The Dissertation Committee for Chao-Yeh Chen
certifies that this is the approved version of the following dissertation:

Learning Human Activities and Poses with Interconnected Data Sources

Committee:

Kristen Grauman, Supervisor

J. K. Aggarwal

Raymond J. Mooney

Deva Ramanan

Peter Stone

Learning Human Activities and Poses with Interconnected Data Sources

by

Chao-Yeh Chen, B.S.E.E.; M.S.E.

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

May 2016

Dedicated to my family.

Acknowledgments

I would like to thank my advisor Kristen Grauman. She showed me how to be a good researcher and advisor. She always had passion to research, dedicated her time to students, encouraged me to explore various directions, gave me insight advice in our meetings, and pushed me to work hard by working harder than us. Furthermore, from her, I learned how to be patient when the result isn't good, improve my presentation skill, inspire students in the class, and be a professional researcher. She has been the best supervisor and mentor I could ever dream of.

I am also grateful to my thesis committee members, Professors Jake Aggarwal, Raymond Mooney, Deva Ramanan, and Peter Stone for their insightful comments and feedback that strengthened this thesis.

My labmates deserve much thanks for making my PhD student life an enjoyable and unforgettable one. I have learned a lot from my current and former labmates (Sudheendra, YongJae, Jaechul, SungJu, Adriana, Sunil, Lu Zheng, Suyog, Dinesh, Aron, Bo, Yu-Chuan, and Danna). I will always remember the time we spent together in the lab, especially the time *before* and *after* paper deadlines, and outside the lab, especially the time we spent debating about a random topic for hours. Thank you for giving me advice in research and presentation, complaining about bad reviews together, and any help when I needed. I hope we will have lots of opportunities to see each other again in future.

My friends in Austin also deserve great thanks for my PhD. Thanks to Cho-Jui, Si-Si, Kai-Yang, Yi-Hung, Wei-Lun, Wei-Ju, Hsiang-Fu, and Wei-Gi in CS/ECE department for our lunch meetings, helping me with the homework and exams, helping me prepare the job interviews, and playing online games together.

I am also thankful to Professor Sheng-Jyh Wang in National Chia-Tung University, Taiwan. I learned a lot from his classes. He gave me an opportunity to experience research as an undergraduate, and that inspired me to pursue a PhD.

Finally, this thesis would not have been possible without the unconditional love and support of my family: my father Jung-Hsien Chen, my mother Rui-Huang Hong, and my brother Hsin-Yeh Chen. I dedicate this thesis to them.

Last but not least, special thanks to Chia-I Lin, who has been by my side throughout my entire graduate study. This thesis would not have been possible without her help. Thank you for your love, support, and encouragement.

Learning Human Activities and Poses with Interconnected Data Sources

Publication No. _____

Chao-Yeh Chen, Ph.D.
The University of Texas at Austin, 2016

Supervisor: Kristen Grauman

Understanding human actions and poses in images or videos is a challenging problem in computer vision. There are different topics related to this problem such as action recognition, pose estimation, human-object interaction, and activity detection. Knowledge of actions and poses could benefit many applications, including video search, surveillance, auto-tagging, event detection, and human-computer interfaces.

To understand humans' actions and poses, we need to address several challenges. First, humans are able to perform an enormous amount of poses. For example, simply to move forward, we can do crawling, walking, running, and sprinting. These poses all look different and require examples to cover these variations. Second, the appearance of a person's pose changes when looking from different viewing angles. The learned action model needs to cover the variations from different views. Third, many actions involve interactions between people and other objects,

so we need to consider the appearance change corresponding to that object as well. Fourth, collecting such data for learning is difficult and expensive. Last, even if we can learn a good model for an action, to localize when and where the action happens in a long video remains a difficult problem due to the large search space.

My key idea to alleviate these obstacles in learning humans' actions and poses is to discover the underlying patterns that connect the information from different data sources. Why will there be underlying patterns? The intuition is that all people share the same articulated physical structure. Though we can change our pose, there are common regulations that limit how our pose can be and how it can move over time. Therefore, all types of human data will follow these rules and they can serve as prior knowledge or regularization in our learning framework. If we can exploit these tendencies, we are able to extract additional information from data and use them to improve learning of humans' actions and poses. In particular, we are able to find patterns for how our pose could vary over time, how our appearance looks in a specific view, how our pose is when we are interacting with objects with certain properties, and how part of our body configuration is shared across different poses. If we could learn these patterns, they can be used to interconnect and extrapolate the knowledge between different data sources.

To this end, I propose several new ways to connect human activity data. First, I show how to connect snapshot images and videos by exploring the patterns of how our pose could change over time. Building on this idea, I explore how to connect humans' poses across multiple views by discovering the correlations between different poses and the latent factors that affect the viewpoint variations. In

addition, I consider if there are also patterns connecting our poses and nearby objects when we are interacting with them. Furthermore, I explore how we can utilize the predicted interaction as a cue to better address existing recognition problems including image re-targeting and image description generation. Finally, after learning models effectively incorporating these patterns, I propose a robust approach to efficiently localize when and where a complex action happens in a video sequence. The variants of my proposed approaches offer a good trade-off between computational cost and detection accuracy.

My thesis exploits various types of underlying patterns in human data. The discovered structure is used to enhance the understanding of humans' actions and poses. By my proposed methods, we are able to 1) learn an action with very few snapshots by connecting them to a pool of label-free videos, 2) infer the pose for some views even without any examples by connecting the latent factors between different views, 3) predict the location of an object that a person is interacting with independent of the type and appearance of that object, then use the inferred interaction as a cue to improve recognition, and 4) localize an action in a complex long video. These approaches improve existing frameworks for understanding humans' actions and poses without extra data collection cost and broaden the problems that we can tackle.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In our daily life, we observe and participate in different kinds of poses, actions and interactions such as walking, sitting, or reading papers. Learning and understanding humans' actions and poses has received increasing attention in recent years in computer vision [3, 97, 164, 178, 52, 102, 176, 132]. It involves recognizing human actions, multi-view pose and action recognition, human-object interaction modeling, human image description, and human activity detection. For each of the topics, today's methods typically learn the action/pose/interaction model from labeled image or video data and use the model to infer the related knowledge in novel images or videos.

While we are seeing good progress in analyzing people's actions and poses, particularly with learning based methods, there are still three main obstacles to be solved:

1. **The variation in appearance.** There can be large variations in appearance for poses that belong to the same action category, not to mention instances from different action categories. For example, different people can have different ways of walking, and the pose of walking is different from the pose of sitting. To learn a robust model for each action or pose, we would need a

large amount of data [92, 47]. Another factor that changes the appearance of pose is viewpoint variation [52, 53, 102]. It is clear that a pose may look quite different between two different views since we are living in a 3D world. An intuitive solution is to learn a model for each view independently. However, this would require examples for each of the views and significantly increase the cost for data collection.

2. **Increased complexity due to interaction.** Many actions are defined as an interaction between a person and an object [36, 132, 177, 33]. To learn the interaction, we need to consider the appearance of the person's pose and the appearance of the object together. We can utilize one as the other's context such as their spatial relationship to further improve the system. However, the complexity of the model also increases by incorporating this additional information, and we will need examples to learn the interactions between different poses and objects.

3. **Difficulty in data collection.** It is hard to get sufficient data to learn and analyze human actions and poses. Collecting action related data is more expensive than collecting data for object recognition tasks. There are two major reasons: (a) video data typically requires more labelling time than data in image format, and (b) the definition of some actions can be vague. For example, it's hard to define the boundary between the transition between two actions and there can be overlap between different action categories. These two aspects make it harder to have good quality data for action recognition tasks

than typical object recognition tasks. Therefore, many existing datasets only provide examples for a limited number of actions [47, 179, 97] or are biased toward one popular (frontal) viewing angle [110, 96, 97].

Intuitively, the first two obstacles can be eased by having enough examples to cover the variation in appearance and learn complex models. However, the third obstacle makes this solution difficult or expensive. In order to break through these obstacles together and provide a realistic solution, my key idea is to exploit some specific properties of human data which provide underlying structures that allow us to connect data in different aspects and extract additional information from them. These structures are all based on the fact that humans share an articulated physical architecture. Though we can control our torso, limbs, and head, they have to follow certain rules limited by our body's physical structure. Besides, there are also regulations and patterns for how we can move our body. For example, to jump, we need to first bend our knees and then make them straight. All the jumping related actions will follow this rule. Furthermore, our pose can only change continuously in time.

These structures exist ubiquitously in all human related data. We can use them to improve the way we learn and understand human action and pose in different fronts. In particular, I explore *the latent structure that connects the examples between snapshots and video clips, examples from different views, and examples from various interactions.* The key to achieve these connections and extrapolation between different data is the discovered pattern of how our pose can vary over time, how our pose is affected by viewpoint changes, and how we interact with objects with similar size in similar location.

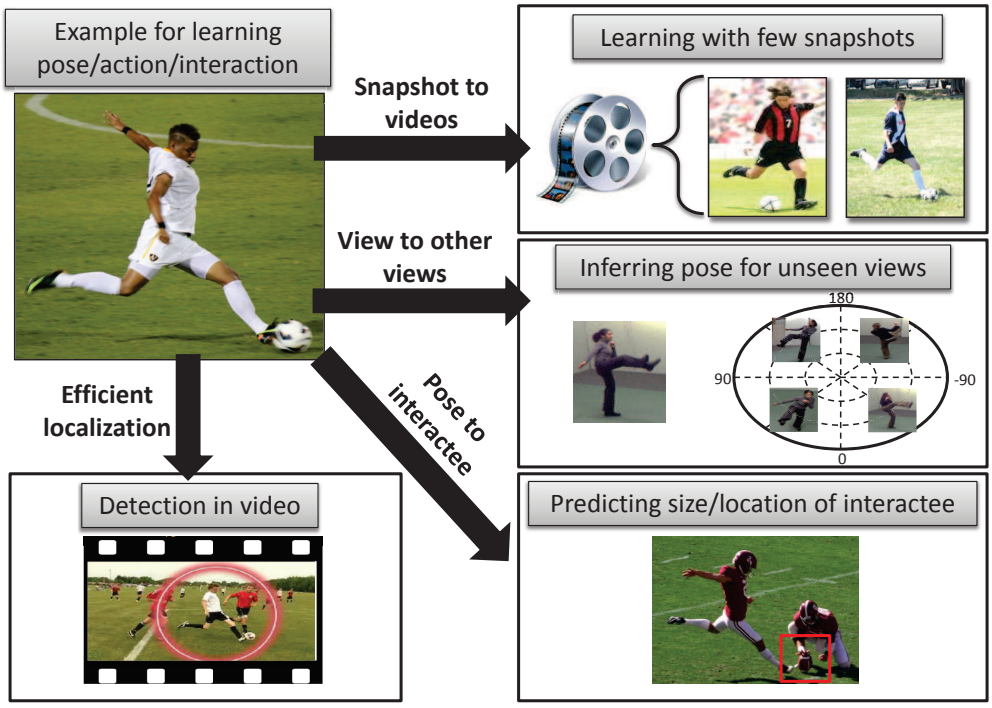


Figure 1.1: Overview of my dissertation. I explore different ways to use articulated human pose patterns to connect limited examples to other rich resources to improve the learning and understanding of humans' actions and poses. Given an example for learning humans' pose/action/interaction, as depicted in top right, we can connect the a few snapshots to the unlabeled video pool to learn the action (Chapter 3), and as depicted in middle right, we can enlarge the available pose data by inferring it in different views (Chapter 4). Furthermore, as depicted in bottom right, we can use the person's pose to predict where the object he is interacting with by connecting the pose to interactee (Chapter 5), and as depicted in bottom left, we can efficiently localize when and where this action happened in the video (Chapter 6).

In my dissertation, I explore how to utilize the underlying patterns that enable us to analyze human action and pose efficiently by connecting data in different aspects. First, I connect the labeled snapshots to the examples in unlabeled video pools. These linked examples from the video pool can help us learn new action

models with very few examples [22]. Second, I explore the latent factors that connect the properties of pose across different views. Then we can use the discovered latent factors to infer how a pose would look in an unseen view [23]. Third, I exploit the pattern between body pose and certain properties of the object we are interacting with. By the learned model, we are able to predict the location and the size of the object independent of its appearance and category [21]. Besides, I also explore four different applications by utilizing the inferred interaction localization as a cue for where to focus in the image. Finally, I propose a new framework to localize when and where the action happens in a long video sequence by transforming the localization problem into a maximum weighted subgraph searching problem. With this framework, we are able to localize the action much faster and in a more flexible scope [20]. See Figure 1.1 for the outline of these four proposed approaches.

In the following sections of this chapter, I will overview each of four major components (learning human actions from few labelled snapshots, inferring human pose in unseen views, predicting location of interactees and its applications, detecting activity with max-subgraph search) of my dissertation. Chapter 2 discusses related work. Chapter 3 through 6 will then give technical details on these ideas and present my results. Chapter 7 will describe future work inspired by this thesis.

1.1 Learning Human Actions from Few Labelled Snapshots

Existing methods require large amount of examples to handle the variations between different instances. In particular, recent recognition approaches [91, 79, 99] get significant progress by using deep convolutional neural networks to learn

an object/action model. To learn the network parameters, a large amount of labeled training data is required. Furthermore, people are able to understand a human action with just a few static snapshots. Presumably, the reason humans are able to do this is that we have strong prior knowledge of how human poses vary over time. The limited information from these snapshots is connected to our previous experience of human pose changes and that knowledge is used to expand our understanding of the action without seeing more examples.

Building on this intuition, I propose an approach to learn action categories from a small number of static images by leveraging *prior observations of generic human motion* to augment the training process. Given unlabelled video, the system first learns how body pose changes over time. We assume this video has some human activity in it, and that humans are often detectable when present, but otherwise make no assumptions about *which* actions are present in the data. Then, given a small set of labelled images for an action category, the system uses the generic knowledge obtained from watching the video to extrapolate beyond those exemplars during training. In particular, it augments its labelled set with "synthetic" examples, which depict poses that could immediately precede or follow the given examples in time. In this way, we expand the training set without requiring additional manually labelled examples.

In my dissertation, I propose two ways to implement this idea. The first uses an example-based representation of pose dynamics; we match the labelled training images to unlabelled video frames based on their pose similarity, and then augment the training set with the poses appearing before and after the matched frames. The

second technique uses a manifold-based representation; we learn a nonlinear manifold over body poses, relying on the temporal nearness of the video frames to establish which should maintain proximity. Then, we map the static training instances to the manifold, and explore their neighborhoods on the manifold to augment the training set. In both cases, we adopt a part-based representation of pose, and use domain adaptation to account for the mismatch between the source images and the unlabelled video. We show that our synthetic expansions to the training set yield more accurate predictions, especially when labeled data is quite sparse. Notably, the gains come at no additional labelling cost, since we make no assumptions about which actions appear in the unlabelled video.

I demonstrate the proposed approach to recognize actions in both static images and videos from multiple challenging datasets. The results show that by letting the system first "watch" generic video, it can successfully infer additional plausible poses that bolster training. For our target scenario where training examples are very few, my approach outperforms both a method limited to the original static exemplars, as well as alternative methods to pad the data by introducing appearance variation.

1.2 Inferring Human Pose in Unseen Views

While my idea for connecting snapshots to video (above) exploits patterns to extrapolate to nearby poses over time, the next major part of the thesis studies how to connect multiple views and image sources to extrapolate to nearby viewpoints in space.

After alleviating the requirement for a large number of examples to learn an action, another obstacle comes from the viewpoint variation. The appearance of people's pose varies between views. If we want to fully understand an action, we would need to collect examples from all different views. However, most available examples are collected from certain *canonical* views, such as facing toward the camera, making it hard to find examples to cover the remaining views. On the other hand, the data that does have pose examples observed from different views tend to come from artificial lab environments.

To overcome this data dilemma, I propose an approach to discover the latent factors that correlate poses across different views. The proposed method takes as input images of person organized by their approximate viewpoint. We construct a 3D tensor indexed by the image examples, their viewpoints, and the spatial image positions. Each entry in the tensor records the appearance observed at those coordinates. Notably, many entries are unobserved in the input data. I show that a probabilistic tensor factorization technique can discover the latent factors governing how all three observed dimensions jointly determine appearance. Intuitively, those factors might correspond to things like the type of clothing, body weight, lighting, or partial pose fragments. Using them, we impute missing entries in the tensor, thereby inferring the image descriptors for unobserved views of people that, during learning, may have been observed from just one camera viewpoint.

In the experiment result, I show that the inferred views are both visually and quantitatively accurate, which lets us expand existing datasets to fuller viewpoint coverage. I demonstrate the impact for two practical applications. First, I show that

the inferred virtual views let the system learn an action category in a viewpoint for which it has never seen any real exemplars, yielding results that are competitive with recent cross-view recognition methods. Second, I show that by using the virtual views to augment real training images, we can predict a person's orientation more accurately in novel images. In both cases, the inferred views help make statistical appearance-based methods robust to viewpoint. While existing methods are often forced to choose between data that is either realistic or multi-view, our virtual views offer both, thereby allowing greater robustness to viewpoint in novel images.

1.3 Predicting Locations of Interactees

As the two approaches devised above extrapolate and interconnect human actions across viewpoints and over time, next I explore another aspect of how human actions relate to each other: *interaction*. A large portion of human actions involve *interactions* between a person and an object, or scene, or another person(s). For example, a person *reading* reads a book or paper; a person *discussing* chats with other people nearby; a person *eating* uses utensils to eat food from a plate. In any such case, the person and the "interactee" object (i.e., book, other person, food and utensils, etc.) are closely intertwined; together they define the story portrayed in the image or video.

Existing research in human action recognition aims to exploit this close connection [127, 65, 36, 177, 176, 76, 132, 32]. Their goal is to improve recognition by leveraging human action in concert with the object being manipulated by the human. However, these prior methods assume that during training it is possible to

learn patterns between a particular action and the particular object category it involves, and thus it is assumed that the data can cover all possible actions and object categories.

In my thesis, I seek to relax these assumptions in order to make predictions about novel, unseen human-object interactions. In particular, I consider the following question: *Given a person in a novel image, can we predict the location of that person's "interactee"—the object or person with which he interacts—even without knowing the particular action being performed or the category of the interactee itself?* Why should the goal be possible? We can do so because we have a model of certain pose, gaze, and scene layout patterns that exist when people interact with a person/object in a similar relative position and size. This is done without knowing the category of the object, and even without (necessarily) being able to name the particular action being performed.

Based on this intuition, my proposed idea is to learn from data how the properties of a person relate to the interactee localization parameters. Given instances labeled with both the person and interactee outlines—from a variety of activities and objects—we train a probabilistic model that can map observed features of the person to a distribution over the interactee's position and scale. Then, at test time, given a novel image and a detected person, we predict the most likely places the interactee will be found.

The proposed approach addresses a number of challenges. They include designing a reliable data collection procedure to handle this somewhat unusual annotation task; developing a bank of descriptors to capture the "meta-cues" about

human appearance that signal localized interactions; and presenting applications to exploit the interactee predictions.

Whereas the methods devised above explore how to extrapolate and interconnect human actions across viewpoints and over time, this component of my thesis explores how to interconnect human poses across various interacting objects in a category-independent manner.

For applications, I explore different ways to utilize the interactee localization as a cue to guide the system for focusing on important object/area(s) in the scene. By focusing attention on regions in the image that are prominently involved in a human interaction, my method can be used to improve object detection speed/accuracy, image retargeting, and image description. In all such cases, I show how to utilize interactee localization as person-centric view of importance.

1.4 Detecting Activity with Max-Subgraph Search

The three proposed methods above improve the understanding of human action, pose, and interaction by connecting the knowledge between different aspects. After we learn these models, we can apply them to new data to detect when and where these activities happened. In the final main component of my thesis, I focus on the detection strategy itself.

The activity detection problem entails both *recognizing* and *localizing* categories of activity in an ongoing (meaning "untrimmed") video sequence. In other words, a system must not only be able to recognize a learned activity in a new

clip; it must also be able to isolate the (potentially small) portion of a long input sequence that contains the activity. Reliable activity detection would have major practical value for applications such as video indexing, surveillance and security, and video-based human computer interaction.

While the recognition portion of the problem has received increasing attention in recent years, state-of-the-art methods largely assume that the space-time region of interest to be classified has already been identified. However, for most realistic settings, a system must not only name what it sees, but also partition out the temporal or spatio-temporal extent within which the activity occurs. The distinction is non-trivial; in order to properly recognize an action, the spatio-temporal extent usually must be known *simultaneously*.

To meet this challenge, existing methods tend to separate activity detection into two distinct stages: the first generates space-time candidate regions of interest from the test video, and the second scores each candidate according to how well it matches a given activity model (often a classifier). Most commonly, candidates are generated either using person-centered tracks [116, 134, 175, 87] or using exhaustive sliding window search through all frames in the video [84, 43, 143]. Both face potential pitfalls. On the one hand, a method reliant on tracks is sensitive to tracking failures, and by focusing on individual humans in the video, it overlooks surrounding objects that may be discriminative for an activity (e.g., the car a person is approaching). On the other hand, sliding window search is clearly a substantial computational burden, and its frame-level candidates may be too coarse, causing clutter features to mislead the subsequent classifier. In both cases, the scope of

space-time regions even considered by the classifier is artificially restricted, e.g., to person bounding boxes or a cubic subvolume.

I propose an efficient approach that unifies activity categorization with space-time localization. The main idea is to pose activity detection as a maximum-weight connected subgraph problem over a learned space-time graph constructed on the test sequence. I show this permits an efficient branch-and-cut solution for the best-scoring and possibly non-cubically shaped portion of the video for a given activity classifier. The upshot is a fast method that can evaluate a broader space of candidates than was previously practical, which I find often leads to more accurate detection.

The proposed approach has several important properties. First, for the specific case where our space-time nodes are individual video frames, the detection solution is equivalent to that of exhaustive sliding window search, yet costs orders of magnitude less search time due to the branch-and-cut solver. Second, we show how to create more general forms of the graph that permit "non-cubic" detection regions, and even allow hops across irrelevant frames in time that otherwise might mislead the classifier (e.g., due to a temporary occluding object). This effectively widens the scope of candidate video regions considered beyond that allowed by any prior methods; the upshot is improved accuracy. Third, we explore a two-stage search extension that increases the speed of the proposed subgraph search for long videos, and show its generality for detecting multiple activity instances in a single input sequence. Finally, the method accommodates a fairly wide family of features and classifiers, making it flexible as a general activity detection tool. To illustrate

this flexibility, we devise a novel high-level descriptor amenable to subgraph search that reflects human poses and objects as well as their relative temporal ordering.

Having summarized the main technical threads of my thesis, I will next overview related work.

Chapter 2

Related Work

In this chapter, I will review related work to the research presented in my dissertation. First I will review existing work related to general human action recognition, including learning from static images and videos (Sec. 2.1). For the second topic, I am going to review the work relates to a specific difficulty in action and pose recognition: multi-view problems (Sec. 2.2). Next, in addition to learning the action and pose model only from human examples, I will go over the work that models the interaction between humans and objects or treats each other as context information (Sec. 2.3). As interactions play an important rule in how we describe the content in an image, I will review existing methods for image description. (Sec. 2.4). Next, I will review the existing techniques for using synthetic data and matrix completion (Sec. 2.5)—two of the important building blocks of portions of my proposed methods. As some of my proposed methods can be seen as a way to reduce data labeling costs, I will also discuss the difference between my approaches and existing transfer learning methods (Sec. 2.6). Finally, I provide an overview of existing methods for activity detection in video clips (Sec. 2.7).

After each subsection, I briefly highlight the most important differences between the prior work and my own.

2.1 Human Action Recognition

Activity recognition and human motion analysis have a rich literature [3]. To learn activities from video, earlier work emphasized tracking and explicit body-part models (e.g., [116, 135, 134]). In parallel, many methods to estimate body pose have been developed, including techniques using nonlinear manifolds to represent the complex space of joint configurations [62, 165, 14, 100, 152, 153]. Such methods assume silhouette (background-subtracted) inputs and/or derive models from mocap data, and are often intended for motion synthesis applications. More recently, researchers have considered how activity classes can be learned directly from lower-level spatio-temporal appearance and motion features—for example, based on bag-of-words models for video (e.g., [97, 164]). By sidestepping tracking and pose, this general strategy offers robustness and can leverage strong learning algorithms; on the other hand, the lack of top-down cues suggests more data is critical to learn the needed invariance.

In addition to learning human activities from video, recent work considers action recognition in *static* images, where image data is much easier to collect and can benefit from existing image-based object recognition techniques. During both training and testing, these algorithms use only static snapshots of the actions of interest. Most current methods rely on a combination of pose- and appearance-based descriptors [173, 110, 33, 179, 178]. In particular, "poselets" [16]—local part-based features mined for their consistency with fragments of body pose—have proven to be a promising representation [173, 110, 33], as well as high-level descriptors that also incorporate interactions with objects [36, 177, 33, 179].

Discussion: The work described above largely focuses on providing better features to model human action. They assume there are enough data to learn the action of interest. However, this constraint limits the number of actions we could learn, and some methods require labeled video data, which is much more expensive to collect. In contrast, I propose a novel approach in Chapter 3 that is able to learn a action model with only a few labeled snapshots by interconnecting the pose in snapshots to a pool of unlabeled videos.

2.2 Multi-View Human Pose Analysis

The work discussed in Sec. 2.1 aims at providing pose related features that are discriminative across different human actions and consistent within each action category, yet people who perform similar pose/action could have quite different appearance when observed from different viewpoints. Intuitively, if we can have enough examples to cover all possible views for all poses, those methods in Sec 2.1 could learn the action model well. However, this would increase the requirement of data and limit the number of actions that we are able to learn.

To handle this problem, some existing works adapt viewpoint-invariant models to avoid the need of examples from all views. The view-invariant methods develop features that remain stable across camera views (e.g., [125, 136, 178, 107]), but they require reliable body joint detection. When multi-view data is available, 3D reconstruction can be used to form 3D exemplars [166] or view-invariant features [171], though their view assumptions and computational demands may be too high for many applications. Multiple action recognition methods *transfer* features

between viewpoints, learning the "domain shift" between pairs of views [52, 70, 102, 106, 185]. These methods construct the features to make them invariant to viewpoint changes.

Discussion: The view-invariant methods require synchronized multi-view data during training, which restricts the type of data that can be used. Furthermore, none of these methods are able to hallucinate unseen views, such as visualization (e.g., helping an artist sketch an actor from a new viewpoint). In Chapter 4, I propose a new learning based approach that implicitly captures geometry through its knowledge about discrete viewpoints. The proposed approach is able to leverage any available views and infer the pose in unseen views.

2.3 Human-Object Interaction

Many human activities are related to the interaction between humans and other objects/humans. When we are interacting with other objects, to capture the characteristic of this action, the model has to incorporate the features from the person *and* the interactee. Besides, the information provided from the interactee can serve as a cue to recognize this interaction or for other applications.

A great deal of recent work aims to jointly model the human and the objects with which he or she interacts [127, 65, 36, 177, 176, 76, 132, 32]. The idea is to use the person's appearance (body pose, hand shape, etc.) and the surrounding objects as mutual context—knowing the action helps predict the object, while knowing the object helps predict the action or pose. For example, the Bayesian model in [65]

integrates object and action recognition to resolve cases where appearance alone is insufficient, e.g., to distinguish a spray bottle from a water bottle based on the way the human uses it. Similarly, structured models are developed to recognize manipulation actions [85] or sports activities [177, 36] in the context of objects. Novel representations to capture subtle interactions, like playing vs. holding a musical instrument, have also been developed [176]. Object recognition itself can benefit from a rich model of how human activity [127] or pose [32] relates to the object categories. While most such methods require object outlines and/or pose annotations, some work lightens the labeling effort via weakly supervised learning [76, 132].

To utilize the interactee as context information, the interactee (object or person) needs to be reasonably localized. For localizing objects, there is work focusing on carried object detection [67, 30]. They assume a static video camera, which permits good background subtraction and use of human silhouette shapes to find outliers. These approaches are specialized for a single action (carrying) only. As for the case where the interactee is another person, there are methods for analyzing social interactions that estimate who is interacting with whom [118, 112, 54], or categorize the type of physical interaction [174]. These social interaction works can leverage rules from sociology [118] or perform geometric intersection of mutual gaze lines [112, 54].

One can also use the knowledge of interactees to help us improve the action recognition task. For example, methods to predict object affordances consider an object [89, 35] or scene [66] as input, and predict which actions are possible as output. They are especially relevant for robot vision tasks, letting the system pre-

dict, for example, which surfaces are sittable or graspable. While these approaches utilize interactees to help them solve a problem, in my dissertation, I consider the inverse task: given a human pose as input, I want to predict the localization parameters of the object defining the interaction as output.

Discussion: The existing methods model the interaction between humans and objects based on the type of interactee and type of action, such as "picking up the phone" or "reading book". Picking and reading are types of actions, while phone and book are types of interactees. One limitation of these settings is that they need to learn different models for interactions of different pose and interactee types. In Chapter 5, I propose a new approach that is *action*- and *object-independent*. The cues our method learns cross activity boundaries, such that we can predict where a likely interactee will appear even if we have not seen the particular activity (or object) before. This is valuable because it could make the learned model generalize to various data and reduce the labeling cost.

2.4 Describing Images

In Chapter 5 of my dissertation, I use interaction as a cue to guide the system for image description tasks. Recent work explores ways to produce a sentence describing an image [50, 93, 181, 122, 40, 49, 83] or video clip [63]. Such methods often smooth the outputs of visual detectors, making them better agree with text statistics [93, 63, 129] or a semantic ontology [181]. One general approach is to produce a sentence by retrieving manually captioned images that appear to match

the content of the novel query [50, 122, 37]. Another is to employ language models to generate novel sentences [93, 49, 94].

Other methods explore various criteria for *selectively* composing textual descriptions for images. In [140], the system composes a description that best discriminates one image from others in a set, thereby focusing on the "unexpected". In [129], a language model is used to help infer a person's motivation, i.e., the purpose of their actions. In [121], a mapping is learned from specific object categories to natural sounding entry-level category names (e.g., dolphin vs. grampus griseus).

Most related to the part of my work of using interaction to guide the system to focus on a part of image are methods that model *importance* [150, 71, 151]. They attempt to isolate those objects within a scene that a human would be most likely to notice and mention. Using compositional cues like object size and position [150] as well as semantic cues about object categories, attributes, and scenes [151], one can learn a function that ranks objects by their importance, or their probability of being mentioned by a human.

Discussion: The key difference between existing methods and my proposed method in helping image description is that we consider the novel cue: the interactee's localization. While a few existing methods employ human activity detectors [122, 63, 129], they do not represent human-object interactions, as I propose.

My contribution in Chapter 5 is not a new way to infer sentences. Rather, it is a new way to infer importance, which can be valuable to description methods. The existing sentence generation methods are primarily concerned with generating

a factually correct sentence; the question of "what to mention" is treated only implicitly via text statistics. While we show the impact of our idea for retrieval-based sentence generation, it has potential to benefit other description algorithms too.

As compared to the methods focused on selecting the important objects for description, we propose a novel basis for doing so—the importance signals offered by a human-object interaction. In addition, unlike methods that exploit object category-specific cues [151, 71, 150], we learn a category-independent metric to localize a probable important object, relative to a detected person.

2.5 Incorporating Synthetic Data

As I will discuss in Chapters 3, 4 and 3, to understand a human action, pose, or interaction efficiently without collecting a large amount of data, we aim to increase the size of training set with realistic but synthetically generated data.

A standard way to expand training data in object recognition is by mirroring the images along the vertical axis (e.g., [123] and many others). This trick has even been employed to produce flipped versions of video sequences for activity recognition [164]. The availability of humanoid models in graphics software, together with mocap data, make it possible to generate synthetic images useful for training action recognition [114] and pose estimation methods [148, 62, 149]. Web images noisily labeled by tags can also serve as a "free" source of data for action classification [75].

Another aspect to expand data synthetically is to infer the data for different views. Existing view synthesis methods originate from image-based render-

ing [82], where, rather than explicitly construct a 3D model, new views are synthesized directly from multiple 2D views. Typically point correspondences are estimated between views, and then intermediate views are synthesized by warping the pixels appropriately, leveraging insights from projective or multi-view geometry (e.g., [146, 6]). The resulting virtual views can be used to augment training data for object recognition [25], or to reposition the viewpoint at test time [147, 144]. Image-based models of pedestrians using calibrated, synchronized cameras are explored in [147, 62]. These view synthesis methods rely on geometry and warping. They make strong assumptions about calibrated cameras and/or simultaneous multi-view capture and require information of point correspondences, which is difficult to estimate reliably.

To infer missing data from the structure of observed data, matrix completion methods have been studied extensively [113, 90, 141, 142, 170] often for applications in collaborative filtering. While the standard completion problem can be treated in 2D, there are also approaches developed to model the 3D structure, e.g. to represent trends over time [170]. However, there is limited work exploring matrix or tensor completion for visual data. Existing methods infer missing pixels in a *single* source image/video, e.g., for in-painting [105], or infer new 3D face meshes captured with a structured light scanner for video puppetry [159]. By exploring linear factors across classes, [128] learns low rank bilinear discriminative classifiers for matrix or tensor visual data. The factorized models have also been used for bilinear models for separating style and content of visual data [60].

Discussion: In Chapter 4, I propose a novel way to infer and incorporate synthetic data using the properties in human action and pose related data. I demonstrate we can improve existing frameworks by adding the synthetic data together with original data. Rather than infer new view using traditional geometry based methods, I show how we can *learn* the latent factors that connect the poses across different views and use them to infer the pose in unseen views.

2.6 Transfer Learning

Related to my works on augment training data in Chapters 3 and 4, transfer learning technique can reduce the need for a large amount of labeled training data. It has been explored for object recognition [55, 9, 172, 133, 162, 154, 103, 7], where the goal is to learn a new object category with few labeled instances by exploiting its similarity to previously learned class(es). While often the source and target classes must be manually specified [9, 162, 7], some techniques automatically determine which classes will benefit from transfer [154, 103, 78].

Discussion: Both transfer learning and my proposed method can reduce the labelling cost of data. However, compared to existing forms of transfer learning, where they focus on exploring the correspondence between different object classes, my proposed methods in Chapters 3 and 4 connect the information through different data sources. Existing transfer learning techniques propagate the knowledge from a source class to a target class. On the other hand, my proposed methods do not require knowledge of class labels, but instead explore the underlying pattern existing

in all types of human related data sources. For example, instead of "transferring" a learned model of an action from a particular viewpoint to another viewpoint, my proposed method in Chapter 4 generates synthetic pose examples for an unseen viewpoint with knowledge from available pose examples of all viewpoints.

2.7 Human Activity Detection

Whereas the work in Sec. 2.1 above largely focuses on the activity *recognition* problem, we are also interested in *detection*. Detecting human activity means localizing when and where a specific action is in a video sequence. One class of methods tackles detection by explicitly tracking people, their body parts, and nearby objects (e.g., [116, 134, 87]). Tracking "movers" is particularly relevant for surveillance data where one can assume a static camera. Conscious of the difficulty of relying on tracks, another class of methods has emerged that instead treats activity classes as learned space-time appearance and motion patterns. The bag of space-time interest point features model is a good example [97, 145]. In this case, at detection time the classifier is applied to features falling within candidate subvolumes within the sequence. Typically the search is done with a sliding window over the entire sequence [84, 43, 143], or in combination with person tracks [87].

Given the enormous expense of such an exhaustive search, some recent work explores branch-and-bound solutions to efficiently identify the subvolume that maximizes an additive classifier's output [184, 183, 17]. This approach offers fast detection and can localize activities in both space and time, whereas sliding windows localize only in the temporal dimension.

An alternative way to avoid exhaustive search is through voting algorithms. Recent work explores ways to combine person-centric tracks or pre-classified sequences with a Hough voting stage to refine the localization [175, 115], or to use voting to generate candidate frames for merging [168]. Like any voting method, such approaches risk being sensitive to noisy background descriptors that also cast votes, and in particular will have ambiguity for actions with periodicity. Furthermore, in contrast to our algorithm, they cannot guarantee to return the maximum scoring space-time region for a classifier.

Discussion: Relying on tracks to detect action can be limiting; it makes the detector sensitive to tracking errors, which are expected in video with large variations in backgrounds or rapidly changing viewpoints (e.g., movies or YouTube video). As for existing branch-and-bound methods, they are restricted to searching over *cubic* subvolumes in the video; that limits detections to cases where the subject of the activity does not change its spatial position much over time. In Chapter 6 of my dissertation, I propose an efficient approach that is able to provide flexible non-cubical detection subvolume and reduce the computational cost to search over long video sequences.

Chapter 3

Learning Human Actions from Few Labeled Snapshots

[1]People can understand a human action by looking at just a few static snapshots. If we are able to develop a system that is capable of learning an activity model with a few images, it would be quite easy to collect the required data to train the system. However, today's systems typically require hundreds, if not thousand of such exemplars to learn an action category well. Human viewers have an important advantage, however: prior knowledge of how human poses tend to vary in time. This undoubtedly helps "fill the gaps" between a sparse set of snapshots, and thereby improves generalization. See Figure 3.1.

Recent existing methods [91, 79, 99] get significant progress by using deep convolutional neural networks (CNN) to learn object and action model. Whether training a CNN or other model, to learn the model parameters in the training stage, a large amount of labeled data is required. However, as we address in Chapter 1, it is expensive to get sufficient data to learn and analyze human actions and poses. In addition, there is also the "long tails" problem in the data, where the number of

[1]The work in this chapter was supervised by Dr. Grauman and originally published in: Watching Unlabeled Video Helps Learn New Human Actions from Very Few Labeled Snapshots. C.-Y. Chen and K. Grauman. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, OR, June 2013.

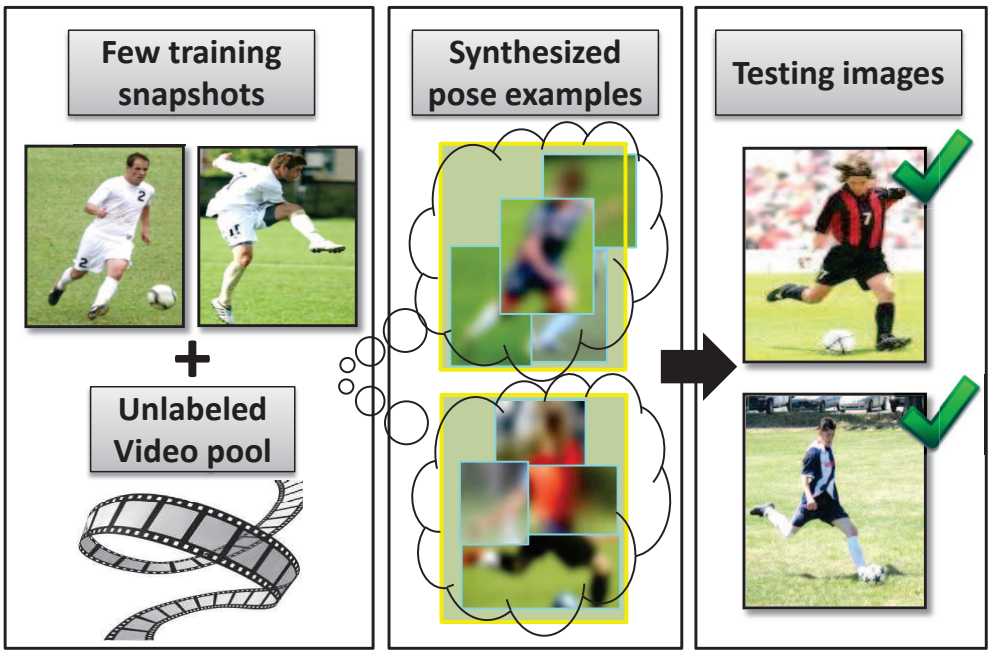


Figure 3.1: The proposed approach learns about human pose dynamics from unlabeled video, and then leverages that knowledge to train novel action categories from very few static snapshots. The snapshots and video (left) are used together to extrapolate "synthetic" poses relevant to that category (center), augmenting the training set. This leads to better generalization at test time (right), especially when test poses vary from the given snapshots.

training examples is highly imbalanced between different categories. For some actions or poses in a given viewpoint, we may not be able to gather enough examples for learning an accurate model.

In this chapter, I will present a novel method that learns human action with very few labelled snapshots incorporating a pool of unlabelled videos. As described in Chapter 1, the intuition is to let the system connect a few snapshots to the unlabelled video pool. Then, it expands the understanding of the action by utilizing the

temporal dependency of pose changes for poses in the videos.

3.1 Approach for Learning Human Actions from Few Labeled Snapshots

Our approach augments a small set of static snapshots by introducing realistic but synthetically generated body pose examples. The synthetic examples extend the real ones locally in time, so that we can train action classifiers on a wider set of poses that are (likely) relevant for the actions of interest. We first define the representation we use for pose. Then, after describing our video data requirements, I present two methods to infer synthetic pose examples; one is example-based, the other is manifold-based. Finally, I explain how we use a mix of real and synthetic data to train a classifier that can predict actions in novel static images.

3.1.1 Representing Body Pose

We use a part-based representation of pose called a *poselet activation vector* (PAV), adopted from [110]. A poselet [16] is an SVM classifier trained to fire on image patches that look like some consistent fragment of human body pose. For example, one poselet might capture arms crossed against the chest, or a left leg bent at the knee, or even the whole body of a seated person. The PAV records the "activation strength" of all poselets appearing within a person bounding box. Specifically, after running a bank of P poselet classifiers on an image, we take those poselet detections that overlap with a person bounding box, and record a vector $\boldsymbol{p} = [p_1, \ldots, p_P]$ where p_i is the sum of the i-th classifier's probability outputs.

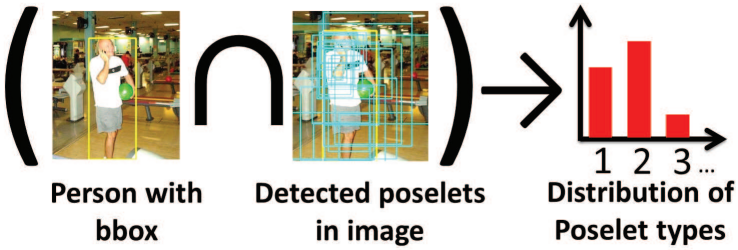


Figure 3.2: The PAV representation summarizes those detected poselets in the image that overlap with the person bounding box.

Figure 3.2 shows this process, and the blurry images in Figure 3.3 depict example poselets in terms of the averaged image patches used to train them. We use the $P = 1200$ poselets provided by [110].

We use this descriptor because it captures human body pose at a high level, and it is robust to occlusion and cluttered backgrounds. While it is quite simple—essentially a histogram of local pose estimates—it is also powerful. The poselets themselves offer a rich encoding of diverse poses, and they are detectable in spite of differences in appearance (e.g., clothing, race). Further, since they are specific to body configurations, the PAV implicitly captures spatial layout. Since 2D HOG descriptors underly the poselet classifiers, they are naturally sensitive to substantial 3D viewpoint changes. This is fine for our data-driven approach, which will synthesize poses that expand exemplars as viewed from a similar viewpoint.

3.1.2 Unlabeled Video Data

My method requires access to unlabeled videos containing human activity. The video has no action category labels associated with it, and the activity

is not segmented in any way. In particular, we do *not* assume that the activities present belong to the same categories as we will observe in the static training images. The category-independence of the video data is crucial. We would like the system to build a model of human motion dynamics—typical changes of body pose over time—without knowing in advance what novel actions it will be asked to learn from snapshots. Intuitively, this suggests that a large and diverse set of clips would be ideal, as we cannot hope to extrapolate poses for inputs that are unlike anything the system has seen before. In our current implementation, we use video from the Hollywood dataset [97] to form the unlabeled pool.

We assume that the humans appearing in the video can often be detected and tracked, i.e., using state-of-the-art human detectors and tracking algorithms, so that we can extract pose descriptors from human bounding boxes. We also expect that the video and snapshots come from roughly similar sensor types, meaning that we would not attempt to use dynamics learned from overhead aerial video (where people are blobs of tens of pixels) to help recognition with snapshots taken on the ground (where people have substantially greater resolution and body parts are visible). This is a very mild requirement, since plenty of ground video is available to us via YouTube, Hollywood movies, and so on. In fact, our method explicitly builds in some flexibility to data source mismatches due to its use of domain adaptation, as we will discuss later.

To pre-process the unlabeled video, we 1) detect people and extract person tracks, 2) compute a PAV pose descriptor for each person window found, and 3) either simply index those examples for our exemplar-based method or else compute

a pose manifold for our manifold-based method (both are defined in Sec. 3.1.3). Note that because this video is unlabeled, our method will enhance the training set with no additional manual effort.

3.1.3 Generating Synthetic Pose Examples

Our key idea is to expand limited training data by exploring unlabeled video, which implicitly provides rules governing how human pose changes over time for various activities. Thus, the heart of our method is to generate synthetic pose examples. We investigate two strategies: example-based and manifold-based.

Let $\mathcal{S} = \{(\boldsymbol{p}_1^i, y_1), \ldots, (\boldsymbol{p}_N^i, y_N)\}$ denote the N training snapshots our system receives as input, where the superscript i denotes *image*, and each $\boldsymbol{p}_j^i \in \mathbb{R}^P$ is a PAV descriptor with an associated action class label $y_j \in \{1, \ldots, C\}$ (e.g., running, answering phone, etc). Let $\{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_K\}$ denote the K person tracks from the unlabeled video, and let each track $\boldsymbol{t}_k$ be represented by a sequence of PAV descriptors, $\boldsymbol{t}_k = (\boldsymbol{p}_{k_1}^v, \ldots, \boldsymbol{p}_{k_M}^v)$, where superscript v denotes *video*, and k_M is the number of frames in the k-th track.

3.1.3.1 Example-based Strategy

Our example-based method treats the video as a non-parametric representation of pose dynamics. For each training snapshot pose $\boldsymbol{p}_j^i$, we find its nearest neighbor pose in any of the video tracks, according to Euclidean distance in PAV space. Denote that neighbor $\boldsymbol{p}_{j*}^v$. Then, we simply sample temporally adjacent poses to $\boldsymbol{p}_{j*}^v$ to form synthetic examples that will "pad" the training set for class y_j.

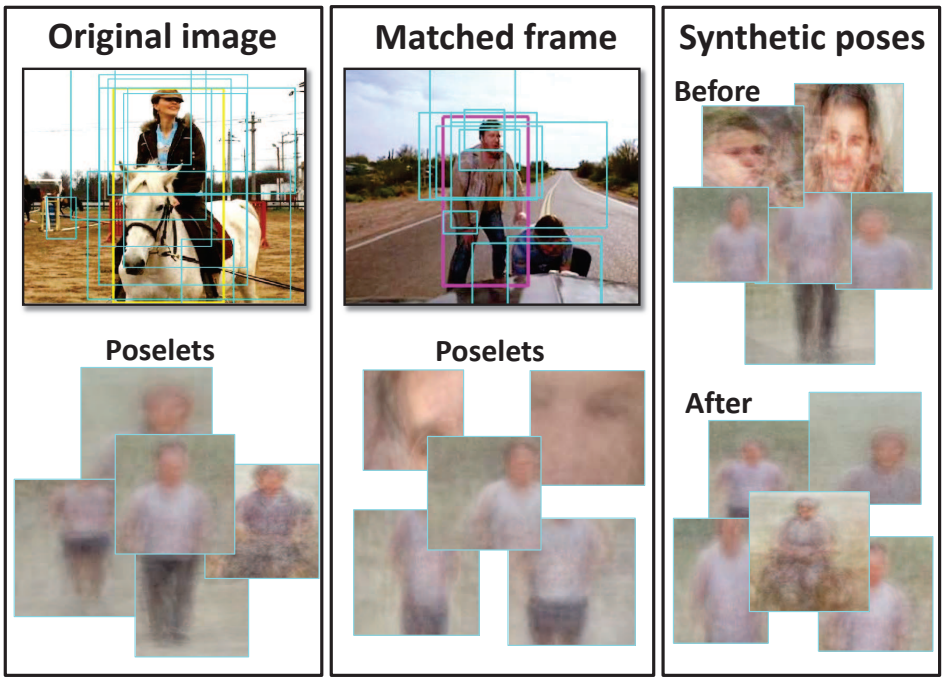


Figure 3.3: For each labeled training snapshot (top left), we use its pose description (depicted in bottom left) to find a neighbor in the unlabeled video (center panel). Then we synthesize additional training poses based on its temporal neighbors or nearby instances on a pose manifold (right panel). Best in color.

Specifically, we take $\boldsymbol{p}^v_{j^*-T}$ and $\boldsymbol{p}^v_{j^*+T}$, the poses T frames before and T frames after the match (accounting for boundary cases if the neighbor starts or ends a track). See Figure 3.3.

We repeat this process for all training snapshots, yielding an expanded training set $\mathcal{S}^+$ with two new synthetic examples for each original snapshot: $\mathcal{S}^+ = \{\mathcal{S} \cup \{(\boldsymbol{p}^v_{j^*-T}, y_j), (\boldsymbol{p}^v_{j^*+T}, y_j)\}_{j=1}^N\}$. In our experiments, we set $T = 10$ in order to get frames showing poses that would occur just before or after the matched pose, without being too visually redundant. In preliminary tests, we found the method

is not very sensitive to this parameter within the range $T = 5, \ldots, 20$, and simply fixed it at 10.

3.1.3.2 Manifold-based Strategy

We also explore a method to extrapolate poses using a nonlinear pose manifold. Whereas the example-based method extrapolates pose solely in the temporal dimension—and solely using one sequence at a time—the manifold variant unifies connections in both appearance and dynamics, and it effectively samples synthetic examples from a mix of sequences at once.

To construct the manifold, we use the locally linear embedding (LLE) algorithm [139]. LLE constructs a neighborhood-preserving embedding function that maps high-dimensional inputs in $\mathbb{R}^P$ to a low-dimensional nonlinear manifold in $\mathbb{R}^d$. The manifold is represented as a set of globally consistent linear subspaces, and the solution to minimize its reconstruction error relies on an eigenvalue problem. The algorithm takes as input a set of data points and their respective k nearest neighbors, and returns as output all points' low-dimensional coordinates.

We use the PAVs from the unlabeled video to build the manifold. Recall that $\boldsymbol{p}_{k_q}$ denotes the PAV for the q-th frame within the k-th track in the unlabeled video (dropping the superscript v for clarity). We determine neighbors for LLE using a similarity function capturing both temporal nearness and pose similarity: $A(\boldsymbol{p}_{k_q}, \boldsymbol{p}_{j_r}) =$

$$\lambda \, \exp\left(-\left\|\boldsymbol{p}_{k_q} - \boldsymbol{p}_{j_r}\right\| / \sigma_p\right) + (1-\lambda) \exp\left(-\left\|q - r\right\| / \sigma_t\right),$$

where $\|q - r\| = \infty$ if $k \neq j$, that is, if the two inputs are from different tracks. Here σ_p and σ_t are scaling parameters, set to the average distance between all PAVs and frame numbers, respectively, and the weight λ controls the influence of the two terms. Note that an example's neighbors under A can span poses from both the same and different tracks. After applying the LLE embedding, each original PAV $\boldsymbol{p}^v \in \mathbb{R}^P$ has a low-dimensional counterpart $\hat{\boldsymbol{p}}^v \in \mathbb{R}^d$.

Next, for each training snapshot, we find nearby poses on the manifold to generate synthetic examples. Specifically, for snapshot $\boldsymbol{p}^i_j$ with nearest neighbor $\boldsymbol{p}^v_{j*}$ in PAV space, we take the associated $\hat{\boldsymbol{p}}^v_{j*}$ manifold coordinate, and compute its closest two embedded points from the video.[2] (We choose two simply to be consistent with the example-based method above.) Finally, we augment the training set similarly to above, putting the original PAVs for those two instances labeled with the snapshot's category into $\mathcal{S}^+$.

Discussion Whether example- or manifold-based, we stress that the synthetic examples exist in *pose* space—not raw image space. Thus, we are padding our training set with plausible poses that could immediately precede or follow the observed static snapshot poses, and ignoring surrounding context, objects, etc. Furthermore, it is entirely possible that the action the person in the video was performing when taking on that pose was *not* the action labeled in the static snapshot. Our idea is that the generic human motion dynamics gleaned from the unlabeled video allow us to

[2] One could alternatively use an out-of-sample extension to LLE [10] when collecting the manifold neighbors.

extrapolate the poses observed in novel static images, at least to very near instants in time. This allows, for example, the system to infer that a kicking action could take on more diverse poses than the few available in the training set (compare left and right panels in Figure 3.1).

The proposed approach can be seen as a novel form of transfer or semi-supervised learning in that my proposed method incorporates unlabeled data. The synthetic examples are technically unlabeled data, but our approach refers their label with underlying pose dynamic patterns existing in video data. While transfer learning adapts a learned model to a new category with a few labeled examples from that category, my method synthetically picks the images across various categories as additional labeled data to improve training.

3.1.4 Training with a Mix of Real and Synthetic Poses

Finally, we use the augmented training set $\mathcal{S}^+$ to train SVM action classifiers to predict the labels of novel images. Rather than directly use the data as-is, we specifically account for the uncertainty in the synthetic examples in two ways. First, we employ domain adaptation to account for the potential mismatch in feature distributions between the labeled snapshots and unrelated video. Second, we use penalty terms in the SVM objective that put more emphasis on satisfying the label constraints for the real data examples compared to the synthetic ones.

Domain adaptation (DA) techniques are useful when there is a shift between the data distributions in a "source" and "target" domain. They typically transform the data in some way that accounts for this discrepancy—for example, by mapping

to an intermediate space that shares characteristics of both domains. In our case, we can think of the static snapshots (whether training or testing) as the target domain, and the unlabeled video as the source domain.

We use the "frustratingly simple" DA approach of [31]. It maps original data in $\mathbb{R}^P$ to a new feature space of dimension $\mathbb{R}^{3P}$, as follows. Every synthetic (source) pose example $\boldsymbol{p}^v$ is mapped to $\boldsymbol{p}^{v'} = [\boldsymbol{p}^v, \boldsymbol{p}^v, \mathbf{0}]$, where $\mathbf{0} = [0, \ldots, 0] \in \mathbb{R}^P$. Every real (target) pose example is mapped to $\boldsymbol{p}^{i'} = [\boldsymbol{p}^i, \mathbf{0}, \boldsymbol{p}^i]$. This augmentation expands the feature space into a combination of three versions of it: a general version, a source-specific version, and a target-specific version. The classifier benefits from having access to all versions to find the most discriminative decision function.

Given the domain-adapted features, we train one-vs.-all SVM classifiers. During training, we want to reflect our lower confidence in the synthetic training examples, as well as account for the fact that they will outnumber the real examples. Thus, we use two separate constants for the slack penalty C in the standard SVM objective, in order to penalize violating label constraints on real data more heavily. Specifically, the cost for label errors on the real examples C_{real} is set to 1, while the cost for synthetic examples $C_{synth} \leq 1$ (set via cross-validation). This weighting, combined with the soft-margin SVM, will give some resilience to off-base synthetic pose examples wrongly hypothesized by our method. This can occur, for example, if the nearest PAV or manifold neighbor is quite distant and thus serves as a weak proxy for the training snapshot's pose.

3.2 Experimental Results

I demonstrate my approach on three datasets for recognizing activities in both images and videos.

3.2.1 Datasets

For the unlabeled video data, we use the training and testing clips from the Hollywood Human Actions dataset [97]. We stress that none of the activity labels are used from these clips. In fact, only one label in Hollywood overlaps with any of the data below (*phoning* is in both PASCAL and Hollywood). To get person tracks, we use the annotation tool provided by [161]. This allows us to focus our evaluation on the impact of our method, as opposed to the influence of a particular person tracking method.

For the recognition task with static test images, we test on both the 9 actions in the PASCAL VOC 2010 dataset [47] *(phoning, playing instrument, reading, riding bike, riding horse, running, taking photo, using computer, walking)* as well as 10 selected verbs from the Stanford 40 Actions dataset [179] *(climbing, fishing, jumping, playing guitar, riding a bike, riding a horse, rowing a boat, running, throwing frisbee, walking the dog)*. While the latter has 40 total verbs, we limit our experiments to those 10 where the baseline has reasonable precision using a body pose descriptor alone; many of the others are strongly characterized by the objects that appear in the scene. We call it Stanford 10. For PASCAL, we use (maximally) the 301 persons from the training set to train, and the 307 persons in the validation set to test. For Stanford 10, we randomly select (maximally) 250 and 1672 persons

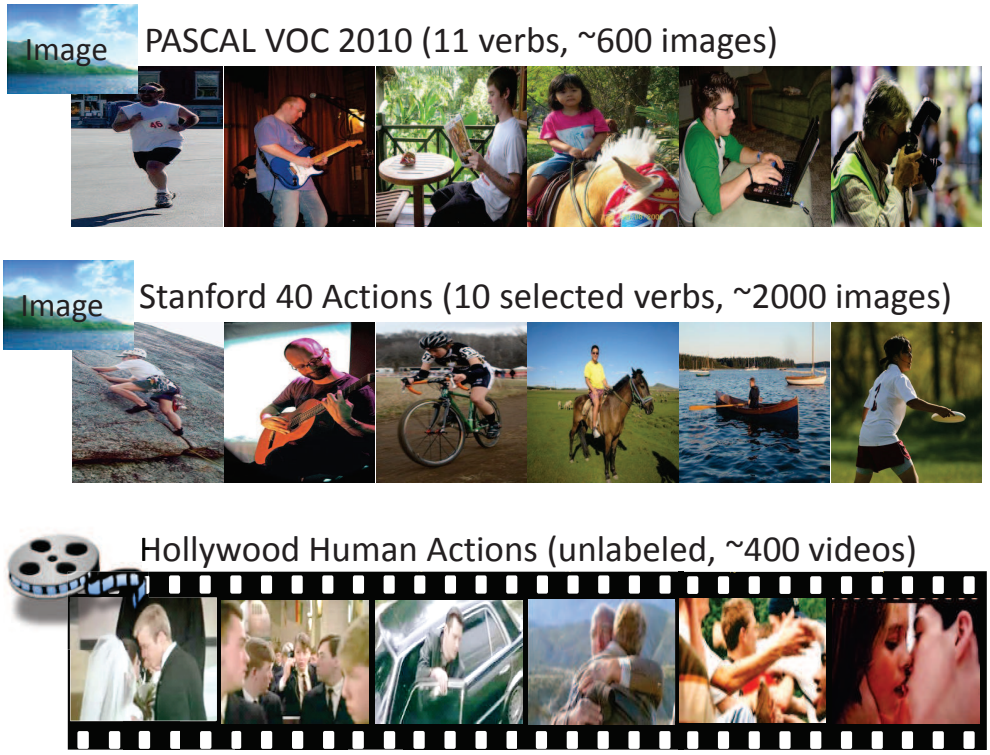


Figure 3.4: Examples of PASCAL, Stanford 40 Actions, and Hollywood Human Action datasets.

for training and testing, respectively, based on the train/test split suggested by the authors. See Figure 3.4 for example images of these three datasets.

For the video recognition task, we compile a test set from multiple video sources, since no existing video dataset has both images and videos for a set of action labels. We gather 78 test videos from the HMDB51 [92], Action Similarity Labeling Challenge [92], and UCF Sports [138] datasets that contain activities also appearing in PASCAL: *phoning*, *riding bike*, *riding horse*, *running*, and *walking*. Note that the unlabeled video source remains the Hollywood data for this task; in all

cases, the only labels our method gets are those on the static snapshots in PASCAL.

We fix the dimensionality for LLE $d = 10$, and the affinity weight $\lambda = 0.7$. We use χ^2-kernels for the SVMs, and set the SVM penalty $C_{synth} = 0.1$ for image recognition and $C_{synth} = 0.5$ for video recognition, based on validation with the PASCAL training data.

3.2.2 Recognizing Activity in Novel Images

The primary comparison of interest is to see whether recognition improves when adding our synthetic training data, versus a baseline that does everything else the same (i.e., PAV representation, SVM, etc.), but uses only the original training snapshots. This baseline corresponds to the state-of-the-art method of [110], and we denote it **Original** throughout. In addition, we provide two more baselines to help isolate the reason for our method's advantage. The first, **Original+random**, replaces our method's nearest neighbor selection with a randomly selected video pose. The second, **Original+synthetic-current-frame**, uses only the matched neighbor to synthesize an example (i.e., it lets $T = 0$). This baseline is useful to see the extent to which we need to extrapolate poses across *time* (dynamics), versus merely padding the data with variations in *appearance* (similar instances of the same pose).

Figure 3.5 shows the mean average precision (mAP) test accuracy as a function of the number of training images, for both static image datasets. To robustly estimate accuracy with few training samples, we run the experiment five times with different randomly sampled training images (when using less than all the data) and report the average. Our approach substantially boosts accuracy when few training

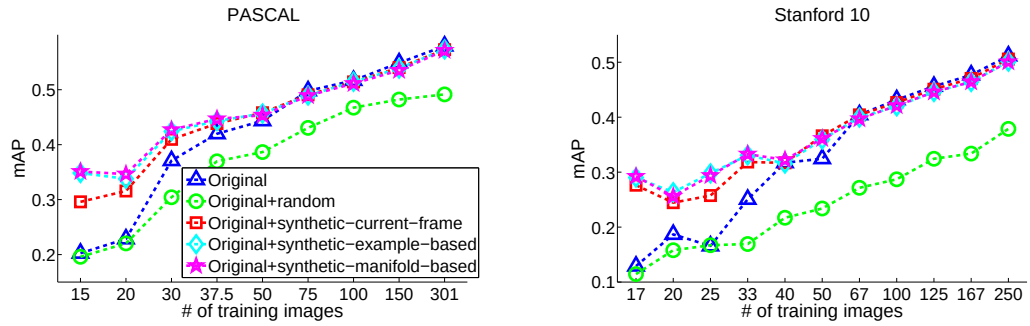


Figure 3.5: Accuracy on static action recognition datasets as a function of the number of training images. Our method shows dramatic gains with very few labeled snapshots, and maintains similar accuracy to the baseline when training exemplars are plentiful.

snapshots are available. As expected, having only few exemplars accentuates our method's ability to "fill in" the related poses. On the other hand, when training examples are plentiful (hundreds), there is less to be gained, since more variation is already visible in the originals; in fact, our results are comparable to the baseline's in the rightmost part of the plots.[3] Adding poses from random frames degrades accuracy across the board, confirming that our method's gain is not due to having *more* pose examples; rather, it synthesizes *useful* ones relevant to the recognition task. Adding a pose from the neighbor frame itself ("current") increases the baseline's accuracy by synthesizing more varied appearances of the poses in the training set, but it is inferior to using the pose dynamics as proposed.

Figure 3.6 shows examples of images responsible for synthetic poses added to the original training set for PASCAL. We see how useful poses can be found

[3] And our numbers roughly replicate those reported in [110] for PASCAL—we obtain 57.94 vs. 59.8 mAP when using all training data.

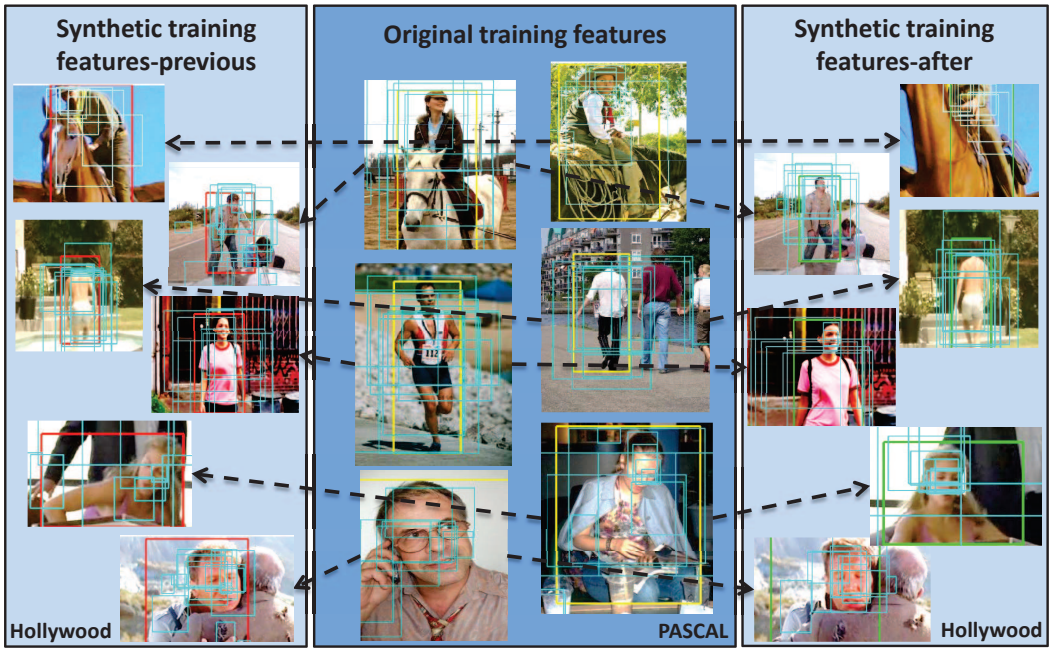


Figure 3.6: Six real examples showing the frames our method found in unlabeled video (left and right panels) and used to expand the original training poses in snapshots (center panel). Each pose in the center panel finds a neighbor in the unlabeled video $\boldsymbol{p}^v_{j^*}$, which generates a synthetic example for what could come immediately before ($\boldsymbol{p}^v_{j^*-T}$, left) and after ($\boldsymbol{p}^v_{j^*+T}$, right) that pose. Red/yellow/green boxes denote person bounding boxes, and smaller cyan boxes denote poselet detections. Dotted arrows connect to corresponding synthetic frames.

across activity categories. For example, the bottom image of a man phoning has synthetic poses generated from a man who is not phoning—but who nonetheless takes on poses and facial expressions that could have been (were the objects in the scene different). In the special case that a familiar action actually appears in the unlabeled video, it too can help, as we see in the horse-riding and walking examples. In all examples, notice how the synthetic examples simulate slight variations over time. This is how our approach fleshes out the training set.

Note that our improvements are in spite of the fact that only one label overlaps between PASCAL and Hollywood, and zero overlap between Stanford 10 and Hollywood. We find that for the largest training set size on PASCAL ($N = 301$), 23 PASCAL images match to a Hollywood clip that shows the verb phoning. Among those 23, only two of them are themselves *phoning*. Hence, our results clearly show the category-independent nature of our approach. Poses from distinct actions are relevant to connect the dots between sparse exemplars.

Next we compare our example- and manifold-based strategies for gathering pose neighbors. The mAP averaged over all classes (Fig. 3.5) is fairly similar for both. Figure 3.7 shows the AP gain of our two methods (compared to Original) for each individual class in PASCAL when training with $N = 20$ examples (ignore the x dimension for now). Indeed, for many classes their gains are similar. However, manifold-based has a noted advantage over example-based for the actions *running* and *using computer*. On Stanford 10, it is stronger for *running* and *climbing* (not shown). What these actions seem to have in common that they entail some repeated motion. We hypothesize the manifold does better in these cases since it captures both temporally nearby poses and appearance variations.

Figure 3.7 also shows that there is a correlation between those classes most benefited by our method and their lack of diversity. We measure diversity by the average inter-PAV distance among training examples. Low distance means low diversity. Just as a training set that is too small needs our method to fill in intermediate poses, so too a class whose examples are too tightly clustered in pose space (e.g., due to a dataset creator's unintentional bias towards "canonical poses") may benefit

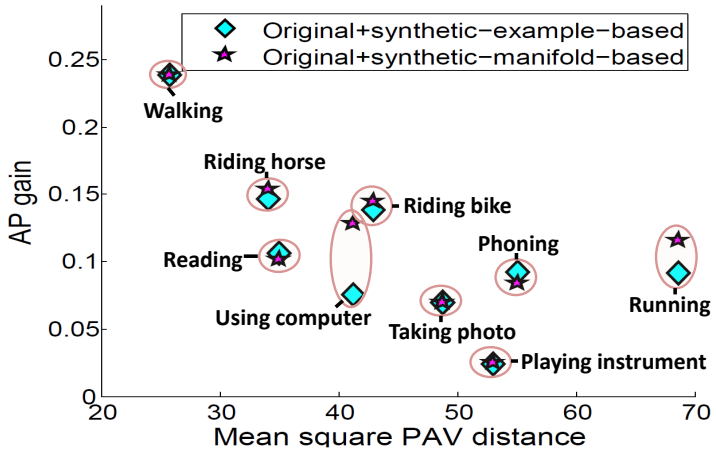


Figure 3.7: Per class accuracy gains by our methods as a function of the diversity of the original PASCAL data. See text.

Dataset	PASCAL		Stanford 10	
Domain-adaptation?	No	Yes	No	Yes
Example-based	0.4243	**0.4320**	0.3308	**0.3378**
Manifold-based	0.4271	**0.4327**	0.3328	**0.3404**

Table 3.1: Impact on mAP of domain adaptation on the static datasets.

most from our method.

Table 3.1 isolates the impact of domain adaptation on our results, when the number of training examples $N = 30$. (The impact is very similar no matter the training set size.) We see that DA gives a modest but noticeable gain in accuracy for both variants of our method, showing it is worthwhile to model the potential data mismatch between the unlabeled video and training snapshots. We suspect the PAV pose descriptors are also playing a role in accounting for the domain shift, since they abstract away some nuisance factors that could differ between the two sources (e.g., lighting, scale).

	Original	Original+synthetic example-based	Original+synthetic manifold-based
Without DA	0.3846	**0.5128**	0.4872
With DA	N/A	**0.5382**	0.5128

Table 3.2: Accuracy of video activity recognition on 78 test videos from HMDB51, ASLAN, and UCF data.

3.2.3 Recognizing Activity in Novel Video

Next, we apply our method to predict activities in novel *video*, still using the same static image training set idea (see dataset details in Sec. 3.2.1). We use a simple voting approach to predict the label for the entire video. First, we classify each frame independently, generating a probability for each possible label $1, \ldots, C$. Then, we sum the probabilities across all frames to get the final prediction. Note that this test should allow our method to shine, since the novel videos will exhibit many intermediate poses that the original snapshots did not cover—but that our method will (ideally) synthesize. For this experiment, we transform the domain adapted features using $\boldsymbol{p}^{v'} = [\boldsymbol{p}^v, \mathbf{0}, \mathbf{0}]$, since the train, test, and synthetic data are all from different domains.

Table 3.2 shows the results. We compare our method to the Original baseline, and also show the impact of domain adaptation. Our method makes a substantial improvement in accuracy. Its synthetic padding of the data makes the training set less sparse, yielding more reliable predictions on the video frames. Domain adaptation again boosts the accuracy further.

3.3 Conclusions

In this chapter, I propose a framework to augment training data for learning human actions without additional labeling cost. My approach leverages knowledge of human pose patterns over time, as represented by an unlabeled video repository. To implement our idea, we explore simple but effective example- and manifold-based representations of pose dynamics, and combine them with a domain adaptation feature mapping that can connect the real and generated examples.

Our results classifying activities in three datasets show that the synthetic poses have significant impact when the labeled training examples are sparse. We demonstrate the benefits with a state-of-the-art local pose representation; however, our idea is not coupled specifically with that method, and it has potential to boost alternative descriptors in similar ways.

I have shown how to interconnect the patterns of pose changes over time in unlabelled video pools to expand our knowledge of human action with very few snapshots. Next, I study how to relate human poses across different views.

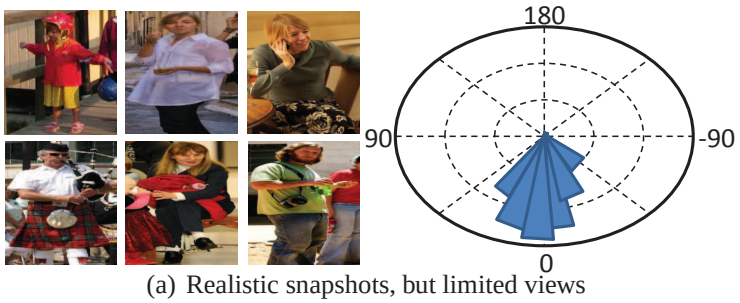
Chapter 4

Inferring Human Pose in Unseen Views

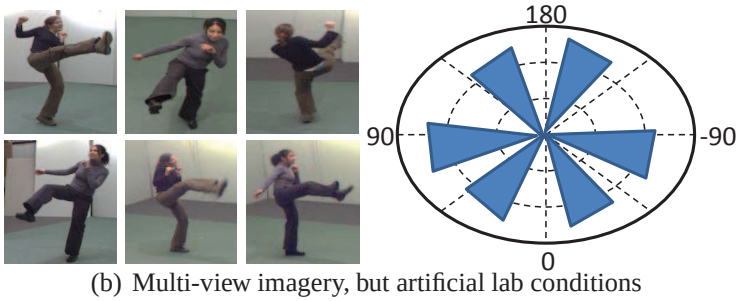
[1]In Chapter 3, my proposed approach connects static snapshots to unlabelled video sequences for better coverage of pose variations. With the help of the proposed approach, we are able to learn a new human actions efficiently with very few examples. In this chapter, I am going to further explore another obstacle in understanding human poses: viewpoint variation.

Since we are living in 3D space, the appearance of our pose would look quite different in different views. To learn an action model for each of multiple views, we would seemingly require examples collected from each of the views. Currently, internet images and movies offer abundant realistic examples of humans performing various actions, but they are naturally biased towards certain viewpoints (see Figure 4.1(a)). This is to be expected, since humans tend to take photos of other humans as they face the camera. As a result, nice "in the wild" examples are sparse for many other viewpoints, and today's challenge datasets (e.g., PASCAL Actions [47]) are restricted to canonical viewpoints. On the other hand, efforts to collect data specifically from multiple views are prone to scripted behavior and

[1]The work in this chapter was supervised by Dr. Grauman and originally published in: Inferring Unseen Views of People. C.-Y. Chen and K. Grauman. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, June 2014.



(a) Realistic snapshots, but limited views



(b) Multi-view imagery, but artificial lab conditions

Figure 4.1: The data dilemma for human images. (a) Single view images are often realistic and "unstaged", but populate only a sparse set of camera viewing angles. (b) Multi-view data give full view coverage, but are more artificial in terms of acted poses and simplistic backgrounds. Our method makes use of any available images to envision seen poses in unseen viewpoints.

artificial lab environments (see Figure 4.1(b)). This is also to be expected, since the actors must be instructed to do certain actions while in the special synchronized multi-camera rig.

How can we overcome this dilemma? How can we obtain realistic human image data from varied viewpoints? Rather than physically place more cameras around subjects, my goal is to use whatever viewpoints we *do* have to generate virtual views in those we do not. To this end, I propose a view synthesis approach based on tensor completion. The key idea is to recover the latent factors that relate

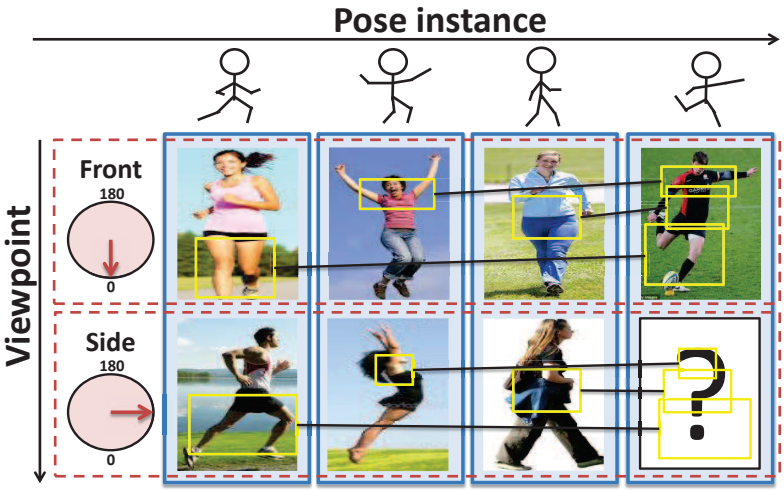


Figure 4.2: The proposed approach discovers the latent factors that relate viewpoint and body pose, and uses them to infer unseen views. For example, despite never seeing a kicking pose from any view but frontal (top right image), it hallucinates what it will look like from the side (bottom right). The key is to learn connections between similar looking parts in different poses (here marked with lines for illustration only).

viewpoint and body pose *without* observing the two neatly varying together—that is, *without observing each pose in all views during training*. We observe that from the same viewpoint, people look similar in certain portions of the image, even when they are performing different actions or poses (see Figure 4.2). Using a latent factor model, I aim to discover these relationships and use them to infer appearance in unseen views.

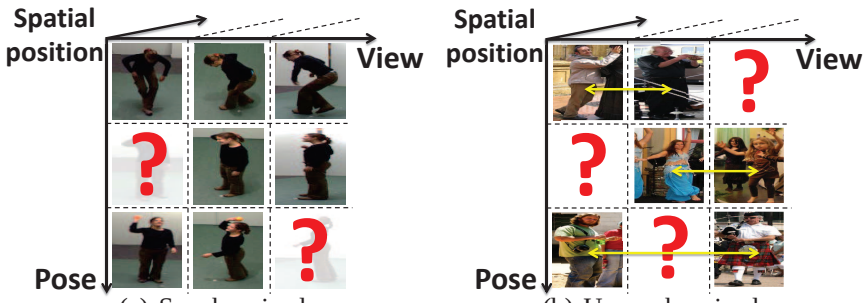
4.1 Approach for Inferring Unseen Views

I pose unseen view inference as a tensor completion problem. Throughout, we consider a set of discrete viewpoints consisting of M orientations of the person with respect to the camera (facing front, front-left, etc.). As input, our method takes cropped images of people organized by their discrete viewpoint ($M = 5$ or 8 in our datasets). As output, our method returns image descriptors capturing the appearance of those same people in each viewpoint from which they were not observed.

I consider two scenarios: *synchronized* and *unsynchronized*. For the synchronized case, the input images include (at least some) examples of people observed simultaneously by multiple cameras. Any subset of the M views might be present for a given instance, and *the poses in the examples are not annotated in any way* (i.e., no stick figures are given). See Figure 4.3(a). For the unsynchronized case, the input images are single-view snapshots, such as those one might typically find in online photo collections. See Figure 4.3(b). In this case, we assume each training image is annotated with body pose (joint positions). In either case, we assume the inputs contain a variety of body poses, though there may be an imbalanced representation of certain poses and viewpoints.

4.1.1 Discovering the Latent Factors

Our model represents human appearance as a function of pose, viewpoint, and position in the image. The goal is to fit a low-dimensional factor model to the observed data, such that the spatially varying appearance can be approximated as a combination of some latent pose and viewpoint factors. As discussed above, the



(a) Synchronized (b) Unsynchronized

Figure 4.3: Visualizing the 3D tensor $\mathbf{X}$ in the synchronized (left) and unsynchronized (right) cases. (We display a whole image for visualization purposes, though really its descriptor extends out in the third dimension of the tensor.)

fact that some local appearance patterns re-occur between different poses suggests that such latent factors exist. Intuitively, they might correspond to things like local body configurations (arm outstretched, knee bent, etc.), lighting conditions, or body types.

For each input image, we first extract its K-dimensional appearance descriptor. We use Histograms of Oriented Gradients (HOG) [29], which offer robustness to small shifts and rotations. HOG pools the gradients within a grid of cells, and histograms the pixels per cell into orientation bins; each block of HOG descriptor dimensions originates from a particular spatial region in the image, and adjacent blocks originate from adjacent regions (except for boundary cells).

Different from Chapter 3, where I use poselet to represent person's pose, here I choose HOG because each feature dimension of HOG corresponds to the content of a spatial region in the image. In contrast, recall that poselet descriptors discard the spatial information and only keep the histogram count of detected body

parts. As shown in Figure 4.2, the spatial correlation between different parts is also a key to our proposed method. For poselet feature, each feature dimension corresponds to the histogram count of a type of poselet and the spatial information is less preserved in the descriptor.

Then, we assign each image to one of the M viewpoints. We currently use ground truth orientation data for this step, as it is available with multiple public datasets [166, 16]; however, automatic methods are also possible, e.g., [110].

Let $i = 1, \ldots, N$ index the input data, where each i corresponds to a unique moment in time—that is, a single snapshot, or a set of multi-view images taken simultaneously. For each of the N inputs, we thus have a descriptor for some number between 1 and M of the total possible viewpoints. Each i captures a distinct pose, whatever pose the human is doing. Thus, we stress that while we refer to the N inputs as "poses", if at least some inputs are multi-view, we do not require pose *annotations* for the input data.

Using this data, we construct a 3D tensor $\mathbf{X} \in \mathbb{R}^{N \times M \times K}$, where entry x_{ij}^k corresponds to the image descriptor value in the i-th pose, the j-th view, and the k-th feature dimension (which reflects image position). Let $\mathbf{P} \in \mathbb{R}^{D \times N}$, $\mathbf{V} \in \mathbb{R}^{D \times M}$, and $\mathbf{S} \in \mathbb{R}^{D \times K}$ denote matrices whose columns are the D-dimensional latent feature vectors for each pose, view, and spatial position, respectively. We suppose that x_{ij}^k can be expressed as an inner product of latent factors, $x_{ij}^k \approx \langle P_i, V_j, S_k \rangle$, where a subscript denotes a column of the matrix. In matrix form, this means $\mathbf{X} \approx \sum_{d=1}^{D} P_{d,:} \circ V_{d,:} \circ S_{d,:}$, where a subscript $d, :$ denotes the d-th row in the matrix, and $\circ$ is the outer product.

To recover the latent factors, we use the Bayesian probabilistic tensor factorization approach of [170], which extends probabilistic matrix factorization [141, 142] to accommodate time-evolving consumer data for movie recommendation tasks. To account for uncertainty, we represent the likelihood distribution for the observed descriptors by

$$p(\mathbf{X}|\mathbf{P},\mathbf{V},\mathbf{S},\alpha) = \Pi_{i=1}^{N}\Pi_{j=1}^{M}\Pi_{k=1}^{K}\left[\mathcal{N}(x_{ij}^{k}|\langle P_i, V_j, S_k\rangle, \alpha^{-1})\right]^{I_{ij}},$$

where $\mathcal{N}(x|\mu,\alpha)$ denotes a Gaussian with mean μ and precision α, and I_{ij} is an indicator variable equal to 1 if pose i appears in view j, and 0 otherwise. We use Gaussian priors for each of the latent factors P_i, V_j, S_k. For pose and viewpoint we use independent Gaussians, while for the spatial factors we use the prior $S_k \sim \mathcal{N}(S_{k-1}, \Sigma_S)$, for $k = 2,\ldots,K$, which reflects that descriptor values are likely to vary smoothly in spatially close regions.[2] Let Θ denote a set of random variables comprised of the mean and covariance of all three factors, including Σ_S. For all Gaussian prior hyper-parameters (α and the variables in Θ), we use conjugate distributions as priors to facilitate subsequent sampling steps.

Following [142, 170], we integrate out all the model parameters and hyper-parameters to obtain a predictive distribution for an unseen view given all observed input images:

[2]Accounting separately for the boundary cells (which need not be smooth a priori) would add complexity to the model, and we find it is sufficient in practice not to.

$$p(\hat{x}_{ij}^k|\mathbf{X}) = \int p(\hat{x}_{ij}^k|P_i, V_j, S_k, \alpha)p(\mathbf{P}, \mathbf{V}, \mathbf{S}, \alpha, \Theta|\mathbf{X})\ d\{\mathbf{P}, \mathbf{V}, \mathbf{S}, \alpha, \Theta\}.$$

Compared to solving for a single point estimate for the MAP factors $\mathbf{P}^*$, $\mathbf{V}^*$, $\mathbf{S}^*$, this helps prevent overfitting to poorly tuned hyper-parameters. It is approximated using Markov chain Monte Carlo (MCMC) sampling:

$$p(\hat{x}_{ij}^k|\mathbf{X}) \approx \sum_{l=1}^{L} p(\hat{x}_{ij}^k|P_i^{(l)}, V_j^{(l)}, S_k^{(l)}, \alpha^{(l)}), \tag{4.1}$$

where L denotes the number of samples. The samples $\{P_i^{(l)}, V_j^{(l)}, S_k^{(l)}, \alpha^{(l)}\}$ are generated with Gibbs sampling on a Markov chain whose stationary distribution is the posterior over the model parameters and hyper-parameters $\{\mathbf{P}, \mathbf{V}, \mathbf{S}, \alpha, \Theta\}$. Sampling is initialized using the MAP estimates of the three factor matrices. See [170] for details.

With this tensor formulation, we capture the global influence that image position has on all the poses and viewpoints, which is very informative for cropped person images. For example, the model can learn that the presence of strong -45 degree gradients in cells in the bottom right of the person bounding box when viewed from the front (due to an extended left leg) suggests the likely presence of 45 degree gradients within the associated bottom left cells if he were viewed from behind.

We choose to infer descriptors, rather than raw pixels. The gradient-based HOGs offer robustness to low-level appearance differences (e.g., clothing), such that we can expect to learn latent factors with less input data than would be needed

for raw pixels. Inferring pixel intensities, though in principle possible with the same approach, would likely waste modeling effort on unneeded detail (a typical person bounding box in our datasets contains 6,000 pixels, but only 108 HOG dimensions). In addition, as we demonstrate below, we can use the inferred views directly in later learning tasks, since most vision methods operate in a feature space other than pixels. Plus, to visualize the results, we can "invert" HOG descriptors back into image space with [160].

4.1.2 Learning with Unsynchronized Single-View Images

Next we generalize our approach to handle the challenging case where only unsynchronized single-view data is available. Doing so will allow us to exploit existing realistic data sources, such as photos on Flickr. Presumably humans can infer unseen views because they have seen many individuals in various poses and viewpoints, not because they have seen carefully orchestrated multi-view examples for individual people. They understand the pose associations across individuals. In a similar vein, our idea is to link snapshots that contain *similar* 3D body poses, but *different* viewpoints. In this way, a pose "instance" in the tensor can be comprised of different individual people (as depicted in Figure 4.3(b)).

This variant requires pose-labeled training data, using either manual or automatic annotations. Good tools are available to semi-automate pose labeling [16], making this requirement manageable.

Let $\boldsymbol{p}_q \in \mathbb{R}^{3J}$ denote the normalized body pose configuration for image q. Its $3J$ elements are the 3D positions of J body joints, normalized to a common

coordinate system where they can be meaningfully compared. Specifically, we shift the raw skeleton to place the center of the hips at the origin, rotate it to align the plane connecting the hips and neck to be orthogonal to the z axis, and scale it to the average head-to-toe height. We estimate the pose distance between two images as $d(q,r) = ||\boldsymbol{p}_q - \boldsymbol{p}_r||_2$. Then we sort all training pairs by $d(q,r)$, and take any pairs whose pose distance is less than 0.2 times the average distance. Each such pair provides two K-dimensional HOG entries for the tensor, placed at the appropriate two columns based on their viewpoints.[3] Once the linked pairs are entered into the tensor, we perform inference as described above.

With this extension, even if an "in the wild" snapshot was observed from just a single viewpoint, we can infer its appearance in novel views. As such, our method provides downstream estimation tasks (e.g., action recognition) with data that is both more complete *and* realistic. Furthermore, while our current implementation focuses on the multi-view and single-view cases separately, our approach naturally supports a mix of both types of data. In that case, the algorithm will learn the multi-view constraints from synchronized instances and propagate them to single-view instances during inference.

4.2 Experimental Results

We validate our approach on two public datasets. The first, INRIA Xmas Motion Acquisition Sequences (IXMAS) [166], contains multi-view synchronized

[3]Preliminary tests in which we link beyond pairs of examples did not show a noticeable difference in results.

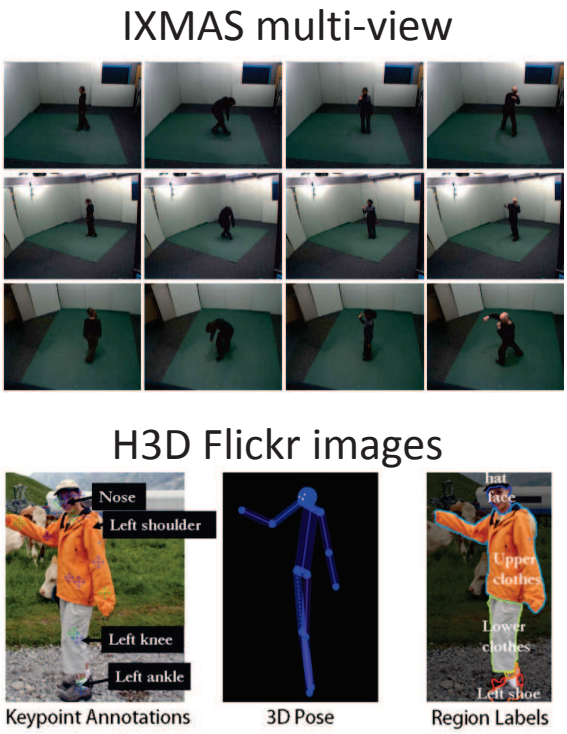


Figure 4.4: Examples of IXMAS and H3D datasets.

data from $M = 5$ cameras, with 11 actions (check watch, cross arms, kick, etc.) performed by 10 actors, for 16,800 total images. The second, Humans in 3D (H3D) [16], contains 2,378 single-view Flickr images, with people doing various unscripted poses (reaching, walking, riding a bike, etc.), and has 3D pose annotations for $J = 33$ joints done by MTurkers. We use the viewpoint annotations of [110]. See Figure 4.4 for example images of these two datasets.

We extract HOG with 9 cells and 12 bin histograms per cell, yielding a $K = 108$ dimensional descriptor per image. We use the factorization code of [170], and fix the latent factor dimensionality to $D = 500$ and the number of samples

$L = 500$, based on cross-validation on training data, and $\alpha = 2$ as default. We clip inferred outputs to $[0, 1]$, the valid HOG range. With these parameters, and with $N = 2,200$ instances, learning the latent factors takes about 6 hours. Inferring feature values requires only two inner products, which takes < 1 ms.

We evaluate how well our inferred views match the (withheld) ground truth images. In addition, we compare to a variety of state-of-the-art view-invariant recognition methods as well as two baseline techniques for virtual view creation: 1) MEMORY, a memory-based tensor completion approach and 2) COPY, a method that copies observed images from nearby views. For MEMORY, we adapt a neighborhood approach in collaborative filtering [90] to our problem setting. For COPY, we find the observed image in the training data *for the very same pose instance* that is nearest in viewpoint to the desired unseen view, and copy its HOG descriptor. For example, if the needed view j were frontal, and the view 45 degrees off of frontal appears in the training set, that would be the estimate. Note that a traditional warping approach is inapplicable for these tests, since it demands multi-view calibrated data, and can warp only to fairly nearby views (i.e., not ground to overhead).

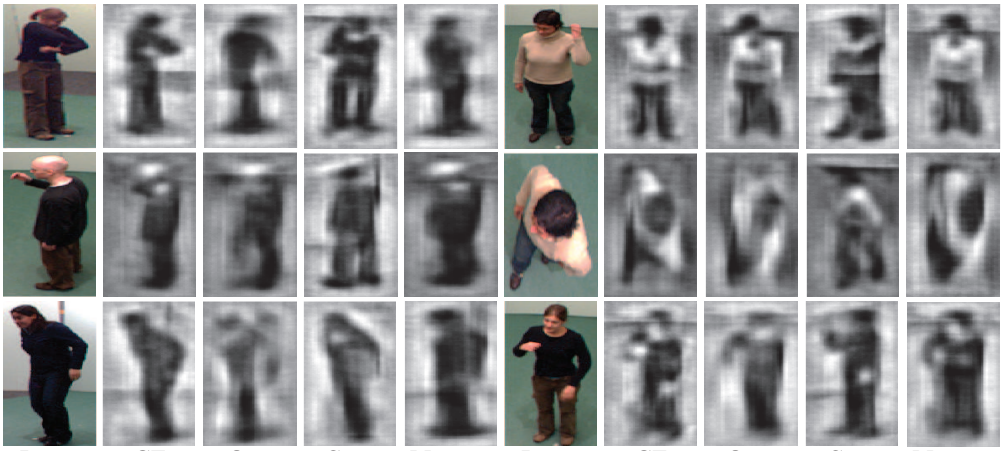
In the following, we first evaluate the inferred views' accuracy (Sec. 4.2.1). Then we use the virtual views for two applications: action recognition (Sec. 4.2.2) and viewpoint estimation (Sec. 4.2.3).

4.2.1 Accuracy of Inferred Views

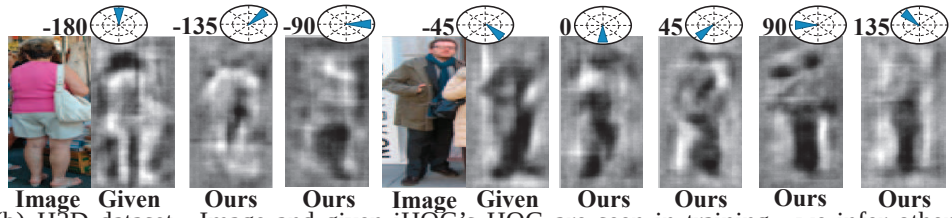
Figure 4.5 visualizes inferred views using the "HOG goggles" inverted-HOG (iHOG) technique, which inverts a HOG descriptor back to a natural im-

age [160]. Here we use HOG descriptors with higher dimension (90 cells$\times$12 bins =2970) to provide detailed visualization. We compare the view inferred by our method to the iHOG for the real ground truth (GT) image, which is the upper bound on quality. The two often look quite similar, which means our method infers the true appearance well. While COPY's results can look realistic—after all, they originate from HOGs on real images—they are not as accurate as ours. This underscores the value in modeling the latent factors for all observations, rather than simply matching to the nearest available view. Our advantage is most striking in the most difficult cases, such as inferring the overhead view (middle row, right side of (a)). For poses that appear similar between views (bottom row, left side of (a)), COPY is competitive, as expected. The H3D visualizations (b) are noisier due to fewer observed features and cluttered backgrounds, yet we still capture the shape of the person and some articulated details of the pose (e.g., see the bent arm in far right). (Note, on H3D COPY simply returns the given iHOG for all other views.) See Supp. for more examples.

Figure 4.6 quantifies these observations. We randomly sample 200 images for each action in IXMAS, for a total of 2,200 images. Then for each action in turn, we withhold all images for that action in a given view, apply factorization, and compare the inferred unseen views to the withheld ground truth. We plot the Summed Square Difference (SSD) error between inferred and actual views, for each view in IXMAS. (H3D lacks the ground truth to make this evaluation possible.) Our factorization method outperforms both baselines. As to be expected, view 5, the overhead view, is most difficult for all methods; nonetheless, our inferred views



(a) IXMAS dataset. Image and its GT iHOG are not seen in training—we infer it.



(b) H3D dataset. Image and given iHOG's HOG are seen in training—we infer other unseen views.

Figure 4.5: Visualization of inferred views using inverted HOGs. Best viewed on pdf.

remain 74% better than COPY and 6% better than MEMORY.

These results validate the main goal of our approach: to accurately map seen poses to unseen views, even when training examples are single-view, asynchronous, and captured in complex environments. In the remaining results, we will further demonstrate that having estimated the unseen views well, we are better positioned to train viewpoint-sensitive models for recognition tasks.

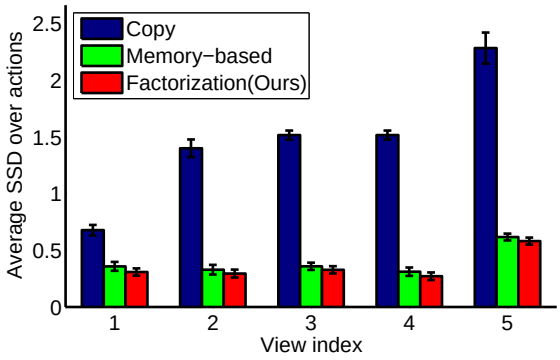


Figure 4.6: Error in inferred views.

4.2.2 Recognizing Actions in Unseen Views

Next, we use our inferred views to train a system to recognize actions from a viewpoint it never observed in the training images. As above, for each IXMAS action label in turn, we hold out all its images in a given viewpoint, and then infer the unseen views. We use those inferred HOGs to train a viewpoint-specific one-vs.-rest SVM action classifier for that action category; the positive exemplars are all synthetic, while the negative exemplars are real images from all other action labels. We evaluate accuracy on a test set of single-view static images consisting of 200 real positives and 2000 real negatives.

Table 4.1 shows the results. Our method significantly outperforms the baselines. Compared to MEMORY, our recognition advantage is much greater than our SSD advantage in Figure 4.6, which suggests the perceptual quality differences are greater than what SSD captures. We also show an upper bound—the accuracy that would be obtained if the *real* images had been available, rather than inferred ("Ground truth"). Naturally, the accuracy is higher using real training images; still,

COPY	MEMORY	Ours	Ground truth
15.08 (2.45)	20.39 (2.49)	**34.32 (3.47)**	60.36 (2.51)

Table 4.1: Action recognition accuracy (mAP) in an unseen viewpoint on IXMAS. Numbers in parens are standard errors.

we more than double the accuracy of a method that uses the nearest available real view (COPY).

Figure 4.7 evaluates the impact of input data sparsity. We repeat the recognition task above, but now with an increasingly sparse set of real input views for training. To increase sparsity, we remove views at random. Our method's accuracy is fairly stable up until about 40% (i.e., when 60% of the tensor is unobserved), showing the power of the latent factors with rather incomplete data. While our accuracy starts to decline when the observed features comprise less than half of the tensor entries, it is still substantially better than the baselines. With only 20% observed data, all methods do similarly, indicating insufficient information about the feature correlations between the views. COPY's standard error increases with sparsity; it suffers once fewer nearby views are available.

Next, we demonstrate how our method can infer missing views in the face of partial occlusions. Table 4.2 shows the results, for action recognition on the first five IXMAS actions. The columns compare our method's accuracy in three scenarios: 1) with no occlusions, 2) when training examples are partially occluded, and 3) when test examples are partially visible. To generate the training set occlusions, we randomly remove 20% of the HOG cells; to generate the test set occlusions, we omit the lower body region. Comparing columns 1 and 2, we see our method

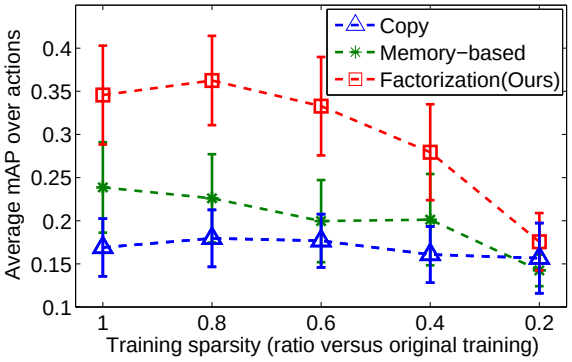


Figure 4.7: Accuracy in unseen views as a function of tensor sparsity.

No occlusions	Occluded training	Partially visible testing
37.7 (3.06)	36.9 (3.03)	52.6 (2.07)

Table 4.2: Testing the impact of occlusions (average mAP)

maintains its accuracy in spite of occluded training examples, showing the latent factors have a similar effect for missing data within an image, not just within the viewpoints. Comparing columns 1 and 3, we see that if the unobserved views are partially visible, our method can even more precisely complete them.

Finally, we use our inferred views to compare to several existing methods for cross-view action recognition. We follow the standard leave-one-action-out IXMAS protocol [52]. We train an action class using the HOG features from all frames, and predict the action label of a test clip by voting. Table 4.3 shows the results. They are quite encouraging. Despite using a rather simple frame-based HOG classifier, our inferred views lead to recognition accuracy better than four existing methods that devise sophisticated features or learning algorithms specifically for this recognition task. This shows that explicitly estimating missing views can offer

	View 0	View 1	View 2	View 3	View 4
Farhadi 08 [52]	61	67	61	63	40
Junejo 08 [81]	63.0	64.3	64.5	58.9	46.6
Farhadi 09 [53]	74	77	76	73	72
Liu 11 [106]	79.0	74.7	75.2	76.4	71.2
Li 12 [102]	83.4	79.9	82.0	85.3	75.5
Zhang 13 [185]	**88.3**	**83.0**	**87.7**	**88.3**	**81.9**
COPY	59.9	56.5	53.4	59.8	41.2
MEMORY	67.7	63.0	58.6	65.0	48.9
Ours	79.9	80.8	79.0	80.2	74.2

Table 4.3: Cross-view action recognition accuracy on IXMAS.

advantages over using view-invariant descriptors. That said, we do underperform two of the methods. We suspect our static frame HOG representation is a handicap, as the other methods use temporal features. It will be interesting future work to generalize our idea to the temporal domain.

On top of its good performance on this specific task, our method offers functionality the prior work does not: 1) it can translate seen images to images in new viewpoints, whereas the prior methods produce invariant features, which cannot be used in support of other prediction tasks, and 2) it can leverage any available views during learning, whereas the prior methods focus on learning connections only between pairs of views.

4.2.3 Estimating Body Orientation

Next we test our unsynchronized method (Sec. 4.1.2) on H3D. We quantize the torso orientations into $M = 8$ discrete views. We use views inferred by our method to augment a training set of real images, then learn viewpoint classifiers.

(a) Average mAP, compared to view synthesis baselines

Orig	Orig+COPY	Orig+MEMORY	Orig+Ours
17.29	14.77	19.94	**20.30**

(b) Classification accuracy vs. state-of-art

Poselet activations+SVM [110]	Ours
48.4%	**49.9%**

Table 4.4: Viewpoint estimation accuracy on H3D when we augment real training images with inferred views, compared to alternative view synthesis methods (a) and a state-of-the-art technique (b).

We form a 75%-25% train-test split, and balance the training images per view, since highly imbalanced training images would favor our approach. We train SVMs with χ^2 kernels for all methods. Given a novel test image, we need to decide which way the person is facing. Table 4.4(a) shows the mAP results. Adding the view-specific training instances created by our method, accuracy is better than training with the real images alone. Furthermore, our factorization approach is again stronger than both baselines.

Next, we compare our viewpoint estimation to an existing method based on poselets [110]. We use the same features, classifier, and experimental setup described in that paper. We train one classifier with the real H3D images, and another with those same images plus our inferred views. Table 4.4(b) shows the classification accuracy results.[4] We see our virtual views boost the accuracy of this state-of-the-art approach for viewpoint estimation.

Both these H3D results are encouraging. Not only can we infer how a person

[4]Note that the numbers in (a) and (b) are not comparable to each other due to differences in features and experimental setup.

will appear in other viewpoints having seen him in only a single view, but doing so improves robustness for appearance-based viewpoint estimation.

4.3 Conclusions

In this chapter, I proposed a novel approach for inferring human appearance in unseen viewpoints. Whereas existing methods tackle the problem using geometry and image warping, my method offers a new perspective based on learning. I show how to cast the problem in terms of tensor completion, and adapt a factorization approach to accommodate both synchronized and unsynchronized single-view images. Our results on two challenging datasets show that not only can we infer unseen views, but that doing so is useful for practical human analysis tasks.

So far I have demonstrated two approaches that expand existing human action recognition frameworks to deal with learning from few available instances and learning from few available views. Next, I study the human actions that involve *interaction* with other objects or another person. Existing work understands human-object interaction by treating the person and the object as context for each other. This framework increases the complexity of the model and would require a large amount of examples for learning. To solve this problem, in next chapter, I aim to develop an approach that connects the person's pose and the object's property in a category independent way. Thus the proposed approach can deal with broader or more general cases in modelling interactions.

Chapter 5

Predicting the Location of "Interactees"

[1]In the previous chapters, I connect observed human data to other available data with underlying patterns such as temporal dependencies, viewpoint correlations, and partially overlapping poses. In this chapter, I shift to another type of connection: the interaction between a human and another object or another person (in this work, we call it an *interactee* as described in Chapter 1).

Here my goal is to discover the patterns that link our pose and certain properties of the interactee such as its size and location. Existing work models the interaction between a person and an interactee with dependence on the person's pose and interactee's category. In this chapter, I propose to model the interaction between a person and that person's interactee in a *category independent* way. For any kind of interaction, our system can predict the location and the size of the interactee by observing the cues from the person's pose, orientation, and scene layout.

In particular, I consider the following question: *Given a person in a novel image, can we predict the location of that person's "interactee"—the object or person with which he interacts—even without knowing the particular action being*

[1]The work in this chapter was supervised by Dr. Grauman and originally published in: Predicting the location of "interactees" in novel human-object interactions. C.-Y. Chen and K. Grauman. In Proceedings of Asian Conference on Computer Vision (ACCV), Singapore, November, 2014.

performed or the category of the interactee itself? Critically, by posing the question in this manner, our solution cannot simply exploit learned action-specific poses and objects. Instead, I aim to handle the open-world setting and learn generic patterns about human-object interactions. In addition, I widen the traditional definition of an interactee to include not only directly manipulated objects, but also untouched objects that are nonetheless central to the interaction (e.g., the poster on the wall the person is reading).

Why should the goal be possible? Are there properties of interactions that transcend the specific interactee's category? Figure 5.1 suggests that, at least for humans, it is plausible. In these examples, without observing the interactee object or knowing its type, one can still infer the interactee's approximate position and size. For example, in image A, we may guess the person is interacting with a small object in the bottom left. We can do so because we have a model of certain pose, gaze, and scene layout patterns that exist when people interact with a person/object in a similar relative position and size. This is done without knowing the category of the object, and even without (necessarily) being able to name the particular action being performed.

After building a system for predicting the location of an interactee, I explore how the inferred interactee localization can be used as a cue to guide the system for focusing on important object/area(s) in the scene and provides four different applications as following. In the first task, I use interactee localization to improve the accuracy or speed of an existing object detection framework by guiding the detector to only focus on areas that are involved in the interaction. Next, I use the

Figure 5.1: Despite the fact we have hidden the remainder of the scene, can you infer where is the object with which each person is interacting? Our goal is to predict the position and size of such "interactee" objects in a *category-independent* manner, without assuming prior knowledge of the specific action/object types.

interactee prediction to assist image retargeting. In this task, the image is resized by removing the unimportant content and preserving the parts related to the person and interactee. Furthermore, I use inferred interactee location as importance prediction for person-centric view of what to mention in an image. For example, given a novel image containing one or more people, can we predict which objects in the scene are essential to generating an informative description? Our key hypothesis is that a person's *interactions* give vital cues. As shown in Figure 5.2, each image contains a dozen or more recognizable objects, but a human viewer has bias towards noticing the object with which each person interacts: *the baby is eating cake* or *the boy is reaching for the frisbee*. Notably, not only do we focus on people and their activity—what they are doing, we also focus on the direct object of that activity—what they are doing it with/to. By applying my proposed approach, we can leverage these interactees to rank detected objects by their importance or perform retrieval-based image captioning.

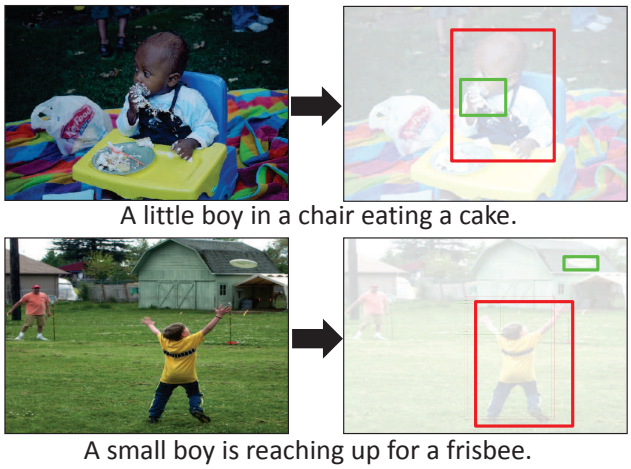


Figure 5.2: When describing an image, people usually mention the object with which the person is interacting, even if it may be small or appear non-salient to traditional metrics. For example, here the "interactee" objects are the cake and the frisbee.

5.1 Approach of Predicting the Location of Interactees

To predict the location of an interactee, I explore two different methods: a interaction embedding based non-parametric approach and a network based probabilistic model. In the following, I first precisely define what qualifies as an interactee and interaction and describe our data collection effort to obtain annotations for training and evaluation. Then, I explain the two proposed learning and prediction procedures. Finally, I overview the four applications that exploit my method's interactee predictions.

5.1.1 Definition of Human-Interactee Interactions

First we must define precisely what a human-interactee[2] interaction is. This is important both to scope the problem and to ensure maximal consistency in the human-provided annotations we collect.

Our definition considers two main issues: (1) the interactions are not tied to any particular set of activity categories, and (2) an interaction may or may not involve physical contact. The former simply means that an image containing a human-object interaction of any sort qualifies as a true positive; it need not depict one of a predefined list of actions (in contrast to prior work [179, 47, 65, 36, 177, 76, 132]). By the latter, we intend to capture interactions that go beyond basic object manipulation activities, while also being precise about what kind of contact does qualify as an interaction. For example, if we were to define interactions strictly by cases where physical contact occurs between a person and object, then walking aimlessly in the street would be an interaction (interactee=road), while reading a whiteboard would not. Thus, for some object/person to be an interactee, the person ("interactor") must be paying attention to it/him and perform the interaction with a purpose.

Specifically, we say that an image displays a human-interactee interaction if either of the following holds:

1. The person is watching a specific object or person and paying specific atten-

[2] An interactee refers to the thing a particular person in the image is interacting with; an interactee could be an object, a composition of objects, or another person.

tion to it. This includes cases where the gaze is purposeful and focused on some object/person within 5 meters. It excludes cases where the person is aimlessly looking around.

2. The person is physically touching another object/person with a specific purpose. This includes contact for an intended activity (such as holding a camera to take a picture), but excludes incidental contact with the scene objects (such as standing on the floor, or carrying a camera bag on the shoulder).

An image can contain multiple human-interactee relationships. We assume each person in an image has up to one interactee. At test time, our method predicts the likely interactee location for each individual detected person in turn.

5.1.2 Interactee Dataset Collection

Our method requires images of a variety of poses and interactee types for training. We found existing datasets that contain human-object interactions, like the Stanford-40 and PASCAL Actions [179, 47], were somewhat limited to suit the category-independent goals of our approach. Namely, these datasets focus on a small number of specific action categories, and within each action class the human and interactee often have a regular spatial relationship. Some classes entail no interaction (e.g., *running*, *walking*, *jumping*) while others have a low variance in layout and pose (e.g., *riding horse* consists of people in fairly uniform poses with the horse always just below). While our approach would learn and benefit from such consistencies, doing so would essentially be overfitting, i.e., it would fall short of demonstrating action-independent interactee prediction.

Therefore, we curated our own dataset and gathered the necessary annotations. We use selected images from three existing datasets, SUN [169], PASCAL 2012 [47], and Microsoft COCO dataset [104]. SUN is a large-scale image dataset containing a wide variety of indoor and outdoor scenes. Using all available person annotations[3], we selected those images containing more than one person. The SUN images do not have action labels; we estimate these selected images contain 50-100 unique activities (e.g., *talking*, *holding*, *cutting*, *digging*, and *staring*). PASCAL is an action recognition image dataset. We took all images from those actions that exhibit the most variety in human pose and interactee localization—*using computer* and *reading*. We pool these images together; our method does not use any action labels. This yields a large number (¿100) of unique activities including skiing, skateboarding, throwing, batting, holding, etc. For COCO, we consider the subset of COCO training images that contain at least one person with area $>$5,000 pixels and more than 4 out of 5 annotators report there is an interaction.

We use Amazon Mechanical Turk (MTurk) to get bounding box annotations for the people and interactees in each image. Figure 5.3 shows the instructions collecting the interactee localization in the form of bounding boxes. We again define what interaction means in our task, and we show examples of good localizations in the instruction. Figure 5.4 shows an example annotation task.

We get each image annotated by 7 unique workers (due to the large amount of image in COCO, we have 5 unique workers for this dataset), and keep only

[3] `http://groups.csail.mit.edu/vision/SUN/`

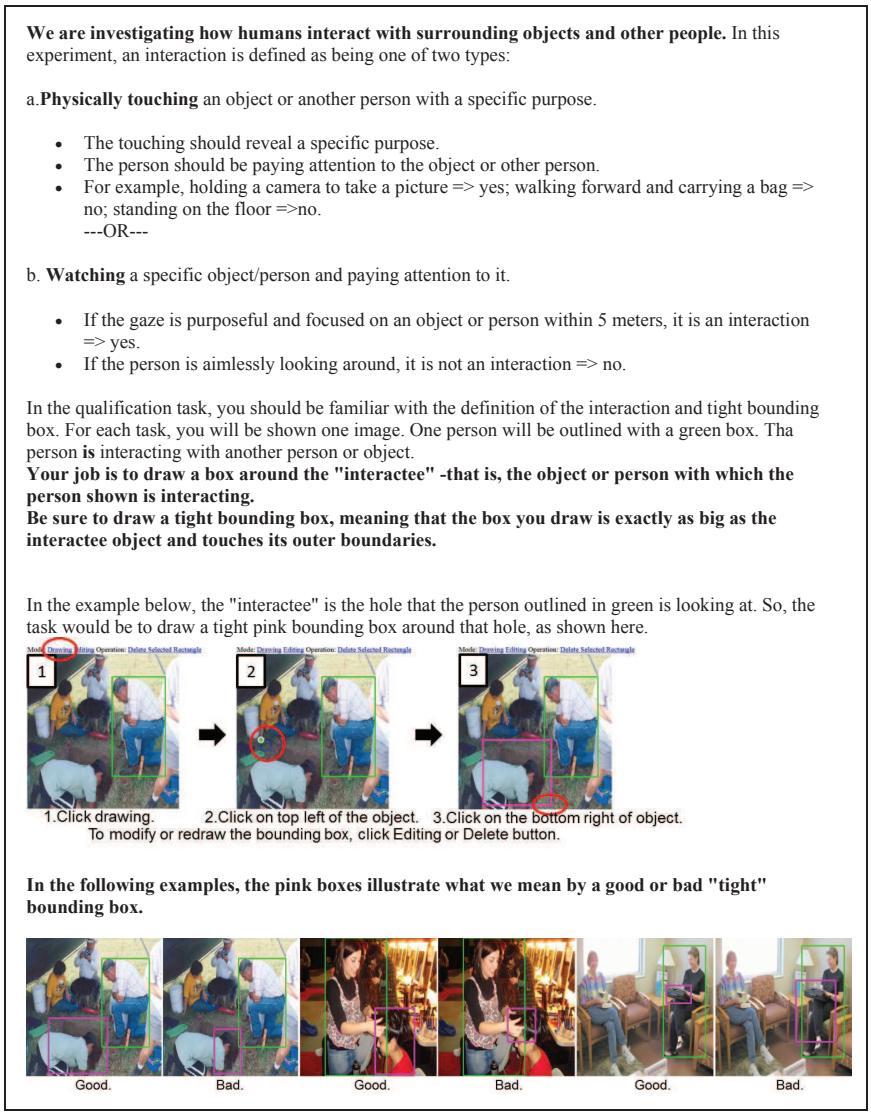


Figure 5.3: Instruction for localizing interactee in images.

those images for which at least 4 workers said it contained an interaction. This left 355/754/10,147 images from SUN/PASCAL/COCO respectively.

The precise location and scale of the various annotators' interactee bounding

Figure 5.4: Example task for localizing interactee in images.

boxes will vary. Thus, we obtain a single ground truth interactee bounding box via an automatic consensus procedure. First, we apply mean shift to the coordinates of all annotators' bounding boxes. Then, we take the largest cluster, and select the box within it that has the largest mean overlap with the rest.

The interactee annotation task is not as routine as others, such as tagging images by the objects they contain. Here the annotators must give careful thought to which objects may qualify as an interactee, referring to the guidelines we provide them. In some cases, there is inherent ambiguity, which may lead to some degree of subjectivity in an individual annotator's labeling. Furthermore, there is some variability in the precision of the bounding boxes that MTurkers draw (their

notion of "tight" can vary). This is why we enlist 7 unique workers on each training example, then apply the consensus algorithm to decide ground truth. Overall, we observe quite good consistency among annotators. The average standard deviation for the center position of bounding boxes in the consensus cluster is 8 pixels. See Figure 5.9, columns 1 and 3, for examples.

5.1.3 Localizing Interactees in Novel Images

I explore two different methods for interactee localization: (1) a interaction embedding based non-parametric regression approach and (2) a network based probabilistic model. I will go over the details of both approaches in the following.

In both methods, to capture the properties of the interactee in a category-independent manner, we represent its layout with respect to the interacting person. In particular, an interactee's localization parameters consist of $\boldsymbol{y} = [x, y, a]$, where (x, y) denotes the displacement from the person's center to the interactee box's center, and a is the interactee's area. Both the displacement and area are normalized by the scale of the person, so that near and far instances of a similar interaction are encoded similarly. Given a novel image with a detected person, we aim to predict $\boldsymbol{y}$, that person's interactee, as I explain next.

5.1.3.1 Non-parametric Regression with Interaction-guided Embedding

My first method for this task predicts the interactee in a novel image using a learned *interaction-guided embedding* together with non-parametric regression. Our goal is to estimate the approximate position and area of the interactee based on

any relevant visual cues in the image.

To learn the relationship between the interactee's location $\boldsymbol{y}$ and the image content, we extract three types of features.

First, we learn *interaction-guided deep person features*. Inspired by the idea that lower layer neurons in a CNN tend to capture the *general* representation and the higher layer neurons tend to capture the representation *specific* to the target task [182], we fine-tune a deep CNN for interactee localization. In particular, as shown in Figure 5.6, we quantize the space of interactee localization parameters, then fine-tune a pre-trained object recognition network [91] to produce the proper discretized parameters when given a detected person (bounding box). The last layer provides the learned feature map, $\boldsymbol{x}_{cnn-p}$. This embedding discovers features informative for an interaction, which may include body pose cues indicating where an interactee is situated (e.g., whether the arms are outstretched, the legs close together, the torso upright or leaning, etc.), as well as attentional cues from the person's head orientation, eye gaze, or body position. As shown in Figure 5.5, given a query pose, our $\boldsymbol{x}_{cnn-p}$ feature is able to precisely retrieve training examples that involve in similar interactions as the query example (e.g., riding and holding bat) while HOG feature could be misleading in some cases.

In a similar manner, we also learn *interaction-guided deep scene features*. As shown in Figure 5.6 we fine-tune a scene recognition network [186] to discover features about the entire scene that are useful for predicting the interaction, yielding an interaction-guided scene descriptor $\boldsymbol{x}_{cnn-s}$. Intuitively, this embedding learns cues surrounding the person that are relevant to his interactee's placement, such

Figure 5.5: Example of nearest pose neighbors by our $\boldsymbol{x}_{cnn-p}$ feature versus HOG feature.

as context for the activities that might be taking place. It is also free to capture the appearance of the interactee itself (though due to the cross-category nature of interactions discussed above, this may or not be learned as useful.)

Finally, we augment the learned features with several standard descriptors possibly indicative of interactees. For pose, we take the Histogram of Oriented Gradients (HOG) $\boldsymbol{x}_h$ computed in the person bounding box, plus the box's aspect ratio ($\boldsymbol{x}_a = \frac{h}{w}$) (e.g., the aspect ratio will be large for a standing person, smaller for a sitting person). For additional features about the scene, we take a GIST descriptor, $\boldsymbol{x}_g$, and the person's normalized position within the image, $\boldsymbol{x}_p$. The latter captures how the person is situated within the scene, and thus where there is "room" for an interactee. For attention, we use poselets [110] to estimate the head and torso orientation, θ_h and θ_t, to capture the direction of attention, whether physical or non-physical. The head orientation offers coarse eye gaze cues, while the torso tells us which objects the person's body is facing.

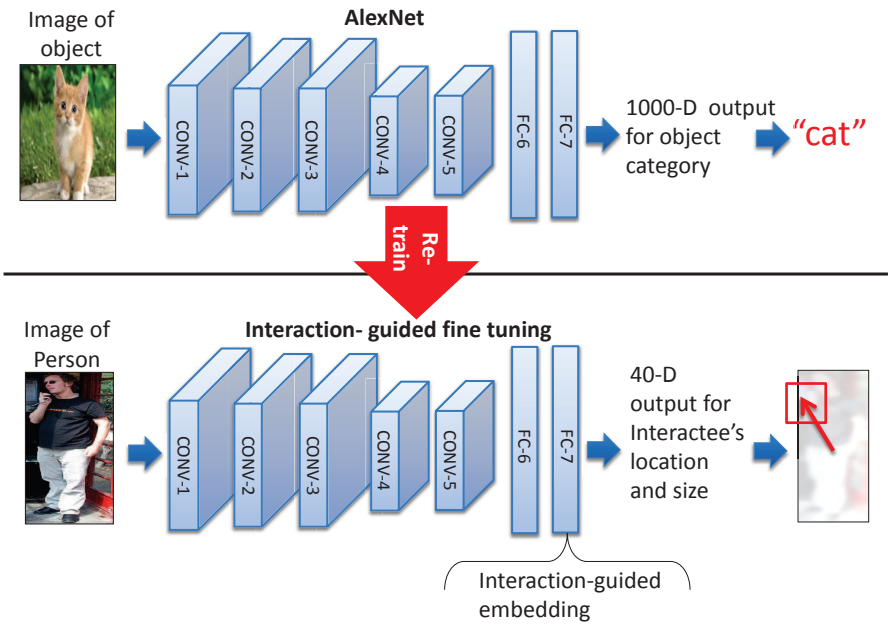


Figure 5.6: Interaction-guided fine-tuning and network architecture of our interaction-guided embedding.

Combining these features, we have the feature vector

$$\boldsymbol{x} = [\theta_h, \theta_t, \boldsymbol{x}_h, \boldsymbol{x}_a, \boldsymbol{x}_g, \boldsymbol{x}_p, \boldsymbol{x}_{cnn-p}, \boldsymbol{x}_{cnn-s}]. \tag{5.1}$$

We compute and store this descriptor for each interactee-annotated training image, yielding a set of N training pairs $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$. To infer the interactee parameters $\hat{\boldsymbol{y}}_{\boldsymbol{q}} = [\hat{x}_q, \hat{y}_q, \hat{a}_q]$ for a novel query image $\boldsymbol{x}_q$, we use non-parametric locally weighted regression. The idea is to retrieve training images most similar to $\boldsymbol{x}_q$, then combine their localization parameters. Rather than simply average them, we attribute a weight to each neighbor that is a function of its similarity to the query. In particular, we retrieve the K nearest neighbors $\boldsymbol{x}_{n_1}, \ldots, \boldsymbol{x}_{n_K}$ from the training set based on their Euclidean distance to $\boldsymbol{x}_q$. We normalize distances

per feature by the standard deviation of the L_2 norms between training features of that type. Then, the estimated interactee parameters are $\hat{\boldsymbol{y}}_{\boldsymbol{q}} = \sum_{i=1}^{K} w_i \boldsymbol{y}_{n_i}$, where $w_i = \exp(-\|\boldsymbol{x}_q - \boldsymbol{x}_{n_i}\|)$.

Note that while interactees are a function of the action being performed, there is not a one-to-one correspondence. That is, the same action can lead to different interactees (e.g., climb a *tree* vs. climb a *ladder*), and vice versa (e.g., *climb* a tree vs. *trim* a tree). This supports our use of a category-independent spatial representation of the interactee. Our method can benefit from any such sharing across verbs; we may retrieve neighbor images that contain people doing activities describable with distinct verbs, yet that are still relevant for interactee estimation. For example, a person cutting paper or writing on paper may exhibit both similar poses and interactee locations, regardless of the distinct action meanings. Thus, there is value here in not collapsing the dataset to verb-specific models.

A natural question is whether one could simply learn the localization parameters "end-to-end" from the image rather than using the person/scene embeddings as an intermediary to a non-parametric learning approach. In practice, we found such an approach inferior to ours. This indicates there is value in 1) separating the person and scene during feature learning (more data would likely be needed if one wanted to treat the person as a latent variable) and 2) augmenting the learned features with semantically rich features like gaze.

5.1.3.2 Probabilistic Model with Mixture Density Network

We expect the non-parametric method described above to fare best when there is ample labeled data for learning. Since this is not always the case, we also consider a parametric model to represent interactee localization. As an alternative to the above proposed non-parametric method, I also explore another way to localize interactees using the Mixture Density Network (MDN) [13] to build a predictive distribution for the interactee localization parameters. An MDN is a neural network that takes as input the observed features, their associated parameters, and as output produces a network able to predict the appropriate Gaussian mixture model (GMM) parameters for a novel set of observed features.

To build a predictive distribution for the interactee localization parameters, we want to represent a conditional probability density $P(\boldsymbol{y}|\boldsymbol{x})$. Here we model density as a mixture of Gaussians with m modes:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{5.2}$$

where α_i denotes the prior mixing proportion for component i, $\boldsymbol{\mu}$ is its mean, and $\boldsymbol{\Sigma}_i$ is its covariance matrix. We use the N labeled training examples $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, (\boldsymbol{x}^N, \boldsymbol{y}^N)\}$ to train the MDN.

In testing, given a novel test image, we extract the descriptors from the person bounding box in the image. Then, we use the learned MDN to generate the GMM $P(\boldsymbol{y}^t|\boldsymbol{x}^t)$ representing the most likely positions and scales for the target interactee.

In this way, we can assign a probability to any candidate position and scale in the novel image. To estimate the single most likely parameters, we use the center of the mixture component with the highest prior (α_i), following [13]. The output interactee box is positioned by adding the predicted $(\hat{x}, \hat{y})$ vector to the person's center, and it has side lengths of $\sqrt{\hat{a}}$.

5.1.4 Applications of Interactee Prediction

Our method is essentially an object saliency metric that exploits cues from observed human-interactions. Therefore, it has fairly general applicability. To make its impact concrete, aside from analyzing how accurate its predictions are against human-provided ground truth, we also study four specific applications that can benefit from such a metric.

In the first task, I use the interactee localization to improve the accuracy or speed of existing object detection framework by guiding the detector to only focus on areas that involved in the interaction. In the second task, I use the interactee prediction to assist image retargeting. In this task, the image is resized by removing the unimportant content and preserving the parts related to the person and interactee. In the third and fourth tasks, I explore how to leverage inferred interactees to detect important objects and generate image descriptions. These tasks aim to mimic human-generated image descriptions by focusing on the prominent object(s) involved in an interaction. Well-focused descriptions can benefit image retrieval applications, where it is useful to judge similarity not purely on how many objects two images share, but rather on how many *important* objects they share.

5.1.4.1 Task 1: Interactee-aware Contextual Priming for Object Detection

First, we consider how interactee localization can prime an object detector. The idea is to use our method to predict the most likely place(s) for an interactee, then focus an off-the-shelf object detector to prioritize its search around that area. This has potential to improve both object detection accuracy and speed, since one can avoid sliding windows and ignore places that are unlikely to have objects involved in the interaction. It is a twist on the well-known GIST contextual priming [155], where the scene appearance helps focus attention on likely object positions; here, instead, the cues we read from the person in the scene help focus attention. Importantly, in this task, our method will look at the person, but will *not* be told which action is being performed; this distinguishes the task from the methods discussed in related work, which use mutual object-pose context to improve object detection for a particular action category.

To implement this idea, we run the Deformable Part Model (DPM) [57] object detector on the entire image, then we apply our method to discard the detections that are outside the 150% enlarged predicted interactee box (i.e., scoring them as $-\infty$). To alternatively save run-time, one could apply DPM to only those windows near the interactee.

5.1.4.2 Task 2: Interactee-aware Image Retargeting

As a second application, we explore how interactee prediction may assist in image retargeting. The goal is to adjust the aspect ratio or size of an image without distorting its perceived content. This is a valuable application, for example, to allow

dynamic resizing for web page images, or to translate a high-resolution image to a small form factor device like a cell phone. Typically retargeting methods try to avoid destroying key gradients in the image, or aim to preserve the people or other foreground objects. Our idea is to protect not only the people in the image from distortion, but also their predicted interactees. The rationale is that both the person and the focus of their interaction are important to preserve the story conveyed by the image.

To this end, we consider a simple adaption of the Seam Carving algorithm [5]. Using a dynamic programming approach, this method eliminates the optimal irregularly shaped "seams" from the image that have the least "energy". The energy is defined in terms of the strength of the gradient, with possible add-ons like the presence of people (see [5] for details). To also preserve interactees, we augment the objective to increase the energy of those pixels lying within our method's predicted interactee box. Specifically, we scale the gradient energy g within both person and interactee boxes by $(g + 5) * 5$.

5.1.4.3 Task 3: Interactees as Important Objects

The third application uses interactees to gauge object *importance* within a scene. Following prior work [150, 151], we define "important" objects as those mentioned by a human describing an image. Our intuition that people tend to mention interactees is supported by data; in COCO, 80% of true interactees appear in the human descriptions.

We use predicted interactees to generate important object hypotheses, as

follows. Given a detected person, we project the predicted interactee bounding box (square box with the predicted area) into the query image. This is essentially a saliency map of where, given the scene context and body pose, we expect to see an object key to the person's interaction. Then, we sort all recognized objects in the scene by their normalized overlap with the interactee regions. The first object in this list is returned as an important object.

Whereas past work [150, 71, 151] focuses on composition cues (like size or position) and semantic cues (like the type of object or attribute), the novelty of our approach is to inject human-object interaction cues into the predictions.

5.1.4.4 Task 4: Interactees in Sentence Generation

In the fourth task, we generate sentences for the query image that account for its interactee. While the importance task above focuses solely on the question of whether an object is important enough to mention, the sentence task entails also describing the activity and scene.

We take a retrieval-based approach, inspired by [122, 37]. Again we use a non-parametric model. Intuitively, if the content of a query image closely resembles a database image, then people will describe them with similar sentences.

Given a novel query, we compute $\boldsymbol{x}$ and its estimated interactee spatial parameters $\hat{\boldsymbol{y}}$, and use them together to retrieve the K_s nearest images in a database annotated with human-generated sentences. In particular, we use Euclidean distance to sort the neighbors, normalizing distances for $\boldsymbol{x}$ and $\hat{\boldsymbol{y}}$. Then, we simply adopt the sentence(s) for the query that are associated with those nearest examples.

Figure 5.7: Examples of SUN, PASCAL-Action, and MS-COCO datasets.

5.2 Experimental Results

We evaluate three things: (1) how accurately do we predict interactees, compared to several baselines? (Sec. 5.2.2), (2) how well can humans perform this task? (Sec. 5.2.3), (3) the four applications of interactee localization (Sec. 5.2.4).

5.2.1 Datasets and Implementation Details

We experiment with images containing people from three datasets: PASCAL Actions 2012 [48], SUN [169], and COCO [104]. All three consist of natural, real-world snapshots with a wide variety of human activity. See Figure 5.7 for example images of these three datasets.

For PASCAL and SUN, we use the subsets collated for human interactions, containing 754 and 355 images, respectively, and the publicly available interactee annotations. As PASCAL Actions and SUN do not have sentence data, we use them solely to evaluate interactee localization accuracy. For COCO, we use the 10,147 total images for which we obtained interactee bounding box annotations on MTurk

(see Sec. 5.1.2). COCO contains 5 human-written sentences per image, as well as object boundaries for 80 common object categories, which we exploit below. See Figure 5.7 for example images of these three datasets.

For the feature embeddings, we fine-train AlexNet [91] and Places-CNN [186] with the Caffe deep learning toolbox [79], using SGD solver with 10,000 iterations and a learning rate of 0.001. To form the target labels, we quantize the interactee's displacement and area into 10 and 4 bins, respectively, so the network provides 40 outputs in the last layer. We extract the features from the 7th layer (fc7) as $\boldsymbol{x}_{cnn-p}$ and $\boldsymbol{x}_{cnn-s}$ from each network. For HOG, each box is resized to 80×80 and we use cell size 8.

We localize interactee regions of interest automatically with our two proposed methods. The inferred interactee localization guides us where to focus in the image for our four applications. In particular, for results in the importance and sentence tasks, we refer to the ground truth person boxes and object outlines when deciding what word to use for a predicted region of interest. This lets us focus evaluation on the "what to mention" task, independent of the quality of the visual detectors. We set $K = 20$ and $K_s = 5$ when retrieving the near neighbor interactions and images, respectively. We fixed K after initial validation showed values between 5-50 to perform similarly. For PASCAL and SUN we use a random 75%-25% train-test split and for COCO we use a random 80%-20% train-test split.

5.2.2 Accuracy of Interactee Localization

First we evaluate the accuracy of our interactee predictions. Given an image, our system predicts the bounding box where it expects to find the object that is interacting with the person. We quantify error in the size and position of the box. In particular, we report the difference in position/area between the predicted and ground truth boxes, normalized by the person's size. We also evaluate the accuracy of our method and baselines by the interaction over union (IOU) between the inferred and ground truth interactee bounding boxes.

We compare to three baselines: (1) the Objectness (Obj) [4] method, which is a category-independent salient object detector; (2) a "Near Person" baseline, which simply assumes the interactee is close to the person[4]; and (3) a Random baseline that randomly generates a position and size. While our methods exploit cues about the person, the Objectness method is completely generic and does not. We score each method's most confident estimate.

Table 5.1 shows the result. On three datasets, both of our methods offer significant improvement in position and size error over the baselines. The margins are largest on the most diverse COCO dataset, where our data-driven approach (Ours-embedding) benefits from the large training set (COCO has more than 10 times the labeled instances than PASCAL or SUN). Our interaction embedding method provides 12% lower errors over our MDN method on average. This indicates the strength of our learned features and data-driven estimation approach. Our error re-

[4]It predicts a box centered on the person, with a scale ~ 0.74 of its area (a parameter set by validation on the training data).

Metric	Dataset	Ours-embedding	Ours-MDN	Obj [4]	Near Person	Random
Position error	COCO	**0.2256**	0.3058	0.3569	0.2909	0.5760
	PASCAL	**0.1632**	0.1926	0.2982	0.2034	0.5038
	SUN	0.2524	**0.2331**	0.4072	0.2456	0.6113
Size error	COCO	**38.17**	47.16	263.57	65.12	140.13
	PASCAL	**27.04**	34.39	206.59	31.97	100.31
	SUN	**33.15**	33.19	257.25	39.51	126.64
IOU	COCO	**0.1989**	0.1153	0.0824	0.1564	0.0532
	PASCAL	**0.2177**	0.1369	0.0967	0.1415	0.0552
	SUN	**0.1710**	0.1145	0.0661	0.1504	0.0523

Table 5.1: Average interactee prediction error as measured by position/size and average IOU between prediction and ground truth interactee on all three datasets.

ductions relative to Near Person average 16% overall, and up to 37% on COCO for object size. However, on the SUN dataset, our MDN method is slightly better than our embedding method for interactee position; with only 355 images in SUN, our data-driven approach may suffer. Our gain over Near Person confirms that this is a non-trivial prediction task, particularly when the person is not touching the interactee (see the bottom example in third column in Figure 5.8). As for the IOU metric, our embedding method provides significantly higher accuracy than other methods especially in COCO and PASCAL dataets with the help of larger data size. Our MDN method provides lower average IOU than the Near Person baseline due the low score cases from incorrect interactee localizations.

Figure 5.8 shows example predictions by the embedding variant of our method. We see that our method can often zero in on regions where the interaction is likely to be focused, even when the object may not have been seen in the training examples. On the other hand, we also find failure cases, e.g., when a person's pose is too rare (upside down in the middle of fourth column) or the unusual

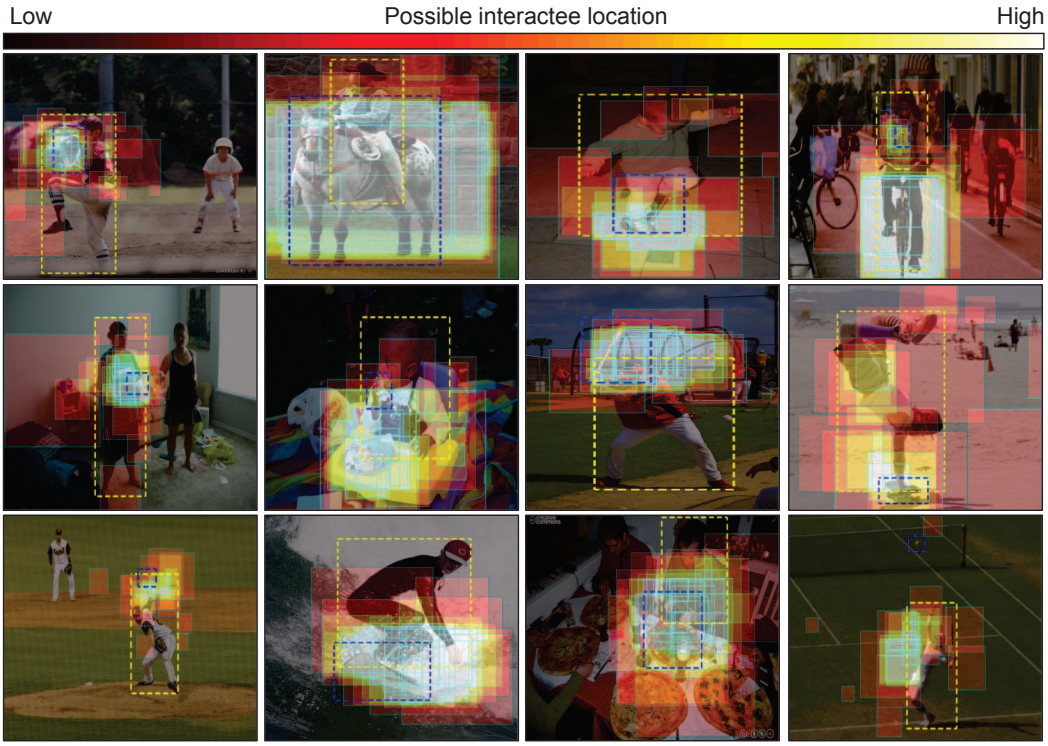


Figure 5.8: Example interactee localizations. We display a heatmap for our embedding method's predictions by overlaying the retrieved training examples, such that they vote on likely areas of interest (white = high confidence). The yellow dotted boxes indicate the main person in the image. The blue box indicates the ground truth interactee location. Our method can infer interactees in spite of varying interactions and object types. The fourth column shows failure cases where there is less confidence in the prediction (see the upside down skater) or errors in unusual cases with multiple interactees (see the guy using the cell phone while riding a bike). Best viewed in color.

cases with multiple interactees (using cell phone while riding bike in the top of fourth column).

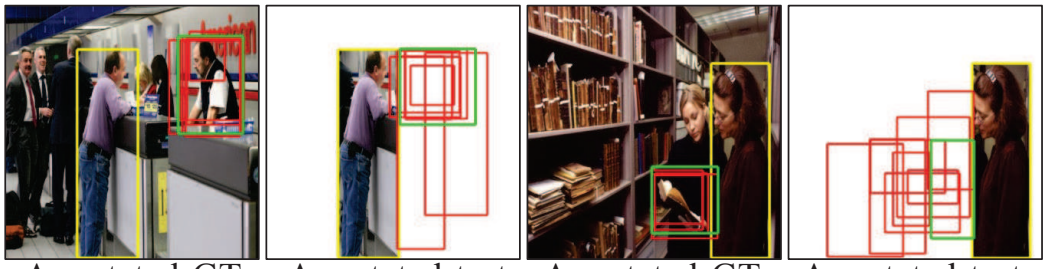


Figure 5.9: We remove the background from the original image and ask human subjects to infer where the interactee might be. Red boxes denote their predictions, green box denotes consensus. Annotated-GT shows the full image (which is the format seen for ground truth collection, cf. Sec. 5.1.2). Annotated-test shows the human subject results. Naturally, annotators can more reliably localize the interactee when it is visible.

5.2.3 Human Subject Experiment

Next we establish an "upper bound" on accuracy by asking human subjects on MTurk to solve the same task. Our MDN method localizes an interactee without observing the background content (outside of the person box) and without knowing what category the interactee belongs to. Thus, we construct an interface forcing humans to predict an interactee's location with a similar lack of information. Figure 5.9, columns 2 and 4, illustrate what the human subjects see, as well as the responses we received from 10 people.

Table 5.2 shows the human subjects' results alongside ours, for the subset of images in either dataset where the interactee is not visible within the person bounding box (since those cases are trivial for the humans and require no inference). The humans' guess is the consensus box found by aggregating all 10 responses with mean shift as before. The humans have a harder time on SUN than PASCAL, due

	Human subject		Ours-MDN	
	Position error	Size error	Position error	Size error
SUN w/o visible	0.1573	28.92	0.2736	36.58
PASCAL w/o visible	0.0952	40.84	0.2961	43.27

Table 5.2: Results of the human subject test.

to its higher diversity of interaction types. This study elucidates the difficulty of the task. It also establishes an (approximate) upper bound for what may be achievable for this new prediction problem.

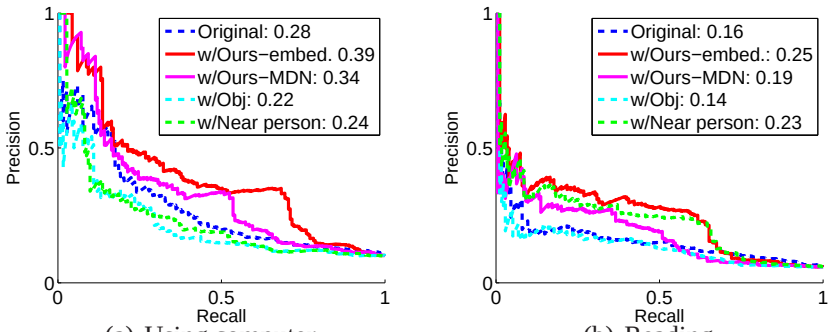
5.2.4 Results for Applications of Interactee Prediction

Finally, we evaluate our idea in the context of the four tasks defined above.

5.2.4.1 Task 1: Interactee-aware object detector contextual priming

We first demonstrate the utility of our approach for contextual priming for an object detector, as discussed in Sec. 5.1.4.1, Task 1. We use the PASCAL training images to train DPMs to find computers and reading materials, then apply our methods and the baselines to do priming.

Figure 5.10 shows the results. We see our methods outperform the baselines, exploiting its inference about the person's attention to better localize the objects. Note that both of our methods don't use the action category labels during training. Again, our interaction embedding method outperforms our MDN method. We also see that Near person fares well for the *reading* instances, since the book or paper is nearly always centered by the person's lap.



(a) Using computer (b) Reading

Figure 5.10: Interactee context helps focus the object detector. Numbers denote mAP.

5.2.4.2 Task 2: Interactee-aware image retargeting

Next, we inject our interactee predictions into the Seam Carving retargeting algorithm, as discussed in Sec. 5.1.4.2, Task 2. Figure 5.11 shows example results. For reference, we also show results where we adapt the energy function using OBJECTNESS's top object region prediction. Both methods are instructed to preserve the provided person bounding box. We retarget the source 500×500 images to 300×300.

We see that our method preserves the content related to both the person and his interactee, while removing some unrelated background objects. In contrast, OBJECTNESS [4], unaware of which among the prominent-looking objects might qualify as an interactee, often discards the interactee and instead highlights content in the background less important to the image's main activity.

Figure 5.11: Interactee-aware image retargeting example results. Our method successfully preserves the content of both the interactee (e.g., BBQ kit, book, painting of horse, laptop) and person, while reducing the content of the background. OBJECTNESS cannot distinguish salient objects that are and are not involved in the activity, and so may remove the informative interactees in favor of background objects. The bottom right example is a failure case for our method, where our emphasis on the interactee laptop looks less pleasing than the baseline's focus on the people.

5.2.4.3 Task 3: Interactees as important objects

Next, we use the interactee region of interest to predict object importance (cf. Sec. 5.1.4.3, Task 3). Following [150, 151], we are given an image plus a list of objects and their categories/locations. Ground truth importance is judged by how often humans mention the object in a caption.

For this task we compare to the existing *Object Prediction* importance method of [151] (Sec. 4.1 in that paper). It trains a logistic regression classifier with fea-

tures based on object size, location, and category. To ensure fair comparison, we use the COCO data to train it to predict the object most often mentioned in the image. We again compare to the Near Person baseline, and two additional baselines: Prior, which looks at all objects present in the image and picks the one most frequently mentioned across all training images, and Majority, which predicts people will mention the object that happens most frequently in the test image.

All methods ignore the persons in the images, since all images have a person. For this result, we discard images with only a person and a single object since all methods can only output that same object, leaving 1,617 test images.

Table 5.3 shows the result of 10 train/test splits. We measure accuracy by the hit rate—the average percentage of ground truth sentences mentioning the object deemed most important, per image. If each of the 5 ground truth captions include the predicted object, the score is 100% for that image. First, we see that interactees are correlated with important objects; the ground truth interactee leads to a hit rate of 78.4. Furthermore, our embedding method predictions outperform the baselines. The nearest competing method is Near Person. Even though the region of interest it predicts is substantially less precise (as we saw above), it does reasonably well because the step of identifying the annotated COCO object nearest to that region is forgiving. Nonetheless, the ground truth upper bound reinforces that better precision does translate to better performance on solving this task.

The state-of-the-art importance method [151] is less accurate than our interactee-based method on this data. We think this is because in the COCO data, an object of the same category, size, and location is sometimes mentioned, sometimes not,

Method	Mention rate (%)
Ground truth interactee	78.4(0.6)
Ours-embedding	**70.5(0.4)**
Importance [151]	65.4(0.4)
Ours-MDN	65.2(0.5)
Near Person	67.5(0.5)
Prior	64.6(0.6)
Majority	51.7(0.6)

Table 5.3: Average hit rates (higher is better) for predicted important objects. Numbers in parens are standard errors.

making the compositional and semantic cues used by that method insufficient. In contrast, our method exploits interactions to learn if an object would be mentioned, independent of its position and category. This result does not mean the properties used in [151] are not valuable; rather, in the case of analyzing images of people involved in interaction, they appear insufficient if taken alone.

5.2.4.4 Task 4: Interactees in sentence generation

Finally, we study how interactee detection can benefit retrieval-based sentence generation (cf. Sec. 5.1.4.4, task 4). For each test image, we retrieve 5 images from the training set, then compute the average similarity between the ground truth query and training sentences. We use the standard BLEU score [124] for n-gram overlap precision.

We compare our interaction embedding based regression approach to two retrieval-based sentence generation methods in prior work [122, 37]. For [122], there are two variations: Global Matching, which retrieves neighbors based on GIST and Tiny Image descriptors, and Global + Content Matching, which reranks

that shortlist with the local image content as analyzed by visual detectors. We were unable to obtain code from the authors, so we implement them ourselves. The Global Matching is straightforward to implement. The Global + Content Matching version involves a series of detectors for objects, stuff, attributes, scene, and actions. We use the same poselet-based action feature [110], which captures cues most relevant to our person-centric approach and utilizes the same *ground truth* person bounding box used by our method.[5] The method of [37] is a retrieval method using CNN features fine-tuned for the caption generation task; we use the features kindly provided by the authors in order to evaluate it on this subset of COCO (all $\sim$10K images with people and interactions).

Table 5.4 shows the results. Our interaction embedding based non-parametric regression method consistently outperforms the baselines and [122], and competes well with [37] despite the fact we do no fine-tuning specific to caption generation for our approach.

Without using CNN feature, our embedding base method (Ours-embedding w/o cnn) outperforms baselines [122]. The result confirms that a person-centric view of "what to mention" is valuable. The local Content Matching does not improve accuracy over Global Matching, and even detracts from it slightly. We suspect this is due to weaknesses in poselets for this data, since the action variation is very high in COCO. The authors also observed only a slight gain with Content Matching

[5] We omit the object, stuff, and attribute detectors because we could not reproduce the implementation (hence the asterisk in the table). In principle, any benefit from additional local content could also benefit us.

	1-Gram BLEU	2-Gram BLEU	3-Gram BLEU	4-Gram BLEU	Combined BLEU
Random	55.19	19.26	4.18	1.26	8.65
Global Matching [122]	63.80	28.02	9.80	3.75	16.01
Global + Content Matching [122](Actions*)	63.19	27.12	9.13	3.41	15.20
Global Match+AlexNet fc7	68.21	33.38	13.32	5.44	20.16
Retrieval fine-tuned [37]	73.05	42.63	22.01	11.19	29.59
Ours-embedding w/o cnn feature	65.08	29.74	11.13	4.56	17.64
Ours-embedding w/cnn-p only	68.03	33.30	13.45	5.64	20.36
Ours-embedding w/cnn-s only	70.78	36.42	15.96	6.87	23.05
Ours-embedding w/all	73.85	40.33	18.88	8.68	26.43
Ours-embedding w/all (fine-tuned) + [37]	73.51	43.03	22.45	11.52	30.07

Table 5.4: Average BLEU scores between query and retrieved sentences (higher = more similar).

in their own results [122].

After incorporating CNN features, our method (Ours-embedding w/all) still provides higher accuracy than the baseline (Global Match+AlexNet fc7) which utilizes CNN feature extracted from the person bounding box. In addition to the main result of our method, we also show results of our methods by only considering the two interaction-guided embedding features for ablation study. As shown in the table, our learned embedding features is helpful for captioning task by guiding the system to focus on the interaction and combining all features provide the highest accuracy.

Finally, following [37], we also tested a variant of our method where we fine-tune our interaction-guided network with training captions. Using our features with those of [37], accuracy is further improved (see last row in Table 5.4). This result shows our person-centric feature provides complementary information to the caption tuned global CNN feature.

Figure 5.12 shows example sentences generated by our method, alongside those of the baselines. We see how modeling person-centric cues of importance

allows our method to find examples with similar interactions. In contrast, the baselines based on global image matching find images focused on total scene similarity. They often retrieve sentences describing similar overall scene contexts, but are unable to properly model the fine-grained interactions (e.g., in third column, riding vs. carrying with a surfboard). The fourth column shows a failure case by our method, where we mispredict the interactee (cyan box) and so retrieve people doing quite different interactions.

5.3 Conclusions

In this chapter, I considered a new problem: how to predict where an interactee object will appear, given cues from content of image. While plenty of work studies action-specific object interactions, predicting interactees in an action-independent manner is both challenging and practical for various applications. The proposed method shows promising results to tackle this challenge. I demonstrate its advantages over multiple informative baselines, including a state-of-the-art object saliency metric, and illustrate the utility of knowing where interactees are for contextual object detection, image retargeting, and how the inferred interactee localization can be used to improve an existing method for describing images by focusing on the interaction.

The proposed methods in the last three chapters have focused on understanding a human's action and pose. After learning a model, next I move on to consider how we can use the learned model to detect when and where it happened in a video sequence.

Figure 5.12: Example sentences generated by our method and the Global Matching method [122] and fine-tuned retrieval system [37]. Blue bbox: true interactee, cyan bbox: our prediction. In the first three examples, ours is better because it correctly predicts the location of the interactee, and then uses the interactee's position and scale relative to the person to retrieve image examples with similar types of interaction. In the last one, our method fails to predict the interactee correctly and thus retrieves poorly matched interactions. See text for details.

Chapter 6

Detecting Activity with Max-Subgraph Search

[1]In the previous chapters, I proposed three approaches to improve the learning framework for understanding people's actions and poses. In this chapter, I focus on how to utilize such a learned model efficiently. I propose an approach to improve the framework of detecting actions in a video sequence.

While the recognition portion of the activity understanding problem has received increasing attention in recent years, state-of-the-art methods largely assume that the space-time region of interest to be classified has already been identified. However, for most realistic settings, a system must not only name what it sees, but also partition out the temporal or spatio-temporal extent within which the activity occurs. The distinction is non-trivial; in order to properly recognize an action, the spatio-temporal extent usually must be known *simultaneously*.

My goal is to unify the classification and localization components into a single detection procedure. We propose an efficient approach that exploits top-down activity knowledge to quickly identify the portion of video that maximizes a classifier's score. In short, it works as follows. Given a novel video, we construct a

[1]The work in this chapter was supervised by Dr. Grauman and originally published in: Efficient Activity Detection with Max-Subgraph Search. C.-Y. Chen and K. Grauman. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, June 2012.

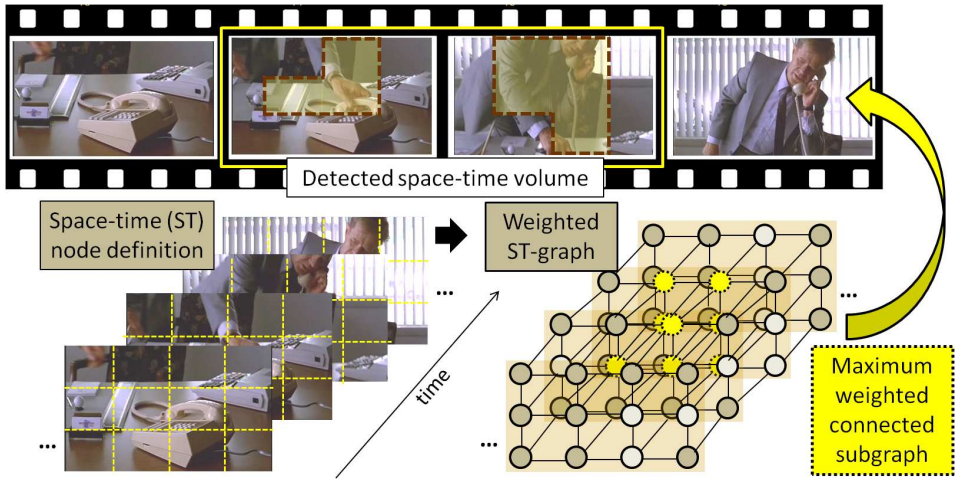


Figure 6.1: My approach constructs a space-time video graph, and efficiently finds the subgraph that maximizes an activity classifier's score. The detection result can take on non-cubic shapes (see dotted shapes in top frames), as demanded by the action.

3D graph in which nodes describe local video subregions, and their connectivity is determined by proximity in space and time. Each node is associated with a learned weight indicating the degree to which its appearance and motion support the action class of interest. Using this graph structure, we show the detection problem is equivalent to solving a *maximum-weight connected subgraph* problem, meaning to identify the subset of connected nodes whose total weight is maximal. For our setting, this in turn is reducible to a prize-collecting Steiner tree problem, for which practical branch-and-cut optimization strategies are available. This means we can efficiently identify both the spatial and temporal region(s) within the sequence that best fit a learned activity model. See Figure 6.1.

I validate the algorithm on four challenging datasets. The results demonstrate its clear speed and accuracy advantages over both standard sliding window

search as well as a state-of-the-art branch-and-bound solution [184].

6.1 Approach of Max-Subgraph Search

My approach first trains a detector using a binary classifier and training examples where the action's temporal extent is known. Then, given test sequences for which we have no knowledge of the start and end of the activity, it returns the subsequence (and optionally, the spatial regions of interest) that maximizes the classifier score. This works by creating a space-time graph over the entire test sequence, where each node is a space-time cube, and the cubes are linked according to their proximity in space and time. Each node is weighted by a positive or negative value indicating its features' contribution to the classifier's score. Thus, the subsequence for which the detector would yield the maximal score is equivalent to the maximum weight connected subgraph. This subgraph can be efficiently computed using an existing branch-and-cut algorithm, thereby finding the optimal solution without exhaustive search through all possible sets of connected nodes.

I first define the classifiers accommodated by our method (Sec. 6.1.1), and the features we use (Sec. 6.1.2). Then I describe how the graphs are constructed (Sec. 6.1.3); I introduce variants of the node structure and linking strategy that allow us to capture different granularities at detection time. Next, I briefly explain the maximum subgraph problem and branch-and-cut search (Sec. 6.1.4). Finally, I devise two extensions of our basic framework that can deal with spatio-temporal detection even in long videos (Sec. 6.1.5) and detection of multiple instances in a single sequence (Sec. 6.1.6).

6.1.1 Detector Training and Objective

We are given labeled training instances of the activity of interest, and train a binary classifier $f : S \rightarrow \mathbb{R}$ to distinguish positive instances from all other action categories. This classifier can score any subvolume S of a novel video according to how well it agrees with the learned activity. To perform activity detection, the goal is to determine the subvolume in a new sequence Q that maximizes the score

$$S^* = \underset{S \in Q}{\operatorname{argmax}} f(S). \tag{6.1}$$

If we were to restrict the subvolume in the spatial dimensions to encompass the entire frame, then S^* would correspond to the output of an exhaustive sliding window detector. More generally, the optimal subvolume S^* is the set of contiguous voxels of arbitrary shape in Q that returns the highest classifier score.

Our approach requires the classifier to satisfy two properties. First, it must be able to score an arbitrarily shaped set of voxels. Second, it must be defined such that features computed within local space-time regions of the video can be combined *additively* to obtain the classifier response for a larger region. The latter is necessary so that we can decompose the classifier response across the nodes of the space-time graph, and thereby associate a single weight with each node. Suitable additive classifiers include linear support vector machines (SVM), boosted classifiers, or Naive Bayes classifiers computed with localized space-time features, as well as certain non-linear SVMs [156].

Our results use a linear SVM with histograms (bags) of quantized space-time descriptors. The bag-of-features (BoF) representation has been explored in a

number of recent activity recognition methods (e.g., [97, 86, 119]), and, despite its simplicity, offers very competitive results. We consider BoF's computed over two forms of local descriptors. The first consists of low-level histograms of oriented gradients and flow computed at space-time interest points; the second consists of a novel high-level descriptor that encodes the relative layout of detected humans, objects, and poses. Both descriptors are detailed below in Sec. 6.1.2.

In either case, we compute a vocabulary of K visual words by quantizing a corpus of features from the training images. A video subvolume with N local features is initially described by the set $S = \{(\boldsymbol{x}_i, \boldsymbol{v}_i)\}_{i=1}^{N}$, where each $\boldsymbol{x}_i = (x_i, y_i, t_i)$ refers to the 3D feature position in space and time, and $\boldsymbol{v}_i$ is the associated local descriptor. Then the subvolume is converted to a K-dimensional BoF histogram $h(S)$ by mapping each $\boldsymbol{v}_i$ to its respective visual word c_i, and tallying the word counts over all N features.

We use the training instances to learn a linear SVM, which means the resulting scoring function has the form:

$$f(S) = \beta + \sum_i \alpha_i \langle h(S), h(S_i) \rangle, \tag{6.2}$$

where i indexes the training examples, and α, β denote the learned weights and bias. This can be rewritten as a sum over the contributions of each feature. Let $h^j(S)$ denote the j-th bin count for histogram $h(S)$. The j-th word is associated with a weight

$$w^j = \sum_i \alpha_i h^j(S_i), \tag{6.3}$$

for $j = 1, \ldots, K$. Thus the classifier response for a subvolume S is:

$$f(S) = \beta + \sum_{j=1}^{K} w^j h^j(S) \tag{6.4}$$

$$= \beta + \sum_{i=1}^{N} w^{c_i}, \tag{6.5}$$

where again c_i is the index of the visual word that feature $\boldsymbol{v}_i$ maps to, $c_i \in [1, K]$. By writing the score of a subvolume as the sum of its N features' "word weights", we now have a way to associate each local descriptor occurrence with a single weight—its contribution to the total classifier score.[2] This same property of linear SVMs is used in [95] to enable efficient subwindow search for object detection, whereas we exploit it to score non-cubic subvolumes in video for action detection.

We stress that our method is not limited to linear SVMs; alternative additive classifiers with the properties described above are also permitted. Our experiments in Sec. 6.2 focus on linear SVMs due to their efficacy. We have also successfully implemented the framework using others, e.g., Naive Bayes, with the same input features. The results are sound, however across the board we find that classifier is less effective than the SVM for our task.

Furthermore, while the additive requirement does lead to an orderless bag-of-features representation, it is still possible to encode temporal ordering into the approach depending on how the local descriptors are extracted. For example, in Sec. 6.1.2.2 we provide one way to record the space-time layout of neighboring objects into high-level visual words.

[2]The bias term β can be ignored for the purpose of maximizing $f(S)$.

6.1.2 Localized Space-Time Features

We consider two forms of localized descriptors for the $\boldsymbol{v}_i$ vectors above: a conventional low-level gradient-based feature, and a novel high-level feature.

6.1.2.1 Low-level Descriptors

For low-level features, we employ an array of widely used local video descriptors from the literature. In general, they capture the texture and motion within localized space-time volumes, either at interest points or dense positions within the video. In particular, we use histograms of oriented gradients (HoG) and histograms of optical flow (HoF) computed in local space-time cubes [97, 86]. The local cubes are centered at either 3D Harris interest points [96] or densely sampled. These descriptors capture the appearance and motion in the video, and their locality lends robustness to occlusions. We also incorporate dense trajectory [163] and motion boundary histogram (MBH) [120] features in a bag-of-features representation. We refer the reader to the original papers about the descriptors for more details.

As is typical in visual recognition, we can expect better accuracy as a function of the greater the variety and complementarity of the features we use, but with some tradeoff in computational cost. Specifically, the main influence the features will have on our method's complexity is their density in the video; while their density will not at all affect the node structure (cf. Sec. 6.1.3), it will dictate how many visual word mappings must be computed. In Sec. 6.2 we provide more discussion about how we select among these descriptors for different datasets; in short, our selection is largely based on empirical findings from previous work about which

are best suited.

6.1.2.2 High-level Descriptors

We introduce a novel descriptor for an alternative high-level representation. While low-level gradient features are effective for activities defined by gestures and movement (e.g., running vs. diving), many interesting actions are likely better defined in terms of the semantic *interactions* between people and objects [65, 36, 132]. For example, "answering phone" should be compactly describable in terms of a person, a reach, a grasp of the receiver, etc.

To this end, we compose a descriptor that encodes the objects and poses occurring in a space-time neighborhood. First, we run a bank of object detectors [57] and a bank of mid-level "poselet" detectors [16] on all frames. To capture human *pose*, we categorize each detected person into one of $P = 15$ "person types". These types are discovered from person detection windows in the training data: for each person window we create a histogram of the poselet activations that overlap it, and then quantize the space of all such histograms with k-means to provide P discrete types. Each reflects a coarse pose—for example, a seated person may cause upper body poselets to fire, whereas a hugging person would trigger poselets from the back.

Given the sparse set of bounding box object detections in a test sequence, we form one neighborhood descriptor per box. This descriptor reflects (1) the type of detector (e.g., person type #3, car) that fired at that position, (2) the distribution of object/person types that also fired within a 50-frame temporal window of it, and

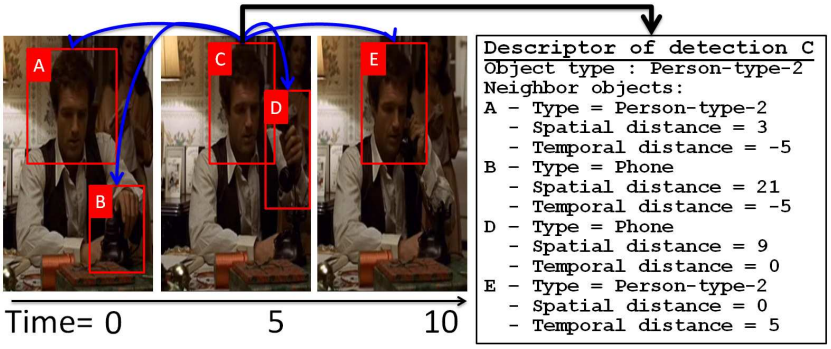


Figure 6.2: Schematic of the data comprising our high-level descriptors. After detecting people and other objects in the video frames, we form semi-local neighborhoods around each detected object that summarize the space-time layout of other nearby detections. To map those neighborhoods into discrete and discriminative visual words, we apply a random forest trained for the action labels (Sec. 6.1.2.2). Here, the left images depict the detected objects surrounding the person detected in bounding box C in the center frame. The right text box displays the information exposed to the random forest feature quantizer, in terms of the neighboring detections and their relative spatial and temporal distance from that person box C.

(3) their relative space-time distances. See Figure 6.2.

To quantize this complex space into discriminative high-level "words", we devise a random forest technique. When training the random forest, we choose spatial distance thresholds, temporal distance thresholds, and object types to parameterize semantic questions that split the raw descriptor inputs so as to reduce action label entropy. Each training and testing descriptor is then assigned a visual word according to the indices of the leaf nodes it reaches when traversing each tree in the forest. Essentially, this reduces each rich neighborhood of space-time object relationships to a single quantized descriptor, i.e., a single index c_i in Eqn. 6.5.

In contrast to the low-level features, this descriptor encodes space-time or-

dering, demonstrating that our max-subgraph scheme is not limited to pure bag-of-words representations. Furthermore, it leads to faster node weight computations, since the number of detected objects is typically much fewer than the number of space-time interest points.

6.1.3 Definition of the Space-Time Graph

So far we have defined the training procedure and features we use. Now we describe how we construct a space-time graph $G = (V, E)$ for a novel test video, where V is a set of vertices (nodes) and E is a set of edges. Recall that a test video is "untrimmed", meaning that we have no prior knowledge about where an action(s) starts or ends in either the spatial or temporal dimensions. Our detector will exploit the graph to efficiently identify the most likely occurrences of a given activity. We present two variants each for the node and link structures, as follows.

6.1.3.1 Node Structure

Each node in the graph is a set of contiguous voxels within the video. In principle, the smallest possible node would be a pixel, and the largest possible node would be the full test sequence. What, then, should be the scope of an individual node? The factors to consider are (1) the granularity of detection that is desired (i.e., whether the detector should predict only when the action starts and ends, or whether it should also estimate the spatial localization), and (2) the allowable computational cost. Note that nodes larger than individual voxels or frames are favorable not only for computational efficiency, but also to aggregate neighborhood statistics to give

better support when the classifier considers that region for inclusion.

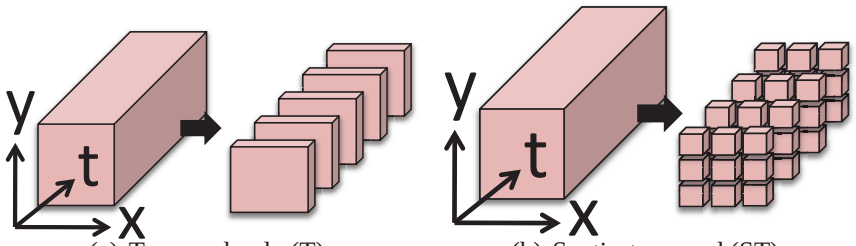
With this in mind, we consider two possible node structures. The first breaks the video into frame-level slabs, such that each node is a sequence of F consecutive frames. The second breaks the video into a grid of $H \times W \times F$ space-time cubes. In all our results, we set $F = 5$ or 10, and let H and W be $\frac{1}{3}$ of the frame dimensions.[3] See Figure 6.3. At detection time, the two forms yield a *temporal subgraph* (**T-Subgraph**) and *spatio-temporal subgraph* (**ST-Subgraph**), respectively. Note that a T-Subgraph will be equivalent to a sliding window search result with a frame step size of F. In contrast, a ST-Subgraph will allow irregular, non-cubic detection results. See the first and last images in Figure 6.7.

After building a graph with either node structure for a test video, we compute the weight for each node v:

$$\omega(\mathrm{v}) = \sum_{\boldsymbol{x}_j \in \mathrm{v}} w^{c_j}, \tag{6.6}$$

where $\boldsymbol{x}_j$ is the 3D coordinate of the j-th local descriptor falling within node $\mathrm{v} \in V$, and c_j is its quantized feature index. We assign the features from Sec. 6.1.2 to their respective graph nodes as follows. For the case of low-level features, $\boldsymbol{x}_j$ is the space-time interest point position. For the case of high-level features, $\boldsymbol{x}_j$ is

[3]Rather than space-time cubes, one could consider using space-time *segments* from a bottom-up grouping algorithm. This would have some potential advantages, including finer-grained localization. However, our preliminary attempts indicated that the regular grid nodes are preferable to segments in practice, for both accuracy and speed. That is because (1) the irregularly shaped segment nodes lead to dense adjacency structures, hurting run-time, and (2) the difficulty in producing quality supervoxels makes it easy to over/under-segment.



(a) Temporal only (T) (b) Spatio-temporal (ST)

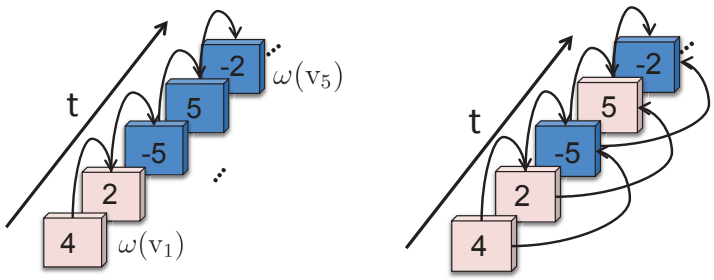
Figure 6.3: The two node structures we consider. (a) A *temporal only* graph simply breaks the video into slabs of frames. Max subgraph search on this graph is equivalent to sliding window in terms of results, but is faster. (b) *Spatio-temporal* graphs further break the frames into spatial cubes, allowing both spatial and temporal localization of the activity in irregular subvolume shapes, at the cost of a denser input graph.

the center of the originating object detection window. In either case, a feature is claimed by the space-time node containing its central position.

Intuitively, nodes with high positive weights indicate that the activity covers that space-time region, while nodes with negative weights indicate the absence of the activity.

6.1.3.2 Linking Strategies

The connectivity between nodes also affects both the shape of candidate subvolumes and the cost of subgraph search. We explore two strategies. In the first, we link only those neighboring nodes that are temporally (and spatially, for the ST node structure) adjacent (see Figure 6.4 (a)). In the second, we additionally link nodes that are within the first two temporal neighbors (see Figure 6.4 (b)); we call this variant **T-Jump-Subgraph**. Since at test time we will seek a maximum scoring *connected* subgraph, the former requires detection subvolumes to be strictly



(a) Neighbors only (b) Neighbors + "Jump"

Figure 6.4: The two linking strategies we consider. (a) The *neighbors only* graph links temporally adjacent (shown here) and optionally spatially adjacent (not shown) nodes. (b) The *temporal "jump"* linking strategy also incorporates edges between non-adjacent nodes, so that the output detection can realize a good connected detection result in spite of intermittent noisy/occlusion features on certain nodes. Here, the numbers shown on nodes indicate weights; white nodes indicate those that would be selected under either linking strategy (see text).

contiguous in time (and thus equates to the options available to a sliding window), while the latter allows subvolumes that "jump" over an adjacent neighbor in time.

By allowing jumps, we can ignore misleading features that may interrupt an otherwise good instance of an action. For example, Figure 6.4 depicts some temporal nodes and their associated weights $\omega(v_i)$'s, under either connectivity scheme. The max subgraph *without* jumps in (a) is the first two nodes only; in contrast, for the same node weights, the max subgraph *with* jumps in (b) extends to include the fourth node, yielding a higher weight subgraph (4+2+5 vs. 4+2). This can be useful when the skipped node(s) contain noisy features, such as an object temporarily blocking the person performing the activity. Like the space-time nodes presented above, the use of temporal jumps further expands the space of candidate subvolumes our method can search, at some additional computational cost.

6.1.4 Searching for the Maximum Weight Subgraph

Having defined the graph constructed on an untrimmed test sequence, we are ready to describe the detection procedure to maximize $f(S)$ in Eqn. 6.1. Our detection objective is an instance of the maximum-weight connected subgraph problem (MWCS): *Given a connected undirected, vertex-weighted graph $G = (V, E)$ with weights $\omega : V \rightarrow \mathbb{R}$, find a connected subgraph $T = (V_T \subseteq V, E_T \subseteq E)$ of G, that maximizes the score $W(T) = \sum_{\mathrm{v} \in V_T} \omega(\mathrm{v})$.* The best-scoring subgraph is the subvolume in the video most likely to encompass the activity of interest. That is the output of our approach. In Sec. 6.1.6 we explain how we iteratively apply the subgraph search procedure to retrieve multiple detections in the same video.

With both positive and negative weights, the problem is NP-complete [74]; an exhaustive search would enumerate and score all possible subsets of connected nodes. However, MWCS can be transformed into an instance of the prize-collecting Steiner tree problem (PCST) [38] which has the same graph structure as original MWCS and vertex profits $p > 0$ and edge costs $c > 0$. This MWCS is solvable by transforming the graph into a directed graph and formulating an integer linear programming (ILP) problem with binary variables for every vertex and edge. Then by relaxing the integrality requirement, the problem can be solved with linear programming using a branch-and-cut algorithm (see [109]). This method gives optimal solutions and is very efficient in practice for the space-time graphs in our setting.

6.1.5 Two Stage Spatio-temporal Detection

Next we describe an extension to the framework that further improves efficiency of spatio-temporal detections, at some loss in search completeness. Basically this extension offers a way to further scale-up our detection strategy for long input videos. It is relevant in the spatio-temporal detection variant of our method (cf. Fig. 6.3(b)), not the temporal-only variant (cf. Fig. 6.3(a)). The fine-grained space-time detection offered by the ST-Subgraph comes from its greater number of nodes and denser connectivity. In particular, in terms of the number of edges as a function of the number of frames, for a temporal-only graph, one more temporal node will add one more edge, in contrast, as for spatio-temporal graph, one more temporal node will add a number of edges quadratic in a function spatial nodes. Thus, to detect the activity efficiently without reducing the granularity of the search scope, we consider how a modest sacrifice on detection accuracy (i.e., giving up the exhaustive search equivalency promised so far) can yield a significantly larger detection speed-up.

To this end, we propose a hierarchical *bottom-up* two stage strategy for the space-time search setting. The basic idea is to first perform space-time detections in each temporal slab, and then propagate those detection results up to a second level of processing that performs temporal detection across the slabs. See Figure 6.5.

Given a test video, we divide the video into spatio-temporal nodes (as depicted in Fig. 6.5, left) and compute their weights as described in Sec. 6.1.3. Next, we search for the best detection volume in two stages: (1) a spatial detection stage and (2) a temporal detection stage. For the spatial detection stage, we connect nodes

in the same temporal slice into a 2D connected weighted graph (see Fig. 6.5, top right). This yields a series of graphs, each of which has nodes representing the features in different spatial positions in the respective temporal slab. We then apply the subgraph search procedure from Sec. 6.1.4 to find the maximum weighted connected subgraph in each slab. Next, the detection score for each 2D subgraph is used to represent the weight of each temporal slab, and these slabs are connected into a 1D temporal graph (see Fig. 6.5, bottom right). Finally, we find the maximum weighted subgraph along the temporal dimension to obtain the detection output. The spatio-temporal detection result is determined by set of spatial-temporal nodes in the 2D max-subgraph that are also selected in 1D max-subgraph.

This hierarchical process reduces the computational cost by dividing the original 3D graph structure into a 2D+1D graph structure. Note, however, that the detection result from the two-stage subgraph search may differ from that returned by the original ST-Subgraph. Whereas the ST-Subgraph is guaranteed to return the same result as an exhaustive search over connected subgraphs, in this modified two-stage procedure, the temporal connection between nodes is always reduced to one edge (vs. nine edges for the original ST-Subgraph). However, the two-stage search process still provides broader searching scope than the simpler T-Subgraph structure.

In practice, when the length of testing video clip is over 1,000 frames, the two-stage subgraph would be preferred over ST-subgraph for efficient spatio-temporal localization. Also, the two-stage subgraph is an approximation of ST-subgraph. If the features are too noisy, the two-stage subgraph may provide lower

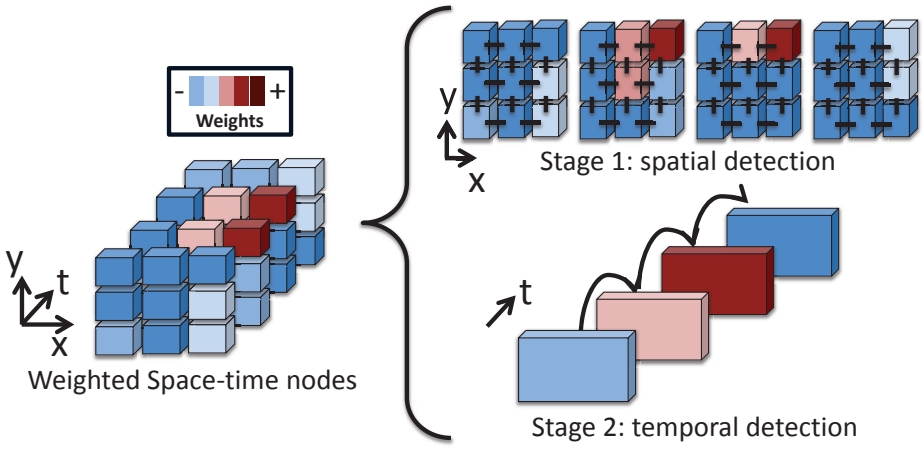


Figure 6.5: Our two stage subgraph search approximates the ST-Subgraph search, allowing efficient spatio-temporal detection even with long test sequences. First we extract the standard space-time cuboid nodes (left). Then, we generate a series of simpler graphs in time (stage 1, top right), and solve for the maximum connected subgraph in each one. This yields a detection region and score for each simpler graph. Finally, we create a graph based on temporal nodes only, which are weighted by the output scores of the previous stage (stage 2, bottom right). The nodes selected in both stages serve as the final output. Best viewed in color.

accuracy since it ignores many edges when computing the maximum weighted subgraph.

6.1.6 Detecting Multiple Activity Instances

Thus far, we have described detection in terms of localizing the single space-time region most likely to contain the activity of interest. In particular, the max-subgraph search returns the subvolume which the trained classifier would score most highly out of all possible subvolumes. To address the scenario where the novel test sequence may contain *multiple* instances of the activity, and/or to provide

multiple confidence-rated hypotheses for the detection output, we extend the max-subgraph search technique as follows.

To detect multiple instances, the main idea is to iteratively run the max-subgraph procedure on adjusted versions of the original input graph, each time adjusting the graph to reflect the most recent detection. The most straightforward approach to modifying the graph would be to take all the nodes selected for the most recent detection and re-weight each one to $-\infty$. Doing so is equivalent to removing those nodes, and it would force the next search iteration to choose other nodes for its next hypothesis. This approach has shortcomings in practice, however. While the max-subgraph output from the first detection is optimal in terms of the classifier and features chosen, it need not be perfect in terms of localizing the actual activity. So, flattening nodes to have weight $-\infty$ leads to fragmented secondary detections.

Therefore, we instead downweight those nodes already involved in a detection, but we do not remove them from the graph entirely. Specifically, each node is re-weighted to 0, as determined empirically on validation data. In this way, the modified graph coming into the next iteration of the max-subgraph computation will favor finding new high-scoring detections, but may still partially re-use portions of the previous detection(s).

The effect of this process is roughly analogous to standard non-maximum suppression (NMS) as applied in object/action detection with sliding windows. With sliding windows, any window with a positive classifier score could be reported as a detection output. However, many windows with positive scores overlap highly

with others, and are actually covering the same object/action instance. To reduce redundant detections, NMS is used to select a single representative output window among a group that highly overlaps. A key parameter that determines the behavior of NMS is the threshold for overlap between detections: candidate windows overlapping with the selected window by more than the selected threshold are not added to the detection output. When the threshold is high, one generates more detection outputs at the risk of redundancy. The re-weighting value applied to nodes in our graphs is analogous to that threshold. A NMS threshold of 0 in traditional sliding windows would correspond to a re-weighting value of $-\infty$ in our setup; a higher NMS threshold corresponds to a higher re-weighting value, allowing some overlap in output detections.

6.2 Experimental Results

We next present experimental results applying our method for activity detection on several public benchmark datasets. We evaluate our approach compared to both sliding window and sliding cuboid baselines as well as an existing state-of-the-art subvolume detection method that exploits branch-and-bound search. Throughout we are interested in both the speed and accuracy attainable. Ideally, we would like to achieve very accurate detection but at a small fraction of the run-time cost incurred by traditional sliding window methods. Furthermore, in some scenarios we hope to improve the accuracy over sliding windows, since our method will permit searching a more complete set of windows than is tractable with a naive search implementation.

In what follows, we first describe the datasets, baselines, and metrics used in our experiments, and we provide implementation details for our approach not already covered above. Then, the next four subsections present results organized around each of the four datasets. This is the most natural organization, since the dataset properties and their respective available ground truth dictate which variants of our approach are relevant for testing (e.g., temporal detection only, fully spatio-temporal, two-stage for spatio-temporal with long sequences, etc.).

6.2.0.1 Activity Detection Datasets

We validate on four datasets, all of which are publicly available:

- **UCF Sports** [138][4]: UCF Sports consists of 10 actions from various sports typically found on TV, such as diving, golf swing, running, and skate boarding. The data originates from stock footage websites like BBC Motion or GettyImages. The provided clips are trimmed to the action of interest, so we expand them into longer test sequences by concatenating clips to form "UCF-Concat" (details below). The ground truth contains the action label and the bounding box annotation of the human.

- **Hollywood Human Actions** [97][5]: The training set contains 219 clips originating from 12 Hollywood movies, and the test set contains 211 clips from a disjoint set of 20 Hollywood movies. The activities are things like answer

[4] http://crcv.ucf.edu/data/UCF_Sports_Action.php
[5] http://www.di.ens.fr/ laptev/actions/

phone, get out of car, shake hands, etc. We test with the noisy "uncropped" versions of the test sequences which are only roughly aligned with the action and contain about 40% extraneous frames. In all data there is a variety of camera motion and dynamic scenes. The ground truth consists of the action label for the clip, as well as the correct temporal boundaries of the activity in the case of the uncropped sequences.

- **MSR Actions** [184][6]: The MSR dataset consists of 16 test clips with three activity classes—hand clapping, hand waving, and boxing—performed in front of cluttered and moving backgrounds. They are performed by 10 subjects, both indoor and outdoor. The ground truth consists of a spatio-temporal bounding box for each action. To our knowledge, this is the only available activity dataset with both spatial and temporal annotations (others are limited to temporal boundaries only). For this dataset, we train the activity classifiers using the disjoint KTH dataset [145], following [184].

- **THUMOS 2014** [80][7]: THUMOS consists of videos collected from YouTube containing 101 different action classes. The emphasis on the THUMOS challenge is to cope with temporally untrimmed videos. Accordingly, the test sequences contain the target actions naturally embedded in other content, and the ground truth includes the temporal boundaries of the true action. Following the localization setting of the winners for the ECCV 2014 workshop's

[6] http://research.microsoft.com/en-us/um/people/zliu/actionrecorsrc/

[7] http://crcv.ucf.edu/THUMOS14/

Dataset	Features	Num test videos	Ave length (#frames)	Ave length of action
UCF-Concat	Dense+HoG3D	12	589	13%
Hollywood uncropped	STIP+HoG/HoF or high-level	211	474	62%
MSR Action	STIP+HoG/HoF	16	756	10%
THUMOS	STIP+HoG/HoF, Trajectory, MBH	111	1717	29%

Table 6.1: Properties of the four datasets. See text for more details.

Figure 6.6: Examples of UCF, Hollywood, MSR Action, and THUMOS datasets.

detection task [1], we divide the 1010 validation videos into two equal parts for testing and training. The test data contains 20 activity classes: baseball pitch, basketball dunk, billiards, clean and jerk, cliff diving, cricket bowling, cricket shot, diving, frisbee catch, golf swing, hammer throw, high jump, javelin throw, long jump, pole vault, shot put, soccer penalty, tennis swing, throw discus, volleyball spiking.

See Table 6.1 for a summary of the dataset properties and Figure 6.6 for example images of these four datasets. In particular, we include each dataset's typical clip lengths and the portion of the sequence occupied by the action to be detected. On average, the action of interest occupies only 28% of the total test sequence, making detection (as opposed to classification) necessary.

6.2.0.2 Baselines

We compare our approach to three baselines:

- **T-Sliding**: a standard temporal sliding window. This is the status quo method in the literature, e.g., [84, 43, 143]. Its results are equivalent to our T-Subgraph variant (using temporal linking structure), but computed with exhaustive search.

- **ST-Cube-Sliding**: a variant of sliding window that searches all cuboid subvolumes having any *rectangular* combination of the spatial-nodes used by our method. Its search scope is similar to our ST-Subgraph, *except* that it lacks all possible spatial links, meaning the detected subvolume cannot shift spatial location over time. While most existing methods simply apply a sliding temporal window, with no spatial localization, we include this baseline as the natural straightforward extension of sliding window search if one wants to obtain localization.

- **ST-Cube-Subvolume**: the state-of-the-art branch-and-bound method of [184]. It considers *all possible* cube-shaped subvolumes, and returns the one maximizing the sum of feature weights inside. Its scope is more flexible than ST-Cube-Sliding. Its objective is identical to ours, *except* that it is restricted to searching cube-shaped volumes that cannot shift spatial location over time. We use the authors' code.[8]

[8] We found its behavior sensitive to its *penalty value* parameter, which is a negative prior on zero-valued pixels [184]. The default setting was weak for our data, so for fairest comparisons, we tuned for best results on UCF.

We stress that our approach is a new strategy for *detection*; results in the literature focus largely on *classification*, and so are not directly comparable. The sliding window and subvolume baselines are state-of-the-art methods for detection, so our comparisons *with identical features and classifiers* will give clear insight into our method's performance.

We consider four variants of our approach: T-Subgraph, T-Jump-Subgraph, ST-Subgraph, and two-stage ST-Subgraph, as defined in Sec. 6.1. Recall that T-Subgraph *provides equivalent accuracy to T-Sliding, but is faster*.[9] The other two variants, T-Jump-Subgraph and ST-Subgraph, provide more flexibility for detection compared to any of the above methods. In particular, the T-Jump-Subgraph variant *allows temporal discontinuities* not permitted by any of the above methods, and the ST-Subgraph variant *allows spatial changes* where the detected content can move spatially within the frame over time. The two-stage ST-Subgraph (cf. Sec. 6.1.5) is like the latter, only computed in an approximate form so as to scale well to longer test sequences.

Figure 6.7 depicts the scope of the regions searched by each method, both ours and the baselines.

[9]For the special case of temporal search, one can obtain equivalent solutions using 1-D branch-and-bound search to detect the max subvector along the temporal axis [11]. In practice we find this method's run-time to be similar or slightly faster than T-Subgraph. Note, however, that it is *not* applicable for any other search scope handled by our approach.

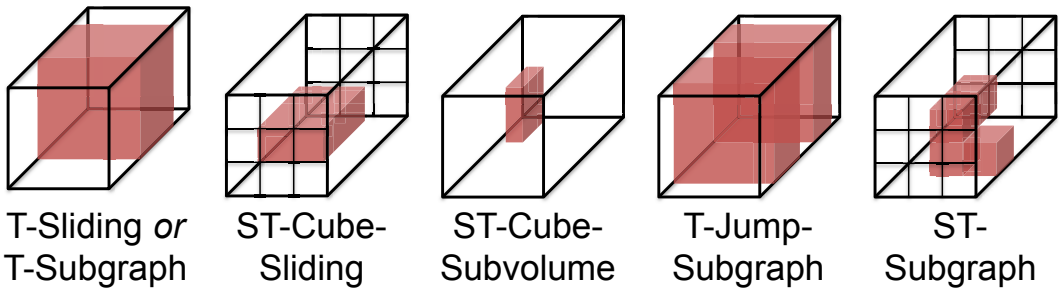


Figure 6.7: Sketch of the candidate subvolume types considered by different methods, ordered approximately from least to most flexible. **T-Sliding or T-Subgraph**: The status quo sliding window search (and the proposed T-Subgraph without jumps) finds the full-frame subvolume believed to contain the activity (leftmost image). **ST-Cube-Sliding**: A variant that performs sliding window on different spatial portions of the frame, with the restriction of cuboid subvolumes. **ST-Cube-Subvolume**: A branch-and-bound search strategy from existing work [184] that considers all possible cube-shaped subvolumes—not just the grid-based subset considered by ST-Cube-Sliding. **T-Jump-Subgraph**: The proposed method using temporal nodes (slabs of frames) only, with additional allowance of temporal "gap(s)" in the output detections. **ST-Subgraph**: The most general form of the proposed method, where we use both spatial and temporal nodes, allowing irregular, non-cubic detection results.

6.2.0.3 Evaluation Metrics

We adopt standard metrics for detection evaluation. Following [175, 87, 184], we use the *mean overlap accuracy*. Whether performing temporal or full spatio-temporal detection, this metric computes the intersection of the predicted detection region with the ground truth, divided by the union.

As for detection speed, we use detection time (on our 3.47GHz Intel Xeon CPUs) to evaluate computational cost. Note that in this work, we focused on improving the speed of the system in testing stage. We use the same feature extraction and classifer training framework for all our methods and baselines. To apply our

method to online system, we will need to add the feature extraction time to our result.

6.2.0.4 Implementation Details

For all datasets, we train a binary SVM to build a detector for each action. We use the descriptors described in Sec. 6.1.2, following the guidance of prior work [164, 163] to select which particular sampling strategies and local space-time descriptors to employ per dataset. In particular, recommendations from [164] lead us to employ HoG/HoF for Hollywood and HoG3D for UCF with dense sampling. For the THUMOS dataset we use the features provided with the dataset, which augments the HoG/HoF set with dense trajectories and MBH. In particular, on THUMOS we train one-versus-all binary SVMs with four types of features: trajectory [163], HOG, HOF, and MBH [120], where the features are quantized to a bag of words representation via k-means with a dictionary size $= 4000$. We use the authors' code for HoG3D/HoG/HoF/trajectory/MBH [97, 86, 163, 120], with default parameter settings. We test the high-level descriptors on Hollywood, since that dataset has substantial person-object interactions, whereas actions in the others are more person-centric (e.g., diving, clapping, skateboarding). We construct our temporal graphs with a node size of 10 frames per slab.

The next four sections describe the results on each dataset in turn.

Verbs	T-Sliding	ST-Cube-Subvol [184]	Our-T-Subgraph	Our-T-Jump-Subgraph
Diving	0.8106	0.7561	0.8106	**0.9091**
Lifting	0.7899	0.8058	0.7899	**0.8096**
Riding	**0.5349**	0.5075	**0.5349**	0.3888
Running	0.4602	0.3269	0.4602	**0.4705**
Skateboard	0.1407	0.1057	0.1407	**0.1803**
Swing-Bench	0.5520	**0.6259**	0.5520	0.4582
Swing-Side	0.6728	0.3478	0.6728	**0.7212**
Walking	0.4085	0.3462	0.4085	**0.4657**

Table 6.2: Mean overlap accuracy for the UCF Sports data.

Detection time (ms)	T-Sliding	ST-Cube-Subvol [184]	Our-T-Subgraph	Our-T-Jump-Subgraph
Mean	1.25×10^5	7.87×10^4	1.02×10^2	6.51×10^2
Stdev	7.52×10^3	3.17×10^4	5.35×10^1	3.17×10^2

Table 6.3: Search time for the UCF Sports data.

6.2.1 Temporal Detection on UCF Sports

Since the UCF clips are already cropped to the action of interest, we modify it to make it suitable for detection. We form 12 test sequences by concatenating 8 different clips each from different verbs. All test videos are totally distinct, and are available on our project website. We train the SVM on a disjoint set of cropped instances. We perform temporal detection only, since the activities occupy the entire frame.

Table 6.2 shows the accuracy results, and Table 6.3 shows the search times. For almost all verbs, our subgraph approaches outperform the baselines. Further, our T-Jump variant gives top accuracy in most cases, showing the advantage of ignoring noisy features (in this data, often found near the onset or ending of the verb). Figure 6.8 shows an example where T-Jump performs robust detection in spite of occlusions, whereas the baseline sliding window or basic T-Sliding fails.

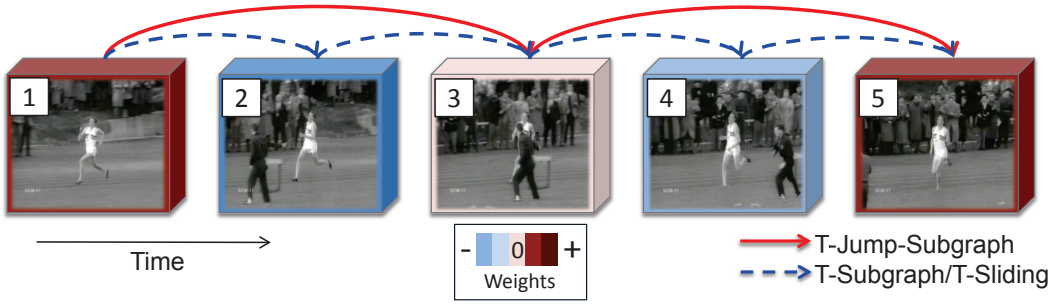


Figure 6.8: Qualitative example showing how our T-Jump method can perform robust detection. The five colored cubes represent the weighted node computed from the extracted features and learned classifier. For the second to fourth nodes, the classifier generates negative weights due to the occlusion. Using T-Sliding or T-Subgraph, the detection output does not cover the first and last cubes due to the negative weights from three cubes in the middle. In contrast, using our T-Jump method, it can skip over the intervening negative weights. This makes the detection framework more robust to noise from occlusion. Best viewed in color.

On this dataset, the ST-Cube-Subvolume baseline is often weaker than sliding window. Upon inspection, we found it often fires on a small volume with highly weighted features when the activity changes in spatial location over time. However, it is best on "Swing-Bench", likely because the backgrounds are fairly static, minimizing misleading features. As we see in Table 6.3, both our subgraph methods are orders of magnitude faster than the baselines. Note that the ST-Cube-Subvolume's higher cost is reasonable since here it is searching a wider space.

6.2.2 Temporal Detection on Hollywood

We next test the Hollywood data, which also permits a study of temporal detection. As noted above, we test with the untrimmed data provided by the dataset creators. Existing work uses this data for classification, and so trains *and* tests with

Verbs	T-Sliding	ST-Cube-Subvol [184]	Our-T-Subgraph	Our-T-Jump-Subgraph
AnswerPhone	0.3968	0.2905	0.3968	**0.3994**
GetOutCar	0.2276	0.2267	0.2276	**0.2921**
HandShake	0.3071	0.3390	0.3071	**0.3663**
HugPerson	0.3869	**0.4486**	0.3869	0.4150
Kiss	0.3822	0.4230	0.3831	**0.4412**
SitDown	**0.3612**	0.2861	**0.3612**	0.3550
SitUp	0.2592	0.2053	0.2592	**0.3255**
StandUp	0.3475	0.3013	0.3475	**0.3775**

Table 6.4: Mean overlap accuracy on uncropped Hollywood data.

Detection Time (ms)	T-Sliding	ST-Cube-Subvol [184]	Our-T-Subgraph	Our-T-Jump-Subgraph
Mean	3.71×10^3	1.70×10^5	6.63×10	5.69×10^2
Stdev	1.03×10^4	5.79×10^5	7.51×10	1.77×10^3

Table 6.5: Search time on uncropped Hollywood data.

the cropped versions. To perform temporal detection, we instead train with the cropped clips, and test with the uncropped clips.

Table 6.4 shows the accuracy results, and Table 6.5 shows the search times. Our T-Jump-Subgraph achieves the best accuracy for 6 of the 8 verbs, with even more pronounced gains than on UCF. This again shows the value of skipping brief negatively weighted portions; e.g., "AnswerPhone" can transpire across several shot boundaries, which tends to mislead the baselines.

As Table 6.5 reveals, our method is again significantly faster than the baselines. Our T-Jump-Subgraph is slower than our T-Subgraph search, given the higher graph complexity (which also makes it more accurate). Hence, which variant to apply depends on how an application would like to make this cost-accuracy tradeoff.

One might wonder whether a naive detector that simply classifies the entire uncropped clip could do as well. To check, we compare *recognition* results when

Test sequence composition	Accuracy
Raw uncropped clips	24.83%
Output from T-Subgraph	29.66%
Manual ground truth	29.97%

Table 6.6: Recognition accuracy on Hollywood as test input varies.

we vary the composition of the test sequence to be either (a) the uncropped clip, (b) the output of our detector, or (c) the ground truth cropped clip. Table 6.6 shows the result. We see indeed that detection is necessary; using our output is much better than the raw untrimmed clips, and only slightly lower than using the manually provided ground truth.

We also test our high-level descriptor (cf. Sec. 6.1.2.2) on Hollywood, since its actions contain human-object interactions. We apply six object detectors—bus, car, chair, dining table, sofa, and phone—to every fifth frame, and use random forests with 10 trees. Table 6.7 shows the results, compared to our method using low-level features. For five of the eight actions, the proposed high-level descriptor improves accuracy. It is best for activities based on the interaction between two people (e.g., kiss) or involving an obvious change in pose (e.g., sit up), showing the strength of the proposed person types to capture pose and temporal ordering. For other verbs with varied objects (answer phone, get out of car), it hurts accuracy, likely due to object detector failures in this dataset. It remains future work outside the scope of this project to bolster the component object detectors fed into this higher-level neighborhood descriptor.

Verbs	T-Subgraph (HoG/HoF)	T-Subgraph (high-level)
AnswerPhone	**0.3968**	0.1741
GetOutCar	**0.2276**	0.1447
HandShake	0.3071	**0.4194**
HugPerson	0.3869	**0.5292**
Kiss	0.3822	**0.4906**
SitDown	0.3612	**0.3753**
SitUp	0.2592	**0.3843**
StandUp	**0.3475**	0.2636

Table 6.7: Mean overlap accuracy on Hollywood for low-level features vs. the object-based high-level descriptors.

6.2.3 Temporal Detection with Multiple Instances on THUMOS

Next we evaluate our approach on the THUMOS dataset. THUMOS allows temporal detection (like UCF Sports and Hollywood), plus, unlike the others, it contains test sequences with multiple instances of the activity. This aspect lets us test our iterative max-subgraph strategy to produce multiple detections, as discussed in Sec. 6.1.6.

In these experiments, the sliding window baseline represents the same search strategy taken by the leading approach on this dataset [1]. As such, we follow the authors' parameter choices for the window search in order to provide a close comparison. That means for the T-Sliding baseline, we use a step size of 10 frames, and evaluate the windows with durations of 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, and 150 frames [1]. We fix the NMS threshold at 0.5 (after we did not observe better results for the baseline shifting this threshold within the range (0,1]), and we fix the node re-weighting value at 0 for our method (cf. Sec. 6.1.6). Note that with a skip size of 10 frames, the sliding window baseline (T-Sliding) does not exhaustively search all subsequences, whereas our method does. For each testing video,

Metric	T-Sliding	Our T-Subgraph	Our T-Jump-Subgraph
mAP	0.1983	**0.2143**	0.1546
Overlap	0.1792	0.2186	**0.2636**

Table 6.8: Recognition accuracy on THUMOS 2014 data.

we return up to 10 positive detection windows.

Table 6.8 shows the accuracy results for T-Sliding and our T-Subgraph method, both in terms of overlap and the mean average precision (mAP) as defined by [80], which is a useful metric for the case when there are multiple instances per testing clip. Our method obtains higher accuracy than the standard sliding window baseline. This is a direct consequence of the efficiency of our approach in considering all possible windows. We also get a noticeable further advantage in overlap accuracy applying our T-Jump variant, yet it harms average precision. Upon inspection, we find that for this challenging data, the classifier scores per node are noisier, which leads T-Jump to cover too many frames; T-Jump can easily find some small-valued positive nodes to skip over highly negative nodes, leading to some poorer detection outputs as seen in the mAP. The high overlapping score of T-Jump confirms this observation and illustrates why mAP is a better metric than overlapping accuracy in multiple instance detection. We also tried a variant of our approach that less aggressively reduces the weights on nodes already involved in a prior iteration's detections: we set the weight of a "used" node to the mean weight of all nodes, with the intent to encourage more overlapping detections. However, this led to slightly worse accuracy for our method (0.2043 overlap accuracy vs. 0.2186 in Table 6.8).

Table 6.9 shows the computation time for both methods. Similar to previous

Time (ms)	T-Sliding	Our T-Subgraph	Our T-Jump-Subgraph
Mean	7.07×10^5	5.34×10^4	4.72×10^4
Stdev	2.26×10^6	2.37×10^5	1.97×10^5

Table 6.9: Search time on THUMOS 2014 data.

results, our T-Subgraph method for detecting multiple instances provides significantly faster running time compared to T-Sliding. For the sliding window method, no matter how many output detections we want, all the candidate window are evaluated. In contrast, for our T-Subgraph, we only return one optimal window in each subgraph search iteration and re-weight the underlying nodes for next iteration. Therefore, in this experiment, we need to run our T-Subgraph 10 times to find top 10 detection windows—yet, in spite of that repetition, it is still about an order of magnitude faster than evaluating all the candidate windows in the T-Sliding method.

Finally, we more closely analyze the behavior of the sliding window baseline (T-Sliding) as it compares to our T-Subgraph. The goal is to see in practice what density of windowed search (skip sizes) is necessary for best results. In other words, if we allow T-Sliding more candidate windows and hence longer running time, at what point does it come close to the optimal result from our method? Since running this experiment is rather costly for the baseline, we limit this test to four of the 20 verbs in the THUMOS test set (chosen randomly: basketball dunk, clean and jerk, cliff diving, and hammer throw).

Figure 6.9 shows the results in terms of the average accuracy over all four actions tested. As expected, increasing the pool of candidate windows searched by T-Sliding increases its accuracy, but at a corresponding linear increase in run-time. At a search time of 200 *ms* per frame, the baseline is searching 35 different window

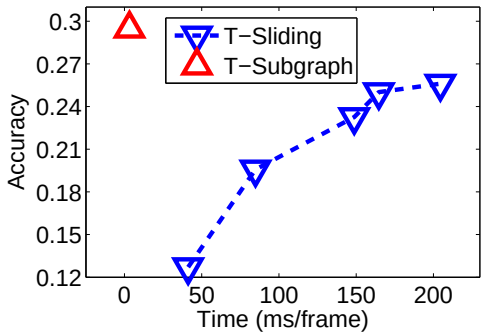


Figure 6.9: Accuracy vs. computation time in temporal search. We compare our T-Subgraph (which produces the optimal detection output for a fixed time) to the standard T-Sliding method (which produces its detection output based on exhaustive search of a pool of candidate windows). Here we increase T-Sliding's accuracy and run-time by increasing that pool of windows.

sizes (out of 300 window sizes for exhausted search) and achieves accuracy of 0.26, nearing but not as good as the result from T-Subgraph of 0.30 accuracy obtained with just a few *ms* per frame.

6.2.4 Space-Time Detection on MSR Actions

As the fourth and final dataset, we experiment with MSR Actions. In contrast to all of the above datasets, MSR Actions contains ground truth for the *spatial* localization of the action—not just the temporal extent. Furthermore, the actors change their position over time and a test sequence may contain multiple simultaneous instances of different actions. Therefore, this dataset is a good testbed to evaluate our ST-Subgraph with the node structure in Figure 6.3(b), where we link neighboring nodes both in space and time. In what follows, we present results with both the exact maximum subgraph from ST-Subgraph as well as its approximate

Verbs	T-Sliding	ST-Cube-Sliding	ST-Cube-Subvol [184]	Our-T-Subgraph	Our-ST-Subgraph	Our-Two-Stage-ST
Boxing	0.0541	0.0717	0.0794	0.0541	0.0989	**0.1188**
Clapping	0.0982	0.0982	0.0602	0.0982	0.1754	**0.1795**
Waving	0.2342	0.2204	0.2669	0.2342	**0.2926**	0.2416

Table 6.10: Mean temporal overlap accuracy on the MSR dataset.

Detection Time (ms)	T-Sliding	ST-Cube-Sliding	ST-Cube-Subvol [184]	Our-T-Subgraph	Our-ST-Subgraph	Our-Two-Stage-ST
Mean	4.2×10^3	5.5×10^4	3.0×10^5	2.8×10^2	3.1×10^6	1.4×10^3
Stdev	3.3×10^3	4.2×10^4	1.6×10^5	2.3×10^2	4.6×10^6	4.1×10^2

Table 6.11: Search time on the MSR dataset.

counterpart, the two-stage search process described in Sec. 6.1.5.

First we isolate temporal detection accuracy alone. We run the temporal and spatio-temporal variants of our method, and project the spatio-temporal results to temporal results. Table 6.10 shows results. Even under the temporal criterion, our ST-Subgraph and two stage ST-Subgraph are most accurate, since they can isolate those nodes that participate in the action. Figure 6.10 illustrates how our space-time node structure succeeds when the location of activity changes over time, whereas ST-Cube-Subvolume may be trapped in cube-shaped maxima. Compared to ST-Subgraph, our two-stage method yields similar accuracy for Boxing and Clapping videos and provides lower accuracy for Waving videos. This result shows the two-stage method is able to provide good approximation to ST-subgraph method.

Next we examine the complete space-time localization accuracy. Table 6.12 shows the results, evaluated under the ground truth annotation for the person who performs the action[10]. Results are mixed between the methods, with a slight edge for our ST-Subgraph. Also, only the non-rectangular shape detection from our ST-

[10]The original ground truth labels only the hand regions (see Figure 6.10), whereas this ground truth labels the whole person performing the action.

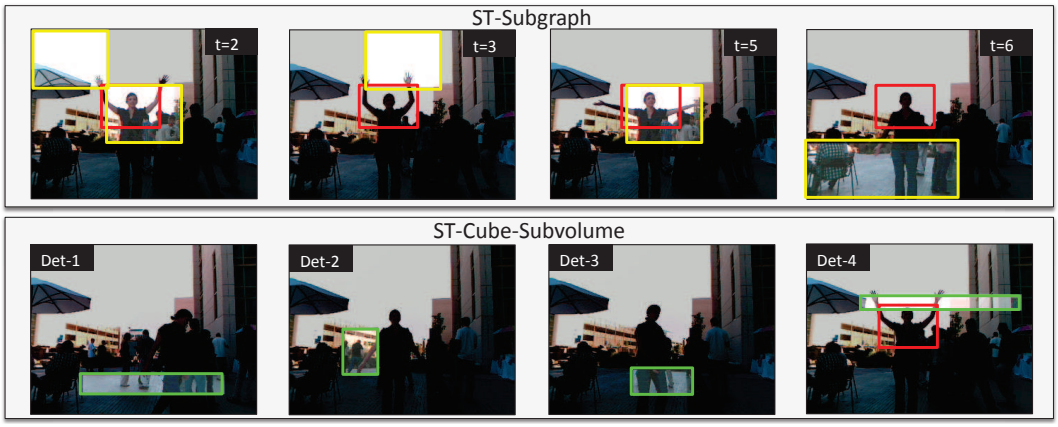


Figure 6.10: Example of ST-Subgraph's top output (top) and the top 4 detections from ST-Cube-Subvolume [184] (bottom). Red rectangles denote ground truth. Brighter areas denote detections.

Verbs	ST-Cube-Sliding	ST-Cube-Subvol [184]	Our-ST-Subgraph	Our-Two-Stage-ST
Boxing	**0.0478**	0.0193	0.0417	0.0296
Hand Clapping	0.0373	0.0071	**0.0630**	0.0425
Hand Waving	0.0851	0.0581	**0.1121**	0.0809

Table 6.12: Mean space-time overlap accuracy on the MSR dataset. (T-Sliding/T-Subgraph are omitted since they don't do spatial localization.)

Subgraph reflects the large spatial motions in actions. As expected, the two-stage search process does detract from the accuracy of the optimal ST-Subgraph result, as we see in the last two columns of Table 6.12.

Finally, we analyze the run-times for all methods tested in Table 6.11. Here we see the substantial practical impact of our two-stage spatio-temporal variant, which yields significantly lower computation time. It is even faster than the sliding temporal window search that produces no spatial localization, and orders of magnitude faster than the existing branch-and-bound subvolume method [184]. The two-stage method is slightly slower than the T-Subgraph variant of our method,

since it requires additional computation for the spatial detection in the first stage for each slab.

As discussed in Sec. 6.1.5, we can achieve efficient spatio-temporal localization with the our proposed two stage subgraph search method. In the previous section, our ST-Subgraph provides more accurate space-time localization of actions with higher computational cost. In this section, we speed up the ST-Subgraph with our two stage subgraph for space time detection on MSR action dataset.

Table 6.12 and Table 6.11 also show the comparison of detection accuracy and search time for our Two-Stage-ST-Subgraph and our original ST-Subgraph. By dividing the node structure into temporal slices, the computation time of the two stage method is reduced by three orders of magnitude compared to the original ST-Subgraph. As expected, the two stage method is slightly slower than the T-Subgraph because it requires additional computation for spatial detection in first stage for each temporal node. For detection accuracy, recall that the two stage method does not guarantee to provide the optimal spatial-temporal volumes since it ignores the temporal link between nodes in the first stage. Thus, it is expected that the two stage method will be less accurate than the ST-Subgraph method. As shown in Table 6.12, Two-Stage-ST method achieves similar accuracy to the ST-Subgraph for hand clapping and hand waving clips, but lower accuracy for boxing clips. It is because the learned activity model for boxing is less accurate than the learned models for other two actions (it provides lower overlap accuracy for ST-Subgraph), and our two stage method is more sensitive to the noisy node score due to the pruned connections between nodes.

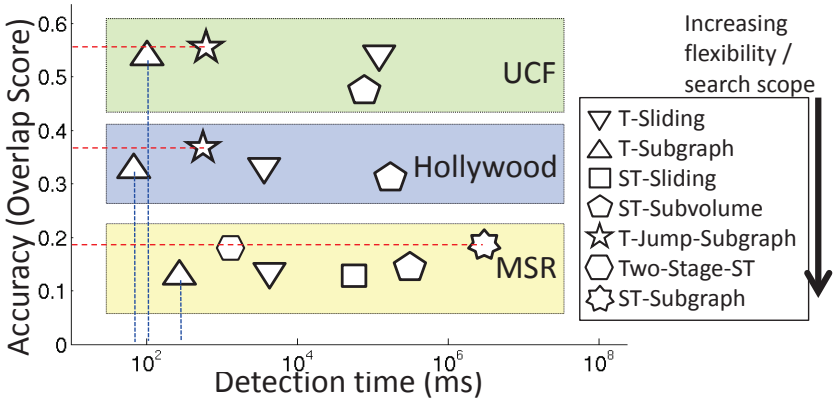


Figure 6.11: Overview of methods on the three datasets.

6.2.5 Summary of Trade-Offs in Results

Having presented all the results, now we step back and attempt to summarize the outcomes succinctly. There are three dimensions of trade-offs between all methods tested: search time, search scope, and detection accuracy.

Figure 6.11 summarizes all trade-offs for three datasets. Here we show the accuracy versus the detection time for each result, and encode the search scope of the method by the complexity of its polygonal symbol. More complex symbols mean wider search scope. For example, recalling Figure 6.7, the least complex search scope is T-Sliding/T-Subgraph, which is plotted as a triangle, whereas the most complex search scope is the ST-Subgraph, which is plotted as a 14-sided star.

Importantly, we see that increased search scope generally boosts accuracy. In addition, the flexibility of the graph structure in our subgraph algorithm allows it to perform best per dataset in terms of *either* speed (see vertical blue dotted lines) or accuracy (see horizontal red dotted lines).

6.3 Conclusions

In this chapter, I presented a novel branch-and-cut framework for activity detection that efficiently searches a wide space of temporal or space-time subvolumes. Compared to traditional sliding window search, it significantly reduces computation time. Compared to existing branch-and-bound methods, its flexible node structure offers more robust detection in noisy backgrounds. Our novel high-level descriptor also shows promise for complex activities, and makes it possible to preserve the spatio-temporal relationships between humans and objects in the video, while still exploiting the fast subgraph search.

With this approach, we can localize the learned action models in new video sequences efficiently. Next, I will discuss about how to extend the ideas in my dissertation to further broaden their applications and develop related novel ideas in this area.

Chapter 7

Future Work

In previous chapters, I described how to interconnect different data sources to overcome the obstacles in learning humans' actions and poses. There are several future directions prompted by this thesis, which could broaden the application of my ideas and reduce the human supervision for solving computer vision problem.

7.1 Exploring Patterns in Videos

In Chapter 3, I showed how temporal dependency allows us to link the snapshots to the images in unlabelled video pools and provide extra information. In Chapter 4, I showed how the correlations between different poses allow us to infer the pose for the unseen views. Next, we could further explore the underlying patterns in human related video clips by combining the dependency across different views *and* temporal frames. These extracted patterns could be used to provide regularization in learning human actions or poses. Once we are able to extract these patterns, we could utilize them as regularization in different tasks.

For example, we could use the videos captured from various views to initialize the weights of a convolutional neural network (CNN) for learning an action/pose model. Then we could fine-tune the model with few labeled video clips or images.

Or the system could infer the spatial-temporal features for unseen video segments for different views. Given two related video clips, the system could connect them by filling in the gap between the two clips. Also, incorporating the temporal dependency, the system could provide inference for pose sequences in different views or handling the occlusion problems.

7.2 Interactions in Wearable Devices

The approaches shown in Chapters 3 and 4 focus on learning the model for a single person, and the approach shown in Chapter 5 considers interaction between a person and a single object. In the future, we could generalize those ideas to interactions between *multiple* people and objects.

One such application is analyzing the interactions for images/videos captured through wearable devices. In our daily life, we might interact with multiple people/objects at the same moment. Besides, the interaction could happen among a group of people instead of as a pairwise interaction as described in Chapter 5. For example, while walking in the street, we step on the pavement, look at the street sign, talk to people, and avoid obstacles. Besides, the people and objects around us can interact with each other. To model the problem, the approach needs to consider the relationship between all visible objects and possible interactions from visual contents captured from multiple devices. The video content captured via wearable devices tend to be hours long thus cannot afford to have detailed annotations. In such case, we would need to combine various sources of data to reduce the labeling cost.

Chapter 8

Conclusion

My thesis presented novel techniques for improving the learning and understanding of human action and pose. The proposed approaches interconnect the data by exploring underlying patterns from articulated human pose structure.

I consider four major components. In the first, I described a novel approach to connect static snapshots to the temporal dependency between poses provided by unlabelled video sequences and utilize the mined information to aid the learning of new human action from just a few snapshots. Second, I used a tensor completion technique to discover the latent factors connecting the human poses across different views. With this method, we are able to learn a human action model from different views without collecting examples for each of the viewing angles. Third, I proposed a new approach that explores the pattern that connects the pose and location of interactees in a category-independent way. With the proposed method, we are able to predict the location and size of an interactee for different types of interaction and objects. In addition, I also explored various applications by using the interactee localization as a cue, including for detection, image retargeting, and image description. Last, after exploring how to better learn models of action and pose, I described a framework to efficiently detect an action in a video sequence. To localize when

and where the action is happened in the video, I transform the problem into finding a maximum weighted subgraph in a flexible graph structure. This method significantly increases the speed for searching and provides diverse localization scope. Furthermore, I developed different variants to handle the trade off between search speed and computation cost for realistic applications.

In summary, the main impact of my thesis is that it shows how we can reduce the data collection cost in human related data. By using the existing labelled data more efficiently with unlabelled data or data from other sources, we can significantly improve the accuracy or speed of existing recognition systems. My work mainly focuses on finding clever and efficient ways of using the human related data, and the approaches I proposed are a promising step in improving action recognition.

Bibliography

[1] *http://crcv.ucf.edu/THUMOS14/papers/INRIA LEAR.pdf*.

[2] *http://www.flickr.com/*.

[3] J. K. Aggarwal and Q. Cai. Human motion analysis: a review. *CVIU*, 73(3):428–440, 1999.

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

[5] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3):10, 2007.

[6] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *CVPR*, 1997.

[7] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV*, 2011.

[8] S. Bandla and K. Grauman. Active learning of an action detector from untrimmed videos. In *ICCV*, 2013.

[9] E. Bart and S. Ullman. Cross-Generalization: Learning Novel Classes from a Single Example by Feature Replacement. In *CVPR*, 2005.

[10] Y. Bengio, J.-F. Paiement, and P. Vincent. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps, and spectral clustering. In *NIPS*, 2003.

[11] J. Bentley. Programming pearls: algorithm design techniques. *Commun. ACM*, 27(9):865–873, Sept. 1984.

[12] A. Berg, T. Berg, H. Daume, J. Dodge, A. Goyal, X. Han, A. Mensch, M. Mitchell, A. Sood, K. Stratos, and K. Yamaguchi. Understanding and predicting importance in images. In *CVPR*, 2012.

[13] C. M. Bishop. Mixture density networks. Technical report, Microsoft Research Cambridge, 1994.

[14] J. Blackburn and E. Ribeiro. Human motion recognition using isomap and dynamic time warping. In *Human Motion*, 2007.

[15] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *ICCV*, 2005.

[16] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009.

[17] L. Cao, Z. Liu, and T. S. Huang. Cross-dataset action detection. In *CVPR*, 2010.

[18] J. Carreira and C. Sminchisescu. Constrained Parametric Min-Cuts for Automatic Object Segmentation. In *CVPR*, 2010.

[19] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. In *ACM Trans Intelligent Sys and Technology*, volume 2, pages 27:1–27:27, 2011.

[20] C.-Y. Chen and K. Grauman. Efficient activity detection with max-subgraph search. In *CVPR*, 2012.

[21] C.-Y. Chen and K. Grauman. Predicting the location of interactees in novel human-object interactions. In *ACCV*, 2012.

[22] C.-Y. Chen and K. Grauman. Watching unlabeled videos helps learn new human actions from very few labeled snapshots. In *CVPR*, 2013.

[23] C.-Y. Chen and K. Grauman. Inferring unseen views of people. In *CVPR*, 2014.

[24] C.-Y. Chen and K. Grauman. Predicting the location of interactees in novel human-object interactions. In *ACCV*, 2014.

[25] H. Chiu, L. Kaelbling, and T. Lozano-Perez. Virtual training for multi-view object class recognition. In *CVPR*, 2007.

[26] W. Choi, K. Shahid, and S. Savarese. What are they doing? : Collective activity classification using spatio-temporal relationship among people. In *Proc. of 9th International Workshop on Visual Surveillance (VSWS09) in conjuction with ICCV*, 2009.

[27] W. Choi, K. Shahid, and S. Savarese. Learning context for collective activity recognition. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.

[28] M. Cristani, L. Bazzani, G. Paggetti, A. Fossati, A. Bue, G. Menegaz, and V. Murino. Social interaction discovery by statistical analysis of fformations. In *BMVC*, 2011.

[29] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, 2005.

[30] D. Damen and D. Hogg. Detecting carried objects in short video sequences. In *ECCV*, 2008.

[31] H. Daume III. Frustratingly easy domain adaptation. In *ACL*, 2007.

[32] V. Delaitre, D. Fouhey, I. Laptev, J. Sivic, A. Gupta, and A. Efros. Scene semantics from long-term observation of people. In *ECCV*, 2012.

[33] V. Delaitre, J. Sivic, and I. Laptev. Learning person-object interactions for action recognition in still images. In *NIPS*, 2011.

[34] M. Denkowski and A. Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*, 2014.

[35] C. Desai and D. Ramanan. Predicting functional regions on objects. In *CVPR Workshop on Scene Analysis Beyond Semantics*, 2013.

[36] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for static human-object interactions. In *Workshop on Structured models in Computer Vision*, 2010.

[37] J. Devlin, S. Gupta, R. Girschick, M. Mitchell, and L. Zitnick. Exploring nearest neighbor approaches for image captioning. In *arXiv*, 2015.

[38] M. T. Dittrich, G. W. Klau, A. Rosenwald, T. Dandekar, and T. Mller. Identifying functional modules in protein-protein interaction networks: an integrated exact approach. *Bioinformatics*, 2008.

[39] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *PAMI*, 34, 2012.

[40] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[41] L. Duan, D. Xu, and S.-F. Chang. Exploiting web images for event recognition in consumer videos: a multiple source domain adaptation approach. In *CVPR*, 2012.

[42] L. Duan, D. Xu, I. W.-H. Tsang, and J. Luo. Visual event recognition oin videos by learning from web data. In *CVPR*, 2010.

[43] O. Duchenne, I. Laptev, J. Sivic, F. R. Bach, and J. Ponce. Automatic annotation of human actions in video. In *ICCV*, 2009.

[44] P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth. Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Im age Vocabulary. In *ECCV*, 2002.

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

[46] E. et al. Pascal voc action classification challenge.

[47] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. In *IJCV*, volume 88, pages 303–338, June 2010.

[48] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results.

[49] H. Fang, S. Gupta, F. Iandola, R. Srivastava, L. Deng, P. Dollar, J. Gao, X. He, M. Mitchell, J. Platt, C. Zitnick, and G. Zweig. From captions to visual concepts and back. In *CVPR*, 2015.

[50] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *ECCV*, 2010.

[51] A. Farhadi and M. Sadeghi. Recognition using visual phrases. In *CVPR*, 2011.

[52] A. Farhadi and M. Tabrizi. Learning to recognize activities from the wrong view point. In *ECCV*, 2008.

[53] A. Farhadi, M. K. Tabrizi, I. Endres, and D. A. Forsyth. A latent model of discriminative aspect. In *ICCV*, 2009.

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

[55] L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. In *ICCV*, 2003.

[56] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[57] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010.

[58] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Pose search: retrieving people using their pose. In *CVPR*, 2009.

[59] D. F. Fouhey, V. Delaitre, A. Gupta, A. A. Efros, I. Laptev, and J. Sivic. People watching: Human actions as a cue for single-view geometry. In *Proc. 12th European Conference on Computer Vision*, 2012.

[60] W. T. Freeman and J. B. Tenenbaum. Learning bilinear models for two-factor problems in vision. In *CVPR*, 1997.

[61] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik. Using k-poselets for detecting people and localizing their keypoints. In *CVPR*, 2014.

[62] K. Grauman, G. Shakhnarovich, and T. Darrell. Inferring 3D structure with a statistical image-based shape model. In *ICCV*, 2003.

[63] S. Guadarrama, N. Krishnamoorthy, G. Malkarnenkar, S. Venugopalan, R. Mooney, T. Darrell, and K. Saenko. Youtube2text: Recognizing and describing arbitrary activities using semantic hierarchies and zero-shot recognition. In *ICCV*, 2013.

[64] A. Gupta and L. S. Davis. Objects in action: An approach for combining action understanding and object perception. In *CVPR*, 2007.

[65] A. Gupta, A. Kembhavi, and L. Davis. Observing human-object interactions: using spatial and functional compatibility for recognition. *PAMI*, 31(10), 2009.

[66] A. Gupta, S. Satkin, A. Efros, and M. Hebert. From 3D scene geometry to human workspace. In *CVPR*, 2011.

[67] I. Haritaoglu, D. Harwood, and L. Davis. W4: real-time surveillance of people and their activities. *PAMI*, 2000.

[68] M. Hoai, Z.-Z. Lan, and F. De la Torre. Joint segmentation and classification of human actions in video. In *CVPR*, 2011.

[69] X. Hou and L. Zhang. Saliency detection: A spectral residual approach. In *CVPR*, 2007.

[70] C.-H. Huang, Y.-R. Yeh, and Y.-C. Wang. Recognizing actions across cameras by exploring the correlated subspace. In *ECCV*, 2012.

[71] S. J. Hwang and K. Grauman. Accounting for the Relative Importance of Objects in Image Retrieval. In *BMVC*, 2010.

[72] S. J. Hwang and K. Grauman. Reading between the lines: Object localization using implicit cues from image tags. In *CVPR*, 2010.

[73] S. J. Hwang and K. Grauman. Learning the relative importance of objects from tagged images for retrieval and cross-modal search. *IJCV*, 100(2):134–153, November 2012.

[74] T. Ideker, O. Ozier, B. Schwikowski, and A. F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 2002.

[75] N. Ikizler-Cinbis, R. G. Cinbis, and S. Sclaroff. Learning actions from the web. In *ICCV*, 2009.

[76] N. Ikizler-Cinbis and S. Sclaroff. Object, scene and actions: Combining multiple features for human action recognition. In *ECCV*, 2010.

[77] L. Itti, C. Koch, and E. Niebur. A Model of Saliency-based Visual Attention for Rapid Scene Analysis. *TPAMI*, 20(11), November 1998.

[78] L. Jacob, J. philippe Vert, and F. R. Bach. Clustered multi-task learning: a convex formulation. In *NIPS*, 2008.

[79] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, 2014.

[80] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`, 2014.

[81] I. Junejo, E. Dexter, I. Laptev, and P. Perez. Cross-view action recognition from temporal self-similarities. In *ECCV*, 2008.

[82] S. B. Kang. A survey of image-based rendering techniques. In *Videometrics SPIE Intl Symp on Elec Imag: Science and Technology*, 1999.

[83] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[84] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *ICCV*, 2005.

[85] H. Kjellstrom, J. Romero, D. M. Mercado, and D. Kragic. Simultaneous visual recognition of manipulation actions and manipulated objects. In *ECCV*, 2008.

[86] A. Kläser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC*, 2008.

[87] A. Kläser, M. Marszałek, C. Schmid, and A. Zisserman. Human focused action localization in video. In *International Workshop on Sign, Gesture, Activity*, 2010.

[88] O. Kliper-Gross, T. Hassner, and L. Wolf. The action similarity labeling challenge. In *PAMI*, 2012.

[89] H. Koppula and A. Saxena. Anticipating human activities using object affordances for reactive robotic response. In *RSS*, 2013.

[90] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[91] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[92] H. Kuehne, H. Jhuang, E. Garrote, T. A. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, 2011.

[93] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. Berg, and T. Berg. Baby talk: Understanding and generating image descriptions. In *CVPR*, 2011.

[94] P. Kuznetsova, V. Ordonez, A. C. Berg, T. L. Berg, and Y. Choi. Collective generation of natural image descriptions. In *ACL*, 2012.

[95] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[96] I. Laptev. On space-time interest points. *IJCV*, 64(2):107–123, 2005.

[97] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008.

[98] I. Laptev and P. Prez. Retrieving actions in movies. In *ICCV*, 2007.

[99] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

[100] C.-S. Lee and A. Elgammal. Human motion synthesis by motion manifold learning and motion primitive segmentation. In *AMDO*, 2006.

[101] Y. J. Lee, J. Kim, and K. Grauman. Key-Segments for Video Object Segmentation. In *ICCV*, 2011.

[102] R. Li and T. Zickler. Discriminative virtual views for cross-view action recognition. In *CVPR*, 2012.

[103] J. J. Lim, R. Salakhutdinov, and A. Torralba. Transfer learning by borrowing examples for multiclass object detection. In *NIPS*, 2002.

[104] T. 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.

[105] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. In *ICCV*, 2009.

[106] J. Liu, M. Shah, B. Kuipers, and S. Savarese. Cross-view action recognition via view knowledge transfer. In *CVPR*, 2011.

[107] Q. Liu and X. Cao. Action recognition using subtensor constraint. In *ECCV*, 2012.

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

[109] I. Ljubic, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Math. Prog.*, 2006.

[110] S. Maji, L. Bourdev, and J. Malik. Action recognition from a distributed representation of pose and appearance. In *CVPR*, 2011.

[111] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011.

[112] M. Marin-Jimenez, A. Zisserman, and V. Ferrari. Here's looking at you kid. detection people looking at each other in videos. In *BMVC*, 2011.

[113] B. Marlin. Modeling user rating profiles for collaborative filtering. In *NIPS*, 2003.

[114] P. Matikainen, R. Sukthankar, and M. Hebert. Feature seeding for action recognition. In *ICCV*, 2011.

[115] K. Mikolajczyk and H. Uemura. Action recognition with motion-appearance vocabulary forest. In *CVPR*, 2008.

[116] D. Moore, I. Essa, and M. Hayes. Exploiting human actions and object context for recognition tasks. In *CVPR*, 1999.

[117] N. R. nd I. Reid. Estimating gaze direction from low-resolution faces in video. In *ECCV*, 2006.

[118] M. C. nd L. Bazzani, G. Paggetti, A. Fossati, A. D. Bue, G. Menegaz, and V. Murino. Social interaction discovery by statistical analysis of f-formations. In *BMVC*, 2011.

[119] J. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *IJCV*, 2008.

[120] D. Oneata, J. Verbeek, and C. Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In *ICCV 2013 - IEEE International Conference on Computer Vision*, pages 1817–1824, Sydney, Australia, Dec. 2013. IEEE.

[121] V. Ordonez, J. Deng, Y. Choi, A. Berg, and T. Berg. From large scale image categorization to entry-level categories. In *ICCV*, 2013.

[122] V. Ordonez, G. Kulkarni, and T. L. Berg. Im2text: Describing images using 1 million captioned photographs. In *NIPS*, 2011.

[123] C. Papageorgiou and T. Poggio. A trainable system for object detection. *IJCV*, 38(1):15–33, 2000.

[124] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: A method for automatic evaluation of machine translation. In *ACL*, 2002.

[125] V. Parameswaran and R. Chellappa. View invariance for human action recognition. *IJCV*, 66(1):83–101, 2006.

[126] H. Park and J. Shi. Social saliency. In *CVPR*, 2015.

[127] P. Peursum, G. West, and S. Venkatesh. Combining image regions and human activity for indirect object recognition in indoor wide-angle views. In *ICCV*, 2005.

[128] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Bilinear classifiers for visual recognition. In *NIPS*, 2009.

[129] H. Pirsiavash, C. Vondrick, and A. Torralba. Inferring the why in images. In *Workshop on Vision Meets Cognition at CVPR*, 2014.

[130] L. Pishchulin, A. Jain, C. Wojek, T. Thormaehlen, and B. Schiele. In good shape: Robust people detection based on appearance and shape. In *BMVC*, 2011.

[131] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012.

[132] A. Prest, C. Schmid, and V. Ferrari. Weakly supervised learning of interactions between humans and objects. *PAMI*, 34(3):601–614, March 2012.

[133] A. Quattoni, M. Collins, and T. Darrell. Transfer learning for image classification with sparse prototype representations. In *CVPR*, 2008.

[134] D. Ramanan and D. Forsyth. Automatic annotation of everyday movements. In *NIPS*, 2003.

[135] C. Rao and M. Shah. View-invariance in action recognition. In *CVPR*, 2001.

[136] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *IJCV*, 50(2):203–226, 2002.

[137] A. Recasens, A. Khosla, C. Vondrick, and A. Torralba. Where are they looking? In *NIPS*, 2015.

[138] M. D. Rodriguez, J. Ahmed, and M. Shah. Action MACH a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008.

[139] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. In *Science*, volume 290, pages 2323–2326, December 2000.

[140] A. Sadovnik, Y.-I. Chiu, N. Snavely, S. Edelman, and T. Chen. Image description with a goal: Building efficient discriminating expressions for images. In *CVPR*, 2012.

[141] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS*, 2007.

[142] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*, 2008.

[143] S. Satkin and M. Hebert. Modeling the temporal extent of actions. In *ECCV*, 2010.

[144] S. Savarese and L. Fei-Fei. View synthesis for recognizing unseen poses of object classes. In *ECCV*, 2008.

[145] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *ICPR*, 2004.

[146] S. Seitz and C. Dyer. View morphing. In *SIGGRAPH*, 1996.

[147] G. Shakhnarovich, L. Lee, and T. Darrell. Integrated face and gait recognition from multiple views. In *CVPR*, 2001.

[148] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *ICCV*, 2003.

[149] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *CVPR*, 2011.

[150] M. Spain and P. Perona. Some objects are more equal than others: Measuring and predicting importance. In *ECCV*, 2008.

[151] K. Stratos, A. Sood, A. Mensch, X. Han, M. Mitchell, K. Yamaguchi, J. Dodge, A. Goyal, H. Daumé III, A. C. Berg, and T. L. Berg. Understanding and predicting importance in images. In *CVPR*, 2012.

[152] N. C. Tang, C.-T. Hsu, T.-Y. Lin, and H.-Y. M. Liao. Example-based human motion extrapolation based on manifold learning. In *ICME*, 2011.

[153] G. W. Taylor, I. Spiro, C. Bregler, and R. Fergus. Learning invariance through imitation. In *CVPR*, 2011.

[154] T. Tommasi, F. Orabona, and B. Caputo. Safety in numbers: learning categories from few examples with multi model knowledge transfer. In *CVPR*, 2010.

[155] A. Torralba. Contextual priming for object detection. *IJCV*, 53(2):169–191, 2003.

[156] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *CVPR*, 2010.

[157] S. Vijayanarasimhan and K. Grauman. Efficient region search for object detection. In *CVPR*, 2011.

[158] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *CVPR*, 2001.

[159] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM Tran on Graphics*, 24(3):426–433, 2005.

[160] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba. HOGgles: Visualizing object detection features. In *ICCV*, 2013.

[161] C. Vondrick, D. Ramanan, and D. Patterson. Efficiently scaling up video annotation with crowdsourced marketplaces. In *ECCV*, 2010.

[162] G. Wang, D. A. Forsyth, and D. Hoiem. Comparative object similarity for improved recognition with few or no examples. In *CVPR*, 2010.

[163] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action Recognition by Dense Trajectories. In *CVPR*, 2011.

[164] H. Wang, M. M. Ullah, A. Klser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009.

[165] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *TPAMI*, pages 283–298, Feb 2008.

[166] D. Weinland, E. Boyer, and R. Ronfard. Action recognition from arbitrary views using 3D exemplars. In *ICCV*, 2007.

[167] D. Weinland, M. Ozuysal, and P. Fua. Making action recognition robust to occlusions and viewpoint changes. In *ECCV*, 2010.

[168] G. Willems, J. Becker, T. Tuytelaars, and L. V. Gool. Exemplar-based action recognition in video. In *BMVC*, 2009.

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

[170] L. Xiong, X. Chen, T. Huang, J. Schneider, and J. Carbonell. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In *SDM*, 2010.

[171] P. Yan, S. Khan, and M. Shah. Learning 4D action feature models for arbitrary view action recognition. In *CVPR*, 2008.

[172] J. Yang, R. Yan, and A. Hauptmann. Cross-domain video concept detection using adaptive svms. In *ACM Multimedia*, 2007.

[173] W. Yang, Y. Wang, and G. Mori. Recognizing human actions from still images with latent poses. In *CVPR*, 2010.

[174] Y. Yang, S. Baker, A. Kannan, and D. Ramanan. Recognizing proxemics in personal photos. In *CVPR*, 2012.

[175] A. Yao, J. Gall, and L. van Gool. A Hough transform-based voting framework for action recognition. In *CVPR*, 2010.

[176] B. Yao and L. Fei-Fei. Grouplet: A structured image representation for recognizing human and object interactions. In *CVPR*, 2010.

[177] B. Yao and L. Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *CVPR*, 2010.

[178] B. Yao and L. Fei-Fei. Action recognition with exemplar based 2.5d graph matching. In *ECCV*, 2012.

[179] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. J. Guibas, and L. Fei-Fei. Action recognition by learning bases of action attributes and parts. In *ICCV*, 2011.

[180] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *The Twenty-Fourth IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

[181] B. Yao, X. Yang, L. Lin, M. Lee, and S.-C. Zhu. I2T: Image parsing to text description. *Proceedings of the IEEE*, 98(8), August 2010.

[182] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014.

[183] G. Yu, J. Yuan, and Z. Liu. Unsupervised random forest indexing for fast action search. In *CVPR*, 2011.

[184] J. Yuan, Z. Liu, and Y. Wu. Discriminative subvolume search for efficient action detection. In *CVPR*, 2009.

[185] Z. Zhang, C. Wang, B. Xiao, W. Zhou, S. Liu, and C. Shi. Cross-view action recognition via a continuous virtual path. In *CVPR*, 2013.

[186] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.

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

Vita

Chao-Yeh Chen was born in Taipei, Taiwan. He attended the National Experimental High School in Hsinchu, from which he received his high school diploma in 2001. He then entered the National Chiao-Tung University, Taiwan, where he received the degree of Bachelor of Science in Electronics engineering in 2005. The same year, he served as an electrical officer for mandatary military service for 15 months and worked as a full time teaching assistant in National Chiao-Tung University for 18 months. In 2008, he entered the Graduate School at the University of Texas at Austin and joined the Computer Vision Group headed by Prof. Kristen Grauman as a graduate research assistant. In 2014, he was an intern at the media analytics group at Nec Labs America.

Permanent contact: chaoyehchen@gmail.com

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.