | 39.35 |
| MCG [6] | 42.23 | 41.15 |
| Ours | **62.12** | **60.18** |

| ImageNet-Segmentation dataset | |
|---|---|
| Jiang et al. [81] | 43.16 |
| Zhang et al. [219] | 45.07 |
| DeepMC [223] | 40.23 |
| DeepSaliency [113] | 62.12 |
| MCG [6] | 39.97 |
| DeepMask [148] | 58.69 |
| SalObj [115] | 41.35 |
| Guillaumin et al. [52] | 57.3 |
| Ours | **64.22** |

Table 5.2: Quantitative results on ImageNet localization and segmentation datasets. Results on ImageNet-Localization (left) show that the proposed model outperforms several state-of-the-art methods and also generalizes very well to unseen object categories (Metric: BBox-CorLoc). It also outperforms all methods on the ImageNet-Segmentation dataset (right) showing that it produces high-quality object boundaries (Metric: Jaccard score).

verify this intuition, we next exclude those ImageNet categories which are directly related to the PASCAL objects, by matching the two datasets' synsets. This results in a total of 3,149 categories which are exclusive to ImageNet ("Non-PASCAL"). See Table 5.2 (left) for the data statistics.

We see only a very marginal drop in performance; our method still significantly outperforms both the saliency and object proposal baselines. This is an important result, because during training the segmentation model *never saw any dense object masks for images in these categories*. Bootstrapping from the pretrained weights of the VGG-classification network, our model is able to learn a transformation between its prior belief on what looks like an object to complete dense foreground segmentations.

**ImageNet-Segmentation:** Finally, we measure the pixel-wise segmentation quality on

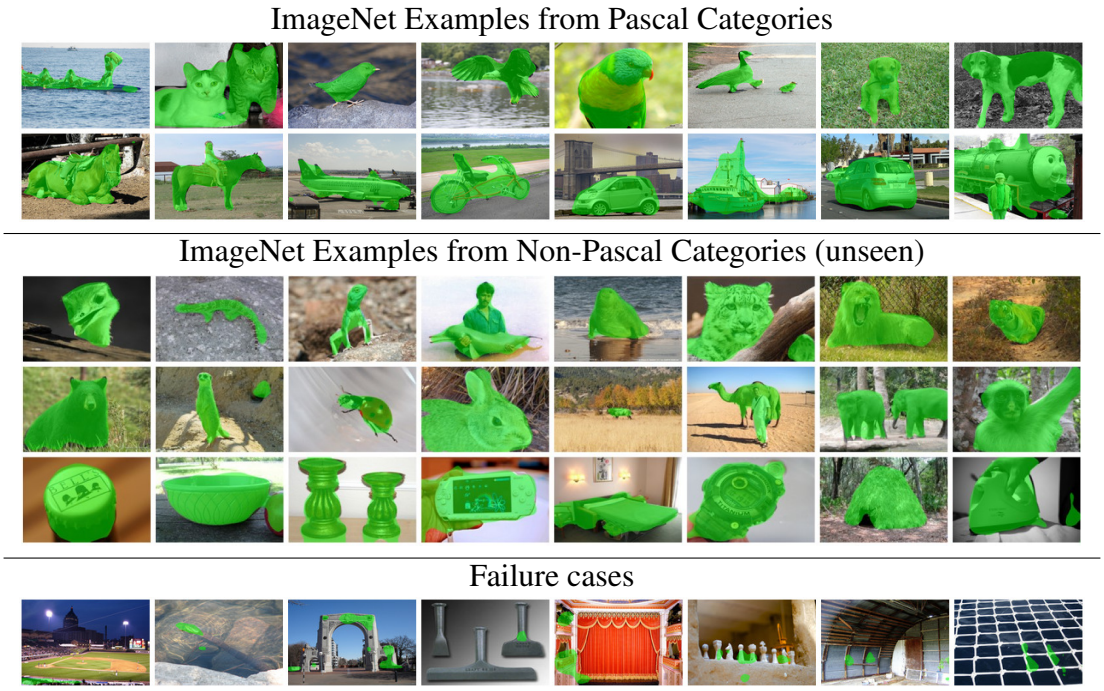


Figure 5.3: Qualitative results: We show qualitative results on images belonging to PASCAL (top) and Non-PASCAL (middle) categories. Our segmentation model generalizes remarkably well even to those categories which were unseen in any foreground mask during training (middle rows). Typical failure cases (bottom) involve scene-centric images where it is not easy to clearly identify foreground objects (best viewed on pdf).

a large scale. For this we use the ground truth masks provided by [52] for 4,276 images from 445 ImageNet categories. The current best reported results are from the segmentation propagation approach of [52]. We found that DeepSaliency [113] and DeepMask [148] further improve it. Note that like us, DeepSaliency [113] also trains with PASCAL [36]. DeepMask [148] is trained with a much larger COCO [117] dataset. Our method outperforms all methods, significantly improving the state-of-the-art (see Table 5.2 (right)). This shows that our method not only generalizes to thousands of object categories but also produces high quality object segmentations.

**Qualitative results:** Fig. 5.3 shows qualitative results for ImageNet from both PASCAL and Non-PASCAL categories. Pixel objectness accurately segments foreground objects from both sets. The examples from the Non-PASCAL categories highlight its strong generalization capabilities. We are able to segment objects across scales and appearance variations, including multiple objects in an image. It can segment even man-made objects, which are especially distinct from the objects in PASCAL. The bottom row shows failure cases. Our model has more difficulty in segmenting scene-centric images where it is more difficult to clearly identify foreground objects.

### 5.2.2 Impact on Downstream Applications

Next we report results leveraging pixel objectness for two downstream tasks on images. Dense pixel objectness has many applications. Here we explore how it can assist in image retrieval and content-aware image retargeting, both of which demand a single, high-quality estimate of the foreground object region.

**Object-aware image retrieval:** First, we consider how pixel objectness foregrounds can assist in image retrieval. A retrieval system accepts a query image containing an object, and then the system returns a ranked list of images that contain the same object. This is a valuable application, for example, to allow object-based online product search. Typically retrieval systems extract image features from the entire query image. This can be problematic, however, because it might retrieve images with similar background, especially when the object of interest is small. We aim to use pixel objectness to restrict the system's attention to the foreground object(s) as opposed to the entire image.

To implement the idea, we first run pixel objectness. In order to reduce false pos-

itive segmentations, we keep the largest connected foreground region if it is larger than $6\%$ of the overall image area. Then we crop the smallest bounding box enclosing the foreground segmentation and extract features from the entire bounding box. If no foreground is found (which occurs in roughly 17% of all images), we extract image features from the entire image. The method is applied to both the query and database images. To rank database images, we explore two image representations. The first one uses only the image features extracted from the bounding box, and the second concatenates the features from the original image with those from the bounding box.

To test the retrieval task, we use the ILSVRC2012 [159] validation set, which contains 50K images and $1,000$ object classes, with $50$ images per class. As an evaluation metric, we use mean average precision (mAP). We extract VGGNet [168] features and use cosine distance to rank retrieved images. We compare with two baselines 1) **Full image**, which ranks images based on features extracted from the entire image, and 2) **Top proposal** (TP), which ranks images based on features extracted from the top ranked MCG [6] proposal. For our method and the Top proposal baseline, we examine two image representations. The first directly uses the features extracted from the region containing the foreground or the top proposal (denoted **FG**). The second representation concatenates the extracted features with the image features extracted from the entire image (denoted **FF**).

Table 5.3 shows the results. Our method with FF yields the best results. Our method outperforms both baselines for many ImageNet classes. We observe that our method performs extremely well on object-centric classes such as animals, but has limited improvement upon the baseline on scene-centric classes (lakeshore, seashore etc.). To

| Method | Ours(FF) | Ours(FG) | Full Img | TP (FF) [6] | TP (FG) [6] |
|---|---|---|---|---|---|
| All | **0.3342** | 0.3173 | 0.3082 | 0.3102 | 0.2092 |
| Obj-centric | **0.4166** | 0.4106 | 0.3695 | 0.3734 | 0.2679 |

Table 5.3: Object-based image retrieval performance on ImageNet. We report average precision on the entire validation set, and on the first 400 categories, which are mostly object-centric classes.

verify our hypothesis, we isolate the results on the first 400 object classes of ImageNet, which contain mostly object-centric classes, as opposed to scene-centric objects. On those first 400 object classes, our method outperforms both baselines by a larger margin. This demonstrates the value of our method at retrieving objects, which often contain diverse background and so naturally benefit more from accurate pixel objectness.

**Foreground-aware image retargeting:** As a second application, we explore how pixel objectness can enhance image retargeting. The goal is to adjust the aspect ratio or size of an image without distorting its important visual concepts. We build on the popular Seam Carving algorithm [7], which eliminates the optimal irregularly shaped path, called a seam, from the image via dynamic programming. In [7], the energy is defined in terms of the image gradient magnitude. However, the gradient is not always a sufficient energy function, especially when important visual content is non-textured or the background is textured.

Our idea is to protect semantically important visual content based on foreground segmentation. To this end, we consider a simple adaption of Seam Carving. We define an energy function based on high-level semantics rather than low-level image features alone. Specifically, we first predict pixel objectness, and then we scale the gradient energy $g$


Figure 5.4: Leveraging pixel objectness for foreground aware image retargeting. Best viewed on pdf.

within the foreground segment(s) by $(g + 1) \times 2$.

We use a random subset of 500 images from the 2014 Microsoft COCO Captioning Challenge Testing Images [117] for experiments. Figure 5.4 shows example results. For reference, we also compare with the original Seam Carving (SC) algorithm [7] that uses image gradients as the energy function. Both methods are instructed to resize the source image to various aspect ratios. Thanks to the proposed foreground segmentation, our method successfully preserves the important visual content (e.g., train, bus, human and dog) while reducing the content of the background. The baseline produces images with important objects distorted, because gradient strength is an inadequate indicator for perceived content, especially when background is textured. The rightmost column is a failure case for our method on a scene-centric image that does not contain any salient objects.

To quantify the results over all 500 images, we perform a human study on Amazon Mechanical Turk. Both methods are instructed to resize the source image to $2/3$ of its

original size. We present image pairs produced by our method and the baseline in arbitrary order and ask workers to rank which image is more likely to have been manipulated by a computer. Each image pair is evaluated by three different workers. Workers found that $38.53\%$ of the time images produced by our method are more likely to have been manipulated by a computer, $48.87\%$ for the baseline; both methods tie $12.60\%$ of the time. Thus, human evaluation with non-experts demonstrates that our method outperforms the baseline. In addition, we also ask a vision expert familiar with image retargeting—but not involved in this project—to score the 500 image pairs with the same interface as the crowd workers. The vision expert found our method performs better for $78\%$ of the images, baseline is better for $13\%$, and both methods tie for $9\%$ images. This further confirms that our foreground prediction can enhance image retargeting by defining a more semantically meaningful energy function.

### 5.2.3 Results on Video Segmentation

Pixel objectness can predict high quality object segmentations and generalize very well to thousands of unseen object categories for image segmentation. We next show, when jointly trained with motion, our method also improves the state-of-the-art results for automatically segmenting generic objects in videos (please see our project home for video results at: `http://vision.cs.utexas.edu/projects/fusionseg/`).

**Datasets and metrics:** We evaluate our method on three challenging video object segmentation datasets: DAVIS [145], YouTube-Objects [151, 75, 183] and Segtrack-v2 [111]. To measure accuracy we again use the standard Jaccard score. The three datasets are:

- **DAVIS [145]:** the latest and most challenging video object segmentation benchmark consisting of 50 high quality video sequences of diverse object categories with $3,455$ densely annotated, pixel-accurate frames. The videos are unconstrained in nature and contain challenges such as occlusions, motion blur, and appearance changes. Only the prominent moving objects are annotated in the ground-truth.

- **YouTube-Objects [151, 75, 183]:** consists of 126 challenging web videos from 10 object categories with more than 20,000 frames and is commonly used for evaluating video object segmentation. We use the subset defined in [183] and the ground truth provided by [75] for evaluation.

- **SegTrack-v2 [111]:** one of the most common benchmarks for video object segmentation consisting of 14 videos with a total of $1,066$ frames with pixel-level annotations. For videos with multiple objects with individual ground-truth segmentations, we treat them as a single foreground for evaluation.

**Semi-supervised methods:** Semi-supervised methods bring a human in the loop. They have some knowledge about the object of interest which is exploited to obtain the segmentation (e.g., a manually annotated first frame). We compare with the following state-of-the-art methods: HVS [51], HBT [47], FCP [146], IVID [163], HOP [75], and BVS [134]. The methods require different amounts of human annotation to operate, e.g. HOP, BVS, and FCP make use of manual complete object segmentation in the first frame to seed the method; HBT requests a bounding box around the object of interest in the first frame; HVS, IVID require a human to constantly guide the algorithm whenever it fails. We also compare with three semi-supervised video segmentation based on deep learning: VPN [78],

MSK [89] and OSVOS [14].

**Baselines:** We compare with several state-of-the-art methods for each dataset as reported in the literature. Here we group them together based on whether they can operate in a fully automatic fashion (automatic) or require a human in the loop (semi-supervised) to do the segmentation:

- **Automatic methods:** Automatic video segmentation methods do not require any human involvement to segment new videos. Depending on the dataset, we compare with the following state of the art methods: FST [144], KEY [109], NLC [37], COSEG [190], MPN [185], and ARP [99]. All use some form of unsupervised motion or objectness cues to identify foreground objects followed by post-processing to obtain space-time object segmentations.
- **Semi-supervised methods:** Semi-supervised methods bring a human in the loop. They have some knowledge about the object of interest which is exploited to obtain the segmentation (e.g., a manually annotated first frame). We compare with the following state-of-the-art methods: HVS [51], HBT [47], FCP [146], IVID [163], HOP [75], and BVS [134]. The methods require different amounts of human annotation to operate, e.g. HOP, BVS, and FCP make use of manual complete object segmentation in the first frame to seed the method; HBT requests a bounding box around the object of interest in the first frame; HVS, IVID require a human to constantly guide the algorithm whenever it fails. We also compare with three semi-supervised video segmentation based on deep learning: VPN [78], MSK [89] and OSVOS [14].

Note that our method requires human annotated data only during training. At test time it operates in a fully automatic fashion. Thus, given a new video, we require equal effort as the automatic methods, and less effort than the semi-supervised methods.

Apart from these comparisons, we also examine some natural baselines and variants of our method:

- **Flow-thresholding (Flow-T):** To examine the effectiveness of motion alone in segmenting objects, we adaptively threshold the optical flow in each frame using the flow magnitude. Specifically, we compute the mean and standard deviation from the L2 norm of flow magnitude and use "mean+unit std." as the threshold.
- **Flow-saliency (Flow-S):** Optical flow magnitudes can have large variances, hence we also try a variant which normalizes the flow by applying a saliency detection method [82] to the flow image itself. We use average thresholding to obtain the segmentation.
- **Probabilistic model for flow (PM) [11]:** We compare with a prior method that uses a probabilistic model [11] to segment objects relying on motion cues only.
- **Appearance model (Ours-A):** To quantify the role of appearance in segmenting objects, we obtain segmentations using only the appearance stream of our model.
- **Motion model (Ours-M):** To quantify the role of motion, we obtain segmentations using only the motion stream of our model.
- **Joint model (Ours-J):** Our complete joint model that learns to combine both motion and appearance together to obtain the final object segmentation.

**Implementation details:** As weak bounding box video annotations, we use the ImageNet-Video dataset [159]. This dataset comes with a total of 3,862 training videos from 30 object categories with 866,870 labeled object bounding boxes from over a million frames. Post refinement using our ground truth generation procedure (see Sec. 5.1.2), we are left with 84,929 frames with good pixel segmentations[4] which are then used to train our motion model. For training the joint model we use a held-out set for each dataset. We train each stream for a total of 20,000 iterations, use "poly" learning rate policy (power = 0.9) with momentum (0.9) and weight decay (0.0005). No post-processing is applied on the segmentations obtained from our networks.

**Quality of training data:** To ascertain that the quality of training data we automatically generate for training our motion stream is good, we first compare it with a small amount of human annotated ground truth. We randomly select 100 frames that passed both the bounding box and optical flow tests, and collect human-drawn segmentations on Amazon MTurk. We first present crowd workers a frame with a bounding box labeled for each object, and then ask them to draw the detailed segmentation for all objects within the bounding boxes. Each frame is labeled by three crowd workers and the final segmentation is obtained by majority vote on each pixel. The results indicate that our strategy to gather pseudo-ground truth is effective. On the 100 labeled frames, Jaccard overlap with the human-drawn ground truth is 77.8 (and 70.2 before pruning with bounding boxes).

**Quantitative evaluation:** We now present the quantitative comparisons of our method with several state-of-the-art methods and baselines, for each of the three datasets in turn.

[4] Available for download on our project website.

| DAVIS dataset (50 videos) | | |
|---|---|---|
| Methods | Human in the loop? | Avg. IoU (%) |
| Flow-T | No | 42.95 |
| Flow-S | No | 30.22 |
| PM [11] | No | 43.4 |
| FST [144] | No | 57.5 |
| KEY [109] | No | 56.9 |
| NLC [37] | No | 64.1 |
| MPN [185] | No | 69.7 |
| ARP [99] | No | **76.3** |
| HVS [51] | Yes | 59.6 |
| FCP [146] | Yes | 63.1 |
| BVS [134] | Yes | 66.5 |
| VPN [78] | Yes | 75 |
| MSK [89] | Yes | **80.3** |
| Ours-A | No | 64.69 |
| Ours-M | No | 60.18 |
| Ours-J | No | 72.82 |

Table 5.4: Video object segmentation results on DAVIS dataset. We show the average accuracy over all 50 videos. Our method outperforms 5 of the 6 fully automatic state-of-the-art methods. The best performing methods grouped by whether they require human-in-the-loop or not during segmentation are highlighted in bold. Metric: Jaccard score, higher is better.

**DAVIS dataset:** Table 5.4 shows the results, with baselines that are the best performing methods taken from the benchmark results [145]. Our method achieves the second best performance among all the fully automatic methods. The best performing method ARP [99], proposed concurrently with our method, segments objects with an iterative augmentation and reduction process. Our method is significantly better than simple flow baselines. This supports our claim that even though motion contains a strong signal about foreground objects in videos, it is not straightforward to simply threshold optical flow and

obtain those segmentations. A data-driven approach that learns to identify motion patterns indicative of objects as opposed to backgrounds or camera motion is required.

The appearance and motion variants of our method themselves result in a very good performance. The performance of the motion variant is particularly exciting, knowing that it has no information about the object's appearance and purely relies on the flow signal. When combined together, the joint model results in a significant improvement, with an absolute gain of up to 11% over the individual streams.

Our method is significantly better than 5 of the 6 fully automatic methods, which typically rely on motion alone to identify foreground objects. This illustrates the benefits of a unified combination of both motion and appearance. Our method also significantly outperforms several semi-supervised techniques, which require substantial human annotation on every video they process. The state-of-the-art human-in-the-loop algorithm MSK [89] achieves better performance than ours. However, their method requires the first frame of the video to be manually segmented, whereas our method uses no human input.

| YouTube-Objects dataset (126 videos) | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Methods | Flow-T | Flow-S | PM [11] | FST [144] | COSEG [190] | HBT [47] | HOP [75] | IVID [163] | OSVOS [14] | Ours-A | Ours-M | Ours-J |
| Human? | No | No | No | No | No | Yes | Yes | Yes | Yes | No | No | No |
| airplane (6) | 18.27 | 33.32 | 25.83 | 70.9 | 69.3 | 73.6 | 86.27 | **89** | 88.2 | **83.38** | 59.38 | 83.09 |
| bird (6) | 31.63 | 33.74 | 26.27 | 70.6 | **76** | 56.1 | 81.04 | 81.6 | **85.7** | 60.89 | 64.06 | 63.01 |
| boat (15) | 4.35 | 22.59 | 12.54 | 42.5 | 53.5 | 57.8 | 68.59 | 74.2 | **77.5** | 72.62 | 40.21 | **72.70** |
| car (7) | 21.93 | 48.63 | 37.90 | 65.2 | 70.4 | 33.9 | 69.36 | 70.9 | **79.6** | 74.50 | 61.32 | **75.49** |
| cat (16) | 19.9 | 32.33 | 30.01 | 52.1 | 66.8 | 30.5 | 58.89 | 67.7 | **70.8** | **67.99** | 49.16 | 67.75 |
| cow (20) | 16.56 | 29.11 | 35.31 | 44.5 | 49 | 41.8 | 68.56 | **79.1** | 77.8 | 69.63 | 39.38 | **70.30** |
| dog (27) | 17.8 | 25.43 | 36.4 | 65.3 | 47.5 | 36.8 | 61.78 | 70.3 | **81.3** | **69.10** | 54.79 | 67.64 |
| horse (14) | 12.23 | 24.17 | 28.09 | 53.5 | 55.7 | 44.3 | 53.96 | 67.8 | **72.8** | 62.79 | 39.96 | **65.05** |
| mbike (10) | 12.99 | 17.06 | 24.08 | 44.2 | 39.5 | 48.9 | 60.87 | 61.5 | **73.5** | 61.92 | 42.95 | **62.22** |
| train (5) | 18.16 | 24.21 | 23.62 | 29.6 | 53.4 | 39.2 | 66.33 | **78.2** | 75.7 | **62.82** | 43.13 | 62.30 |
| Avg. IoU (%) | 17.38 | 29.05 | 28.01 | 53.84 | 58.11 | 46.29 | 67.56 | 74.03 | **78.3** | 68.57 | 49.43 | **68.95** |

Table 5.5: Video object segmentation results on YouTube-Objects dataset. We show the average performance for each of the 10 categories from the dataset. The final row shows an average over all the videos. Our method outperforms all other unsupervised methods, and half of those that require human annotation during segmentation. The best performing methods grouped by whether they require human-in-the-loop or not during segmentation are highlighted in bold. Metric: Jaccard score, higher is better.

| SegTrack-v2 dataset (14 videos) | | |
|---|---|---|
| Methods | Human in the loop? | Avg. IoU (%) |
| Flow-T | No | 37.77 |
| Flow-S | No | 27.04 |
| PM [11] | No | 33.5 |
| FST [144] | No | 53.5 |
| KEY [109] | No | 57.3 |
| NLC [37] | No | **80*** |
| HBT [47] | Yes | 41.3 |
| HVS [51] | Yes | 50.8 |
| MSK [89] | Yes | **67.4** |
| Ours-A | No | 56.88 |
| Ours-M | No | 53.04 |
| Ours-J | No | 64.44 |

Table 5.6: Video object segmentation results on SegTrack-v2. We show the average accuracy over all 14 videos. Our method outperforms most state-of-the-art methods, including the ones which actually require human annotation during segmentation. The best performing methods grouped by whether they require human-in-the-loop or not during segmentation are highlighted in bold. *For NLC results are averaged over 12 videos as reported in their paper [37], whereas all other methods are tested on all 14 videos. Metric: Jaccard score, higher is better.

**YouTube-Objects dataset:** In Table 5.5 we see a similarly strong result on the YouTube-Objects dataset. Our method again outperforms the flow baselines and all the automatic methods by a significant margin. The publicly available code for NLC [37] runs successfully only on 9% of the YouTube dataset (1725 frames); on those, its Jaccard score is 43.64%. Our proposed model outperforms it by a significant margin of 25%. Even among human-in-the-loop methods, we outperform all methods except IVID [163] and OSVOS [14]. However, both methods [163, 14] require manual annotations. In particular, IVID [163] requires a human to consistently track the segmentation performance and correct whatever mistakes the algorithm makes. This can take up to minutes of annotation time for each video. Our method uses zero human involvement but still performs competitively.

**Segtrack-v2 dataset:** In Table 5.6, our method outperforms all automatic methods except NLC [37] on Segtrack. While our approach significantly outperforms NLC [37] on the DAVIS dataset, NLC is exceptionally strong on this dataset. Our relatively weaker perfor-

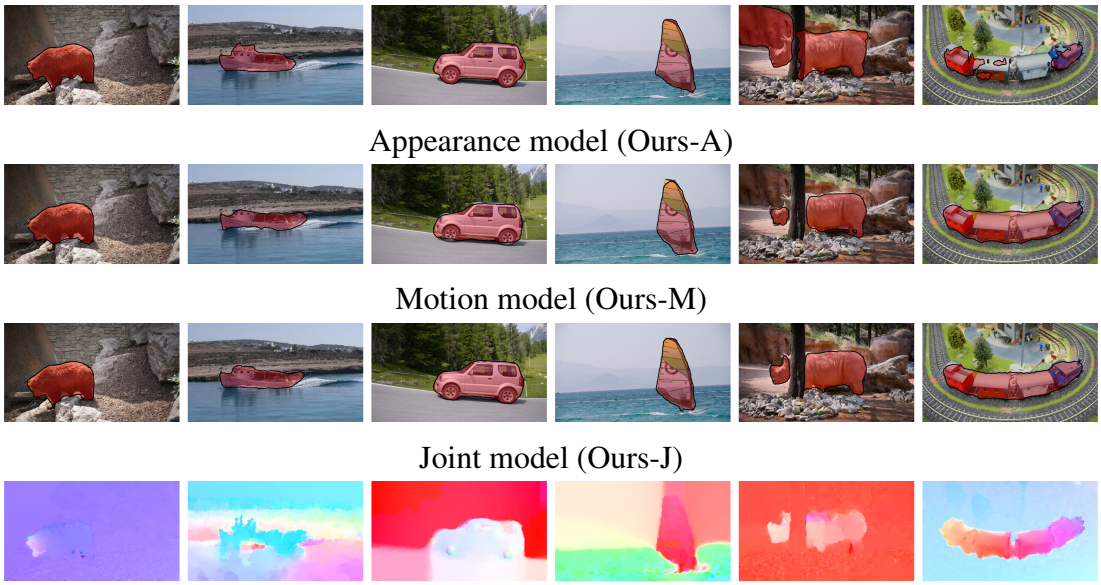


Figure 5.5: We show examples from our appearance, motion, and joint models along with the flow image which was used as an input to the motion network (best viewed on pdf and see text for the discussion). Videos of our segmentation results are available on the project website.

mance could be due to the low quality and resolution of the Segtrack-v2 videos, making it hard for our network based model to process them. Nonetheless, our joint model still provides a significant boost over both our appearance and motion models, showing it again realizes the true synergy of motion and appearance.

**Qualitative evaluation:** Fig. 5.5 shows qualitative results. We show visual comparisons between different components of our method including the appearance, motion, and joint models. We also show the optical flow image that was used as an input to the motion stream. These images help reveal the complexity of learned motion signals. In the bear example, the flow is most salient only on the bear's head, still our motion stream alone is able to segment the bear completely. The boat, car, and sail examples show that even when the flow is noisy—including strong flow on the background—our motion model is able to learn about object shapes and successfully suppresses the background. The rhino and train examples show cases where the appearance model fails but when combined with

the motion stream, the joint model produces accurate segmentations.

## 5.3 Summary

In this chapter, I introduced a novel approach to automatically segment foreground objects in both images and videos. Through experiments on multiple challenging image and video segmentation benchmarks, the proposed method offers consistently strong results and improves the state-of-the-art results for fully automatic segmentation of foreground objects.

The proposed motion stream works well for segmenting salient moving objects. However, the performance suffers if the optical flow estimation is inaccuracy. In addition, my proposed solution for foreground segmentation in videos does not consider temporal consistency, which could be potentially addressed with a memory-based network to further improve performance.

In the next chapter, I address how to predict viewing angles to enhance photo composition with the proposed foreground segmentation method.

# Chapter 6

# Snap Angle Prediction for 360° Panoramas

[1] Building on the proposed foreground segmentation method presented in Chapter 5, I address how to predict snap angles to enhance photo composition after identifying those foreground objects in this chapter. Specifically, I consider snap angle prediction for 360° panoramas, which are a rich medium, yet notoriously difficult to visualize in the 2D image plane.

The goal of snap angle prediction is to find the best rotation angle of the cube that will yield a set of cube faces that, among all possible rotations, most look like nicely composed human-taken photos originating from the given 360° panoramic image. While what comprises a "well-composed photo" is itself the subject of active research [100, 71, 204, 54, 92], I concentrate on a high-level measure of good composition, where the goal is to consolidate each (automatically detected) foreground object within the bounds of one cubemap face.

I concentrate on the cubemap projection [50]. Recall that a cubemap maps the sphere to a cube with rectilinear projection (where each face captures a 90° FOV) and then unfolds the six faces of the cube. The unwrapped cube can be visualized as an unfolded

[1]The work in this chapter was supervised by Prof. Kristen Grauman and originally published in: "Snap angle prediction for 360 panoramas ". Bo Xiong and Kristen Grauman. In Proceedings of the European Conference on Computer Vision, Munich, Germany, September 2018.

box, with the lateral strip of four faces being spatially contiguous in the scene (see Fig. 6.1, bottom). We explore our idea with cubemaps for a couple reasons. First, a cubemap covers the entire 360° content and does not discard any information. Secondly, each cube face is very similar to a conventional FOV, and therefore relatively easy for a human to view and/or edit.

I explore how intelligent rotations of a spherical image may enable content-aware projection with fewer perceptible distortions. Whereas existing approaches assume the viewpoint is fixed [97, 164, 21], intuitively some viewing angles within the sphere preserve high-level objects better than others. To discover the relationship between these optimal snap angles and the spherical panorama's content, I develop a reinforcement learning approach for the cubemap projection model.

I first describe our approach for predicting snap angles in Section 6.1, and then show results in Section 6.2. Please see Section 2.7 for prior work on viewing wide-angle images and panoramas and Section 2.8 on recurrent networks for attention, which motivates our proposed method for snap angle prediction.

## 6.1 Approach

I first formalize snap angle prediction as an optimization problem (Sec. 6.1.1). Then I present the learning framework and network architecture for snap angle prediction (Sec. 6.1.2).

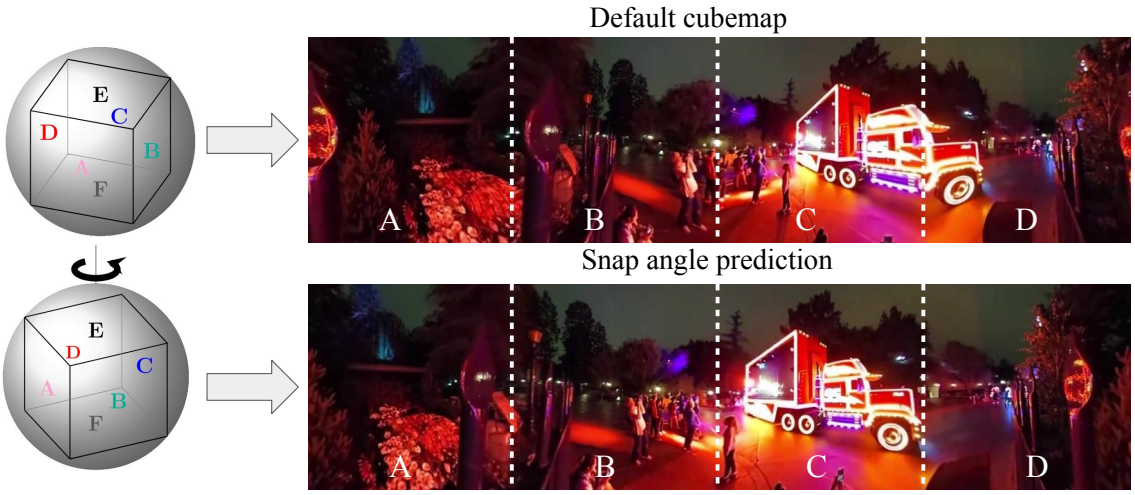


Figure 6.1: Comparison of a cubemap before and after snap angle prediction (dotted lines separate each face). Unlike prior work that assumes a fixed angle for projection, I propose to predict the cube rotation that will best preserve foreground objects in the output. For example, here my method better preserves the truck (third picture C in the second row). We show four (front, right, left, and back) out of the six faces for visualization purposes. Best viewed in color or pdf.

### 6.1.1 Problem Formulation

We first formalize snap angle prediction as an optimization problem. Let $P(I, \theta)$ denote a projection function that takes a panorama image $I$ and a projection angle $\theta$ as input and outputs a cubemap after rotating the sphere (or equivalently the cube) by $\theta$. Let function $F$ be an objective function that takes a cubemap as input and outputs a score to measure the quality of the cubemap. Given a novel panorama image $I$, our goal is to minimize $F$ by predicting the snap angle $\theta^*$:

$$\theta^* =_\theta \quad F(P(I, \theta)). \tag{6.1}$$

The projection function $P$ first transforms the coordinates of each point in the panorama based on the snap angle $\theta$ and then produces a cubemap in the standard manner.

Views from a horizontal camera position (elevation $0^\circ$) are more informative than others due to human recording bias. The bottom and top cube faces often align with the sky (above) and ground (below); "stuff" regions like sky, ceiling, and floor are thus common in these faces and foreground objects are minimal. Therefore, rotations in azimuth tend to have greater influence on the disruption caused by cubemap edges. Hence, without loss of generality, we focus on snap angles in azimuth only, and jointly optimize the front/left/right/back faces of the cube.

The coordinates for each point in a panorama can be represented by a pair of latitude and longitude $(\lambda, \varphi)$. Let $L$ denote a coordinate transformation function that takes the snap angle $\theta$ and a pair of coordinates as input. We define the coordinate transformation function $L$ as:

$$L((\lambda, \varphi), \theta) = (\lambda, \varphi - \theta). \tag{6.2}$$

Note when the snap angle is $90^\circ$, the orientation of the cube is the same as the default cube except the order of front, back, right, and left is changed. We therefore restrict $\theta \in [0, \pi/2]$. We discretize the space of candidate angles for $\theta$ into a uniform $N = 20$ azimuths grid, which we found offers fine enough camera control.

We next discuss our choice of the objective function $F$. A cubemap in its default orientation has two disadvantages: 1) It does not guarantee to project each important object onto the same cube face; 2) Due to the nature of the perspective projection, objects projected onto cube boundaries will be distorted more than objects in the center. Motivated by these shortcomings, our goal is to produce cubemaps that *place each important object in a single face* and avoid placing objects at the cube boundaries/edges.

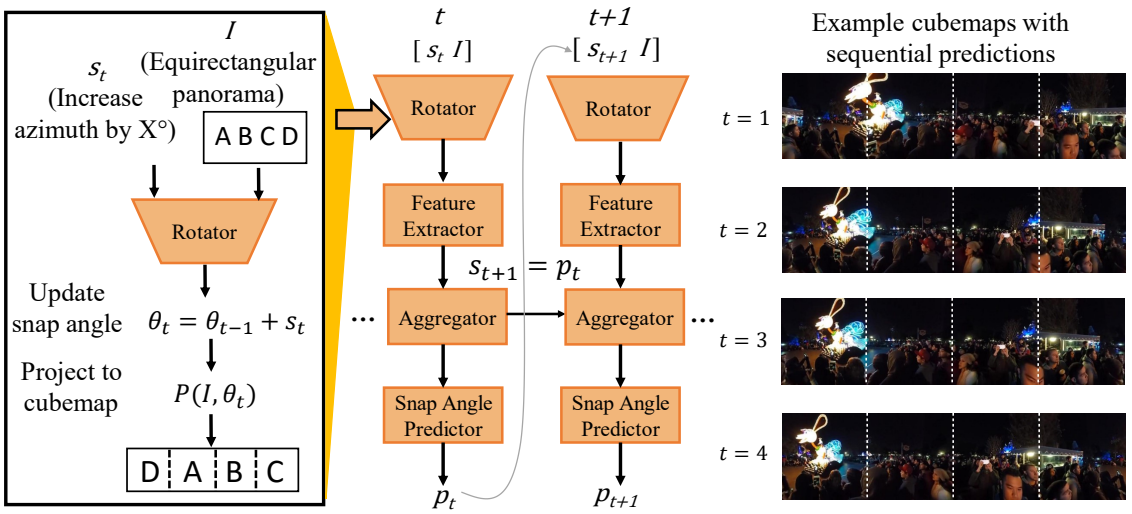


Figure 6.2: We show the rotator (left), our model (middle), and a series of cubemaps produced by our sequential predictions (right). Our method iteratively refines the best snap angle, targeting a given budget of allowed computation.

In particular, we propose to minimize the area of foreground objects near or on cube boundaries. Supposing each pixel in a cube face is labeled as either object or background, our objective $F$ measures *the fraction of pixels that are labeled as foreground near cube boundaries.* A pixel is near cube boundaries if it is less than $A\%$ of the cube length away from the left, right, or top boundary. We do not penalize objects near the bottom boundary since it is common to place objects near the bottom boundary in photography (e.g., potraits).

To infer which pixels belong to the foreground, we use "pixel objectness", presented in Chapter 5. While other foreground methods are feasible (e.g., [225, 18, 83, 147, 125]), we choose pixel objectness due to its accuracy in detecting foreground objects of any category, as well as its ability to produce a single pixel-wise foreground map which can contain multiple objects. Figure 6.3 shows example pixel objectness foreground maps

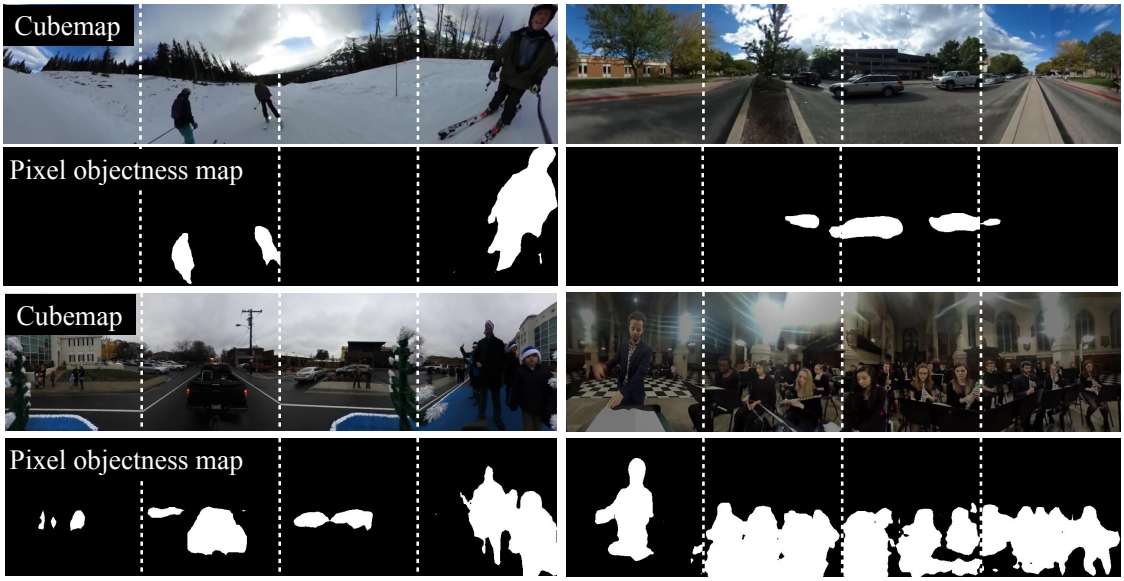


Figure 6.3: Pixel objectness (see Chapter 5) foreground map examples. White pixels in the pixel objectness map indicate foreground. Our approach learns to find cubemap orientations where the foreground objects are not disrupted by cube edges, i.e., each object falls largely within one face.

on cube faces. We apply pixel objectness to a given projected cubemap to obtain its pixel objectness score. In conjunction, other measurements for photo quality, such as interestingness [54], memorability [71], or aesthetics [32], could be employed within $F$.

### 6.1.2 Learning to Predict Snap Angles

On the one hand, a direct regression solution would attempt to infer $\theta^*$ directly from $I$. However, this is problematic because good snap angles can be multi-modal, available at multiple directions in the sphere, and thus poorly suited for regression. On the other hand, a brute force solution would require projecting the panorama to a cubemap and then evaluating $F$ for every possible projection angle $\theta$, which is costly.

We instead address snap angle prediction with reinforcement learning. The task is a time-budgeted sequential decision process—an iterative adjustment of the (virtual) camera rotation that homes in on the least distorting viewpoint for cubemap projection. Actions are cube rotations and rewards are improvements to the pixel objectness score $F$. Loosely speaking, this is reminiscent of how people take photos with a coarse-to-fine refinement towards the desired composition. However, unlike a naive coarse-to-fine search, our approach learns to trigger different search strategies depending on what is observed, as we will demonstrate in results.

Specifically, let $T$ represent the budget given to our system, indicating the number of rotations it may attempt. We maintain a history of the model's previous predictions. At each time step $t$, our framework takes a relative snap prediction $s_t$ (for example, $s_t$ could signal to update the azimuth by 45°) and updates its previous snap angle $\theta_t = \theta_{t-1} + s_t$. Then, based on its current observation, our system makes a prediction $p_t$, which is used to update the snap angle in the next time step. That is, we have $s_{t+1} = p_t$. Finally, we choose the snap angle with the lowest pixel objectness objective score from the history as our final prediction $\hat{\theta}$:

$$\hat{\theta} =_{\theta_t=\theta_1,\ldots,\theta_T} F(P(I,\theta_t)). \tag{6.3}$$

To further improve efficiency, one could compute pixel objectness *once* on a cylindrical panorama rather than recompute it for every cubemap rotation, and then proceed with the iterative rotation predictions above unchanged. However, learned foreground detectors [77, 83, 18, 147, 125] are trained on Web images in rectilinear projection, and

so their accuracy can degrade with different distortions. Thus we simply recompute the foreground for each cubemap reprojection. See Sec. 6.2.1 for run-times.

**Network** We implement our reinforcement learning task within deep recurrent and convolutional neural networks. Our framework consists of four modules: a *rotator*, a *feature extractor*, an *aggregator*, and a *snap angle predictor*. At each time step, it processes the data and produces a cubemap (*rotator*), extracts learned features (*feature extractor*), integrates information over time (*aggregator*), and predicts the next snap angle (*snap angle predictor*).

At each time step $t$, the *rotator* takes as input a panorama $I$ in equirectangular projection and a relative snap angle prediction $s_t = p_{t-1}$, which is the prediction from the previous time step. The *rotator* updates its current snap angle prediction with $\theta_t = \theta_{t-1} + s_t$. We set $\theta_1 = 0$ initially. Then the *rotator* applies the projection function $P$ to $I$ based on $\theta_t$ with Eq 6.2 to produce a cubemap. Since our objective is to minimize the total amount of foreground straddling cube face boundaries, it is more efficient for our model to learn directly from the pixel objectness map than from raw pixels. Therefore, we apply pixel objectness [77] to each of the four lateral cube faces to obtain a binary objectness map per face. The rotator has the form: $\mathbb{I}^{W \times H \times 3} \times \Theta \rightarrow \mathbb{B}^{W_c \times W_c \times 4}$, where $W$ and $H$ are the width and height of the input panorama in equirectangular projection and $W_c$ denotes the side length of a cube face. The *rotator* does not have any learnable parameters since it is used to preprocess the input data.

At each time step $t$, the *feature extractor* then applies a sequence of convolutions to the output of the *rotator* to produce a feature vector $f_t$, which is then fed into the *aggre-*

*gator* to produce an aggregate feature vector $a_t = A(f_1, ..., f_t)$ over time. Our *aggregator* is a recurrent neural network (RNN), which also maintains its own hidden state.

Finally, the *snap angle predictor* takes the aggregate feature vector as input, and produces a relative snap angle prediction $p_t$. In the next time step $t + 1$, the relative snap angle prediction is fed into the *rotator* to produce a new cubemap. The *snap angle predictor* contains two fully connected layers, each followed by a ReLU, and then the output is fed into a softmax function for the $N$ azimuth candidates. The $N$ candidates here are relative, and range from decreasing azimuth by $\frac{N}{2}$ to increasing azimuth by $\frac{N}{2}$. The *snap angle predictor* first produces a multinomial probability density function $\pi(p_t)$ over all candidate relative snap angles, then it samples one snap angle prediction proportional to the probability density function. See Figure 6.2 for an overview of the network. for all architecture details.

**Training** The parameters of our model consist of parameters of the *feature extractor*, *aggregator*, and *snap angle predictor*: $w = \{w_f, w_a, w_p\}$. We learn them to maximize the total reward (defined below) our model can expect when predicting snap angles. The *snap angle predictor* contains stochastic units and therefore cannot be trained with the standard backpropagation method. We therefore use REINFORCE [198]. Let $\pi(p_t|I, w)$ denote the parameterized policy, which is a pdf over all possible snap angle predictions. REINFORCE iteratively increases weights in the pdf $\pi(p_t|I, w)$ on those snap angles that have received higher rewards. Formally, given a batch of training data $\{I_i : i = 1, \ldots, M\}$,

we can approximate the gradient as following:

$$\sum_{i=1}^{M}\sum_{t=1}^{T}\nabla_w \log \pi(p_t^i|I_i, w)R_t^i \tag{6.4}$$

where $R_t^i$ denotes the reward at time $t$ for instance $i$.

**Reward** At each time step $t$, we compute the objective. Let $\hat{\theta}_t =_{\theta=\theta_1,\ldots\theta_t} F(P(I,\theta))$ denote the snap angle with the lowest pixel objectness until time step $t$. Let $O_t = F(P(I,\hat{\theta}_t))$ denote its corresponding objective value. The reward for time step $t$ is

$$\hat{R}_t = \min(O_t - F(P(I,\theta_t + p_t)), 0). \tag{6.5}$$

Thus, the model receives a reward proportional to the decrease in edge-straddling foreground pixels whenever the model updates the snap angle. To speed up training, we use a variance-reduced version of the reward $R_t = \hat{R}_t - b_t$ where $b_t$ is the average amount of decrease in pixel objectness coverage with a random policy at time $t$.

## 6.2 Results

Our results address **four main questions**: 1) How efficiently can our approach identify the best snap angle? (Sec. 6.2.1); 2) To what extent does the foreground "pixel objectness" objective properly capture objects important to human viewers? (Sec. 6.2.2); 3) To what extent do human viewers favor snap-angle cubemaps over the default orientation? (Sec. 6.2.3); and 4) Might snap angles aid image recognition? (Sec. 6.2.4).

**Dataset** We collect a dataset of 360° images to evaluate our approach; existing 360° datasets are topically narrow [203, 175, 69], restricting their use for our goal. We use

YouTube with the 360° filter to gather videos from four activity categories—Disney, Ski, Parade, and Concert. After manually filtering out frames with only text or blackness, we have 150 videos and 14,076 total frames sampled at 1 FPS. The dataset can be found at `http://vision.cs.utexas.edu/projects/snapangle/`.

**Implementation details** We implement our model with Torch, and optimize with stochastic gradient and REINFORCE. We set the base learning rate to 0.01 and use momentum. We fix $A = 6.25\%$ for all results after visual inspection of a few human-taken cubemaps (not in the test set).

### 6.2.1 Efficient Snap Angle Prediction

We first evaluate our snap angle prediction framework. We use all 14,076 frames, 75% for training and 25% for testing. We ensure testing and training data do *not* come from the same video. We define the following baselines:

- RANDOM ROTATE: Given a budget $T$, predict $T$ snap angles randomly (with no repetition).
- UNIFORM ROTATE: Given a budget $T$, predict $T$ snap angles uniformly sampled from all candidates. When $T = 1$, UNIFORM receives the CANONICAL view. This is a strong baseline since it exploits the human recording bias in the starting view. Despite the 360° range of the camera, photographers still tend to direct the "front" of the camera towards interesting content, in which case CANONICAL has some manual intelligence built-in.

- COARSE-TO-FINE SEARCH: Divide the search space into two uniform intervals and search the center snap angle in each interval. Then recursively search the better interval, until the budget is exhausted.
- PANO2VID(P2V) [175]-ADAPTED: We implement a snap angle variant inspired by the pipeline of Pano2Vid [175]. We replace C3D [188] features (which require video) used in [175] with F7 features from VGG [166] and train a logistic classifier to learn "capture-worthiness" [175] with Web images and randomly sampled panorama subviews. For a budget $T$, we evaluate $T$ "glimpses" and choose the snap angle with the highest encountered capture-worthiness score. We stress that Pano2Vid addresses a different task: it creates a normal field-of-view video (discarding the rest) whereas we create a well-oriented omnidirectional image. Nonetheless, we include this baseline to test their general approach of learning a framing prior from human-captured data.
- SALIENCY: Select the angle that centers a cube face around the maximal saliency region. Specifically, we compute the panorama's saliency map [125] in equirectangular form and blur it with a Gaussian kernel. We then identify the $P \times P$ pixel square with the highest total saliency value, and predict the snap angle as the center of the square. Unlike the other methods, this baseline is not iterative, since the maximal saliency region does not change with rotations. We use a window size $P = 30$. Performance is not sensitive to $P$ for $20 \leq P \leq 200$.

We train our approach for a spectrum of budgets $T$, and report results in terms of the amount of foreground disruption as a function of the budget. Each unit of the budget

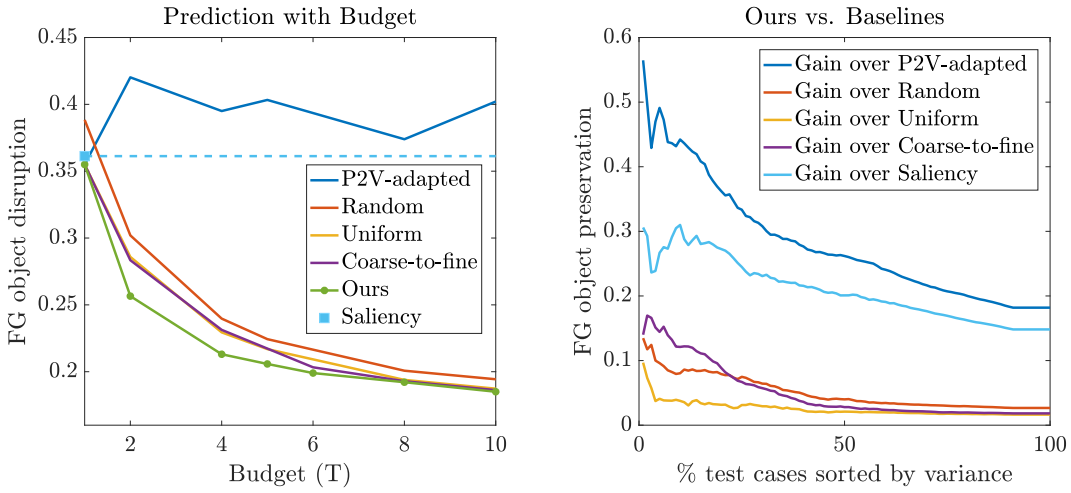


Figure 6.4: Predicting snap angles in a timely manner. Left: Given a budget, our method predicts snap angles with the least foreground disruption on cube edges. Gains are larger for smaller budgets, demonstrating our method's efficiency. Right: Our gain over the baselines (for a budget $T = 4$) as a function of the test cases' decreasing "difficulty", i.e., the variance in ground truth quality for candidate angles. See text.

corresponds to one round of rotating, re-rendering, and predicting foregrounds. We score foreground disruption as the average $F(P(I, \theta_t^*))$ across all four faces.

Figure 6.4 (left) shows the results. Our method achieves the least disruptions to foreground regions among all the competing methods. UNIFORM ROTATE and COARSE-TO-FINE SEARCH perform better than RANDOM because they benefit from hand-designed search heuristics. Unlike UNIFORM ROTATE and COARSE-TO-FINE SEARCH, our approach is content-based and learns to trigger different search strategies depending on what it observes. When $T = 1$, SALIENCY is better than RANDOM but it underperforms our method and UNIFORM. SALIENCY likely has difficulty capturing important objects in panoramas, since the saliency model is trained with standard field-of-view images. Di-

rectly adapting PANO2VID [175] for our problem results in unsatisfactory results. A capture-worthiness classifier [175] is relatively insensitive to the placement of important objects/people and therefore less suitable for the snap angle prediction task, which requires detailed modeling of object placement on *all* faces of the cube.

Figure 6.4 (right) plots our gains sorted by the test images' decreasing "difficulty" for a budget $T = 4$. In some test images, there is a high variance, meaning certain snap angles are better than others. However, for others, all candidate rotations look similarly good, in which case all methods will perform similarly. The righthand plot sorts the test images by their variance (in descending order) in quality across all possible angles, and reports our method's gain as a function of that difficulty. Our method outperforms P2V-ADAPTED, SALIENCY, COARSE-TO-FINE SEARCH, RANDOM and UNIFORM by up to 56%, 31%, 17%, 14% and 10% (absolute), respectively. Overall Figure 6.4 demonstrates that our method predicts the snap angle more efficiently than the baselines.

We have thus far reported efficiency in terms of abstract budget usage. One unit of budget entails the following: projecting a typical panorama of size $960 \times 1920$ pixels in equirectangular form to a cubemap (8.67 seconds with our Matlab implementation) and then computing pixel objectness (0.57 seconds). Our prediction method is very efficient and takes $0.003$ seconds to execute for a budget $T = 4$ with a GeForce GTX 1080 GPU. Thus, for a budget $T = 4$, the savings achieved by our method is approximately 2.4 minutes (5x speedup) per image compared to exhaustive search. Note that due to our method's efficiency, even if the Matlab projections were 1000x faster for all methods, our 5x speedup over the baseline would remain the same. Our method achieves a good tradeoff between speed and accuracy.

### 6.2.2 Justification for Foreground Object Objective

Next we justify empirically the pixel objectness cube-edge objective. To this end, we have human viewers identify important objects in the source panoramas, then evaluate to what extent our objective preserves them.

Specifically, we randomly select 340 frames among those where: 1) Each frame is at least 10-seconds apart from the rest in order to ensure diversity in the dataset; 2) The difference in terms of overall pixel objectness between our method and the canonical view method is non-neglible. We collect annotations via Amazon Mechanical Turk. Following the interface of [69], we present crowdworkers the panorama and instruct them to label any "important objects" with a bounding box—as many as they wish.[2]

Here we consider PANO2VID(P2V) [175]-ADAPTED and SALIENCY as defined in Sec. 6.2.1 and two additional baselines: 1) CANONICAL VIEW: produces a cubemap using the camera-provided orientation; 2) RANDOM VIEW: rotates the input panorama by an arbitrary angle and then generates the cubemap. Note that the other baselines in Sec. 6.2.1 are not applicable here, since they are search mechanisms.

Consider the cube face $X$ that contains the largest number of foreground pixels from a given bounding box after projection. We evaluate the cubemaps of our method and the baselines based on the overlap score (IoU) between the foreground region from the cube face $X$ and the corresponding human-labeled important object, for each bounding box. This metric is maximized when all pixels for the same object project to the same cube face; higher overlap indicates better preservation of important objects.

[2]The 360° sports data [69] annotates only a single point on where annotators think a human should look,

| | CANONICAL | RANDOM | SALIENCY | P2V-ADAPTED | OURS | UPPERBOUND |
|---|---|---|---|---|---|---|
| Concert | 77.6% | 73.9% | 76.2% | 71.6% | **81.5%** | 86.3% |
| Ski | 64.1% | 72.5% | 68.1% | 70.1% | **78.6%** | 83.5% |
| Parade | 84.0% | 81.2% | 86.3% | 85.7% | **87.6%** | 96.8% |
| Disney | 58.3% | 57.7% | 60.8% | 60.8% | **65.5%** | 77.4% |
| All | 74.4% | 74.2% | 76.0% | 75.0% | **81.1%** | 88.3% |

Table 6.1: Performance on preserving the integrity of objects explicitly identified as important by human observers. Higher overlap scores are better. Our method outperforms both baselines.

Table 6.1 shows the results. Our method outperforms all baselines by a large margin. This supports our hypothesis that avoiding foreground objects along the cube edges helps preserve objects of interest to a viewer. Snap angles achieve this goal much better than the baseline cubemaps. The UPPERBOUND corresponds to the maximum possible overlap achieved if exhaustively evaluating *all* candidate angles, and helps gauge the difficulty of each category. Parade and Disney have the highest and lowest upper bounds, respectively. In Disney images, the camera is often carried by the recorders, so important objects/persons appear relatively large in the panorama and cannot fit in a single cube face, hence a lower upper bound score. On the contrary, in Parade images the camera is often placed in the crowd and far away from important objects, so each can be confined to a single face. The latter also explains why the baselines do best (though still weaker than ours) on Parade images.

| | Prefer OURS | Tie | Prefer CANONICAL | Prefer OURS | Tie | Prefer RANDOM |
|---|---|---|---|---|---|---|
| Parade | 54.8% | 16.5% | 28.7% | 70.4% | 9.6% | 20.0% |
| Concert | 48.7% | 16.2% | 35.1% | 52.7% | 16.2% | 31.1% |
| Disney | 44.8% | 17.9% | 37.3% | 72.9% | 8.5% | 18.6% |
| Ski | 64.3% | 8.3% | 27.4% | 62.9% | 16.1% | 21.0% |
| All | 53.8% | 14.7% | 31.5% | 65.3% | 12.3% | 22.4% |

Table 6.2: User study result comparing cubemaps outputs for perceived quality. Left: Comparison between our method and CANONICAL. Right: Comparison between our method and RANDOM.

### 6.2.3 User Study: Perceived Quality

Having justified the perceptual relevance of the cube-edge foreground objective (Sec. 6.2.2), next we perform a user study to gauge perceptual quality of our results. Do snap angles produce cube faces that look like human-taken photos? We evaluate on the same image set used in Sec. 6.2.2.

We present cube faces produced by our method and one of the baselines at a time in arbitrary order and inform subjects the two sets are photos from the same scene but taken by different photographers. We instruct them to consider composition and viewpoint in order to decide which set of photos is more pleasing. To account for the subjectivity of the task, we issue each sample to 5 distinct workers and aggregate responses with majority vote. 98 unique MTurk crowdworkers participated in the study.

Table 6.2 shows the results. Our method outperforms the CANONICAL baseline by more than 22% and the RANDOM baseline by 42.9%. This result supports our claim

but we need to know the spatial extent of the objects.


Figure 6.5: Qualitative examples of default CANONICAL cubemaps and our snap angle cubemaps. Our method produces cubemaps that place important objects/persons in the same cube face to preserve the foreground integrity. Bottom two rows show failure cases. In the bottom left, pixel objectness [77] does not recognize the round stage as foreground, and therefore our method splits the stage onto two different cube faces, creating a distorted heart-shaped stage.

| | Concert | Ski | Parade | Disney | All (normalized) |
|---|---|---|---|---|---|
| Image Memorability [92] | | | | | |
| CANONICAL | **71.58** | 69.49 | 67.08 | 70.53 | 46.8% |
| RANDOM | 71.30 | 69.54 | 67.27 | 70.65 | 48.1% |
| SALIENCY | 71.40 | 69.60 | 67.35 | 70.58 | 49.9% |
| P2V-ADAPTED | 71.34 | 69.85 | 67.44 | 70.54 | 52.1% |
| OURS | 71.45 | **70.03** | **67.68** | **70.87** | **59.8%** |
| UPPER | 72.70 | 71.19 | 68.68 | 72.15 | – |
| Image Aesthetics [100] | | | | | |
| CANONICAL | 33.74 | 41.95 | 30.24 | 32.85 | 44.3% |
| RANDOM | 32.46 | 41.90 | 30.65 | 32.79 | 42.4% |
| SALIENCY | 34.52 | 41.87 | 30.81 | 32.54 | 47.9% |
| P2V-ADAPTED | 34.48 | 41.97 | 30.86 | **33.09** | 48.8% |
| OURS | **35.05** | **42.08** | **31.19** | 32.97 | **52.9%** |
| UPPER | 38.45 | 45.76 | 34.74 | 36.81 | – |

Table 6.3: Memorability and aesthetics scores.

that by preserving object integrity, our method produces cubemaps that align better with human perception of quality photo composition. Figure 6.5 shows qualitative examples. As shown in the first two examples (top two rows), our method is able to place an important person in the same cube face whereas the baseline splits each person and projects a person onto two cube faces. We also present two failure cases in the last two rows. In the bottom left, pixel objectness does not recognize the stage as foreground, and therefore our method places the stage on two different cube faces, creating a distorted heart-shaped stage.

So far, Table 6.1 confirms empirically that our foreground-based objective does preserve those objects human viewers deem important, and Table 6.2 shows that human viewers have an absolute preference for snap angle cubemaps over other projections. As

| | CANONICAL | RANDOM | OURS |
|---|---|---|---|
| Single | 68.5 | 69.4 | **70.1** |
| Pano | 66.5 | 67.0 | **68.1** |

Table 6.4: Image recognition accuracy (%). Snap angles help align the 360° data's statistics with that of normal FOV Web photos, enabling easier transfer from conventional pretrained networks.

a final test of snap angle cubemaps' perceptual quality, we score them using state-of-the-art metrics for *aesthetics* [100] and *memorability* [92]. Since both models are trained on images annotated by people (for their aesthetics and memorability, respectively), higher scores indicate higher correlation with these perceived properties (though of course no one learned metric can perfectly represent human opinion).

Table 6.3 shows the results. We report the raw scores $s$ per class as well as the score over all classes, normalized as $\frac{s-s_{min}}{s_{max}-s_{min}}$, where $s_{min}$ and $s_{max}$ denote the lower and upper bound, respectively. Because the metrics are fairly tolerant to local rotations, there is a limit to how well they can capture subtle differences in cubemaps. Nonetheless, our method outperforms the baselines overall.

### 6.2.4 Cubemap Recognition from Pretrained Nets

Since snap angles provide projections that better mimic human-taken photo composition, we hypothesize that they also align better with conventional FOV images, compared to cubemaps in their canonical orientation. This suggests that snap angles may better align with Web photos (typically used to train today's recognition systems), which in turn could help standard recognition models perform well on 360° panoramas. We present a

preliminary proof-of-concept experiment to test this hypothesis.

We train a multi-class CNN classifier to distinguish the four activity categories in our 360° dataset (Disney, Parade, etc.). The classifier uses ResNet-101 [62] pretrained on ImageNet [160] and fine-tuned on 300 training images per class downloaded from Google Image Search. Note that in all experiments until now, the category labels on the 360° dataset were invisible to our algorithm. We randomly select 250 panoramas per activity as a test set. Each panorama is projected to a cubemap with the different projection methods, and we compare the resulting recognition rates.

Table 6.4 shows the results. We report recognition accuracy in two forms: *Single*, which treats each individual cube face as a test instance, and *Pano*, which classifies the entire panorama by multiplying the predicted posteriors from all cube faces. For both cases, snap angles produce cubemaps that achieve the best recognition rate. That said, the margin is slim, and the full impact of snap angles for recognition warrants further exploration. Still, this result hints at the potential for snap angles to be a bridge between pretrained normal FOV networks on the one hand and 360° images on the other hand.

## 6.3 Summary

In this chapter, I showed how to predict viewing angles to enhance the viewing experience for 360° panoramas. In contrast to previous work that assumes either a fixed or manually supplied projection angle, I propose to automatically predict the angle that will best preserve detected foreground objects.

My solution only considers snap angles for 360° images. Future work will explore

ways to generalize snap angles to video data. In addition, my solution relies on the cube-map projection model and therefore cannot effectively handle the case when a foreground object is too large to fit in a single cube face. Future work will explore ways to address how to handle large objects.

Now I have presented all four components of my thesis. In the next chapter, I will outline some possible directions for future research and conclude my thesis.

# Chapter 7

# Future Work

Passive cameras (e.g., wearable cameras, 360° cameras) offer a more relaxing experience to record our visual world but they do not always capture frames that look like intentional human-taken photos. My thesis aims to narrow the gap between the quality of visual data captured by passive cameras and by intentional human photographers. In the previous chapters, I have described my thesis research which develops a framework to compose photos and videos automatically from passive cameras. In this chapter, I discuss possible directions for future research.

**Leveraging multi-modality data** In my thesis, all the proposed frameworks only consider visual cues to compose photos or videos from passive cameras. However, other forms of auxiliary data could also provide valuable cues to enhance photo and video composition. I discuss two types of auxiliary data for possible future research.

In the recording stage, most passive cameras can also record auxiliary data such as audio, GPS coordinates, and IMU data. Audio data could provide useful cues to understand user activities and user intention. The proposed framework in Chapter 3 only considers visual cues for video highlight detection. However, visual cues alone are not always enough to find the best moments in unedited videos. In a recording of casual conversation or public speech, visual cues often remain similar throughout but audio data can

indicate the best moments. In sports games, the noise from the audience could also indicate the best moments in the games. Audio data provides a complementary source of information to understand videos. I believe understanding audio data and other auxiliary data is a promising direction for future research.

Furthermore, when users share videos in the social media platforms, the videos often come with user generated descriptions such as tags, captions and comments. The descriptions could be utilized to understand the content of the user videos. This opens up possibilities for new frameworks that can leverage both visual data and natural languages. I believe jointly understanding visual data with user generated description is also a promising future direction.

**Moving from passive cameras to active cameras:** Majority of the work in my thesis has employed a "passive online capture followed by an active offline processing" paradigm. The proposed frameworks in this thesis always rely on passive cameras to first capture visual data, and then intelligently compose photos and videos from passively captured data. Much of the effort in my thesis has been focused on how to design intelligent machine systems for offline visual data processing. My proposed frameworks can narrow the gap between the quality of visual data captured by "unintentional" photographers with passive cameras and by intentional human photographers. However, the quality of the visual data after offline processing is still upper bounded by the best quality among all the passively captured data. The first and second components of my thesis presented in Chapter 3 and Chapter 4 aim to find the best moments—in terms of either short video clips or keyframes— from passive cameras. We implicitly assume that passively captured

data should contain some well-composed moments. If the assumption does not hold, my proposed methods become much less effective.

However, the next-generation camera systems do not necessarily need to follow a "passive online capture" paradigm. In the recording stage, passive cameras can be replaced with *active cameras*, which can automatically capture well-composed photos and videos in the recording stage. I believe future research on active cameras will revolutionize the photography experience. Active cameras will provide users with a hands-free way of recording while produce photos of professional quality. Compared to passive cameras, active cameras would automatically composes photos in the recording stage and therefore have several advantages. First, the quality of the photos is no longer limited by the passively captured data. Instead, we directly design intelligent imaging systems and optimize the captured photo quality in the recording stage. Furthermore, active cameras can decide what data to capture and store in the recording stage and therefore do not need extra storage to keep everything they observe. Active cameras can also support other applications. In the case of surveillance videos, active cameras could automatically track abnormal behaviors.

# Chapter 8

# Conclusion

In the previous chapters, I have presented all four components of my thesis on learning to compose photos and videos from passive cameras and possible research directions for future work. In particular, I have presented:

- *Learning highlight detection from video duration*, in Chapter 3
- *Detecting snap points in egocentric video with a Web photo prior*, in Chapter 4
- *Learning to segment generic objects in images and videos*, in Chapter 5
- *Snap angle prediction for 360° panoramas*, in Chapter 6

Wearable and 360° cameras have already revolutionized the photography experience, yet it is still challenging to directly produce professional quality photos from these passive cameras. The main contribution of my thesis is to develop a framework that aims to narrow the gap between the quality of visual data captured by "unintentional" photographers with passive cameras and by intentional human photographers. My thesis provides solutions to the the following problems in the context of passive cameras: 1) what visual data to capture and store, 2) how to identify foreground objects, and 3) how to enhance the viewing experience.

Throughout, I validate the strength of the proposed frameworks on multiple challenging datasets against a variety of previously established state-of-the-art methods and other pertinent baselines. Our experiments demonstrate the following: 1) our method can automatically identify the best moments from unedited videos; 2) our segmentation method substantially improves the state-of-the-art on foreground segmentation in images and videos and also benefits automatic photo composition; 3) our viewing angle prediction for 360° imagery can enhance the viewing experience. Although my thesis mainly focuses on passive cameras, a portion of the proposed methods are also applicable to general user generated videos.

I believe the next-generation cameras will remain light-weight while having the ability to automatically compose photos and videos with quality that can match or even exceed professional human photographer level.

# Bibliography


[1] https://code.facebook.com/posts/1638767863078802/under-the-hood-building-360-video/.

[2] https://www.blog.google/products/google-vr/bringing-pixels-front-and-center-vr-video/.

[3] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[4] B. Alexe, T. Deselaers, and V. Ferrari. What is an Object? In *CVPR*, 2010.

[5] Humam Alwassel, Fabian Caba Heilbron, and Bernard Ghanem. Action search: Learning to search for human activities in untrimmed videos. *arXiv preprint arXiv:1706.04269*, 2017.

[6] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014.

[7] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. In *ACM Trans on Graphics*, 2007.

[8] Vijay Badrinarayanan, Fabio Galasso, and Roberto Cipolla. Label propagation in video sequences. In *CVPR*, 2010.

[9] X. Bai, J. Wang, D. Simons, and G. Sapiro. Video snapcut: Robust video object cutout using localized classifiers. In *SIGGRAPH*, 2009.

[10] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 2011.

[11] Pia Bideau and Erik Learned-Miller. Its moving! a probabilistic model for causal motion segmentation in moving camera videos. In *ECCV*, 2016.

[12] A. Borji, M. M. Cheng, H. Jiang, and J. Li. Salient object detection: A benchmark. *IEEE Transactions on Image Processing*, 2015.

[13] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Patt Rec Letters*, 2009.

[14] 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.

[15] Sijia Cai, Wangmeng Zuo, Larry S Davis, and Lei Zhang. Weakly-supervised video summarization using variational encoder-decoder and web prior. In *ECCV*, 2018.

[16] Juan C Caicedo and Svetlana Lazebnik. Active object localization with deep reinforcement learning. In *ICCV*, 2015.

[17] J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[18] Joao Carreira and Cristian Sminchisescu. Cpmc: Automatic object segmentation

using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[19] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

[20] Robert Carroll, Aseem Agarwala, and Maneesh Agrawala. Image warps for artistic perspective manipulation. In *ACM Transactions on Graphics (TOG)*, 2010.

[21] Robert Carroll, Maneesh Agrawala, and Aseem Agarwala. Optimizing content-preserving projections for wide-angle images. *ACM Transactions on Graphics (TOG)*, 2009.

[22] Che-Han Chang, Min-Chun Hu, Wen-Huang Cheng, and Yung-Yu Chuang. Rectangling stereographic projection for wide-angle image visualization. In *ICCV*, 2013.

[23] C.-Y. Chen and K. Grauman. Clues from the Beaten Path: Location Estimation with Bursty Sequences of Tourist Photos. In *CVPR*, 2011.

[24] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015.

[25] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Enriching Visual Knowledge Bases via Object Discovery and Segmentation. In *CVPR*, 2014.

[26] Ming-Ming Cheng, Guo-Xin Zhang, Niloy J. Mitra, Xiaolei Huang, and Shi-Min Hu. Global contrast based salient region detection. In *CVPR*, 2011.

[27] Wen-Sheng Chu, Yale Song, and Alejandro Jaimes. Video co-summarization: Video summarization by visual co-occurrence. In *CVPR*, 2015.

[28] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[29] F. Crete-Roffet, T. Dolmiere, P. Ladret, and M. Nicolas. The blur effect: Perception and estimation with a new no-reference perceptual blur metric. In *SPIE*, 2007.

[30] 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.

[31] Thomas Deselaers, Bogdan Alexe, and Vittorio Ferrari. Weakly supervised localization and learning with generic knowledge. *IJCV*, 2012.

[32] Sagnik Dhar, Vicente Ordonez, and Tamara L Berg. High level describable attributes for predicting aesthetics and interestingness. In *CVPR*, 2011.

[33] 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.

[34] I. Endres and D. Hoiem. Category independent object proposals. In *ECCV*, 2010.

[35] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *ECCV*, 2016.

[36] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 2010.

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

[38] A. Fathi, M. Balcan, X. Ren, and J. Rehg. Combining self training and active learning for video segmentation. In *BMVC*, 2011.

[39] A. Fathi, A. Farhadi, and J. Rehg. Understanding Egocentric Activities. In *ICCV*, 2011.

[40] A. Fathi, J. Hodgins, and J. Rehg. Social interactions: a first-person perspective. In *CVPR*, 2012.

[41] A. Fathi and J. Rehg. Modeling actions through state changes. In *CVPR*, 2013.

[42] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[43] Juliet Fiss, Aseem Agarwala, and Brian Curless. Candid portrait selection from video. In *TOG*, 2011.

[44] Alex Flint, Ian Reid, and David Murray. Learning texton models for real-time scene context. In *CVPR Workshops*, 2009.

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

[46] Fabio Galasso, Roberto Cipolla, and Bernt Schiele. Video segmentation with superpixels. In *ACCV*, 2012.

[47] Martin Godec, Peter M. Roth, and Horst Bischof. Hough-based tracking of non-rigid objects. In *ICCV*, 2011.

[48] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

[49] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *NIPS*, 2014.

[50] Ned Greene. Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications*, 1986.

[51] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. In *CVPR*, 2010.

[52] Matthieu Guillaumin, Daniel Küttel, and Vittorio Ferrari. Imagenet auto-annotation with segmentation propagation. *IJCV*, 2014.

[53] D. Gurari, S. Jain, M. Betke, and K. Grauman. Predicting if computers or humans should segment images. In *CVPR*, 2016.

[54] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, Fabian Nater, and Luc Van Gool. The interestingness of images. In *ICCV*, 2013.

[55] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In *ECCV*, 2014.

[56] Michael Gygli, Helmut Grabner, and Luc Van Gool. Video summarization by learning submodular mixtures of objectives. In *CVPR*, 2015.

[57] Michael Gygli, Yale Song, and Liangliang Cao. Video2gif: Automatic generation of animated gifs from video. In *CVPR*, 2016.

[58] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, 2018.

[59] Bharath Hariharan, Pablo Arbelaez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision (ICCV)*, 2011.

[60] J. Hays and A. Efros. im2gps: estimating geographic information from a single image. In *CVPR*, 2008.

[61] James Hays and Alexei A Efros. Scene completion using millions of photographs. In *ACM Transactions on Graphics (TOG)*, 2007.

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

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

[64] J. Healey and R. Picard. Startlecam: a cybernetic wearable camera. In *Wearable Computers*, 1998.

[65] S. Hodges, L. Williams, E. Berry, S. Izadi, J. Srinivasan, A. Butler, G. Smyth, N. Kapur, and K. Wood. SenseCam: a retrospective memory aid. In *UBICOMP*, 2006.

[66] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 2007.

[67] Derek Hoiem, Martial Hebert, and Andrew Stein. Learning to find object boundaries using motion cues. *ICCV*, 2007.

[68] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[69] Hou-Ning Hu, Yen-Chen Lin, Ming-Yu Liu, Hsien-Tzu Cheng, Yung-Ju Chang, and Min Sun. Deep 360 pilot: Learning a deep agent for piloting through 360 sports videos. In *CVPR*, 2017.

[70] Maximilian Ilse, Jakub M Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *ICML*, 2018.

[71] Phillip Isola, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. What makes an image memorable? In *CVPR*, 2011.

[72] Phillip Isola, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. What makes an image memorable? In *CVPR*, 2011.

[73] S. Jain and K. Grauman. Predicting sufficent annotation strength for interactive foreground segmentation. In *ICCV*, 2013.

[74] S. Jain and K. Grauman. Active image segmentation propagation. In *CVPR*, 2016.

[75] Suyog Dutt Jain and Kristen Grauman. Supervoxel-consistent foreground propagation in video. In *ECCV*. 2014.

[76] 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.

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

[78] Varun Jampani, Raghudeep Gadde, and Peter V Gehler. Video propagation networks. In *CVPR*, 2017.

[79] Dinesh Jayaraman and Kristen Grauman. Look-ahead before you leap: End-to-end active recognition by forecasting the effect of motion. In *ECCV*, 2016.

[80] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[81] Bowen Jiang, Lihe Zhang, Huchuan Lu, Chuan Yang, and Ming-Hsuan Yang. Saliency detection via absorbing Markov chain. In *ICCV*, 2013.

[82] Bowen Jiang, Lihe Zhang, Huchuan Lu, Chuan Yang, and Ming-Hsuan Yang. Saliency detection via absorbing Markov chain. In *ICCV*, 2013.

[83] P. Jiang, H. Ling, J. Yu, and J. Peng. Salient region detection by ufo: Uniqueness, focusness, and objectness. In *ICCV*, 2013.

[84] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *CVPR*, 2010.

[85] A. Joulin, F. Bach, and J. Ponce. Multi-class cosegmentation. In *CVPR*, 2012.

[86] E. Kalogerakis, O. Vesselova, J. Hays, A. Efros, and A. Hertzmann. Image sequence geolocation with human travel priors. In *ICCV*, 2009.

[87] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[88] Y. Ke, X. Tang, and F. Jing. The design of high-level features for photo quality assessment. In *CVPR*, 2006.

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

[90] A. Khosla, R. Hamid, C.-J. Lin, and N. Sundaresan. Large-scale video summarization using web-image priors. In *CVPR*, 2013.

[91] Aditya Khosla, Raffay Hamid, Chih-Jen Lin, and Neel Sundaresan. Large-scale video summarization using web-image priors. In *CVPR*, 2013.

[92] Aditya Khosla, Akhil S. Raju, Antonio Torralba, and Aude Oliva. Understanding and predicting image memorability at a large scale. In *ICCV*, 2015.

[93] G. Kim, L. Sigal, and E. Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *CVPR*, 2014.

[94] G. Kim and E. Xing. Jointly aligning and segmenting multiple web photo streams for the inference of collective photo storylines. In *CVPR*, 2013.

[95] G.H. Kim, E.P. Xing, L. Fei Fei, and T. Kanade. Distributed cosegmentation via submodular optimization on anisotropic diffusion. In *ICCV*, 2011.

[96] Gunhee Kim and Eric P Xing. Reconstructing storyline graphs for image recommendation from web community photos. In *CVPR*, 2014.

[97] Yeong Won Kim, Dae-Yong Jo, Chang-Ryeol Lee, Hyeok-Jae Choi, Yong Hoon Kwon, and Kuk-Jin Yoon. Automatic content-aware projection for 360 videos. In *ICCV*, 2017.

[98] K. Kitani, T. Okabe, Y. Sato, and A. Sugimoto. Fast unsupervised ego-action learning for first-person sports videos. In *CVPR*, 2011.

[99] Yeong Jun Koh and Chang-Su Kim. Primary object segmentation in videos based on region augmentation and reduction. In *CVPR*, 2017.

[100] Shu Kong, Xiaohui Shen, Zhe Lin, Radomir Mech, and Charless Fowlkes. Photo aesthetics ranking network with attributes and content adaptation. In *ECCV*, 2016.

[101] Johannes Kopf, Dani Lischinski, Oliver Deussen, Daniel Cohen-Or, and Michael Cohen. Locally adapted projections to reduce panorama distortions. In *Computer Graphics Forum*. Wiley Online Library, 2009.

[102] Johannes Kopf, Matt Uyttendaele, Oliver Deussen, and Michael F Cohen. Capturing and viewing gigapixel images. In *ACM Transactions on Graphics*. ACM, 2007.

[103] J. Kosecka and W. Zhang. Video compass. In *ECCV*, 2002.

[104] Daniel Kuettel and Vittorio Ferrari. Figure-ground segmentation by transferring window masks. In *CVPR*, 2012.

[105] Kuan-Ting Lai, Felix X Yu, Ming-Syan Chen, and Shih-Fu Chang. Video event detection by inferring temporal instance labels. In *CVPR*, 2014.

[106] Wei-Sheng Lai, Yujia Huang, Neel Joshi, Chris Buehler, Ming-Hsuan Yang, and Sing Bing Kang. Semantic-driven generation of hyperlapse from 360 video. *CoRR*, 2017.

[107] Y. J. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012.

[108] Yong Jae Lee, Joydeep Ghosh, and Kristen Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012.

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

[110] Dongping Li, Kaiming He, Jian Sun, and Kun Zhou. A geodesic-preserving method for image warping. In *CVPR*, 2015.

[111] Fuxin Li, Taeyoung Kim, Ahmad Humayun, David Tsai, and James M. Rehg. Video Segmentation by Tracking Many Figure-Ground Segments. In *ICCV*, 2013.

[112] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *ECCV*, 2008.

[113] Xi Li, Liming Zhao, Lina Wei, Ming-Hsuan Yang, Fei Wu, Yueting Zhuang, Haibin Ling, and Jingdong Wang. DeepSaliency: Multi-task deep neural network model for salient object detection. *IEEE TIP*, 2016.

[114] Yin Li, Alireza Fathi, and James M. Rehg. Learning to predict gaze in egocentric video. In *ICCV*, 2013.

[115] Yin Li, Xiaodi Hou, Christof Koch, James M. Rehg, and Alan L. Yuille. The secrets of salient object segmentation. *CVPR*, 2014.

[116] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *ICCV*, 2017.

[117] T.Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[118] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: label transfer via dense scene alignment. In *CVPR*, 2009.

[119] Ce Liu. *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, Citeseer, 2009.

[120] D. Liu, G. Hua, and T. Chen. A hierarchical visual model for video object summarization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[121] Nian Liu, Junwei Han, Dingwen Zhang, Shifeng Wen, and Tianming Liu. Predicting eye fixations using convolutional neural networks. In *CVPR*, 2015.

[122] T. Liu and J. Kender. Optimization algorithms for the selection of key frame sequences of variable length. In *ECCV*, 2002.

[123] T. Liu, J. Sun, N. Zheng, X. Tang, and H. Shum. Learning to detect a salient object. In *CVPR*, 2007.

[124] T. Liu and D. Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[125] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[126] W. Liu, T. Mei, Y. Zhang, C. Che, and J. Luo. Multi-task deep visual-semantic embedding for video thumbnail selection. In *CVPR*, 2015.

[127] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, 2015.

[128] Z. Lu and K. Grauman. Story-driven summarization for egocentric video. In *CVPR*, 2013.

[129] Zheng Lu and Kristen Grauman. Story-driven summarization for egocentric video. In *CVPR*, 2013.

[130] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015.

[131] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018.

[132] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. Unsupervised video summarization with adversarial lstm networks. In *CVPR*, 2017.

[133] S. Mann. Wearcam (the wearable camera): Personal imaging systems for long term use in wearable tetherless computer mediated reality and personal photo/videographic memory prosthesis. In *Wearable Computers*, 1998.

[134] Nicolas Märki, Federico Perazzi, Oliver Wang, and Alexander Sorkine-Hornung. Bilateral space video segmentation. In *CVPR*, 2016.

[135] Stefan Mathe, Aleksis Pirinen, and Cristian Sminchisescu. Reinforcement learning for visual object detection. In *CVPR*, 2016.

[136] Engin Mendi, Hélio B Clemente, and Coskun Bayrak. Sports video summarization based on motion analysis. *Computers & Electrical Engineering*, 2013.

[137] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *NIPS*, 2014.

[138] N. Natarajan, I. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. In *NIPS*, 2015.

[139] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015.

[140] Dan Oneata, Jerome Revaud, Jakob Verbeek, and Cordelia Schmid. Spatio-temporal object detection proposals. In *ECCV*, 2014.

[141] Junting Pan, Kevin McGuinness, Elisa Sayrol, Noel O'Connor, and Xavier Giro-i Nieto. Shallow and deep convolutional networks for saliency prediction. 2016.

[142] Rameswar Panda, Abir Das, Ziyan Wu, Jan Ernst, and Amit K Roy-Chowdhury. Weakly supervised summarization of web videos. In *ICCV*, 2017.

[143] Rameswar Panda and Amit K Roy-Chowdhury. Collaborative summarization of topic-related videos. In *CVPR*, 2017.

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

[145] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016.

[146] Federico Perazzi, Oliver Wang, Markus Gross, and Alexander Sorkine-Hornung. Fully connected object proposals for video segmentation. In *ICCV*, 2015.

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

[148] Pedro O. Pinheiro, Ronan Collobert, and Piotr Dollr. Learning to segment object candidates. In *NIPS*, 2015.

[149] Hamed Pirsiavash and Deva Ramanan. Detecting activities of daily living in first-person camera views. In *CVPR*, 2012.

[150] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In *ECCV*, 2014.

[151] Alessandro Prest, Christian Leistner, Javier Civera, Cordelia Schmid, and Vittorio Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012.

[152] Brian L. Price, Bryan S. Morse, and Scott Cohen. Livecut: Learning-based interactive video segmentation by evaluation of multiple propagated cues. In *ICCV*, 2009.

[153] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[154] X. Ren and J. Malik. Tracking as repeated figure/ground segmentation. In *CVPR*, 2007.

[155] Xiaofeng Ren and Chunhui Gu. Figure-ground segmentation improves handled object recognition in egocentric video. In *CVPR*, 2010.

[156] C. Rother, V. Kolmogorov, and A. Blake. Grabcut -interactive foreground extraction using iterated graph cuts. In *SIGGRAPH*, 2004.

[157] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, 2013.

[158] Yong Rui, Anoop Gupta, and Alex Acero. Automatically extracting highlights for tv baseball programs. In *ACM Multimedia*, 2000.

[159] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

[160] 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. *IJCV*, 2015.

[161] M. Ryoo and L. Matthies. First-person activity recognition: What are they doing to me? In *CVPR*, 2013.

[162] G. Shakhnarovich, P. Viola, and T. Darrell. Fast Pose Estimation with Parameter-Sensitive Hashing. In *ICCV*, 2003.

[163] Naveen Shankar Nagaraja, Frank R. Schmidt, and Thomas Brox. Video segmentation with just a few strokes. In *ICCV*, 2015.

[164] Thomas K Sharpless, Bruno Postle, and Daniel M German. Pannini: a new projection for rendering wide angle perspective images. In *International Conference on Computational Aesthetics in Graphics, Visualization and Imaging*, 2010.

[165] I. Simon and S. Seitz. Scene segmentation using the wisdom of crowds. In *ECCV*, 2008.

[166] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[167] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[168] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[169] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *CVPR*, 2016.

[170] John P Snyder. *Flattening the earth: two thousand years of map projections*. University of Chicago Press, 1997.

[171] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. Tvsum: Summarizing web videos using titles. In *CVPR*, 2015.

[172] E. Spriggs, F. De la Torre, and M. Hebert. Temporal segmentation and activity classification from first-person sensing. In *Workshop on Egocentric Vision, CVPR*, 2009.

[173] T. Starner, B. Schiele, and A. Pentland. Visual contextual awareness in wearable computing. In *Intl Symp on Wearable Comp*, 1998.

[174] Yu-Chuan Su and Kristen Grauman. Making 360 video watchable in 2d: Learning videography for click free viewing. In *CVPR*, 2017.

[175] Yu-Chuan Su, Dinesh Jayaraman, and Kristen Grauman. Pano2vid: Automatic cinematography for watching 360 videos. In *ACCV*, 2016.

[176] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

[177] Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Good image priors for non-blind deconvolution. In *ECCV*, 2014.

[178] Libin Sun and James Hays. Super-resolution from internet-scale scene matching. In *IEEE International Conference on Computational Photography*, 2012.

[179] Min Sun, Ali Farhadi, and Steve Seitz. Ranking domain-specific highlights by analyzing edited videos. In *ECCV*, 2014.

[180] P. Sundberg, T. Brox, M. Maire, P. Arbelaez, and J. Malik. Occlusion boundary detection and figure/ground assignment from optical flow. In *CVPR*, 2011.

[181] Hao Tang, Vivek Kwatra, Mehmet Emre Sargin, and Ullas Gargi. Detecting highlights in sports videos: Cricket as a test case. In *ICME*, 2011.

[182] Kevin Tang, Armand Joulin, Li-Jia Li, and Li Fei-Fei. Co-localization in real-world images. In *CVPR*, 2014.

[183] Kevin Tang, Rahul Sukthankar, Jay Yagnik, and Li Fei-Fei. Discriminative segment annotation in weakly labeled video. In *CVPR*, 2013.

[184] Mahdi Abbaspour Tehrani, Aditi Majumder, and M Gopi. Correcting perceived perspective distortions using object specific planar transformations. In *ICCP*, 2016.

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

[186] A. Torralba and A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

[187] A. Torralba, R. Fergus, and W. T. Freeman. 80 million Tiny Images: a Large Dataset for Non-Parametric Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

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

[189] Y.-H. Tsai, M.-H. Yang, and M. J. Black. Video segmentation via object flow. In *CVPR*, 2016.

[190] Y.-H. Tsai, G. Zhong, and M.-H. Yang. Semantic co-segmentation in videos. In *ECCV*, 2016.

[191] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *IJCV*, 2013.

[192] S. Vicente, C. Rother, and V. Kolmogorov. Object cosegmentation. In *CVPR*, 2011.

[193] S. Vijayanarasimhan and K. Grauman. Active frame selection for label propagation in videos. In *ECCV*, 2012.

[194] Jinjun Wang, Changsheng Xu, Chng Eng Siong, and Qi Tian. Sports highlight detection from keyword sequences using hmm. In *ICME*, 2004.

[195] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2015.

[196] Zeyu Wang, Xiaohan Jin, Fei Xue, Xin He, Renju Li, and Hongbin Zha. Panorama to cube: a content-aware representation method. In *SIGGRAPH Asia 2015 Technical Briefs*. ACM, 2015.

[197] T. Weyand and B. Leibe. Discovering favorite views of popular places with iconoid shift. In *ICCV*, 2011.

[198] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

[199] Zhengyang Wu, Fuxin Li, Rahul Sukthankar, and James M. Rehg. Robust video segment proposals with painless occlusion handling. In *CVPR*, 2015.

[200] Fanyi Xiao and Yong Jae Lee. Track and segment: An iterative unsupervised approach for video object proposals. In *CVPR*, 2016.

[201] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. SUN database: large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[202] Jianxiong Xiao. Princeton vision toolkit, 2013. Available from: `http://vision.princeton.edu/code.html`.

[203] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *CVPR*, 2012.

[204] Bo Xiong and Kristen Grauman. Detecting snap points in egocentric video with a web photo prior. In *ECCV*, 2014.

[205] Bo Xiong and Kristen Grauman. Intentional photos from an unintentional photographer: Detecting snap points in egocentric video with a web photo prior. In *Mobile Cloud Visual Media Computing*. 2015.

[206] Bo Xiong and Kristen Grauman. Snap angle prediction for 360 panoramas. In *ECCV*, 2018.

[207] Bo Xiong, Suyog Dutt Jain, and Kristen Grauman. Pixel objectness: Learning to segment generic objects automatically in images and videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[208] Bo Xiong, Yannis Kalantidis, Deepti Ghadiyaram, and Kristen Grauman. Less is more: Learning highlight detection from video duration. In *CVPR*, 2019.

[209] Bo Xiong, Gunhee Kim, and Leonid Sigal. Storyline representation of egocentric videos with an applications to story-based search. In *ICCV*, 2015.

[210] Ziyou Xiong, Regunathan Radhakrishnan, Ajay Divakaran, and Thomas S Huang. Highlights extraction from sports video based on an audio-visual marker detection framework. In *ICME*, 2005.

[211] Chenliang Xu, Caiming Xiong, and Jason J Corso. Streaming Hierarchical Video Segmentation. In *ECCV*, 2012.

[212] Jia Xu, Lopamudra Mukherjee, Yin Li, Jamieson Warner, James M Rehg, and Vikas Singh. Gaze-enabled egocentric video summarization via constrained submodular maximization. In *CVPR*, 2015.

[213] Huan Yang, Baoyuan Wang, Stephen Lin, David Wipf, Minyi Guo, and Baining Guo. Unsupervised extraction of video highlights via robust recurrent auto-encoders. In *ICCV*, 2015.

[214] Zhenheng Yang, Jiyang Gao, and Ram Nevatia. Spatio-temporal action detection with cascade proposal and location anticipation. In *BMVC*, 2017.

[215] Ting Yao, Tao Mei, and Yong Rui. Highlight detection with pairwise deep ranking for first-person video summarization. In *CVPR*, 2016.

[216] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *CVPR*, 2016.

[217] Felix X Yu, Dong Liu, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. $\propto$ svm for learning with label proportions. In *ICML*, 2013.

[218] Lihi Zelnik-Manor, Gabriele Peters, and Pietro Perona. Squaring the circle in panoramas. In *ICCV*, 2005.

[219] Jianming Zhang and Stan Sclaroff. Saliency detection: a boolean map approach. In *ICCV*, 2013.

[220] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *ECCV*, 2016.

[221] Ke Zhang, Kristen Grauman, and Fei Sha. Retrospective encoders for video summarization. In *ECCV*, 2018.

[222] Bin Zhao and Eric P Xing. Quasi real-time summarization for consumer videos. In *CVPR*, 2014.

[223] Rui Zhao, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Saliency detection by multi-context learning. In *CVPR*, 2015.

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

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

# Vita

Bo Xiong was born in Nantong, China on December 2nd 1990, the son of Gongjian Xiong and Dr. Jiangping Wang. He received the Bachelor of Arts degree in computer science and mathematics from Connecticut College in 2013. He then began graduate studies at The University of Texas at Austin, where he has been studying computer vision under the supervision of Prof. Kristen Grauman since August 2013. His research interests are in computer vision and machine learning.

Permanent address: 66 Muxuyuan Avenue
Nanjing, Jiangsu, China 210007

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.