DOCTORAL THESIS

Humans In Their Natural Habitat: Training AI to Understand People

Gunnar A. Sigurdsson
The Robotics Institute
Carnegie Mellon University

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

July 2020

Tech Report Number: CMU-RI-TR-20-23

Abstract

Computer vision has a great potential to help our daily lives by searching for lost keys, watering flowers or reminding us to take a pill. To succeed with such tasks, computer vision methods need to be trained from real and diverse examples of our daily dynamic scenes. First, we need to give computers insight into our world, and our daily lives. Not just through the charade the we present to the world on social media, but through a genuine look at the most boring, mundane, routine aspects of our lives. But how do we model this data? How do we model information over time? How do we harness the richness and complexity of this data to enable understanding?

To provide a lens through which to look at humans in their mundane lives, we explored techniques for crowdsourcing the creation of this data from hundreds of people in their own homes, and analyzed how humans think about activities along with the best strategies for annotating complex data of this nature. Given this insight into human behaviour, we can start understanding where other vision techniques have trouble, understand how to improve them, and which venues are most promising moving forward.

Once we have this kind of data, we can start building algorithms that harness the unique aspects of this data by learning how human activities change over time, and what activities occur with a recognizable temporal structure. We can harness the data to learn how complete human events generally unfold, such as a snowboarding trip, and apply these models to applied problems such as summarizing photo albums. Finally, we combine ideas from our work to demonstrate how these techniques can be used to collect data and modeling human activities from first and third-person at the same time, and unsupervised concept learning from web videos. We hope this kind of realistic bias may provide new insights that aid robots equipped with our computer vision models operating in the real world.

Acknowledgements

I want to thank my advisor, Abhinav Gupta, for his guidance and support throughout my PhD. His energetic and friendly nature continues to inspire his students. I am forever thankful for his wisdom on choosing research problems and pursuing ideas that I am passionate about, and creating and sustaining an environment where I could research anything.

I was fortunate to also have the advice and guidance from numerous other advisors during my PhD. Li Jia, for teaching me scale. Xinlei Chen, for teaching me research coding. Ivan Laptev, for teaching me videos. Ali Farhadi, for teaching me to organize my thoughts and my work. Olga Russakovsky, for teaching me scientific thinking and how to write. Karteek Alahari, for helping me believe in my work. Cordelia Schmid, for teaching me to make things that work. Andrew Zisserman, teaching me high impact research. To my collaborators: Abhinav Gupta, Xinlei Chen, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, Olga Russakovsky, Santosh Divvala, Cordelia Schmid, Karteek Alahari, Jean-Baptiste Alayrac, Aida Nematzadeh, Lucas Smaira, Mateusz Malinowski, João Carreira, Phil Blunsom, Andrew Zisserman. Thank you for your help, your knowledge, and teaching me. And a thank you to all the workers on Amazon Mechanical Turk—I strive to work as hard as you.

I want to thank everyone who got me to where I am today:

- to my family, for teaching me, supporting me, and encouraging me.
- to all my teachers, for your help, your patience, and teaching me to learn, and Ragnar Geir Brynjólfsson for guiding me through my early steps in programming.
- to George Claassen, Júlíana Rún Indriðadóttir, Jón Hrólfur Sigurjónsson, Hróðmar Ingi Sigurbjörnsson, and all my music teachers, for teaching me how to be creative.
- to Magnús Örn Úlfarsson, Jóhannes Rúnar Sveinsson, Helgi Þorbergsson, Gunnlaugur Þór Briem, Einar Örn Sveinbjörnsson, Hrefna Marín Gunnarsdóttir, Páll Melsted, and all my collage professors, for showing me what is possible.
- to Steven Bowers and Ali Hajimiri, your guidance ignited my interest in research.
- to Jerry L. Prince, for believing in me, and teaching me how to do research.
- to René Vidal, John Goutsias, and Trac D. Tran, and the IACL reading group for introducing me to Computer Vision.
- to Suchi Saria, for teaching me that "no research happens in a vacuum" and how to discover and develop ideas.
- to Kris Kitani, Deva Ramanan, Martial Hebert, and all my teachers at CMU who supported me and gave me advice always when I needed.
- to Peter Siska, Mark Rodosky, Ata Murat Kaynar, Jason Hoellwarth, Gele Moloney, Hamza Shaikh, Chris Losego, Kim Maguire, Steve Odezenne, Fred, Jonathan, Adam, their teams, and everyone who pieced me back together and helped me get better.
- to Miriam, for your support.

My labmates, and office neighbors: Carl Doersch, David Fouhey, Abhinav Srivastava, Xinlei Chen, Jacob Walker, Ishan Misra, Xiaolong Wang, Lerrel Pinto, Yin Li, Rohit Girdhar, Olga Russakovsky, Pavel Tokmakov, Dhiraj Gandhi, Tian Ye, Nilesh Kulkarni, Tao Chen, Wenxuan Zhou, Nadine Chang, Kenneth Marino, Adithya Murali, Senthil Purushwalkam, Sam Powers, Victoria Dean, Helen Jiang, Sudeep Dasari, Shikhar Bahl, Yufei Ye, Aayush Bansal, Achal Dave, Brian Okorn, Pratyusha Sharma, Deepthi Hegde, Kit Ham, Nick Rhinehart, Allie Del Giorno, Ruta Desai, Kumar Shaurya Shankar, Preeti Sar, Arun Venkatraman, Jack Valmadre, Hilton Bristow, Matt Barnes, Vincent Chu, Xuehan Xiong, Nicolas Chesneau, Vicky Kalogeiton, Konstantin Shmelkov, Nikita Dvornik, Valentin Gabeur, Dexiong Chen, Andrei Kulunchakov, Alexander Pashevich, Vladyslav Sydorov, Alberto Bietti, and everyone who shared in my ups and downs in research. Suzanne Lyons Muth, Jessica Butterbaugh, Christine Downey, Nathalie Gillot, and everyone that was always there to help.

Chris, Lucia, Jessica, Katie, Nick, Kelvin, Nick, Eir, Lottie, Matt, and my friends in Pittsburgh, Seattle, Grenoble, and London. Thank you for your camaraderie, your company made all of this possible.

Finally, to my family, to Afi Gunnar, who encouraged our education, and Amma Nanna, who believed I could do anything.

Contents

Chapter 1

Introduction

One of the great promises of AI and Robotics is to bring more and more computation and machines into our daily lives and routines. This will enable machines to assist us and work among us. However, a fundamental challenge for enabling machines that live among us, is creating machines that understand us. Where do we start? First, we need to give computers insight into our world, and our daily lives. Not just through the charade the we present to the world on social media, but through a genuine look at the most boring, mundane, routine aspects of our lives. Once we have data that we can give to the computer, the questions are: How do we model this data? How do we model information over time? How do we harness the richness and complexity of this data to enable understanding? For example, looking at Figure 1.1 we can see that our daily activities do not occur in isolation, and tend to be very semantically meaningful, even though the change in appearance is subtle. We perform a sequence of activities, usually with a purpose—an intent. Modelling the complete nature of the activities is an important problem that we can tackle with such a data.

Once we have this kind of data, we can start building algorithms that harness the unique aspects of this data by learning how human activities change over time, and what activities occur with a recognizable temporal structure [217]. We can harness the data to learn how complete human events generally unfold, such as a snowboarding trip, and apply these models to applied problems such as summarizing photo albums [215]. We can combine ideas from our work to demonstrate how these techniques can be used to collect data and modelling human activities from first and third-person at the same time [219]. We hope this kind of realistic bias may provide new insights that aid robots equipped with our computer vision models operating in the real world. Finally, to expand the understanding of our models beyond labelled data, we have explored multi-modal ideas for unsupervised learning of language and translation from hundreds of millions of YouTube video clips [214].

Understanding humans, is as elusive of a task as it is to define. Even for the complexities of human intelligence, even relatively minor damage can significantly impair the ability to understand human behavior. This suggests that simple observations of humans may not be sufficient for full understanding, and if we really want our systems to understand humans we need to teach them to put themselves in our shoes. To fully understand human culture and humans themselves, it may be needed to go beyond building a complete dataset that captures all possible concepts. Thus, venturing forth away from prepared datasets is eventually needed to expand the knowledge of the model. We explore this idea of learning

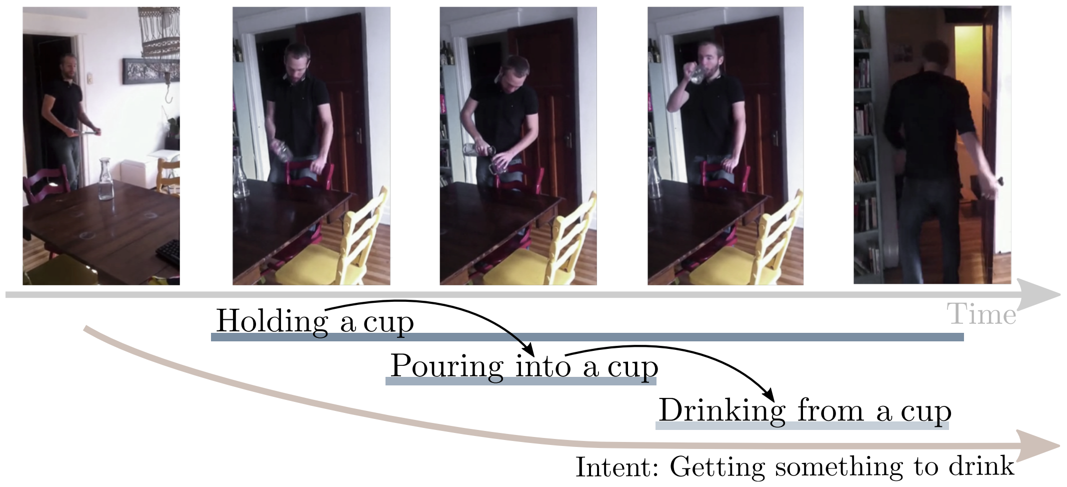


Figure 1.1: Understanding human activities requires understanding the subtle sequence of motions and tasks that make up daily human activities, and how they are tied together with the person's intentions.

concepts in an unsupervised manner from videos uploaded by people to the web.

In this thesis, we explore the full pipeline from acquiring data, building systems using the data, to analyzing the high-level of what these systems are able to do. The thesis is organized into three parts: (I) Getting Data. (II) Building Algorithms. (III) Understanding Humans. These parts are summarized below.

I Data for Activity Understanding. The first part explores how to provide the data for learning about human activities. We explored a new lens through which to look at humans in their daily lives, using techniques for crowdsourcing the creation of this data from hundreds of people in their own homes. Prompted by this novel data, we analyzed how humans think about activities and the best strategies for annotating complex data of this nature. This insight into human behaviour, allowes to to start investigating where other vision techniques have trouble, understand how to improve them, and which venues are most promising moving forward.

II Algorithms for Video Understanding. In the second part, once we have this kind of data, we can start building algorithms that harness the unique aspects of this data by learning how human activities change over time, and what activities occur with a recognizable temporal structure. On an even higher level, we can harness the data to learn how complete human events, such as a snowboarding trip, generally unfold, and apply these models to practical problems such as summarizing photo albums. Building systems that process chaotic real world video data, requires using 3D information, and allocating resolution to where it is needed. Incorporating camera motion into all layers of a network allows for synthetic "eye movements" that can address these problems.

III Understanding Humans from Their Perspective. Finally, in the third part, we combine ideas from our work to demonstrate how these techniques can be used to collect data and modelling human activities from first and third-person at the same time. To extend the understanding from a labelled data to arbitrary human concepts, we demonstrate how concepts can be learned from watching unlabelled videos and that these concepts are consistent across languages.

Concretely, the contributions of this dissertations are: (1) The Charades and Charades-Ego datasets for training and evaluating human activity recognition models, along with techniques for extensive annotation and analysis of such data. (2) Techniques for modeling human activities in videos and photo albums over long time scales. (3) Human inspired techniques for modeling video from first/third person perspectives and world coordinate viewpoints. (4) A method for unsupervised translation using uncurated web videos. These contributions were previously described in [214–223] and are briefly outlined below.

Overview

In Chapter 2 we propose a novel Hollywood in Homes approach to collect any data. Instead of shooting videos in the lab, we ensure diversity by distributing and crowdsourcing the whole process of video creation from script writing to video recording and annotation. Following this procedure we collect a new dataset, *Charades*, with hundreds of people recording videos in their own homes, acting out casual everyday activities. The dataset is composed of 9,848 annotated videos with an average length of 30 seconds.

Chapter 3 then investigates and determines the most cost-effective way of obtaining high-quality multi-label annotations for temporal data such as videos. Watching even a short 30-second video clip requires a significant time investment from a crowd worker; thus, requesting multiple annotations following a single viewing is an important cost-saving strategy. But how many questions should we ask per video? We demonstrate the effectiveness of our method by collecting multi-label annotations of 157 human activities on 1,815 videos.

This then allows us, in Chapter 4, to analyze the current state of human activity understanding in videos, where we examine datasets, evaluation metrics, algorithms, and potential future directions. We look at the qualitative attributes that define activities such as pose variability, brevity, and density.

In Chapter 5 we consider that a thorough understanding of videos requires going beyond appearance modeling and necessitates reasoning about the sequence of activities, as well as the higher-level constructs such as intentions. We propose a fully-connected temporal CRF model for reasoning over various aspects of activities that includes objects, actions, and intentions, where the potentials are predicted by a deep network. To address this challenge, we present an asynchronous variational inference method that allows efficient end-to-end training.

Pushing long-term temporal reasoning to its limits, we explore in Chapter 6 how to automatically learn the temporal aspects, or storylines of visual concepts from web data. Our novel Skipping Recurrent Neural Network (S-RNN) model does not attempt to predict each and every data point in the sequence, like classic RNNs. Rather, S-RNN uses a framework that skips through the images in the photo stream to explore the space of all ordered subsets of the albums via an efficient sampling procedure. We show how our learned storylines can be used to analyze, predict, and summarize photo albums from Flickr.

Exploring space in addition to time, we introduce the idea of WorldFeatures in Chapter 7, where each feature at every layer has a spatial transformation, and the feature map is only transformed as needed. We show that a network built with these WorldFeatures, can be used to model eye movements, such as saccades, fixation, and smooth pursuit, even in a batch setting on pre-recorded video. That is, the network can for example use all 224 by 224 pixels to look at a small detail one moment, and the whole scene the next. We show that typical building blocks, such as convolutions and pooling, can be adapted to support WorldFeatures using available tools.

In Chapter 8 we introduce Charades-Ego, a large-scale dataset of paired first-person and third-person videos, involving 112 people, with 4000 paired videos. This enables learning the link between the actor and observer perspectives. We use this data to learn a joint representation of first and third-person videos, with only weak supervision, and show its effectiveness for transferring knowledge from the third-person to the first-person domain.

Finally, Chapter 9 demonstrates that given powerful video modeling we can now establish a common visual representation between two languages by learning embeddings from *unpaired* instructional videos narrated in the native language. Given this shared embedding we demonstrate that (i) we can map words between the languages, particularly the 'visual' words; (ii) that the shared embedding provides a good initialization for existing unsupervised text-based word translation techniques, forming the basis for our proposed hybrid visual-text mapping algorithm, MUVE; and (iii) our approach achieves superior performance by addressing the shortcomings of text-based methods – it is more robust, handles datasets with less commonality, and is applicable to low-resource languages.

Following is the relevant publication list for each chapter.

Ch. 2 Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding [223] (ECCV 2016)

Ch. 3 Much Ado About Time: Exhaustive Annotation of Temporal Data [221] (HCOMP 2016)

Ch. 4 What Actions are Needed for Understanding Human Actions in Videos? [222] (ICCV 2017)

Ch. 5 Asynchronous Temporal Fields for Action Recognition [217] (CVPR 2017)

Ch. 6 Learning Visual Storylines with Skipping Recurrent Neural Networks [215] (ECCV 2016)

Ch. 7 Beyond the Camera: Neural Networks in World Coordinates [218] (ArXiv 2020)

Ch. 8 Actor and Observer: Joint Modeling of First and Third-Person Video [219] (CVPR 2018)

Ch. 9 Visual Grounding in Video for Unsupervised Word Translation [214] (CVPR 2020)

Part I

Data for Activity Understanding

The first step in most modern AI models is *data*. A model cannot be expected to learn information that is not available, and even generalizing from current data is an unsolved problem. If we want to learn about *humans*, then we need insight into *human lives*. Getting that data requires cleverly harnessing the data humans put on the internet, but also acquiring the data that is not put on the internet. For example, the really boring stuff. We explore these ideas in Chapter 2 and Chapter 3. Utilizing the only expert human understanding that currently exists—*humans*, we need to identify what it means to understand, and what steps are needed to get to that level with our systems. These ideas are explored in Chapter 3 and Chapter 4.

Chapter 2

Crowdsourcing Video Collection

Computer vision has a great potential to help our daily lives by searching for lost keys, watering flowers or reminding us to take a pill. To succeed with such tasks, computer vision methods need to be trained from real and diverse examples of our daily dynamic scenes. While most of such scenes are not particularly exciting, they typically do not appear on YouTube, in movies or TV broadcasts. So how do we collect sufficiently many diverse but *boring* samples representing our lives? We propose a novel Hollywood in Homes approach to collect such data. Instead of shooting videos in the lab, we ensure diversity by distributing and crowdsourcing the whole process of video creation from script writing to video recording and annotation. Following this procedure we collect a new dataset, *Charades*, with hundreds of people recording videos in their own homes, acting out casual everyday activities. The dataset is composed of 9,848 annotated videos with an average length of 30 seconds, showing activities of 267 people from three continents, and over 15% of the videos have more than one person. Each video is annotated by multiple free-text descriptions, action labels, action intervals and classes of interacted objects. In total, Charades provides 27,847 video descriptions, 66,500 temporally localized intervals for 157 action classes and 41,104 labels for 46 object classes. Using this rich data, we evaluate and provide baseline results for several tasks including action recognition and automatic description generation. We believe that the realism, diversity, and casual nature of this dataset will present unique challenges and new opportunities for computer vision community.

2.1 Background

Large scale visual learning fueled by huge datasets has changed the computer vision landscape [36, 299]. Given the source of this data, it's not surprising that most of our current success is biased towards static scenes and objects in Internet images. As we move forward into the era of AI and robotics, however, new questions arise. How do we learn about different states of objects (*e.g.*, cut vs. whole)? How do common activities affect changes of object states? In fact, it is not even yet clear if the success of the Internet pre-trained recognition models will transfer to real-world settings where robots equipped with our computer vision models should operate.

Shifting the bias from Internet images to real scenes will most likely require collection of new large-scale datasets representing activities of our boring everyday life: getting up, getting dressed, putting groceries in fridge, cutting vegetables and so on. Such datasets will allow us to develop new representations and to learn models with the right biases. But more importantly, such datasets representing people interacting with objects and performing natural action sequences in typical environments will finally allow us to learn common sense and contextual knowledge necessary for high-level reasoning and modeling.

But how do we find these boring videos of our daily lives? If we search common activities such as "drinking from a cup", "riding a bike" on video sharing websites such as YouTube, we observe a highly-biased sample of results (see Figure 2.1). These results are biased towards entertainment—boring videos have no viewership and hence no reason to be uploaded on YouTube!

We propose a novel **Hollywood in Homes** approach to collect a large-scale dataset of boring videos of daily activities. Standard approaches in the past have used videos downloaded from the Internet [50,68,106,124,139,232] gathered from movies [128,185,187] or recorded in controlled environments [69,123,165,186,189,205]. Instead, as the name suggests: we take the Hollywood filming process to the homes of hundreds of people on Amazon Mechanical Turk (AMT). AMT workers follow the three steps of filming process: (1) script generation; (2) video direction and acting based on scripts; and (3) video verification to create one of the largest and most diverse video dataset of daily activities.

There are threefold advantages of using the **Hollywood in Homes** approach for dataset collection: (a) Unlike datasets shot in controlled environments (*e.g.*, MPII [189]), crowdsourcing brings in diversity which is essential for generalization. In fact, our approach even allows the same script to be enacted by multiple people; (b) crowdsourcing the script writing enhances the coverage in terms of scenarios and reduces the bias introduced by generating scripts in labs; and (c) most importantly, unlike for web videos, this approach allows us to control the composition and the length of video scenes by proposing the vocabulary of scenes, objects and actions during script generation.

The Charades v1.0 Dataset

Charades is our large-scale dataset with a focus on common household activities collected using the Hollywood in Homes approach. The name comes from of a popular American word guessing game where one player acts out a phrase and the other players guess what phrase it is. In a similar spirit, we recruited hundreds of people from Amazon Mechanical Turk to act out a paragraph that we presented to them. The workers additionally provide action classification, localization, and video description annotations. The first publicly released version of our *Charades* dataset will contain 9,848 videos of daily activities 30.1 seconds long on average (7,985 training and 1,863 test). The dataset is collected in 15 types of indoor scenes, involves interactions with 46 object classes and has a vocabulary of 30 verbs leading to 157 action classes. It has 66,500 temporally localized actions, 12.8 seconds long on average, recorded by 267 people in three continents. We believe this dataset will provide a crucial stepping stone in developing action representations, learning object states, human object interactions, modeling context, object detection in videos, video captioning and many more. The dataset will be publicly available at `http://allenai.org/plato/charades/`.

Contributions The contributions of our work are three-fold: (1) We introduce the Hollywood in Homes approach to data collection, (2) we collect and release the first crowdsourced large-scale dataset of boring household activities, and (3) we provide extensive baseline evaluations.

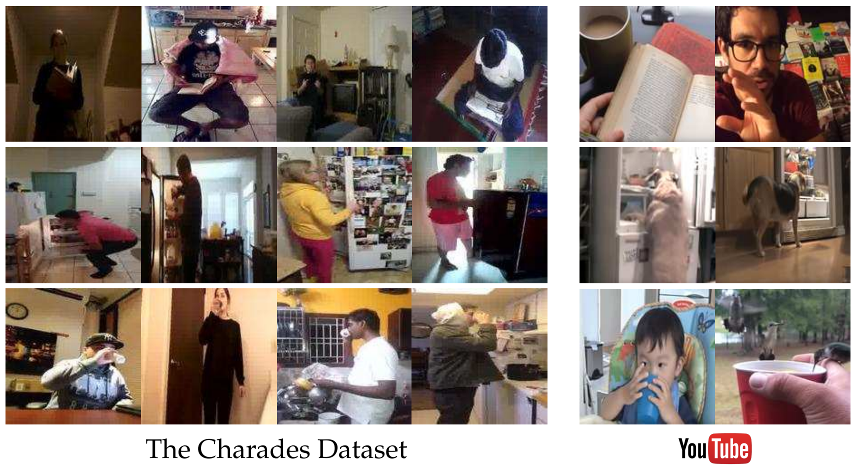


Figure 2.1: Comparison of actions in the Charades dataset and on YouTube: *Reading a book*, *Opening a refrigerator*, *Drinking from a cup*. YouTube returns entertaining and often atypical videos, while *Charades* contains typical everyday videos.

The KTH action dataset [205] paved the way for algorithms that recognized human actions. However, the dataset was limited in terms of number of categories and enacted in the same background. In order to scale up the learning and the complexity of the data, recent approaches have instead tried collecting video datasets by downloading videos from Internet. Therefore, datasets such as UCF101 [232], Sports1M [106] and others [68, 124, 139] appeared and presented more challenges including background clutter, and scale. However, since it is impossible to find boring daily activities on Internet, the vocabulary of actions became biased towards more sports-like actions which are easy to find and download.

There have been several efforts in order to remove the bias towards sporting actions. One such commendable effort is to use movies as the source of data [58, 147]. Recent papers have also used movies to focus on the video description problem leading to several datasets such as MSVD [19], M-VAD [242], and MPII-MD [187]. Movies however are still exciting (and a source of entertainment) and do not capture the scenes, objects or actions of daily living. Other efforts have been to collect in-house datasets for capturing human-object interactions [72] or human-human interactions [196]. Some relevant big-scale efforts in this direction include MPII Cooking [189], TUM Breakfast [123], and the TACoS Multi-Level [186] datasets. These datasets focus on a narrow domain by collecting the data in-house with a fixed background, and therefore focus back on the activities themselves. This allows for careful control of the data distribution, but has limitations in terms of generalizability, and scalability. In contrast, PhotoCity [246] used the crowd to take pictures of landmarks, suggesting that the same could be done for other content at scale.

Another relevant effort in collection of data corresponding to daily activities and objects is in the domain of ego-centric cameras. For example, the Activities of Daily Living dataset [173] recorded 20 people performing unscripted, everyday activities in their homes in first person, and another extended that idea to animals [93]. These datasets provide a challenging task but fail to provide diversity which is crucial for generalizability. It should however be noted that these kinds of datasets could be crowdsourced similarly to our work.

	Actions per video	Classes	Labelled instances	Total videos	Origin	Type	Temporal localization
Charades v1.0	6.8	157	67K	10K	267 Homes	Daily Activities	Yes
ActivityNet [50]	1.4	203	39K	28K	YouTube	Human Activities	Yes
UCF101 [232]	1	101	13K	13K	YouTube	Sports	No
HMDB51 [124]	1	51	7K	7K	YouTube/Movies	Movies	No
THUMOS'15 [68]	1-2	101	21K+	24K	YouTube	Sports	Yes
Sports 1M [106]	1	487	1.1M	1.1M	YouTube	Sports	No
MPII-Cooking [189]	46	78	13K	273	30 In-house actors	Cooking	Yes
ADL [173]	22	32	436	20	20 Volunteers	Ego-centric	Yes
MPII-MD [187]	Captions	Captions	68K	94	Movies	Movies	No

Table 2.1: Comparison of Charades with other video datasets.

The most related dataset is the recently released ActivityNet dataset [50]. It includes actions of daily living downloaded from YouTube. We believe the ActivityNet effort is complementary to ours since their dataset is uncontrolled, slightly biased towards non-boring actions and biased in the way the videos are professionally edited. On the other hand, our approach focuses more on action sequences (generated from scripts) involving interactions with objects. Our dataset, while diverse, is controlled in terms of vocabulary of objects and actions being used to generate scripts. In terms of the approach, Hollywood in Homes is also related to [303]. However, [303] only generates synthetic data. A comparison with other video datasets is presented in Table 2.1. To the best of our knowledge, our approach is the first to demonstrate that workers can be used to collect a vision dataset by filming themselves at such a large scale.

2.2 Hollywood in Homes

We now describe the approach and the process involved in a large-scale video collection effort via AMT. Similar to filming, we have a three-step process for generating a video. The first step is generating the script of the indoor video. The key here is to allow workers to generate diverse scripts yet ensure that we have enough data for each category. The second step in the process is to use the script and ask workers to record a video of that sentence being acted out. In the final step, we ask the workers to verify if the recorded video corresponds to script, followed by an annotation procedure.

2.2.1 Generating Scripts

We focus on indoor scenes, hence, we group together rooms in residential homes (*Living Room*, *Home Office*, etc.). We found 15 types of rooms to cover most of typical homes, these rooms form the scenes in the dataset. In order to generate the *scripts* (a text given to workers to act out in a video), we use a vocabulary of objects and actions to guide the process. To understand what objects and actions to include in this vocabulary, we analyzed 549 movie scripts from popular movies in the past few decades. Using both term-frequency (TF) and TF-IDF [203] we analyzed which nouns and verbs occur in those rooms in these movies. From those we curated a list of 40 objects and 30 actions to be used as seeds for script generation, where objects and actions were chosen to be generic for different scenes.

To harness the creativity of people, and understand their bias towards activities, we crowdsourced the script generation as follows. In the AMT interface, a single scene, 5 randomly selected objects, and 5 randomly selected actions were presented to workers. Workers were asked to use two objects and two actions to compose a short paragraph about activities of one or two people performing realistic and commonplace activities in their home. We found this to be a good compromise between controlling what kind of words were used and allowing the users to impose their own human bias on the generation. Some examples of generated scripts are shown in Figure 2.2. (see the website for more examples). The distribution of the words in the dataset is presented in Figure 2.3.

2.2.2 Generating Videos

Once we have scripts, our next step is to collect videos. To maximize the diversity of scenes, objects, clothing and behaviour of people, we ask the workers themselves to record the 30 second videos by following collected scripts.

AMT is a place where people commonly do quick tasks in the convenience of their homes or during downtime at their work. AMT has been used for annotation and editing but can we do content creation via AMT? During a pilot study we asked workers to record the videos, and until we paid up to \$3 per video, no worker picked up our task. (For comparison, to annotate a video [221]: 3 workers $\times$ 157 questions $\times$ 1 second per question $\times$ \$8/h salary = \$1.) To reduce the base cost to a more manageable \$1 per video, we have used the following strategies:

Worker Recruitment. To overcome the inconvenience threshold, worker recruitment was increased through sign-up bonuses (211% increased new worker rate) where we awarded a \$5 bonus for the first submission. This increased the total cost by 17%. In addition, "recruit a friend" bonuses (\$5 if a friend submits 15 videos) were introduced, and were claimed by 4% of the workforce, generating indeterminate outreach to the community. US, Canada, UK, and, for a time, India were included in this study. The first three accounted for estimated 73% of the videos, and 59% of the peak collection rate.

Worker Retention. Worker retention was mitigated through performance bonuses every 15th video, and while only accounting for a 33% increase in base cost, significantly increased retention (34% increase in come-back workers), and performance (109% increase in output per worker).

Each submission in this phase was manually verified by other workers to enforce quality control, where a worker was required to select the corresponding sentence from a line-up after watching the video. The rate of collection peaked at 1225 per day from 72 workers. The final cost distribution was: 65% base cost per video, 21% performance bonuses, 11% recruitment bonuses, and 3% verification. The code and interfaces will be made publicly available along with the dataset.

2.2.3 Annotations

Using the generated scripts, all (verb,proposition,noun) triplets were analyzed, and the most frequent grouped into 157 action classes (*e.g.*, *pouring into cup*, *running*, *folding towel*, etc.). The distribution of those is presented in Figure 2.3.

For each recorded video we have asked other workers to watch the video and describe

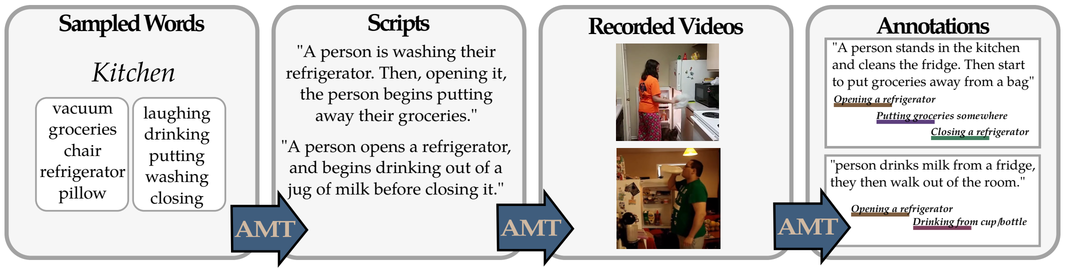

Sampled Words
Kitchen
vacuum
groceries
chair
refrigerator
pillow
laughing
drinking
putting
washing
closing
AMT
Scripts
"A person is washing their refrigerator. Then, opening it, the person begins putting away their groceries."
"A person opens a refrigerator, and begins drinking out of a jug of milk before closing it."
AMT
Recorded Videos
AMT
Annotations
"A person stands in the kitchen and cleans the fridge. Then start to put groceries away from a bag"
Opening a refrigerator
Putting groceries somewhere
Closing a refrigerator
"person drinks milk from a fridge, they then walk out of the room."
Opening a refrigerator
Drinking from cup/bottle

Figure 2.2: An overview of the three Amazon Mechanical Turk (AMT) crowdsourcing stages in the *Hollywood in Homes* approach.

what they have observed with a sentence (this will be referred to as a *description* in contrast to the previous *script* used to generate the video). We use the original script and video descriptions to automatically generate a list of interacted objects for each video. Such lists were verified by the workers. Given the list of (verified) objects, for each video we have made a short list of 4-5 actions (out of 157) involving corresponding object interactions and asked the workers to verify the presence of these actions in the video.

In addition, to minimize the number of missing labels, we expanded the annotation procedure to exhaustively annotate all actions in the video using state-of-the-art crowdsourcing practices [221], where we focused particularly on the test set.

Finally, for all the chosen action classes in each video, another set of workers was asked to label the starting and ending point of the activity in the video, resulting in a temporal interval of each action. A visualization of the data collection process is illustrated in Figure 2.2. On the website we show numerous additional examples from the dataset with annotated action classes.

2.3 Charades v1.0 Analysis

Charades is built up by combining 40 objects and 30 actions in 15 scenes. This relatively small vocabulary, combined with open-ended writing, creates a dataset that has substantial coverage of a useful domain. Furthermore, these combinations naturally form action classes that allow for standard benchmarking. In Figure 2.3 the distributions of action classes, and most common nouns/verbs/scenes in the dataset are presented. The natural world generally follows a long-tailed distribution [224,302], but we can see that the distribution of words in the dataset is relatively even. In Figure 2.3 we also present a visualization of what scenes, objects, and actions occur together. By embedding the words based on their co-occurance with other words using T-SNE [250], we can get an idea of what words group together in the videos of the dataset, and it is clear that the dataset possesses real-world intuition. For example, *food*, and *cooking* are close to *Kitchen*, but note that except for *Kitchen*, *Home Office*, and *Bathroom*, the scene is not highly discriminative of the action, which reflects common daily activities.

Since we have control over the data acquisition process, instead of using Internet search, there are on average 6.8 relevant actions in each video. We hope that this may inspire new and interesting algorithms that try to capture this kind of context in the domain of action

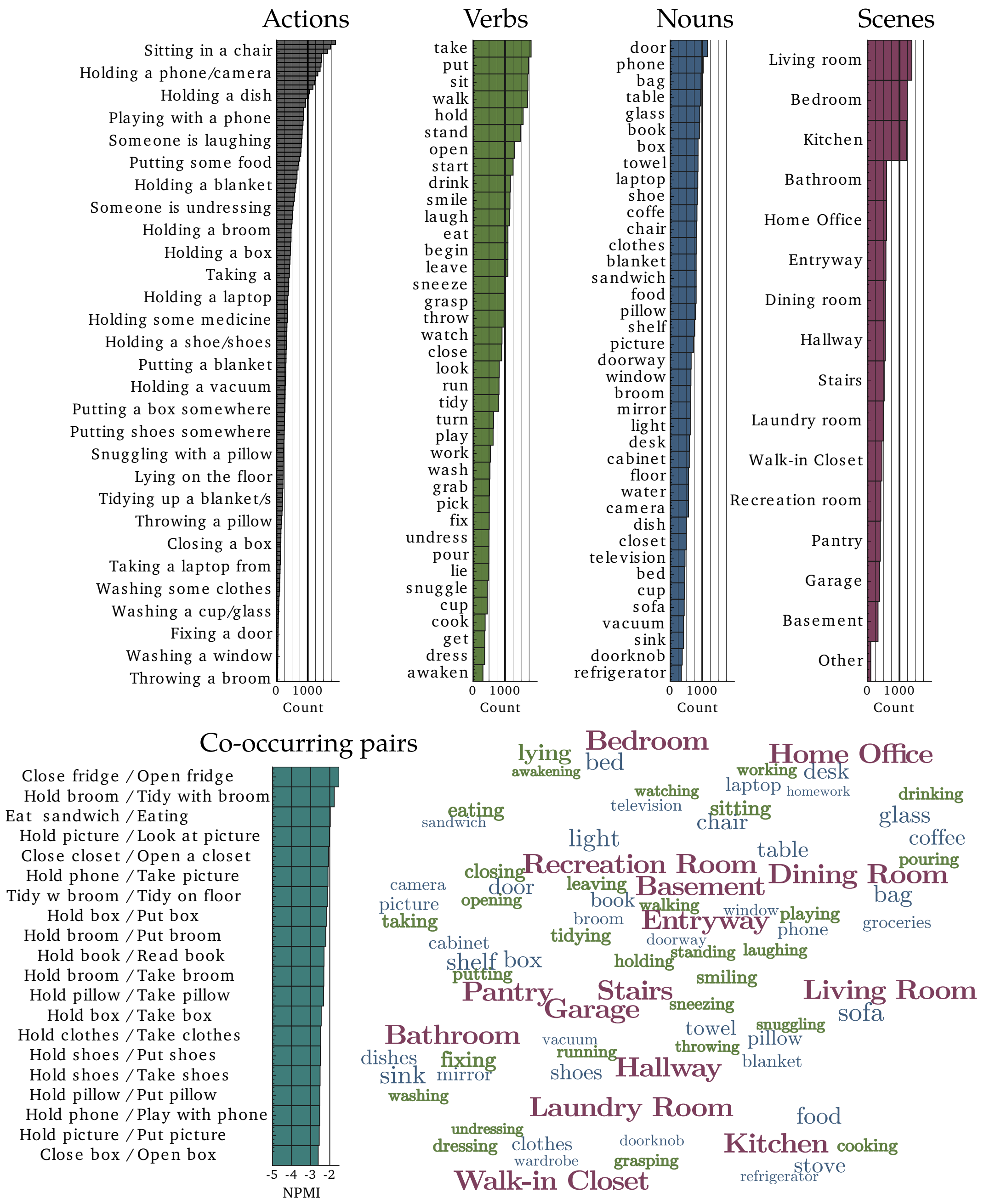


Figure 2.3: Statistics for actions (gray, every fifth label shown), verbs (green), nouns (blue), scenes (red), and most co-occurring pairs of actions (cyan). Co-occurrence is measured with normalized pointwise mutual information. In addition, a T-SNE embedding of the co-occurrence matrix is presented. We can see that while there are some words that strongly associate with each other (*e.g.*, lying and bed), many of the objects and actions co-occur with many of the scenes. (Action names are abbreviated as necessary to fit space constraints.)

recognition. Some of the most common pairs of actions measured in terms of normalized pointwise mutual information (NPMI), are also presented in Figure 2.3. These actions occur in various orders and context, similar to our daily lives. For example, in Figure 2.4 we can see that among these five videos, there are multiple actions occurring, and some are in common. We further explore this in Figure 2.5, where for a few actions, we visualize the most probable actions to precede, and most probable actions to follow that action. As the scripts for the videos are generated by people imagining a boring realistic scenario, we find that these statistics reflect human behaviour.

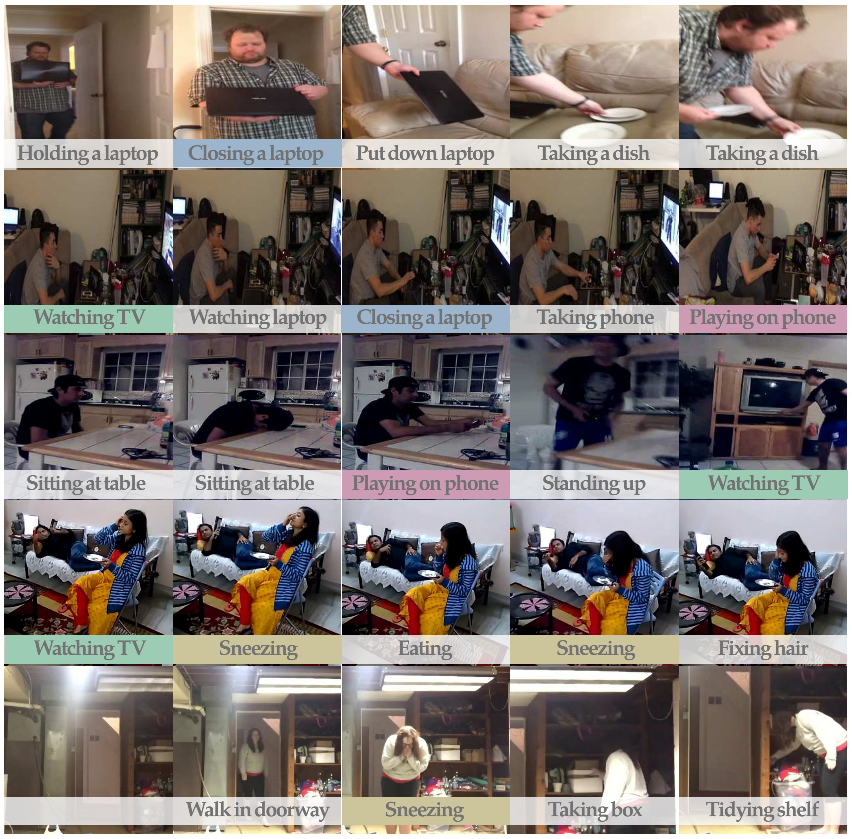


Figure 2.4: Keyframes from five videos in *Charades*. We see that actions occur together in many different configurations. (Shared actions are highlighed in color).

2.4 Applications

We run several state-of-the-art algorithms on Charades to provide the community with a benchmark for recognizing human activities in realistic home environments. Furthermore, the performance and failures of tested algorithms provide insights into the dataset and its properties.

Train/test set. For evaluating algorithms we split the dataset into train and test sets by

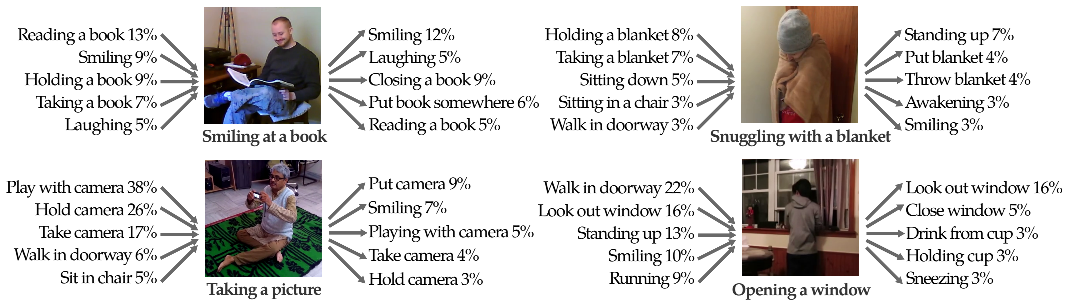


Figure 2.5: Selected actions from the dataset, along with the top five most probable actions before, and after the action. For example, when *Opening a window*, it is likely that someone was *Standing up* before that, and after opening, *Looking out the window*.

Random	C3D	AlexNet	Two-Stream-B	Two-Stream	IDT	Combined
5.9	10.9	11.3	11.9	14.3	17.2	18.6

Table 2.2: mAP (%) for action classification with various baselines.

considering several constraints: (a) the same worker should not appear in both training and test; (b) the distribution of categories over the test set should be similar to the one over the training set; (c) there should be at least 6 test videos and 25 training videos in each category; (d) the test set should not be dominated by a single worker. We randomly split the workers into two groups (80% in training) such that these constraints were satisfied. The resulting training and test sets contain 7,985 and 1,863 videos, respectively. The number of annotated action intervals are 49,809 and 16,691 for training and test.

2.4.1 Action Classification

Given a video, we would like to identify whether it contains one or several actions out of our 157 action classes. We evaluate the classification performance for several baseline methods. Action classification performance is evaluated with the standard mean average precision (mAP) measure. A single video is assigned to multiple classes and the distribution of classes over the test set is not uniform. The label precision for the data is 95.6%, measured using an additional verification step, as well as comparing against a ground truth made from 19 iterations of annotations on a subset of 50 videos. We now describe the baselines.

Improved trajectories. We compute improved dense trajectory features (IDT) [264] capturing local shape and motion information with MBH, HOG and HOF video descriptors. We reduce the dimensionality of each descriptor by half with PCA, and learn a separate feature vocabulary for each descriptor with GMMs of 256 components. Finally, we encode the distribution of local descriptors over the video with Fisher vectors [172]. A one-versus-rest linear SVM is used for classification. Training on untrimmed intervals gave the best performance.

Static CNN features. In order to utilize information about objects in the scene, we make use of deep neural networks pretrained on a large collection of object images. We experiment with VGG-16 [225] and AlexNet [122] to compute fc_6 features over 30 equidistant frames in

	HOG	HOF	MBH	HOG+MBH	HOG+HOF+MBH
K=64	12.3	13.9	15.0	15.8	16.5
K=128	12.7	14.3	15.4	16.2	16.9
K=256	13.0	14.4	15.5	16.5	17.2

Table 2.3: Action classification evaluation with the state-of-the-art approach on Charades. We study different parameters for improved trajectories, by reporting for different local descriptor sets and different number of GMM clusters. Overall performance improves by combining all descriptors and using a larger descriptor vocabulary.

the video. These features are averaged across frames, L2-normalized and classified with a one-versus-rest linear SVM. Training on untrimmed intervals gave the best performance.

Two-stream networks. We use the VGG-16 model architecture [225] for both networks and follow the training procedure introduced in Simonyan et al. [227], with small modifications. For the spatial network, we applied finetuning on ImageNet pre-trained networks with different dropout rates. The best performance was with 0.5 dropout rate and finetuning on all fully connected layers. The temporal network was first pre-trained on the UCF101 dataset and then similarly finetuned on conv4, conv5, and fc layers. Training on trimmed intervals gave the best performance.

Balanced two-stream networks. We adapt the previous baseline to handle class imbalance. We balanced the number of training samples through sampling, and ensured each minibatch of 256 had at least 50 unique classes (each selected uniformly at random). Training on trimmed intervals gave the best performance.

C3D features. Following the recent approach from [244], we extract fc_6 features from a 3D convnet pretrained on the Sports-1M video dataset [106]. These features capture complex hierarchies of spatio-temporal patterns given an RGB clip of 16 frames. Similar to [244], we compute features on chunks of 16 frames by sliding 8 frames, average across chunks, and use a one-versus-rest linear SVM. Training on untrimmed intervals gave the best performance.

Action classification results are presented in Table 2.2, where we additionally consider **Combined** which combines all the other methods with late fusion.

Notably, the accuracy of the tested state-of-the-art baselines is much lower than in most currently available benchmarks. Consistently with several other datasets, IDT features [264] outperform other methods by obtaining 17.2% mAP. To analyze these results, Figure 2.6(left) illustrates the results for subsets of best and worst recognized action classes. We can see that while the mAP is low, there are certain classes that have reasonable performance, for example *Washing a window* has 62.1% AP. To understand the source of difference in performance for different classes, Figure 2.6(right) illustrates AP for each action, sorted by the number of examples, together with names for the best performing classes. The number of actions in a class is primarily decided by the universality of the action (can it happen in any scene), and if it is common in typical households (writer bias). It is interesting to notice, that while there is a trend for actions with higher number of examples to have higher AP, it is not true in general, and actions such as *Sitting in chair*, and *Washing windows* have top-15 performance.

Delving even further, we investigate the confusion matrix for the *Combined* baseline in Figure 2.7, where we convert the predictor scores to probabilities and accumulate them for each class. For clearer analysis, the classes are sorted by the object being interacted with.

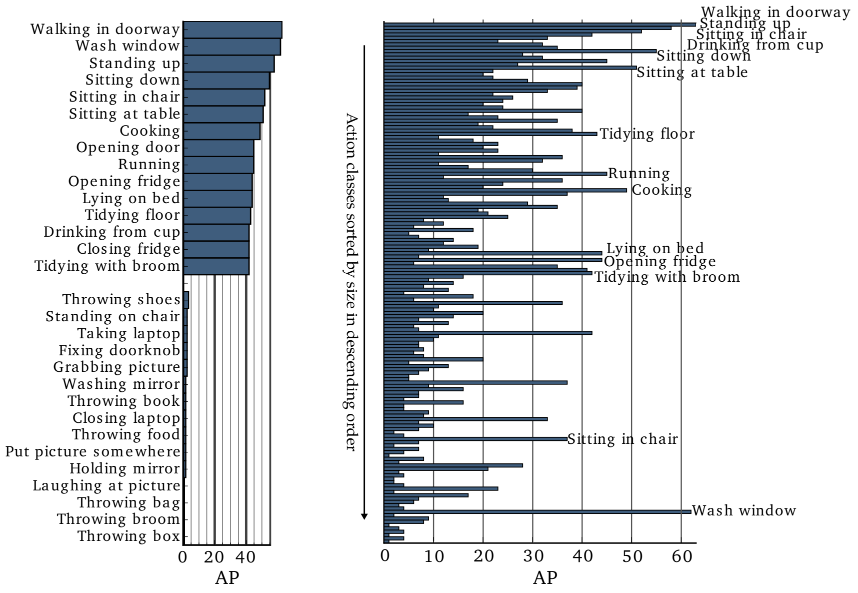


Figure 2.6: On the left classification accuracy for the 15 highest and lowest actions is presented for *Combined*. On the right, the classes are sorted by their size. The top actions on the left are annotated on the right. We can see that while there is a slight trend for smaller classes to have lower accuracy, many classes do not follow that trend.

The first aspect to notice is the squares on the diagonal, which imply that the majority of the confusion is among actions that interact with the same object (*e.g.*, *Putting on clothes*, or *Taking clothes from somewhere*), and moreover, there is confusion among objects with similar functional properties. The most prominent squares are annotated with the object being shared among those actions. The figure caption contains additional observations. While there are some categories that show no clear trend, we can observe less confusion for many actions that have no specific object of interaction. Evaluation of action recognition on this subset results in 38.9% mAP, which is significantly higher than average. Recognition of fine-grained actions involving interactions with the same object class appears particularly difficult even for the best methods available today. We hope our dataset will encourage new methods addressing activity recognition for complex person-object interactions.

2.4.2 Sentence Prediction

Our final, and arguably most challenging task, concerns prediction of free-from sentences describing the video. Notably, our dataset contains sentences that have been used to create the video (*scripts*), as well as multiple video *descriptions* obtained manually for recorded videos. The scripts used to create videos are biased by the vocabulary, and due to the writer's imagination, generally describe different aspects of the video than descriptions. The description of the video by other people is generally simpler and to the point. Captions are evaluated using the CIDEr, BLEU, ROUGE, and METEOR metrics, as implemented in the COCO Caption Dataset [20]. These metrics are common for comparing machine translations to ground truth, and have varying degrees of similarity with human judgement.

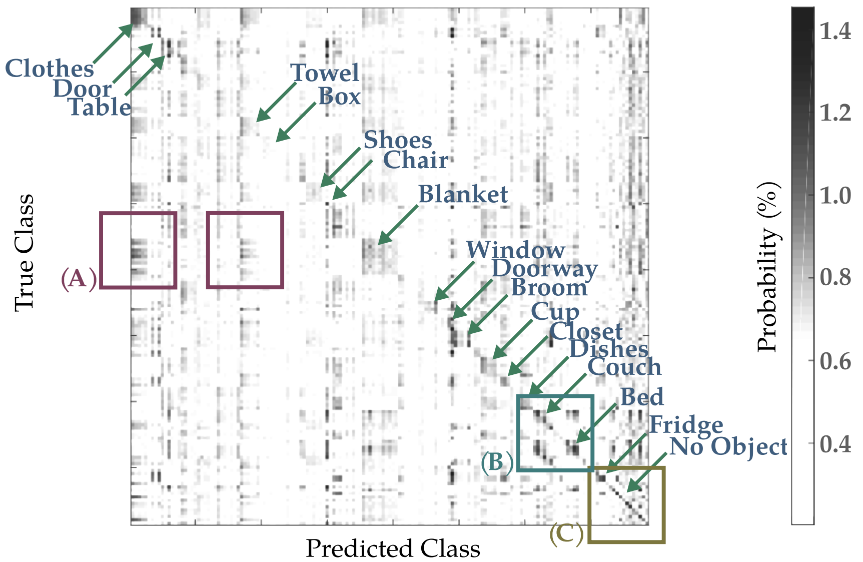


Figure 2.7: Confusion matrix for the *Combined* baseline on the classification task. Actions are grouped by the object being interacted with. Most of the confusion is with other actions involving the same object (squares on the diagonal), and we highlight some prominent objects. Note: (A) High confusion between actions using *Blanket*, *Clothes*, and *Towel*; (B) High confusion between actions using *Couch* and *Bed*; (C) Little confusion among actions with no specific object of interaction (*e.g. standing up*, *sneezing*).

	Script					*Description*				
	RW	Random	NN	S2VT	Human	RW	Random	NN	S2VT	Human
CIDEr	0.03	0.08	0.11	0.17	0.51	0.04	0.05	0.07	0.14	0.53
BLEU$_4$	0.00	0.03	0.03	0.06	0.10	0.00	0.04	0.05	0.11	0.20
BLEU$_3$	0.01	0.07	0.07	0.12	0.16	0.02	0.09	0.10	0.18	0.29
BLEU$_2$	0.09	0.15	0.15	0.21	0.27	0.09	0.20	0.21	0.30	0.43
BLEU$_1$	0.37	0.29	0.29	0.36	0.43	0.38	0.40	0.40	0.49	0.62
ROUGE$_L$	0.21	0.24	0.25	0.31	0.35	0.22	0.27	0.28	0.35	0.44
METEOR	0.10	0.11	0.12	0.13	0.20	0.11	0.13	0.14	0.16	0.24

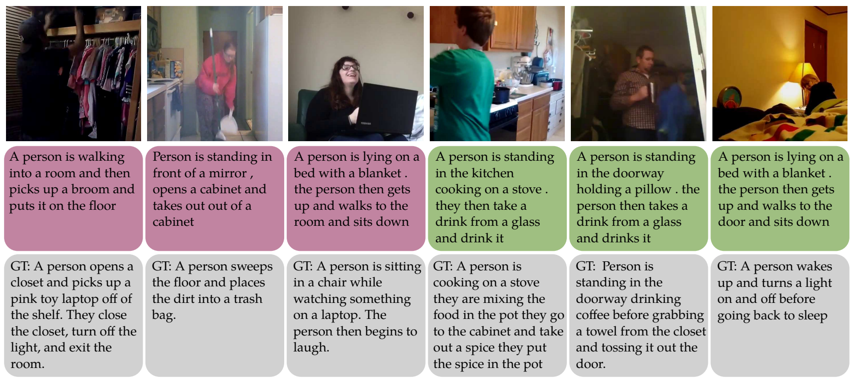
Table 2.4: Sentence Prediction. In the *script* task one sentence is used as ground truth, and in the *description* task 2.4 sentences are used as ground truth on average. We find that S2VT is the strongest baseline.

For comparison, human performance is presented along with the baselines where workers were similarly asked to watch the video and describe what they observed. We now describe the sentence prediction baselines in detail:

Random Words (RW): Random words from the training set.

Random Sentence (Random): Random sentence from the training set.

Nearest Neighbor (NN): Inspired by Devlin et al. [38] we simply use a 1-Nearest Neighbor baseline computed using AlexNet fc$_7$ outputs averaged over frames, and use the caption from that nearest neighbor in the training set.



A person is walking into a room and then picks up a broom and puts it on the floor

GT: A person opens a closet and picks up a pink toy laptop off of the shelf. They close the closet, turn off the light, and exit the room.

Person is standing in front of a mirror , opens a cabinet and takes out out of a cabinet

GT: A person sweeps the floor and places the dirt into a trash bag.

A person is lying on a bed with a blanket . the person then gets up and walks to the room and sits down

GT: A person is sitting in a chair while watching something on a laptop. The person then begins to laugh.

A person is standing in the kitchen cooking on a stove . they then take a drink from a glass and drink it

GT: A person is cooking on a stove they are mixing the food in the pot they go to the cabinet and take out a spice they put the spice in the pot

A person is standing in the doorway holding a pillow . the person then takes a drink from a glass and drinks it

GT: Person is standing in the doorway drinking coffee before grabbing a towel from the closet and tossing it out the door.

A person is lying on a bed with a blanket . the person then gets up and walks to the door and sits down

GT: A person wakes up and turns a light on and off before going back to sleep

Figure 2.8: Three generated captions that scored low on the CIDEr metric (red), and three that scored high (green) from the strongest baseline (S2VT). We can see that while the captions are fairly coherent, the captions lack sufficient relevance.

S2VT: We use the S2VT method from Venugopalan et al. [251], which is a combination of a CNN, and a LSTM.

Table 2.4 presents the performance of multiple baselines on the caption generation task. We both evaluate on predicting the *script*, as well as predicting the *description*. As expected, we can observe that descriptions made by people after watching the video are more similar to other descriptions, rather than the scripts used to generate the video. Table 2.4 also provides insight into the different evaluation metrics, and it is clear that CIDEr offers the highest resolution, and most similarity with human judgement on this task. In Figure 2.8 few examples are presented for the highest scoring baseline (S2VT). We can see that while the language model is accurate (the sentences are coherent), the model struggles with providing relevant captions, and tends to slightly overfit to frequent patterns in the data (*e.g.*, *drinking from a glass/cup*).

2.5 Discussion

We proposed a new approach for building datasets. Our Hollywood in Homes approach allows not only the labeling, but the data gathering process to be crowdsourced. In addition, *Charades* offers a novel large-scale dataset with diversity and relevance to the real world. We hope that Charades and Hollywood in Homes will have the following benefits for our community:

(1) *Training data*: Charades provides a large-scale set of 66,500 annotations of actions with unique realism.

(2) *A benchmark*: Our publicly available dataset and provided baselines enable benchmarking future algorithms.

(3) *Object-action interactions*: The dataset contains significant and intricate object-action relationships which we hope will inspire the development of novel computer vision techniques targeting these settings.

(4) *A framework to explore novel domains*: We hope that many novel datasets in new domains

can be collected using the Hollywood in Homes approach.

(5) *Understanding daily activities*: Charades provides data from a unique human-generated angle, and has unique attributes, such as complex co-occurrences of activities. This kind of realistic bias, may provide new insights that aid robots equipped with our computer vision models operating in the real world.

Chapter 3

Annotation of Video Data

Large-scale annotated datasets, like the one introduced in the previous chapter, allow AI systems to learn from and build upon the knowledge of the crowd. Many crowdsourcing techniques have been developed for collecting image annotations. These techniques often implicitly rely on the fact that a new input image takes a negligible amount of time to perceive. In contrast, we investigate and determine the most cost-effective way of obtaining high-quality multi-label annotations for temporal data such as videos. Watching even a short 30-second video clip requires a significant time investment from a crowd worker; thus, requesting multiple annotations following a single viewing is an important cost-saving strategy. But how many questions should we ask per video? We conclude that the optimal strategy is to ask *as many questions as possible* in a HIT (up to 52 binary questions after watching a 30-second video clip in our experiments). We demonstrate that while workers may not correctly answer all questions, the cost-benefit analysis nevertheless favors consensus from multiple such cheap-yet-imperfect iterations over more complex alternatives. When compared with a one-question-per-video baseline, our method is able to achieve a 10% improvement in recall (76.7% ours versus 66.7% baseline) at comparable precision (83.8% ours versus 83.0% baseline) in about half the annotation time (3.8 minutes ours compared to 7.1 minutes baseline). We demonstrate the effectiveness of our method by collecting multi-label annotations of 157 human activities on 1,815 videos from our Charades dataset, which forms the test set.

Large-scale manually annotated datasets such as ImageNet [36] led to revolutionary development in computer vision technology. In addition to playing a critical role in advancing computer vision, crowdsourced visual data annotation has inspired many interesting research questions: How many exemplars are necessary for the crowd to learn a new visual concept [169]? How can image annotation be gamified [255, 256]? How can we provide richer annotators in the form of visual attributes [170] or object-object interactions [121]? How can we exhaustively annotate all visual concepts present in an image [37]?

Much of the work on visual data annotation has focused on images, but many real-world applications require annotating and understanding *video* rather than image data. A worker can understand an image in a few hundred milliseconds [241]. Naïvely applying image annotation techniques to data that takes longer to understand, such as data involving time, is prohibitively expensive. Developing effective strategies for temporal annotation is important for multiple domains that require watching, listening, or reading: musical attributes or

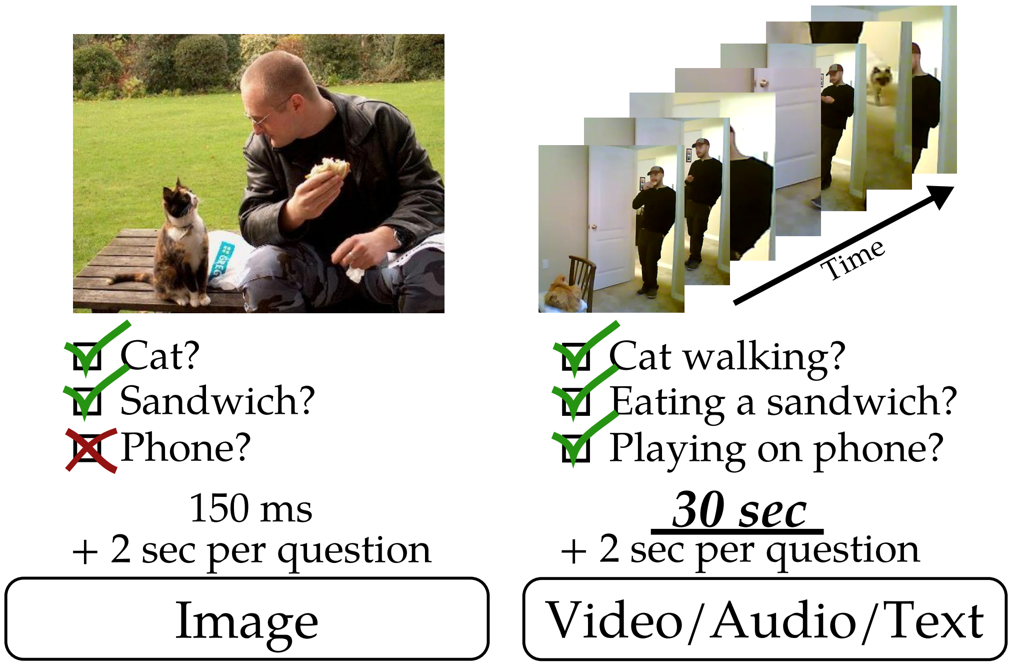


Figure 3.1: Time data (e.g., video) is fundamentally different from image data. In this chapter, we explore cost-optimal strategies for exhaustively annotating video data.

emotion on songs [134], web page categorization [248], news article topics [204], and video activity recognition [223].

We are interested in the following annotation task illustrated in Fig. 3.1: given a video and a set of visual concepts (such as a set of objects or human actions or interesting events), label whether these concepts are present or absent in the video. Efforts such as Glance [129] focus on quickly answering a question about a video by parallelizing the work across the crowd workforce in 30-second video clips. They are able to get results in near real-time, allowing for *interactive* video annotation. In contrast, we are interested in annotating a large-scale video dataset where multiple questions (known apriori) need to be answered about each video. Even for a short 30-second video clip, it takes at least 15 seconds at double speed for an annotator to watch the video; thus, asking only a single question at a time is highly inefficient. Efforts such as [15,36] explore multi-label annotation of images but cannot be directly applied to temporal video data because of this inefficiency.

We thus ask: how many questions should we ask workers when annotating a video? Psychology research shows that only on the order of 7 concepts can be kept in short-term memory [158]. However, our results demonstrate asking many more questions at a time in a single Human Intelligence Task (HIT) can be significantly more efficient. In particular, we demonstrate that asking as many questions as possible, up to 52 questions at a time about a 30-second video in our experiments, provides an optimal tradeoff between accuracy and cost. When compared with a one-question-at-a-time baseline, our method achieves a 10% improvement in recall (76.7% ours versus 66.7% baseline) at comparable precision (83.8% ours versus 83.0% baseline) in about half the annotation time (3.8 minutes ours compared to 7.1 minutes baseline). We empirically verify that our conclusions hold for videos of multiple lengths, explore several strategies for reducing the cognitive load on the workers in the context of video annotation and demonstrate the effectiveness of our method by exhaustively annotating a video dataset of [223] enabling computer vision research into multi-label human action understanding.

The annotated data and additional details are available at: `http://allenai.org/plato/charades/`.

3.1 Background

Video annotation applications. Video understanding is important for many applications ranging from behavior studies [27] to surveillance [202] to autonomous driving [62]. Large-scale annotated computer vision video datasets [50,68,124,232,288] enable the development of algorithms that are able to automatically process video collections. However, the lack of large-scale multi-label video datasets makes it difficult to study the intricate interactions between objects and actions in the videos rather than focusing on recognition of one or a handful of concepts.

Efficient video annotation. Video annotation is very time-consuming. Determining the absence of a concept in an image takes on the order of seconds; in contrast, determining the absence of a concept in a video takes time proportional to the length of the video. Efforts such as [252,258,293] exploit temporal redundancy between frames to present cost-effective video annotation frameworks. The approaches of [54,252,260] and others additionally incorporate active learning, where the annotation interfaces learns to query frames that, if annotated, would produce the largest expected change in the estimated object track. However, these methods combine human annotation with automatic computer vision techniques, which causes several problems: (1) these techniques are difficult to apply to challenging tasks such as activity recognition where computer vision models lag far behind human ability; (2) these methods are difficult to apply to scenarios where very short or rare events, such as shoplifting, may be the most crucial, and (3) the resulting hybrid annotations provide unfair testbeds for new algorithms.

Glance [129] focuses on parallelizing video annotation effort and getting an answer to a single question in real-time. Our work can be effectively combined with theirs: they parallelize annotation in 30-second video chunks, while we explore the most effective ways to obtain multiple labels simultaneously for every 30-second video.

Action recognition datasets. Some existing large-scale action datasets such as EventNet [287] or Sports-1M [106] rely on web tags to provide noisy video-level labels; others like THUMOS [68] or MultiTHUMOS [288] employ professional annotators rather than crowdsourcing.

There are two recent large-scale video annotation efforts that successfully utilize crowdsourcing. The first effort is ActivityNet [81] which uses a proposal/verification framework similar to that of ImageNet [36]. They define a target set of actions, query video search engines for proposal videos of those actions and then ask crowd workers to clean up the results. The second effort [223] entirely crowdsources the creation of a video dataset: one worker writes a video script containing a few target objects/actions, another one acts out the script and films the video, and others verify the work. In both these efforts, each video comes pre-associated with one or a handful of action labels, and workers are tasked with verifying these labels. In contrast, we're interested in the much more challenging problem of multi-label video annotation beyond the provided labels.

Multi-label image annotation. Increasingly more complex image annotations are provided in recent dataset [14, 121, 138]. Multi-label image annotation has been studied by e.g., [15, 37, 163, 255, 297]. We incorporate insights from these works into our video annotation framework. We use a hierarchy of concepts to accelerate multi-label annotation following [15, 37]. Inspired by [120], we explore using cheap but error-prone annotation interfaces over thorough but more expensive formulations.

3.2 Method for multi-label video annotation

We are given a collection of M videos and a set of N target labels: for example, a list of target object classes, e.g., "cat," "table," or "tree," or a list of human actions, e.g., "reading a book" or "running." The goal is to obtain $M \times N$ binary labels, corresponding to the presence or absence of each of the N target concepts in each of the M videos. These labels can then be used for a variety of applications from training computer vision models [223] to studying human behavior [27].

We are particularly interested in situations where the label space N is large: $N = 157$ in our experiments. As a result, the key challenge is that workers are not able to remember all N questions at the same time; however every time a worker is required to watch a video of length L during annotation, they have to invest an additional L seconds of annotation time. We focus on video annotation but our findings may be applicable to any media (e.g., audio, text) where a non-trivial amount of time L is required to process each input.

3.2.1 Multiple question strategy

Our strategy is to ask all N target questions at the same time about each video, even if N is much higher than the 7 concepts that people can commit to short-term memory [158]. We randomize the order of questions and ask workers to select only the concepts that occur within the video. This naturally leads to lower recall r than if we ask only a handful of questions that the workers would be more likely to read carefully. However, there are two advantages.

Advantage #1: Low annotation times. Since only one worker has to watch the video instead of asking N different workers to annotate one label each, this recall r is obtained with relatively little time investment t. This makes it a highly effective strategy combined with consensus among multiple workers [211]. Given a fixed time budget T, we can repeat the annotation process $\frac{T}{t}$ times with different workers. Assume the workers are independent and we count the concept as present in the image if at least one worker annotates it. Our expected recall in T time is:

$$\text{ExpectedRecall} = 1 - (1 - r)^{\frac{T}{t}} \tag{3.1}$$

since each worker will miss a concept with $1-r$ probability, and a concept won't be annotated only if all $\frac{T}{t}$ workers independently miss it.

Advantage #2: High precision. The $M \times N$ label matrix is naturally sparse since most concepts do not occur in most videos. When workers are faced with only a small handful of concepts and none of them occur in the video, they may get nervous that they are not doing the task correctly and provide erroneous positive labels. However, when they are faced with many concepts at the same time and asked to select the ones that occur in the video, they get satisfaction out of being able to annotate several target concepts and are less likely to erroneously select additional concepts.

3.2.2 Practical considerations

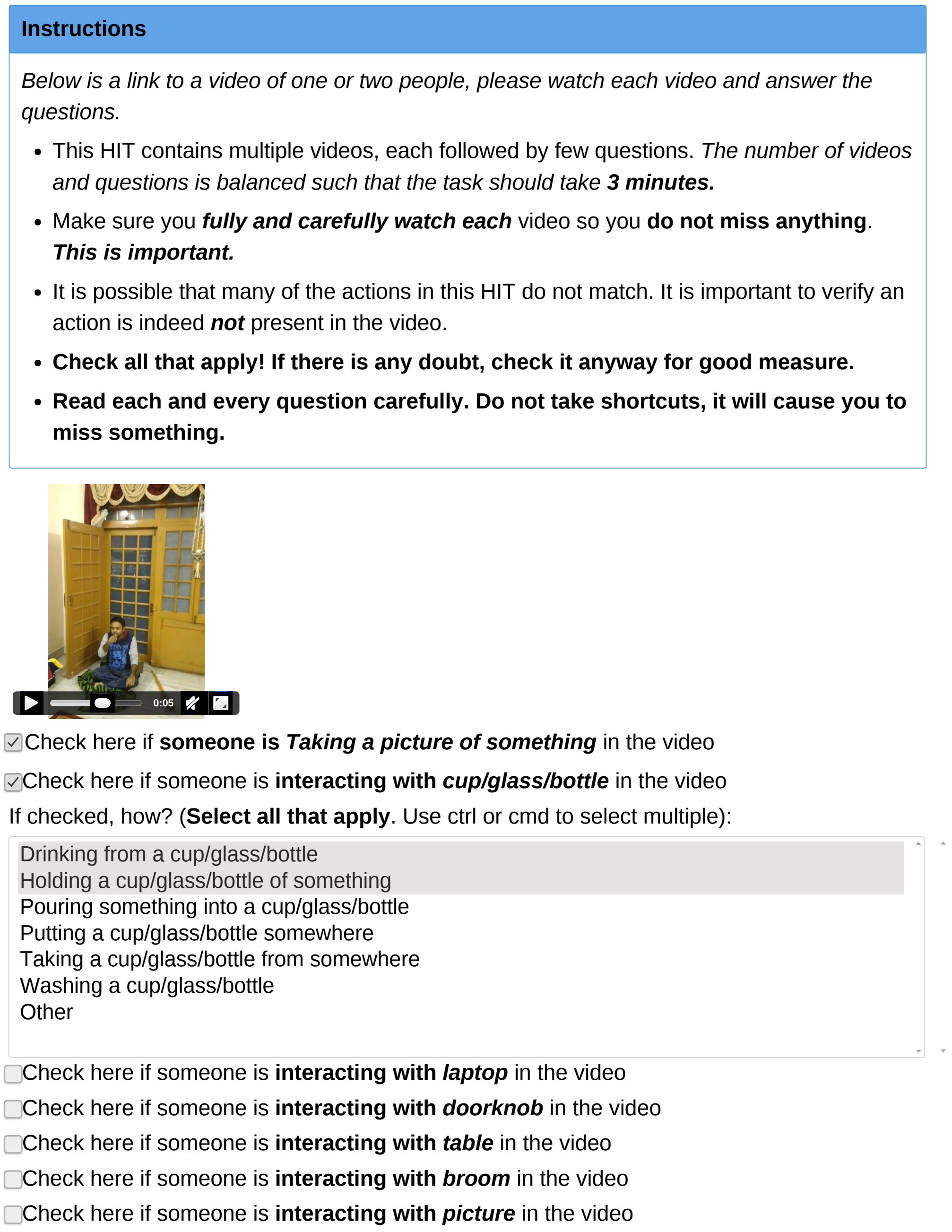
In designing an effective multi-question video annotation interface shown in Fig. 3.2, we incorporate insights from image annotation [37] to reduce the space of N labels and from video annotation [129] to compress the video length L.

Semantic hierarchy. Following [37] we create a semantic hierarchical grouping of concepts to simplify the multi-label annotation. However, [37] use the hierarchy differently. They ask one question at a time about a matrix of images, e.g., "click on all images which contain an animal." They then ask a low-level question, e.g., "click on all images which contain a dog," on a smaller matrix of images which were positive for the prior question. In contrast, we use the concept hierarchy similar to [279] to simplify our annotation interface on a single video.

Playback speed. Videos of average length of 30 seconds are played at 2x speed following [129]. In this way, worker time is not unnecessarily wasted but they are able to perceive and accurately annotate the target concepts.

Instructions. Workers are instructed to carefully watch each video and select all concepts that occur. Since most concepts do not occur in the video, workers are asked to only check the boxes for the ones that do occur and to ignore the others. We experimentally verify this design choice below.

Time vs cost. We use human time and cost interchangeably throughout. We group together multiple videos into a single HIT such that each HIT takes approximately the same time to complete: e.g., an interface with fewer questions will allow for more videos to be annotated in a single HIT. We pay a uniform amount per HIT. Thus, an interface that takes 3x less time will allow for 3x more videos per HIT which will allow for 3x fewer HITs to annotate the full dataset, which will in turn translate to a 3x reduction in cost when annotating a large-scale video dataset.



Instructions

Below is a link to a video of one or two people, please watch each video and answer the questions.

- This HIT contains multiple videos, each followed by few questions. *The number of videos and questions is balanced such that the task should take **3 minutes.***
- Make sure you ***fully and carefully watch each*** video so you **do not miss anything**. ***This is important.***
- It is possible that many of the actions in this HIT do not match. It is important to verify an action is indeed ***not*** present in the video.
- **Check all that apply! If there is any doubt, check it anyway for good measure.**
- **Read each and every question carefully. Do not take shortcuts, it will cause you to miss something.**

[✓] Check here if **someone is *Taking a picture of something*** in the video

[✓] Check here if someone is **interacting with *cup/glass/bottle*** in the video

If checked, how? (**Select all that apply**. Use ctrl or cmd to select multiple):

[✓ Drinking from a cup/glass/bottle]
[✓ Holding a cup/glass/bottle of something]
Pouring something into a cup/glass/bottle
Putting a cup/glass/bottle somewhere
Taking a cup/glass/bottle from somewhere
Washing a cup/glass/bottle
Other

[ ] Check here if someone is **interacting with *laptop*** in the video

[ ] Check here if someone is **interacting with *doorknob*** in the video

[ ] Check here if someone is **interacting with *table*** in the video

[ ] Check here if someone is **interacting with *broom*** in the video

[ ] Check here if someone is **interacting with *picture*** in the video

Figure 3.2: Our multi-question video annotation interface.

3.3 Experiments

We begin by describing the setup used to evaluate our method, including steps taken to control for factors of variation across different crowdsourcing experiments. We then present a series of smaller-scale experiments on 100-150 videos at a time investigating (1) varying the number of questions in the annotation interface, and (2) strategies for reducing cognitive load on workers during annotation. We conclude by bringing our findings together and evaluating our large-scale multi-label video annotation pipeline.

3.3.1 Data and evaluation setup

We use the large-scale video dataset [223] with a focus on common household activities. The target labels are 157 activity classes such as *Someone is running* and *Putting a cup somewhere* provided with the dataset. The videos are associated with some labels apriori, similar to ImageNet [36] and ActivityNet [50]. Fig. 3.3 shows some examples. This misses additional activities also present in the video, making it difficult to evaluate computer vision algorithms and to study interactions between different actions. We demonstrate how to cost-effectively collect exhaustive annotations for this dataset and exhaustively annotate the test set consisting of 1,815 videos.

Evaluating recall. We use the originally provided action labels to evaluate the recall of our multi-label annotation algorithms. There were on average 3.7 activities labeled per video in this dataset. The activities follow a long-tailed distribution: some occur in as many as 1391 videos, others in only 33. Each activity occurs in 42 videos on average.

Evaluating precision. Precision is more difficult to evaluate since to the best of our knowledge no large-scale video dataset is annotated with hundreds of visual concepts. Annotating the videos in this dataset exhaustively in a straight-forward way is prohibitively expensive, which is exactly what we are trying to address. We adopt a middle ground. After obtaining a set of candidate labels from the annotators, we perform a secondary verification step. In the verification task, workers have to annotate the temporal extent of the action in the video or specify it is not present in the video. This serves as an evaluation of the precision of multi-label annotation. In addition, this provides temporal action annotations which we also publicly released.

Semantic hierarchy. The 157 target human activities are grouped based on the object being interacted with to simplify the annotation interface. The annotator first sees several questions such as “Check here if someone is interacting with a book in the video” or “Check here if someone is interacting with shoes in the video.” If the annotator says *yes* someone is interacting with a book, s/he will be asked to select one or more of the types of interaction: closing a book? opening a book? holding a book? putting a book somewhere?

We create 33 object groups, each group with 4.2 activities on average. Additionally, 19 activities (such as *Someone is laughing*, *Someone is running somewhere*) do not belong to any group and are asked individually. Thus, we obtain 52 high-level questions which cover all of the label space.

3.3.2 Crowdsourcing setup

During the study, 674 workers were recruited to finish 6,337 tasks on Amazon Mechanical Turk. We summarize some key crowdsourcing design decisions here.

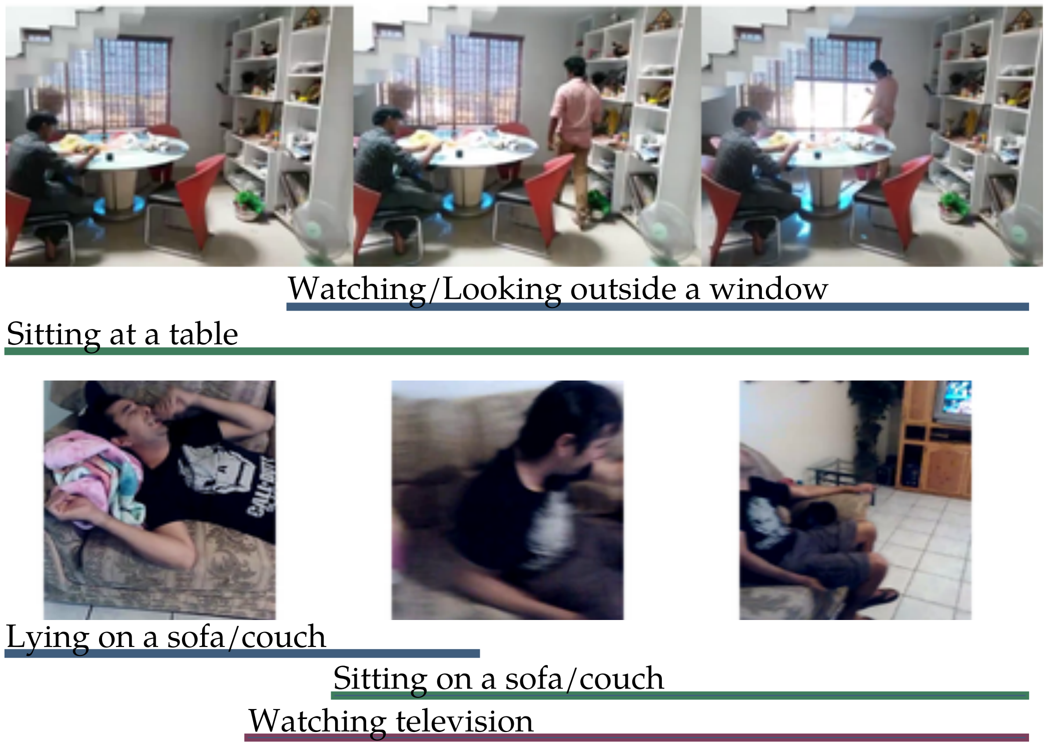


Figure 3.3: Examples from the video dataset of [223]. The videos contain complex human activities that require the annotator to carefully watch each video.

Quality control. Workers were restricted to United States, with at least 98% approval rate from at least 1000 tasks. We used recall, annotation time, and positive rate to flag outliers, which were manually examined and put on a blacklist. To maintain a good standing with the community all work completed without clear malice was approved, but bad workers were prohibited from accepting further work of this type.

In Fig. 3.4 the relationship between how much time an individual worker spends on a task and quality of the annotation is presented. We can see that apart from clear outliers, there is no significant difference, and we treat the worker population as following the same distribution, and focus on the time difference between different methods.

Uncontrolled factors. There are many sources of variation in human studies, such as worker experience (we observed worker quality increasing as they became more familiar with our tasks) or time of day (full-time workers might primarily be available during normal business hours). We attempted to minimize such variance by deploying all candidate methods at the same time within each experiment.

Payment. In order to verify our hypothesis that it is best to ask multiple questions about a video simultaneously, we need to evaluate interfaces with a varying number of questions per video. However, we want to maintain as much consistency as possible outside of the factor

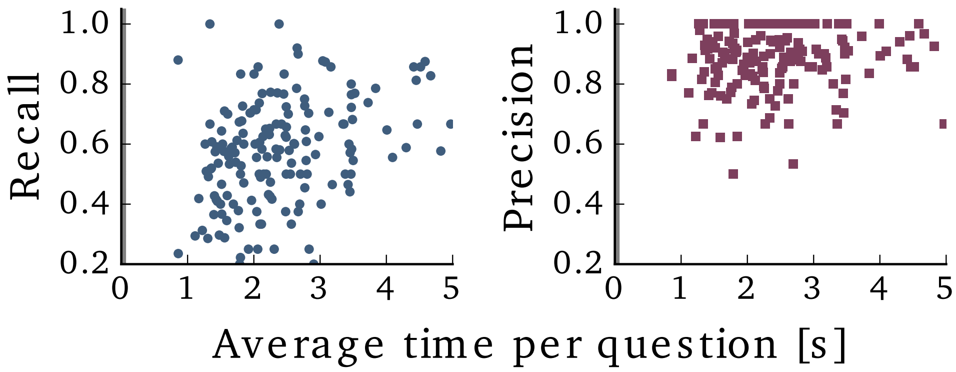


Figure 3.4: Workers that spend more time answering questions have marginally higher accuracy (Pearson's correlation of time with recall is 0.227 and with precision is 0.036). However this trend is so slight that we ignore it and instead focus on improving the annotation workflow as a whole.

we're studying. We use a single type of HIT where workers are provided with V videos and Q questions for each video using the interface of Fig. 3.2. When we increase the number of questions Q per video, we decrease the number of videos V to keep the expected annotation effort consistent within the HIT.

To do this, we ran some preliminary experiments and analyzed the average amount of time it takes to label a video in our dataset with Q questions. Fig. 3.5 shows the relationship between number of questions Q and time. The least-squares line of best fit to this data is

$$T = 14.1 + 1.15Q \tag{3.2}$$

Thus it takes an average of 14.1 seconds to watch a video and 1.15 seconds to answer each question. This is consistent with our expectations: our average video is 30.1 seconds long played at double speed, and binary questions take on the order of 1-2 seconds to answer [121].

We varied the number of videos in each HIT using Eqn. 3.2 to target about 150 seconds of expected annotation effort. We paid $\$0.40$ per HIT, amounting to about $\$9.60$ per hour.

Multiple question interface. We report results on annotating the 157 activities using the 52-question semantic hierarchy.[1] Our method solicits labels for all 52 questions and corresponding sub-questions in the same interface as shown in Fig. 3.2. When evaluating interfaces with a smaller number of questions k, we partition the 52 questions into $\frac{52}{k}$ subsets randomly. Multiple workers then annotate the video across $\frac{52}{k}$ tasks, and we accumulate the results.[2] An *iteration* of annotation refers to a complete pass over the 52 questions for each video. We can then directly compare the annotations resulting from interfaces with different values of k.

[1]We additionally verified that all conclusions hold if we are interested in only the 52 high-level activities as well.

[2]Some of the questions take longer than others, and thus some subsets may take longer to annotate than others. However, we report cumulative results after all subsets have been annotated and thus the variations in time between the subsets is irrelevant.

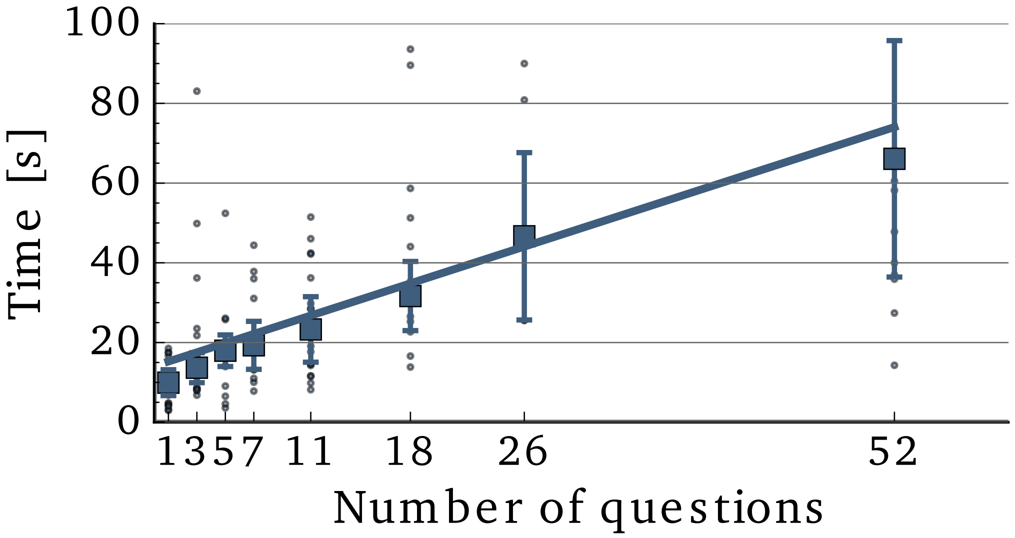


Figure 3.5: The relationship between number of questions in the interface and the amount of time it takes. We use it to maintain a consistent amount of annotation effort across HITs while varying the number of questions in the interface. Error bars correspond to 1 standard deviation.

3.3.3 Effect of varying the number of questions

So far we described the data, the evaluation metrics and the crowdsourcing setup. We are now ready to begin experimenting with different annotation strategies.

We begin by varying the number of questions the workers are asked after watching each video: from only 1 question per video (very time-inefficient since 52 workers have to independently watch the video) up to all 52 questions at the same time (potentially daunting for the workers). We run the annotation experiment on 140 videos, and report the time, recall and precision after one iteration of annotation, i.e., after workers answer all 52 questions about each video.

Advantages of asking multiple questions. Fig. 3.6 demonstrates two advantages to asking multiple questions together rather than one-at-a-time. The first advantage is *low annotation time*: the time for one iteration of annotation drastically decreases as the number of questions increases. Concretely, it takes 8.61 minutes per video with the 1-question interface versus only 1.10 minutes per video with the 52-question interface (since the time to watch the video gets amortized).

The second advantage to asking multiple questions together is *increased precision* of annotation. Concretely, precision is only 81.0% with the 1-question interface but rises up to 86.4% with the 52-question interface. When only one question per video is asked, all answers in a HIT will likely be negative since only a handful of target activities occurs in each 30-second video. Workers report being concerned when all answers are negative. We hypothesize that as a result they may erroneously answer positively if they have any suspicions about the activity being present, which decreases the precision of annotation in

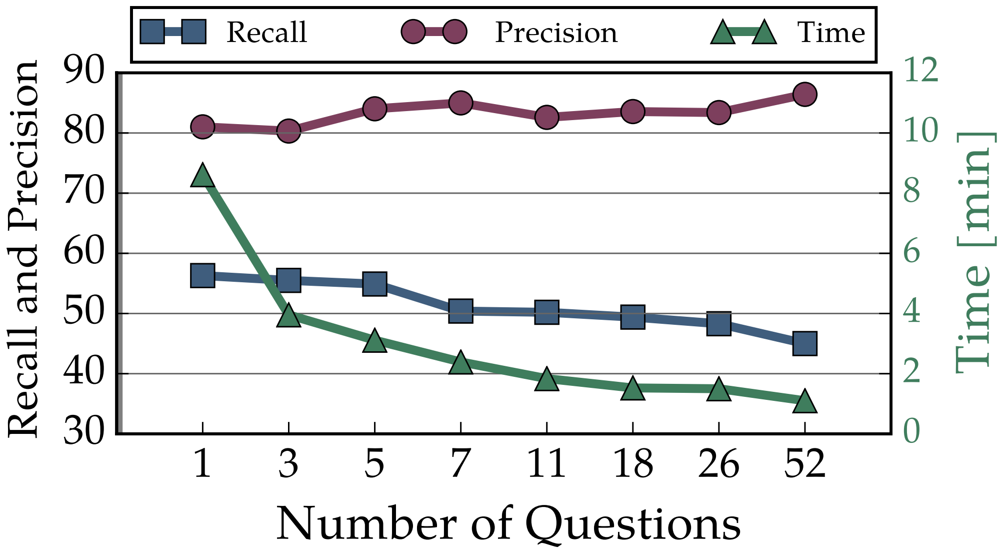


Figure 3.6: Accuracy (*left axis*) and time (*right axis*) of annotation as a function of the number of questions in the interface (*x-axis*). While recall is higher with fewer questions, this is at the cost of significantly higher annotation time.

the few-question interfaces.

Drawback of asking multiple questions. The one drawback of asking multiple questions is *decreased recall*. When asked only one question per video, workers achieve 56.3% recall compared to only 45.0% recall when asked all 52 questions at once. This is because it is challenging to keep 52 questions in memory while watching the video or the entire video in memory while answering the questions. Interestingly, Fig. 3.6 shows a sharp drop in recall beyond 5-7 questions in the interface, which is the number of concepts people can keep in short-term memory [158].

Fixing the drawback. Even though recall is lower when asking multiple questions about a video, it is obtained in significantly less annotation time. Given a fixed time budget, we can compute the expected recall if we were to ask multiple workers to do the annotation by referring back to Eqn. 3.1. In particular, assume we are given 8.61 minutes that it takes to fully annotate a video using the 1-question interface. In this amount of time, we can ask at least 7 workers to annotate it with the 52-question interface (since it only takes 1.10 minutes per iteration). Fig. 3.7 reports the expected recall achievable in 8.61 minutes using the different interfaces. We conclude that the many-question interfaces are better than the few-question interfaces not only in terms of time and precision, but also in terms of recall for a fixed time budget. We will revisit this in later experiments.

Worker behavior. Besides quantitatively evaluating the different interfaces according to the standard metrics, it is also informative to briefly look into annotator behavior.

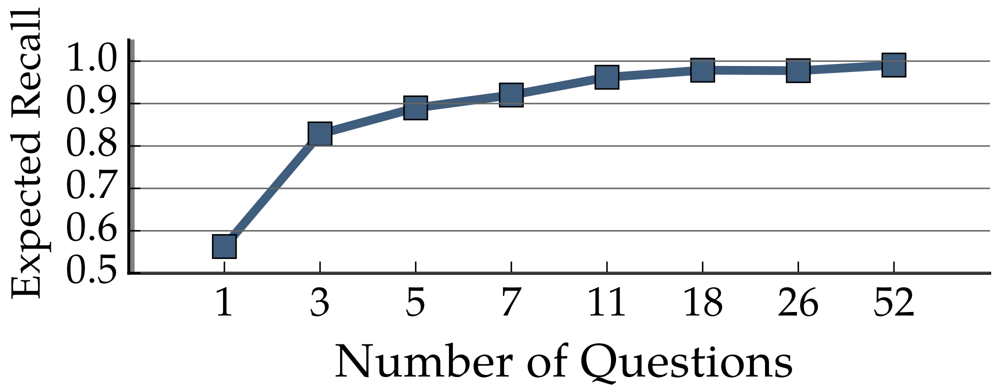


Figure 3.7: Expected recall given a fixed time budget (simulated using Eqn. 3.2) for interfaces with a varying number of questions. The budget is 8.61 minutes per video, enough to run 1 iteration of annotation with the 1-question interface.

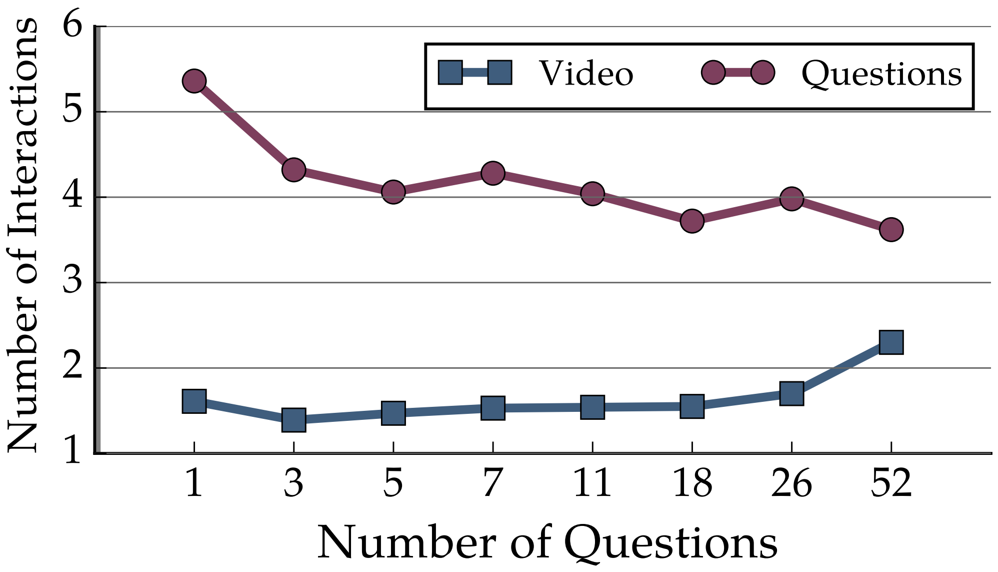


Figure 3.8: The number of times workers paused or synced the video (*video*) and the number of questions answered affirmatively after an iteration of annotation (*questions*) as a function of the number of questions in the interface.

Fig. 3.8 reports the number of interactions of workers with the video: i.e., the number of times they pause or seek the video. We observe that the interactions with the video generally increase with the question count, suggesting that workers may be watching the video more carefully when asked more questions. Interestingly, however, with only a single question the users seem to hurry through the video.

Fig. 3.8 additionally reports the average number of questions answered affirmatively by the workers after an iteration of annotation. As the number of questions in the interface increases, the average number of affirmative answers after 52 questions have been answered decreases from 5.36 to 3.62. We hypothesize that when multiple questions are presented to the workers simultaneously, they feel satisfied once they are able to answer a handful of them positively; when faced with only a small number of questions, they feel increased pressure to find more positive answers. This contributes to both the increase in recall and the drop in precision.

Worker feedback. Finally, we asked workers to report their enjoyment of the task on a scale of 1 (lowest) to 7 (highest). Average enjoyment ranged from 5.0 to 5.3 across the different interfaces, indicating that workers were equally pleased with both few-question and many-question tasks.[3]

3.3.4 Targeting the UI for different number of questions

So far we investigated the effect that number of questions have on the accuracy and efficiency of annotation, while keeping all other factors constant. However, using the same user interface and annotation workflow for different numbers of questions may not be optimal. For example workers tend to worry when asked too many negative questions in a row in an interface with a few questions, or may not read all questions in detail in an interface with many questions.

In this section, we use the 3-question interface for the *few-questions* setting, and the 26-question interface for the *many-questions* setting. We run a series of experiments investigating strategies for improving the UI. We discover two strategies for improving the few-questions interface and conclude that our many-questions interface is optimal.

Positive bias. When using the few-questions interface, most answers within a HIT are expected to be negative since most target activities are not present in the videos. This has two undesirable effects: (1) workers may start paying less attention, and (2) workers may get nervous and provide erroneous positive answers, lowering the annotation precision.

To overcome this, we duplicate questions known to be positive and inject them such that approximately 33% of the questions are expected to be positive. This forces the workers to pay closer attention and be more active in the annotation; on the downside, this increases the number of questions per annotation from 52 to 78 including the duplicates.

In an experiment on 150 videos, injecting such positive bias into the few-questions interface improves on all three metrics: recall, precision and time of annotation. Recall increases from 53.2% to 57.9% with positive bias,[4] precision increases slightly from 79.0% to 81.3% with positive bias, and time for an iteration of annotation drops from 4.6 minutes to 3.6 minutes, likely because workers trust their work more and thus are able to annotate faster. Workers also report slightly higher enjoyment: on a scale of 1 (lowest) to 7 (highest), they report 5.8 enjoyment of the task with positive bias versus 5.5 without. We incorporate positive bias into the few-question interface in future experiments.

[3]In our preliminary experiments we did not use Eqn. 3.2 to control for the amount of work within each HIT; worker enjoyment was then strongly inversely correlated with the amount of work.

[4]To maintain a fair comparison, answers to duplicate questions are ignored during evaluation. Thus the time it takes to answer them is also ignored when computing annotation time per iteration.

Grouping. Prior work such as [36] demonstrated that asking about the same visual concepts across multiple images reduces the cognitive load on workers and increases annotation accuracy. In our second experiment, we apply the same intuition to videos: we randomly group questions together and make sure that all questions are the same for all videos within a single HIT. Residual question not part of the groups, and groups too small to fill a whole task were discarded, but each question was presented both in the context of grouping and not, for a fair comparison.

In the few-questions interface, grouping improves the precision and the time of annotation, albeit at a slight reduction in recall. Specifically, in an experiment on 100 videos, precision increases from 77.7% to 81.4% when grouping is added. Annotation time per iteration drops from 5.9 minutes to 5.1 minutes with grouping; however, recall also drops from 70.4% to 67.2% with grouping. To determine if the drop in recall is a concern, we refer back to Eqn. 3.1 to compute the expected recall for a fixed time budget. In 5.9 minutes (enough for one iteration without grouping), we expect a recall of 72.3% with grouping, higher than 70.4% recall without. Thus, we conclude that grouping is strictly beneficial in the few-question setting as hypothesized, and we use it in future experiments.

We also investigated the effect of grouping in the many-question interface, but concluded it's unhelpful. Recall with grouping is 55.2%, much lower than 62.0% without grouping. Even though annotation time is faster (1.4 minutes per iteration with grouping compared to 1.6 minutes per iteration without), this is not enough to compensate for the drop in recall: the expected recall given a budget of 1.6 minutes of annotation is still only 61.2% with grouping compared to 62.0% without. Further, precision is also lower with grouping: 79.0% with grouping compared to 80.2% without. We hypothesize that this is because workers are not able to remember all 26 questions anyway, so grouping only provides a false sense of security (as evidenced by the speedup in annotation time). We do not use grouping in the multi-question interface in future experiments.

Note that grouping had no effect on worker enjoyment. On a scale of 1 (lowest) to 7 (highest), workers reported 5.30 enjoyment with grouping and 5.24 without. We believe this is because while grouping makes the task easier, the workers are also less engaged since it is more repetitive.

Video summary. Having discovered two strategies for improving the few-question interface (positive bias and grouping), we turn our attention to strategies targetting the multi-question setup. The main challenge in this setting is that workers may be overwhelmed by the number of questions and may not read them all carefully.

To better simulate a scenario where the worker has to pay careful attention to the video, we add an additional prompt to the many-questions interface. In an experiment on 100 videos, workers were asked to "please describe with approximately 20 words what the person/people are doing in the video." This adds on average 36 seconds per iteration, yielding 2.1 minutes of annotation time with the additional prompt versus 1.5 without. However, the extra time does not translate to noticeable benefits in annotation accuracy: recall drops slightly to 53.2% with the prompt compared to 54.2% without, although precision increases slightly to 88.3% with the prompt compared to 87.1% without. We conclude that adding the prompt has no significant impact on the accuracy of annotation despite a 1.4x increase in annotation time.

Forced responses. The final investigation into improving the many-questions interface is asking workers to actively select a yes/no response to every question rather than simply checking a box only if an action is present. Intuitively this forces the workers to pay attention to every question. However, this again produces no improvements in accuracy, indicating that workers are already working hard to provide the most accurate responses and are only confused by the additional forced responses.

Concretely, we experimented on 100 videos and observed a drop in recall to 55.7% with the forced responses compared to 63.3% without as well as a drop in precision to 84.6% with forced responses compared to 88.8% without. Further, annotation time increases to 2.2 minutes per video with forced responses versus 1.6 minutes without. Thus forcing workers to read every question is in fact appears harmful: it is better for them to focus on watching the video and only skim the questions.

Conclusions. We thoroughly examined the annotation interface in the few-questions and many-questions setting. We discover that positive bias and grouping are effective strategies for improving the few-questions UI, and incorporate them in future experiments. For the many-questions setting, simply randomizing the questions and allowing the workers to select the actions that appear in the video is shown to be more effective than any other baseline.

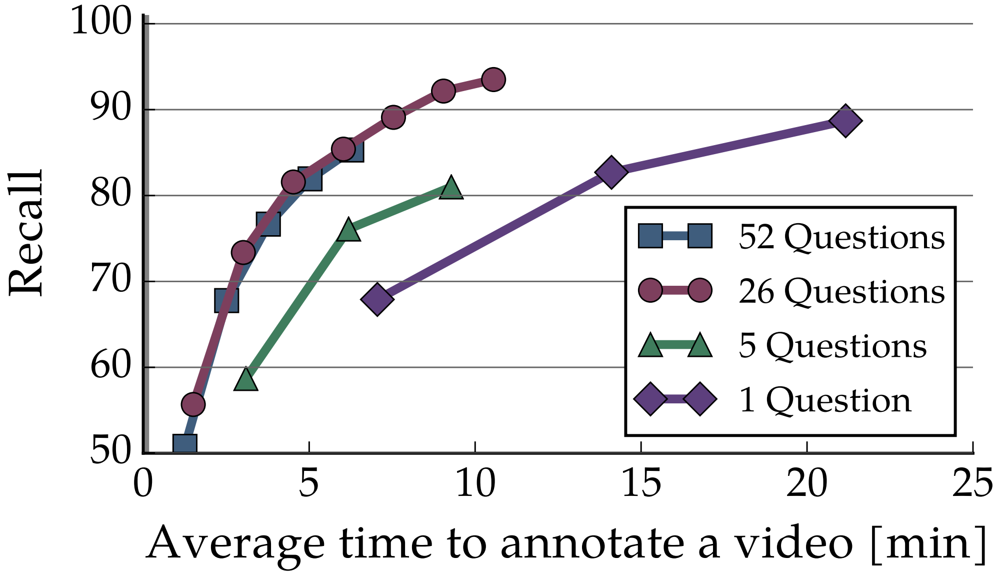
3.3.5 Multi-iteration video annotation

So far we established that (1) the many-questions interface provides a more effective accuracy to annotation cost tradeoff on expectation than the few-questions interface when all other factors are kept the same, (2) the few-questions interface can be further improved by the addition of positive bias and grouping, and (3) the many-questions interface we proposed is optimal as is. In this section we bring all these findings together and conclusively demonstrate that our many-question annotation strategy is strictly better than the few-questions alternatives for practical video annotation.

Advantages of asking multiple questions. In previous sections we computed the expected recall across multiple iterations of annotations for a fixed time budget to compare different methods; here, we report the results in practice. We run multiple iterations of annotation and consider a label positive if at least one worker marks it as such. Thus, recall steadily increases with the number of iterations while precision may drop as more false positives are added.

Fig. 3.9 reports recall and precision as a function of annotation time. For the few-question interfaces (5-questions and 1-question) we include the positive bias and grouping strategies found helpful above. Nevertheless, we observe a clear advantage of the multi-question methods.

For example, given 7.1 minutes required to annotate a video with the 1-question interface, we are able to run two iterations with the 5-question interface (taking up 6.2 minutes), and five iterations with 52-questions (taking up 6.3 minutes). With this annotation budget, the 52-question interface obtains a recall of 85.3%, which is 10.5% higher than the 74.8% recall with 5-questions and 18.6% higher than the 66.7% recall with 1-question. Further, the 52-question interface obtains precision of 81.2%, which is 6.6% higher than the 74.6%



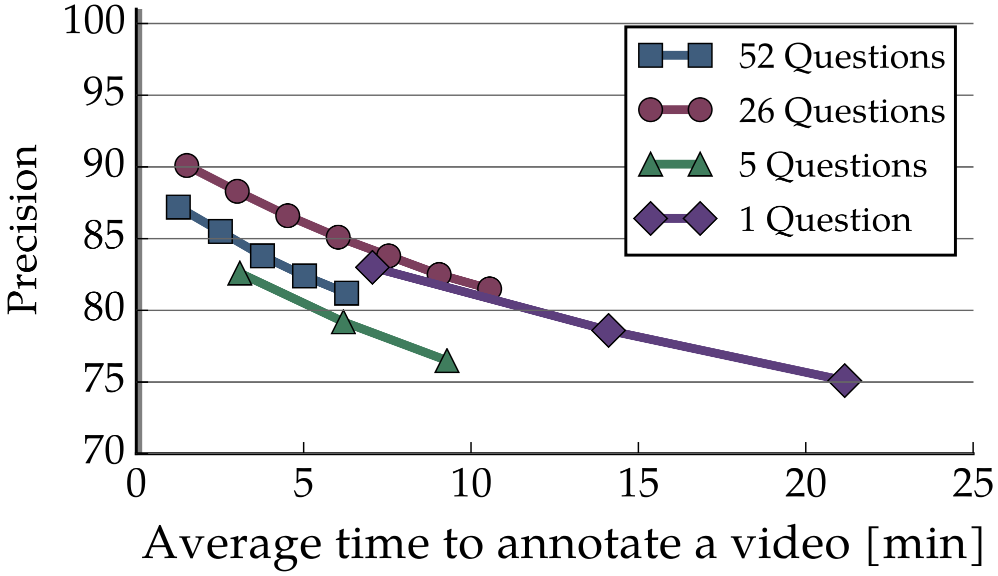


Figure 3.9: Recall (*top*) and precision (*bottom*) with multiple iterations of annotation. Each square represents one iteration. We can see that since each annotation iteration with the 52-question interface is much cheaper, it quickly matches the performance of the more time-costly alternatives.

precision with 5-questions and slightly lower by 1.8% than the 83.0% precision with the 1-question interface.

In another example, in about half the annotation time (3.8 versus 7.1 minutes) we achieve a 10% improvement in recall (76.7% with three iterations of 52-questions versus 66.7% with one iteration of 1-question) at comparable precision (83.8% with 52-questions versus 83.0%

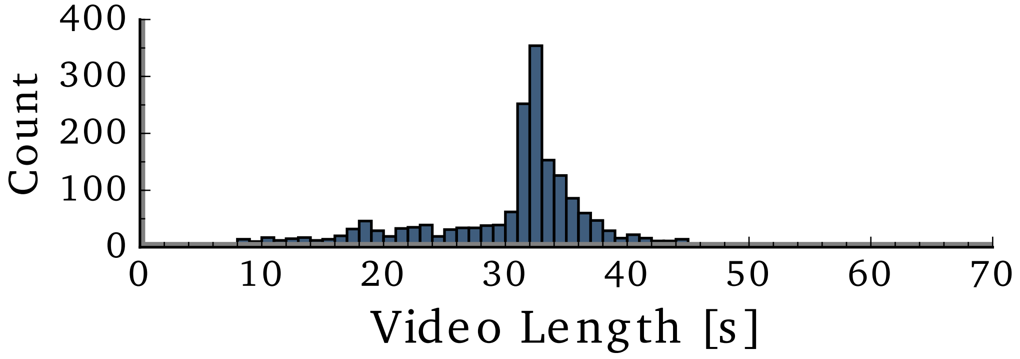


Figure 3.10: Statistics from the dataset. Histogram of the lengths of the videos, where we can see that the videos have various lengths enabling analysis based on content length.

with 1-question). The improvement in recall is statistically significant at 0.01 level using a one-tailed unequal variance t-test.

We conclude that simultaneously asking multiple questions per video, as many as 26 or even 52, is significantly more effective than asking only a handful of questions. When comparing the 26-question and 52-question interfaces in Fig. 3.9, the results are remarkably similar: recall per unit time is almost identical, although precision is slightly (statistically insignificantly) higher for 26-questions. Thus while there are diminishing returns with asking more questions in the same interface, asking "too many" questions per video does not appear to be harmful to annotation quality. Studying the exact point at which the number of questions becomes overwhelming for workers is important future work.

Effect of video length. We investigate whether these conclusions hold for different video lengths – for example, an image is just a zero-length video, so would our conclusions still apply? Our dataset contains videos of varying length as shown in Fig. 3.10 and we group the videos into three groups: 0-20 seconds, 20-40 seconds and 40-60 seconds long.

Fig. 3.11 reports the recall of the different methods for each of the three groups, following the same experimental design as before. For shorter videos that require little time to process, the exact annotation interfaces make little difference. This suggests that in the case of images our method would be as effective as the standard one-question baseline.

Importantly, as the content gets longer the benefit of our method becomes more pronounced. For example, on 40-60 second videos for a fixed annotation budget of 4.4 minutes (enough to run one iteration with the 5-question interface), our 52-question method achieves 62.7% recall compared to only 37.4% with the 5-question baseline (a 25.3% improvement!) and 83.1% precision compared to only 79.4% precision of the 5-question baseline.

3.3.6 Annotated dataset

We used our annotation strategy to collect additional annotations for the video dataset of [223]. This amounted to $443{,}890$ questions answered, resulting in $1{,}310{,}014$ annotations for the $1{,}815$ videos. This increased the density of annotation on the dataset from 3.7 labels per video on average (which were available apriori based on the data collection procedure) to 9.0 labels per video. When evaluating the precision of annotation we additionally collected temporal annotation of *when* the actions took place in the video. This yielded 66,963 action instances. We verified the quality of temporal annotations by collecting duplicate annotations on a subset of the data. Agreement among the workers for temporal annotation was 82.8% using 0.1 temporal intersection-over-union overlap.

Using these temporal annotations, we verify that using our method we are able to successfully annotate both actions that are long and short in the video. For every one of the 157 target actions, we compute the average (median) length of its instances in the videos as well as the recall of our annotations. Fig. 3.12 plots recall as a function of action duration. As expected, recall tends to be slightly higher for actions that are longer in the video but not significantly (Pearson correlation of 0.178). We conclude our method is effective at annotating both long and short events.

3.4 Discussion

We explored the challenging problem of multi-label video annotation. In contrast to insights obtained from studying crowdsourcing of video annotation, we demonstrated that asking multiple questions simultaneously about a video provides the most effective tradeoff between annotation time and accuracy. While we observed that accuracy decreases with additional questions for each video, this drop was not sufficient to warrant the significant cost of only a few questions per video. Furthermore, we observed that the performance gap between cheap fast methods over slow careful methods grows with increasing content length. In conclusion, our results suggest that optimal strategy of annotating data involving time is to minimize the cost in each iteration through sufficiently many questions, and simply run multiple iterations of annotation.

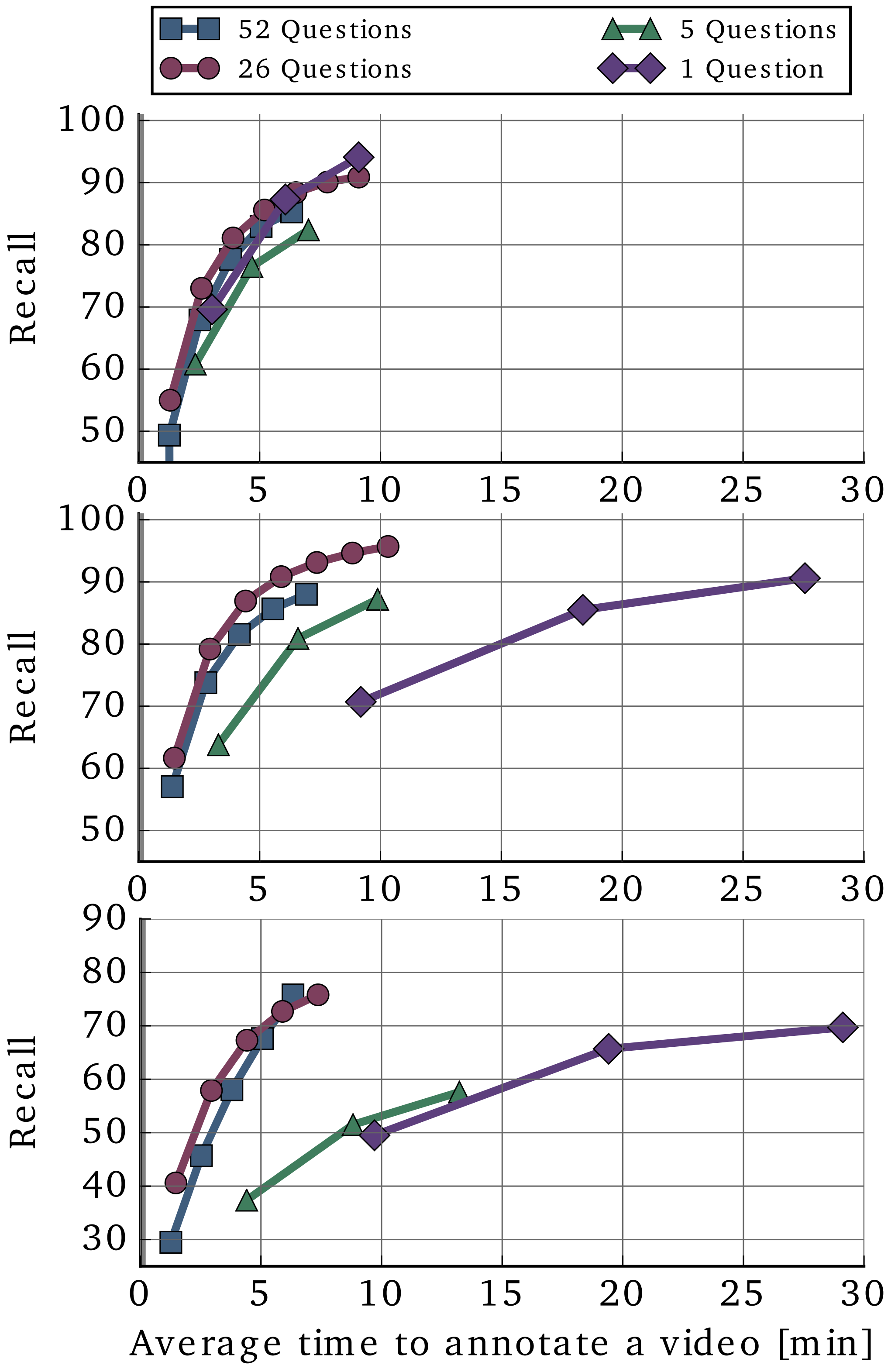


Figure 3.11: Breakdown of Fig. 3.9 for different video lengths: (*top*) 0-20 second videos, (*middle*) 20-40 second videos, (*bottom*) 40-60 second videos. The benefit of the many-question interfaces is more prominent with increased content length.

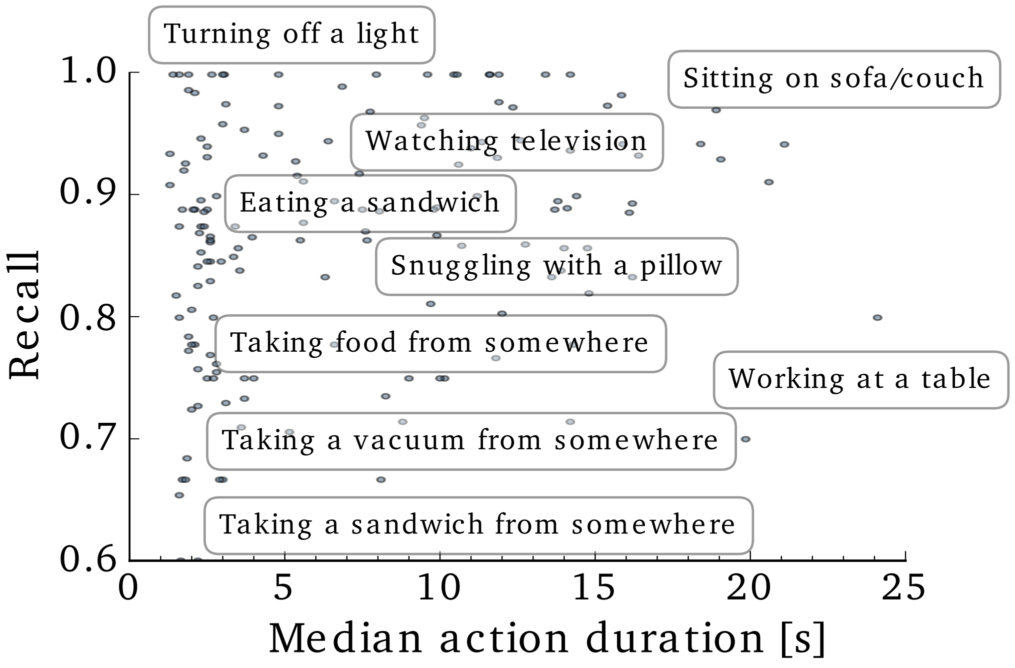


Figure 3.12: Annotation recall (*y-axis*) as a function of the average duration in the video (*x-axis*) for every one of the target 157 actions. Our method for multi-label video annotation is effective for labeling both long- and short-duration events.

Chapter 4

Modeling Human Actions in Videos

After creating, annotating, and benchmarking our video dataset in the previous chapters, we can start investigating where methods succeed and where they fail. That is, what is the right way to reason about human activities? What directions forward are most promising? In this chapter, we analyze the current state of human activity understanding in videos. The goal is to examine datasets, evaluation metrics, algorithms, and potential future directions. We look at the qualitative attributes that define activities such as pose variability, brevity, and density. The experiments consider multiple state-of-the-art algorithms and multiple datasets. The results demonstrate that while there is inherent ambiguity in the temporal extent of activities, current datasets still permit effective benchmarking. We discover that fine-grained understanding of objects and pose when combined with temporal reasoning is likely to yield substantial improvements in algorithmic accuracy. We present the many kinds of information that will be needed to achieve substantial gains in activity understanding: objects, verbs, intent, and sequential reasoning. The software and additional information will be made available to provide other researchers detailed diagnostics to understand their own algorithms.

4.1 Background

Over the last few years, there has been significant advances in the field of static image understanding. There is absolutely no doubt that we are now closer to solving tasks such as image classification, object detection, and even semantic segmentation. On the other hand, when it comes to video understanding we are still struggling to figure out basic questions such as: What is an activity and how should we represent it? Do activities have well-defined spatial and temporal extent? What role do goals and intentions play in defining and understanding activities?

A significant problem in the past has been the absence of good datasets for activity detection and recognition. Most of the major advances in the field of object recognition have come with the creation of generic datasets such as PASCAL [48], ImageNet [36] and COCO [138]. These datasets helped define the problem scope and the evaluation metrics, as

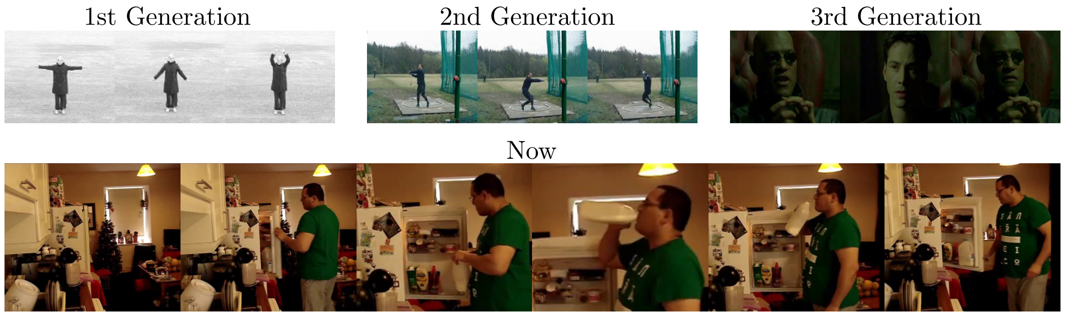


Figure 4.1: Now that the field of activity recognition has moved on from simple motions (KTH [205]), sports (UCF101 [232]), and isolated activities in movies (HMDB51 [124]) to cluttered sequences of home activities (Charades [223]), how should we think about activities? What are the categories? Do activities have well-defined boundaries? In the highlighted video from Charades, a person "walks to the kitchen," "opens the fridge," "grabs some milk," "opens the bottle," "drinks from the bottle," "puts it back," and "closes the fridge." We discuss the problem of how to think about when, where, and what the person is doing at any given time.

well as revealed the shortcomings of existing approaches.

On the other hand, historically video datasets such as UCF101 [232] have been biased and have corresponded to activities that are hardly seen in daily lives. However, in recent years, things have started to change with the arrival of large-scale video datasets depicting a variety of complex human activities in untrimmed videos. Datasets such as ActivityNet [50], Sports1M [106], Charades [223], and MultiTHUMOS [68, 288] have revitalized activity recognition and inspired new research and new ideas.

But before we move forward and define the benchmarks, we believe it is worth pausing and thoroughly analyzing this novel domain. What does the data show about the right categories for recognition in case of activities? Do existing approaches scale with increasing complexity of activities categories, video data, or temporal relationships between activities? Are the hypothesized new avenues of studying context, objects, or intentions worthwhile: Do these really help in understanding videos?

This chapter provides an in-depth analysis of the new generation of video datasets, human annotators, activity categories, recognition approaches, and above all possible new cues for video understanding. Concretely, we examine:

What are the right questions to ask?. Here we investigate some fundamental questions regarding the problem of activity recognition. First, we ask what are the right categories for activities? Unlike objects where semantic categories are somewhat well-defined, activity categories defined by verbs are relatively few in number. Should we model one activity category called "open" or should we model "open suitcase", "open windows" and "open curtain" by different categories. We also ask a fundamental question: whether we should perform classification or localization. Do humans agree on temporal activity boundaries? And if they do not, is it worth exploring localization at all?

What are existing approaches learning, and are those the right things?. In the next set of analyses, we explore the current algorithms for activity classification and localization. We define attributes for quantifying video and category complexity and evaluate algorithms with respect to these attributes. Do current approaches work better when there are multiple activities per video by exploiting the context? Does large variation in pose among categories help in activity classification? The analyses presented in this section have been combined into a single open-sourced tool that automatically renders a summary of the diagnostics and suggest improvements to existing algorithms.

What directions seem most promising?. By considering various ideal components of activity understanding, we evaluate how can we get the next big gain. Should we explore intentions, verbs, and sequences of activities, or only focus on jointly reasoning over objects and activities?

4.2 Evolution of Activity Recognition

Starting with the KTH action dataset [205] we have observed significant advances in understanding activities through new datasets and algorithms. KTH combined with the idea that activities are *motions* sparked numerous advances [31,115,117,127,128]. With increasingly complex datasets such as UCF101 [232] and others [68,106,124,139] came new challenges, including scale, background clutter, and action complexity, in turn leading to improved algorithms [100,166,227,264]. However, these datasets still focused on short individual clips of activities, and commonly sports, which encouraged the next wave of datasets that focus on sequences of everyday activities [50,173,223,288].

These recent datasets present different challenges for activity understanding. ActivityNet [50] includes many activity categories and a dense hierarchy of activities, although each video is only associated with one activity label. THUMOS [68] and its extension MultiTHUMOS [288] provide multi-label sports videos. Charades [223] contains diverse and densely annotated videos of common daily activities occurring in a variety of scenes.

Our setup:. Much of our evaluation focuses on understanding the scope of and the interaction between different activities. Thus we choose Charades [223] as the best testbed for our analysis, and introduce MultiTHUMOS [288], THUMOS [68], and ActivityNet [50] as needed to establish the generality of our conclusion. All datasets use the same normalized mAP metric that is robust to different ratios of positives to negatives as well as different number of categories [88]. Charades contains $9{,}848$ videos, split into $7{,}985$ for training and $1{,}863$ for testing. It contains annotations of 157 activity categories such as "drinking from a cup" and "watching the tv" which happen naturally along with other categories. "watching the tv" might for example occur with "lying on the couch" and "snuggling with a blanket", or "drinking from a cup". That is, activity categories have moved from capturing *verbs* to capturing a variety of (*object,verb*) pairs, and we begin our investigation by analyzing this distinction in more detail.

Baselines:. We evaluate Two-Stream [227][1], Improved Dense Trajectories (IDT) [264], LSTM on top of VGG-16 [292], and two recent approaches: ActionVLAD [65] with sophisticated spatio-temporal pooling and Asynchronous Temporal Fields [217] with a deep structured model.[2]

4.3 What are the right questions to ask?

To start our discussion about activities, let us establish what we want to learn. When we talk about activities, we are referring to anything a person is doing, regardless of whether the person is intentionally and actively altering the environment, or simply sitting still. In this section, we will first look at how to define activity categories, and then investigate the temporal extents of activities.

4.3.1 What are the right activity categories?

Should we focus our analysis on general categories such as "drinking", or more specific, such as "drinking from cup in the living room"? Verbs such as "drinking" and "running" are unique on their own, but verbs such as "take" and "put" are ambiguous unless nouns and even prepositions are included: "take medication", "take shoes", "take off shoes". That is, nouns and verbs form atomic units of actions.

To verify this property of verbs, we ran a human study using workers on Amazon Mechanical Turk. We presented them with Charades videos [223], and asked workers to select which of the 157 activities are present in the video. We looked at the likelihood a worker will choose an activity B when activity A is present in the video but not B. We found that people considered verbs to be relatively more ambiguous. That is, given the verb, workers confused activities with different objects, e.g., "holding a cup" vs "holding a broom," only 0.3% of the time. However given the object, there was more confusion among different verbs, e.g., "holding a cup" vs "drinking from a cup," 1.3% of the time. A similar pattern of confusion between similar categories is seen by state-of-the-art algorithms in Sec. 4.4.1.

To quantitatively determine the distinctiveness of the different types of categories, we looked at pose similarity (Sec. 4.4.4) across categories on the Charades dataset, both for verb supercategories, and (verb,noun) categories. As expected, even with this simple metric there was more distinctiveness across categories for (noun,verb) than verbs ($p<0.01$). This reinforces that verbs alone lack the clarity to provide clear supervision for learning.

4.3.2 Do Activities Have Temporal Extents?

Physical objects have clear boundaries defined by their physical extent, which is visible as a depth change with respect to the viewer. Activities, however, have few such clear boundaries. How should we evaluate *when* each activity is happening? Would two people even agree on this?

[1]The Two-Stream network uses a soft-max loss function and randomly samples positive training examples from each class at training time.

[2]We primarily use the test set predictions of these models released by [217] available at `github.com/gsig/temporal-fields`; for ActionVLAD we use predictions provided by [65].

We looked at how well human annotators agreed with the activity boundaries by asking Amazon Mechanical Turk workers to re-annotate the extent of actions in the Charades and MultiTHUMOS videos. We make three observations.

First, the average agreement with ground truth was only 72.5% intersection-over-union (IOU)[3] for Charades and 58.7% IOU in MultiTHUMOS, indicating that temporal extent of an activity is ambiguous even for humans. The median starting error was less than the ending error (0.9 ± 0.8 sec compared to 1.4 ± 1.4 sec in Charades), which suggests that more of the confusion is about the end of the activity.

Second, there is a significant positive correlation between IOU agreement and the duration of activity ($\rho{=}0.50$) suggesting that longer activities tend to be easier for humans to localize. Further, humans tend to better agree on the starting point of longer activities compared to shorter activities: the starting error in seconds decreases with longer duration ($\rho{=}0.18$). This suggests that categories of temporally brief activities may require more careful annotation.

Finally, the difference in IOU agreement between categories in Charades was lower than average IOU agreement (13.0% IOU standard deviation compared to 27.5% average IOU), implying that the ambiguity in temporal boundaries is a common problem for many activity categories.

This analysis suggests that there is inherent ambiguity in precisely localizing activities, which primarily depends on the length of the activity, and evaluation metrics must account for this. Furthermore, this suggests that algorithms might benefit from treating the activity boundaries as fluid, which is related to jointly reasoning about the whole video and the boundaries therein.

Can we evaluate temporal localization?. This raises a natural question: if there is such inherent ambiguity in localizing activities, is evaluating localization simply too erroneous? To analyze this, we experimented with omitting ambiguous boundary regions from the test set. Concretely, for a ground truth action instance with temporal extent $[t_1, t_2]$, we define the boundary as $B{=}[t_1{-}\alpha, t_1{+}\alpha] \cup [t_2{-}\alpha, t_2{+}\alpha]$ with $\alpha{=}(t_2{-}t_1)/3$.

We again consider human annotations and compute consensus IOU ignoring the boundary region in both the intersection and the union. The human consensus increases from 72.5% IOU to 79.8% IOU. We found that the center (33%) of an activity was likely to be agreed upon, where 82.7% of the center was covered by a subsequent annotator.[4]

To investigate if this is significant on algorithmic evaluation, we use a Two-Stream network [227] (Sec. 4.2), which outputs per-frame predictions for 157 activities. We evaluate its per-frame accuracy following the Charades metric [223] ignoring the ambiguous boundary regions. Accuracy increases from 9.6% mAP to 10.9% mAP (0.1-4.3% increase on other datasets). Looking at state-of-the-art baselines on Charades (Sec. 4.2) we find that they increase by $0.96 \pm 0.33\%$ mAP where 0.33% mAP is less than the average difference between methods 1.26% mAP. Thus, this does not affect order for evaluation. This suggests that despite boundary ambiguity, current datasets allow us to understand, learn from, and evaluate the temporal extents of activities.

[3]Equivalently, only 76.1% were over 50% IOU.

[4]Training a Two-Stream (RGB) [227] baseline on only this part of the activity yielded an improvement in classification, 15.9% to 16.7% mAP.

Should we evaluate temporal localization?. We ask one final question. When videos are short, is it perhaps unnecessary to worry about localization at all? We measured how well a perfect video classifier emitting the same binary prediction for every frame would fare on localization accuracy. In Charades, videos are 30.1 seconds on average and localization accuracy of this oracle was 56.9% mAP. That is, a perfect classifier would automatically do 5 times better than current state-of-the-art [217] on activity localization. Similarly, a perfect classifier improves over our localization baseline on all the datasets.

This suggests that focusing our attention on gaining more insight into activity classification would naturally yield significant improvements in localization accuracy as well. Further, as we will see, understanding temporal relationships is important for perfect classification; with the complete understanding of activities and their temporal relationships needed to get perfect classification accuracy, we stand to benefit in terms of localization as well. In the rest of this chapter, we focus on classification accuracy, but report localization accuracy to gain a deeper understanding.

4.4 What are existing approaches learning?

Having concluded that (1) we should be reasoning about activities as (*verb,object*) pairs rather than just *verb*, that (2) temporal boundaries of activities are ambiguous but nevertheless meaningful, and that (3) classification of short videos is a reasonable proxy for temporal localization, we now dig deeply into the state of modern activity recognition. What are our algorithms learning? In Sec. 4.4.1 we start by analyzing the errors made by state-of-the-art approaches; in Sec. 4.4.2 we analyze the effect of training data and activity categories; in Sec. 4.4.3 we look at the importance of temporal reasoning; and in Sec. 4.4.4 we emphasize the importance of person-based reasoning. All annotated attributes along with software to analyze any new algorithm along with diagnostics from the software for various algorithms are available at `github.com/gsig/actions-for-actions`.

Setup:. The models described in Sec. 4.2 are trained on the Charades training videos and we evaluate classification mAP on the Charades test set unless stated otherwise. Accuracy is measured using mAP, which is normalized [88] as to be comparable across different subsets of the data with different numbers of positive and negative examples. Please refer to the Appendix for details.

The category plots are generated from (x, y) pairs where x is a an attribute for a category and y is the classification performance for the category. Finally, the pairs are clustered by the x coordinate and the average of the y coordinates visualized. Error bars for category plots represent one standard deviation around Two-Stream based on the y values in each cluster. In this section we report Pearson's ρ correlation between x and y. The video plots are generated similarly by clustering the videos based on the attributes. Finally the mAP is calculated in that group of videos. Error bars for video plots represent the 95% confidence interval around Two-Stream obtained via bootstapping [49].

4.4.1 Analyzing correct and incorrect detections

What kind of mistakes are made by current methods?

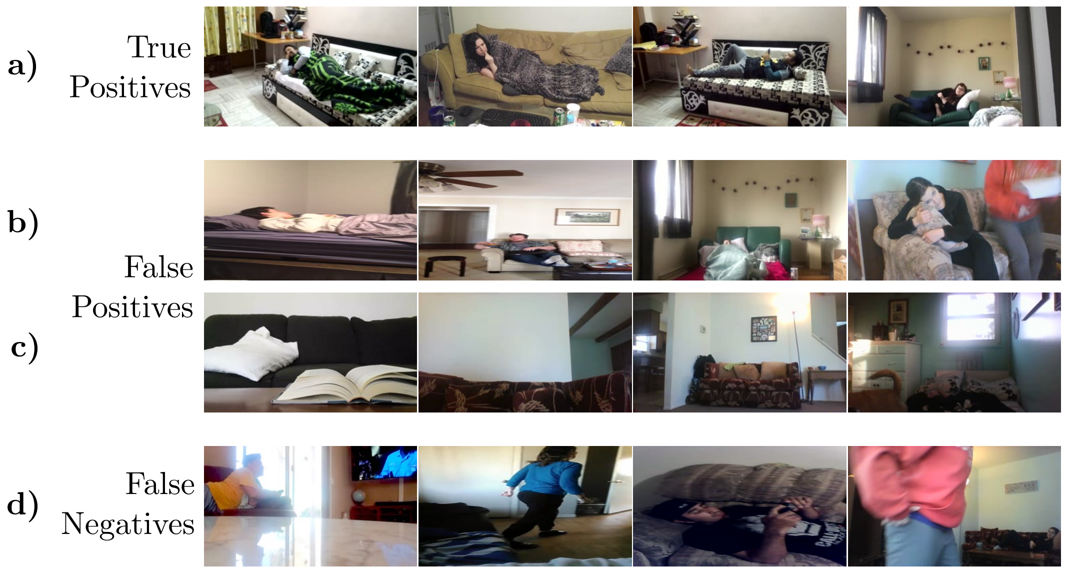


Figure 4.2: Example results from a Two-stream network [227] on per-frame classification of the "Lying on a couch" action in Charades. a) High-scoring true positives commonly include canonical views of both a couch and a person. Top false positives often include b) confusion with other objects (e.g., beds) or verbs (e.g., sitting) and c) the correct scene but with an absent action (e.g., vacant couches). d) Top false negatives include unusual viewpoints.

To understand what current methods are learning and motivate the rest of this section, we start by highlighting some of the errors made by current methods. First, we look at visual examples of the errors that a Two-Stream Network [227] makes on Charades. In Fig. 4.2 we see correct classifications, as well as three types of errors. The figure suggests that 1) models need to learn how to reason about similar categories (Sec. 4.4.2); 2) methods have to develop temporal understanding that can suppress temporally similar but semantically different information (Sec. 4.4.3); and, 3) models need to learn about humans and not assume if couch is detected, then "Lying on a couch" is present (Sec. 4.4.4).

To provide further insight into the algorithms we provide brief overview of the types of errors made by multiple algorithms. In Fig. 4.3 we look at the relative types of errors. First, we note that not many errors are made close to the boundary. However, we can see that significant confusion is among similar categories, both for verbs and objects, interestingly TFields [217] confuse unrelated categories much less (29.1% compared to $41.0 - 44.0\%$), but at the cost of confusing categories with similar object. In fact, for a given category the more categories share the object or verb, the worse the accuracy. This is visualized in Fig. 4.4, where we consider how many categories share an object/verb with the given category (*Object/Verb complexity*). This suggests that moving forward fine-grained discrimination between activities with similar objects and verbs is needed.

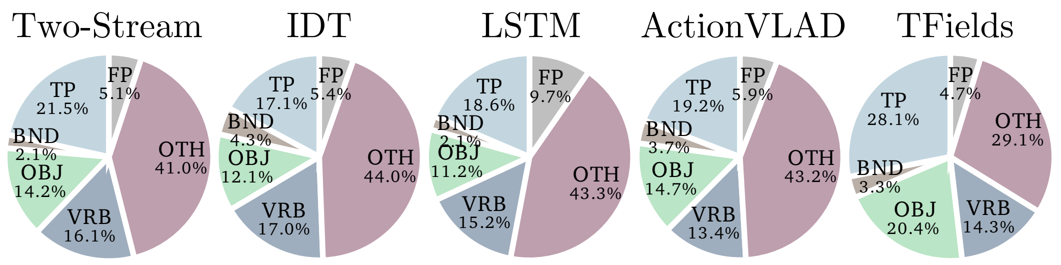


Figure 4.3: Fraction of top ranked predictions for each class that are correct (TP), on the boundary (BND), other class with same object (OBJ), other class with same verb (VRB), other class with neither (OTH), or no class (FP). Inspired by Hoiem et al. [88].

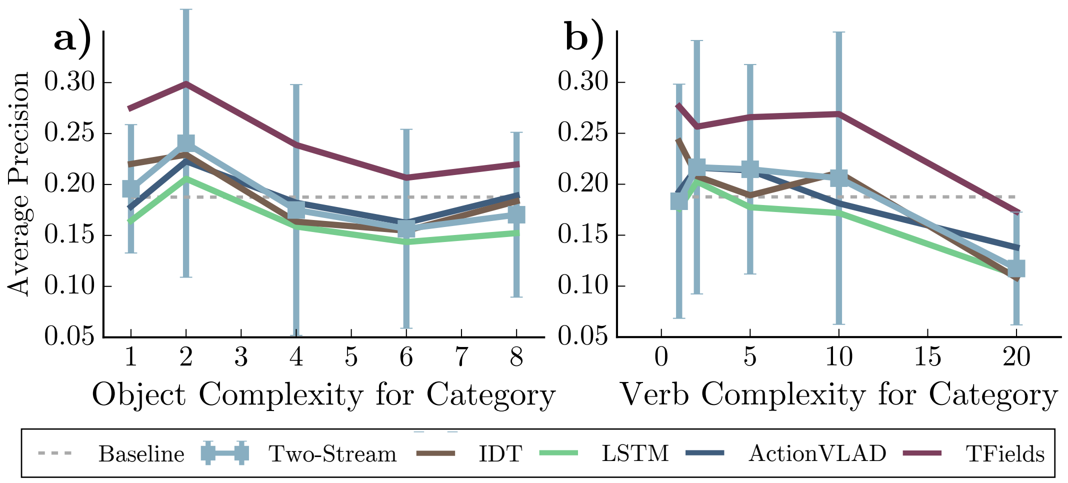


Figure 4.4: Accuracy of activity categories for multiple methods as a function of how many categories share its object/verb.

4.4.2 Training Data

Are current methods limited by the available data? How do we handle imbalanced data and similar categories?

To understand how training data affects current algorithms, we first look at the relationship between amount of data and accuracy to understand how to better make use of data and what kind of data we need more of.

We train a Two-Stream (RGB) [227] baseline and focus on Charades, since it has naturally occurring long-tailed distribution of classes. With all of the available training data we get 15.6% mAP, and with $\frac{7}{10}$ and $\frac{3}{10}$ of the data we get 14.7% and 11.6% mAP. This is expected—more data is better. However, what kind of more data is better? For example there is not a clear relationship for larger categories to have higher accuracy in the datasets (ρ<0.10). Why is more data not helping these categories?

Focus on small categories.. First, we note that most categories seem to benefit from more data even in other categories. Since Charades has different number examples in each class, removing $\frac{1}{2}$ of the data removes many more examples from large categories than small. Even so, small categories see a larger decrease in accuracy (ρ=0.18, p=0.03). In the extreme, limiting all categories to at most 100 examples, there is still no significant relationship between category size and drop in accuracy.

The only significant correlation we found with this drop in accuracy when limiting all categories to at most 100 examples was with the number of similar categories (that share the same object/verb). Categories with more similar categories had more reduction in performance (ρ=0.18, p<0.05). This suggests that *there is no such thing as balanced data*. Any attempts at reweighting and resampling without considering similarity between categories is unlikely to help all small categories uniformly. For example, when limiting categories to at most 100 samples, the range in relative change in accuracy is from a 65.8% *decrease* ("holding a laptop") to 52.2% *increase* ("standing on a chair"). The highest drop is in a category that has 5 categories that share the object, and 18 categories that share the verb. "Standing on a chair" however is a relatively unique activity with only 42 examples. Investigating this large difference in performance when balancing the data will be important to fully harness this kind of activity data.

Focus on large categories.. Categories that are naturally ubiquitous and have many examples in naturally long-tailed datasets seem to have additional complexity that outweighs the advantage of having more training data. For example in Charades, they have slightly more pose diversity: concretely, the correlation between the number of training examples of a category and its pose diversity (as defined in Sec. 4.4.4) is ρ=0.09. Second, they tend to have less inter-class variation. For example, categories with more examples have more categories that share the same object ρ=0.13 ("object complexity" in Sec. 4.4.1). Thus more common actions may in fact be more challenging to learn.

Interestingly, looking at current methods, we find that the main improvement in accuracy does not come from models that are better able to make use of the wealth of data in large categories, but rather in small categories. This is visualized in Fig. 4.5 both for number of training examples as well as training frames. This both suggests that developing models that generalize well across categories is clearly beneficial, but also that more expressive models are needed for large categories, perhaps by dividing them into subcategories.

4.4.3 Temporal Reasoning

How does the temporal extent of activities affect accuracy? Should we think about activities in terms of high-level temporal reasoning?

Given that Two-Stream networks, which operate on isolated RGB and Optical Flow frames, are on par with state-of-the-art on many recent datasets, it is either that temporal reasoning beyond instantaneous motion is not needed for activity recognition, or current methods are missing important pieces of temporal reasoning. In this section we look at current algorithms at increasing granularity of temporal reasoning: motion, continuity, and temporal context.

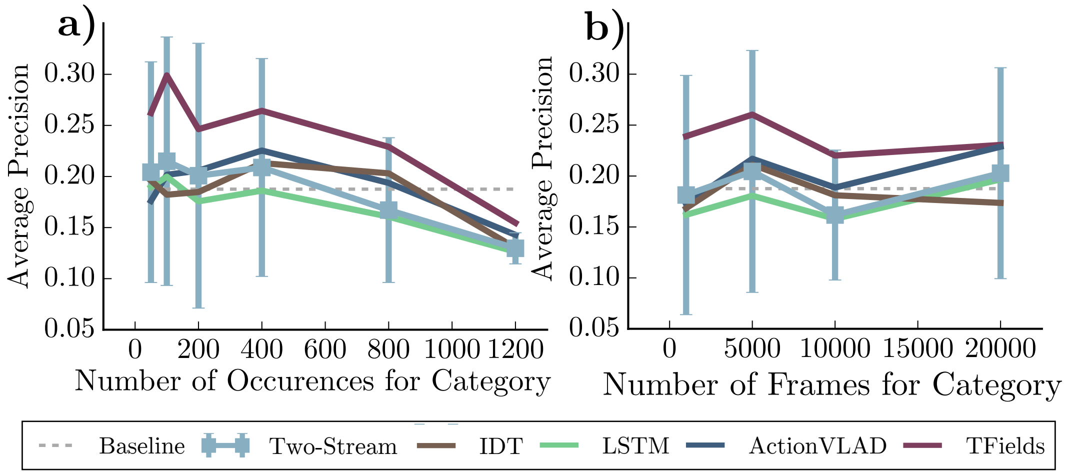


Figure 4.5: Accuracy of activity categories for multiple methods as a function of training examples/frames.

Motion and temporal continuity.. Most activities involve movement that causes blurred frames, intermittent observability, and large visual variation. In theory, algorithmic approaches must be robust to these effects by combining multiple predictions over time. To analyze how well current algorithms combine predictions to, for example, reason in the presence of motion, we consider average amount of optical flow in a given category on average (*Motion for Category*) and the average temporal extent of activities in each category (*Average Extent for Category*). This is visualized in Fig. 4.6. We find that instantaneous motion affects algorithms differently, for example, more temporal reasoning seems to help (IDT [264], TFields [217]) algorithms be robust to motion. Furthermore, short actions do noticeably worse on the datasets (ρ=0.21$-$0.42). We could expect that longer activities indeed do better because they have more training data, but this view was refuted in Sec. 4.4.2. This suggests that current methods are better at modeling longer activities than shorter ones, implying that more emphasis may be needed on understanding shorter activities. This both suggests that brief patterns, motion and short actions, need more temporal reasoning to be understood. One potential avenue for exploration would be combining the trajectory representation in IDT (which appears to help on shorter activities) with the benefits of longer-term pooling of Two-Stream (which works well on longer activities).

How well do these algorithms combine predictions over time? As it turns out, naive temporal smoothing of the predictions helps improve localization and classification accuracy, where all methods except LSTM benefit from averaging predictions over a window 4% of the video size (1.2 sec on average). However, for optimally smoothed Two-Stream for example, the relative change in accuracy for individual classes varies from 24.7% decrease ("Holding a broom") to 32.4% increase ("Closing a window"). Furthermore, smoothing helps larger classes more (ρ=0.22, p<0.01), but does not help classes with much motion. This leads to the conclusion that combining predictions over time is a non-trivial problem that must be addressed in future work.

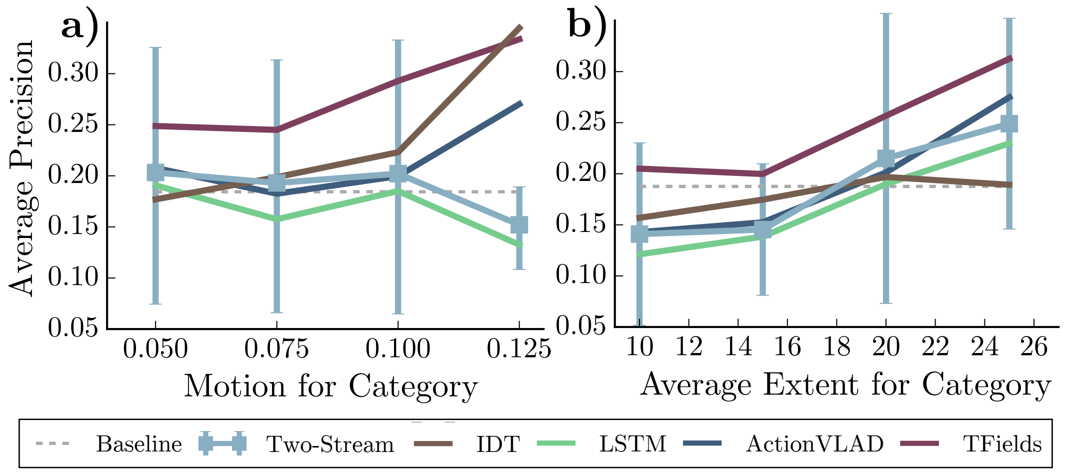


Figure 4.6: Accuracy as a function of motion and temporal extent for categories in Charades.

Temporal Context.. Moving to larger temporal scales, we now investigate how current methods utilize temporal context. We noticed that for videos with 1-4 activities per video, Two-Stream [227] obtains 22.1% mAP, for videos with 14 or more, the accuracy is 16.6% mAP. One could expect that with more activities, the additional context could be used to improve performance, but the additional complexity seems to outweigh any additional context. This pattern also applies to temporal overlap (multiple activities happening at a given instant) where classes that overlap frequently in the training data are significantly more confused at test time (ρ=0.28 correlation between percent of training frames where classes overlap, versus score given to class A when class B is present but not A).

In fact, the methods vary significantly in terms of how much they use context. In Fig. 4.7 we visualize how much each method benefits from context; we measure for each action how many other actions on average increase the prediction confidence of that action by being present in a video. We consider video classification, and since all methods do combine their predictions to make a video prediction, context is being used in some form by all methods. We observe that high-level temporal modeling in TFields [217] helps utilize context, and is important moving forward.

4.4.4 Person-based Reasoning

Should activity recognition be image-based or person-based? Should activity recognition models be explicitly taught what a person is?

In CNNs for object recognition, it has been observed that object parts are discovered automatically [298]. Ideally, we would want the person to be automatically discovered by activity recognition systems. But is this happening now?

Person location in the frame.. First, we look at the average size of a person in a video as measured by the highest-scoring per-frame Faster-RCNN bounding box detection [209] (*Person Size in Pixels*), as well as whether there are more than one person in the video (*More*

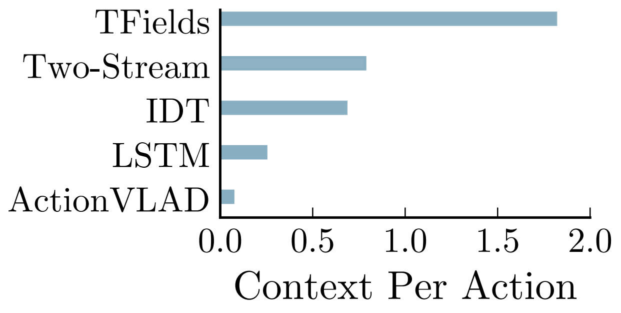


Figure 4.7: How many actions positively influence each action by being present. Presented for multiple algorithms on Charades.

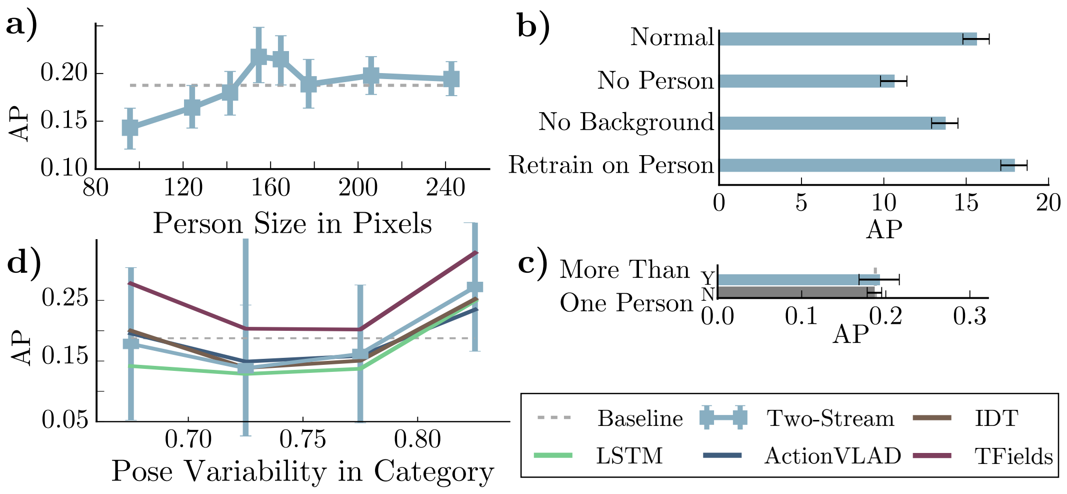


Figure 4.8: Supporting plots for Sec. 4.4.4. Visualizing the impact of attributes on classification performance in the Charades Dataset.

than One Person). We look at how classification performance changes with these attributes. From Fig. 4.8a,c we notice that there is significant dependency on the size of the person, and that there seems to be an optimal size in the middle. This pattern was observed on multiple datasets, where models perform the best on actions with medium-sized people (48.7-78.3 pixels in size) and worse on small or large. Having multiple people in the video does not significantly affect accuracy. This suggests that the networks are not properly latching onto the person in the scene.

To investigate this further we ran ablative studies on the network. We removed the person from the scene (*No Person*) and then removed everything but the person from the scene (*No Background*) and evaluated the network accuracy. The removal was done by using the same Faster-RCNN bounding box detection and by setting all removed pixels to the average value. In Fig. 4.8b, we can see that removing the person from the test images does more damage than removing other parts of the image. Finally, we retrain the Two-stream network [227]

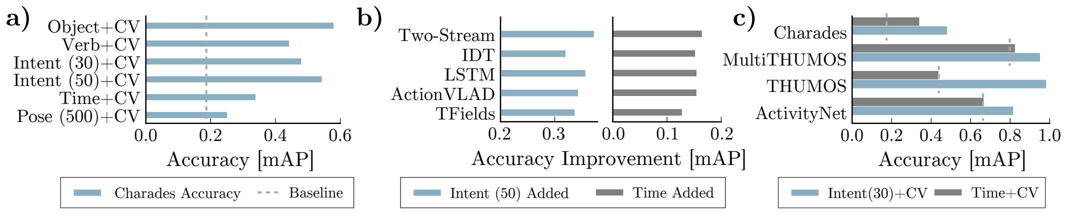


Figure 4.9: Algorithms combined with oracles. a) Multiple types of perfect information combined with baseline on Charades. b) Two types of perfect information combined with different baselines. c) Two types of perfect information combined with a baseline on multiple datasets.

on only the cropped image of the person (*Retrain on Person*). Retraining on the cropped image yields a noticeable improvement in accuracy (15.6% to 17.9% mAP). This suggests that person-focused reasoning may be beneficial to current algorithms.

Pose.. Daily human activities are centered around the person. Even so, many top scoring methods get good performance without explicitly reasoning about human poses. First, we found that the variability of a pose in each category significantly determined the accuracy on Charades (ρ=0.28). Pose variability is the average Procrustes distance [108] between any two poses in the category, which aligns the poses[5] with a linear transformation and euclidean distance between corresponding joints (*Pose variability*). We visualize how the accuracy in a category increases as the category contains more diverse poses in Fig. 4.8d. Second, pose *similarity* between poses in two categories determines how much two categories are confused at test time in Charades (ρ=0.39) where we consider the same metric but across categories. This demonstrates that poses play a significant role in modern human activity recognition and harnessing poses is likely to lead to substantial advances.

4.5 Where should we look next?

Now we have analyzed state-of-the art algorithms in terms of various attributes and identified some strengths and weaknesses. To try to understand what directions look promising, we consider what aspects would help the most if solved perfectly, i.e., with an oracle.

Types of oracles:. We study the effectiveness of five oracles on action recognition in Charades [223], MultiTHUMOS [288], THUMOS [68], and ActivityNet [50] datasets.

(1) *Perfect Object Oracle* assumes the list of objects that the person is interacting with in the test video is given. There are 38 objects in Charades. Given this list of objects, we predict that all actions associated with these objects are present, and all other actions are absent from the video.

(2) *Perfect Verb Oracle* is similar except it assumes the list of verbs that the person is executing in the test video is given. There are 33 verbs in Charades.

[5]Poses were obtained using Convolutional Pose Machines [273].

(3) *Perfect Temporal Oracle* assumes that for each frame of the test video, the last activity to end and the next activity to begin are given. There are 157 activities in Charades, 65 in MultiTHUMOS, 20 in THUMOS, and 200 ActivityNet. From the video annotations we learn the distribution of activities that is likely occur in this frame given this information.[6] This produces a probability distribution of actions in each frame; we max-pool over all frames to obtain predictions for the entire video.

(4) *Perfect Intent Oracle* is the trickiest. Each video contains multiple labels. When thinking about *Intent* we can imagine that these labels occur together in certain ways. For example, "put on clothes" "put on shoes" "open door", might be associated with the intent of leaving the house. We cluster the labels from all the videos into 30 or 50 clusters.[7] Each intent cluster thus corresponds to a distribution of activities. A perfect intent oracle gives which cluster the video corresponds to. Given this cluster, the distribution of activities within the cluster is used as the activity prediction.

(5) *Perfect Pose Oracle* We cluster the poses in all frames into 500 clusters. Given the cluster, the distribution of activities within the cluster is used as the activity prediction.

These oracles should be thought of as lower bounds for these types of information. That is, a method using these types of information should do at least this well on the datasets. However, it is likely that better performance could be obtained with better ways of using perfect information.

Comparing different oracles:. We start by evaluating these oracles on video classification on Charades. To do so, we design a simple video classification method using each oracle that is combined with Two-Stream (RGB) [227] on each of the datasets by multiplying their probabilities; results are presented in Fig. 4.9a. All of these oracles individually are beyond current state-of-the-art of computer vision, which suggests room for improvement in many directions. Object understanding is more effective than temporal reasoning on its own in Charades. This suggests object understanding is important moving forward. Getting a benefit from poses proved challenging, likely because this oracle samples individual poses throughout the video in isolation, and does not consider the motion. The accuracy of only 30 types of intent suggests that more research into how to understand intent is likely to yield substantial gains.[8]

Comparing methods in terms of oracles:. Next, we selected oracles on Charades to highlight differences between current methods. In Fig. 4.9b we see how much is gained by combining perfect intent and time with the baselines. Adding time helps Two-Stream the most, since Two-Stream does not model temporal information beyond optical flow. Interestingly, intent helps IDT [264] the least, which suggests the trajectory information captures high-level information missing from Two-Stream based approaches. To summarize, development of algorithms that combine complementary benefits of IDT, LSTM [292] and global pooling such as ActionVLAD [65] is likely to increase accuracy.

[6]We use simple first-order statistics, i.e., the probability of an action a in the current frame given that action a_p occurred before and action a_n occurs after is $p(a|a_p, a_n) = p(a|a_p)p(a|a_n)$.

[7]Each video can be thought of as a 157 dimensional vector with 0s and 1s based on what categories are in the video. We cluster these vectors with cosine distance and spectral clustering.

[8]We attempted using Two-Stream to predict the 30 intent clusters in Charades (by comparing their label distributions), however this only predicted the right cluster 15.1% of the time, suggesting that intent clustering should be joint with learning discriminative intent clusters.

Comparing datasets in terms of oracles:. Finally, we selected two oracles to compare different datasets; results are presented in Fig. 4.9c. For the datasets with fewer categories, having 30 types of intent given from the oracle, gives almost a perfect score. Although Charades and ActivityNet have 200 classes, they see improvement from only 30 types of intent. Temporal oracle helps the datasets with multiple actions per video, Charades [223] and MultiTHUMOS [288], but not the detection oriented datasets ActivityNet [50] and THUMOS [68]. The datasets address different needs, and these results highlight that some can be useful for developing better high-level temporal modelling and others for better ways of combining predictions to detect activities.

4.6 Discussion

Our analysis of action recognition in videos is inspired by the diagnosis and analysis paper of Hoiem at al. [88] and a long line of meta-analyses that have been done in other domains: e.g., studying dataset bias in image classification [243], analyzing sources of errors in object detection [40,88,92,110,167,194], understanding image segmentation [249], and investigating specific classes of models such as CNNs [295] or LSTMs [104]. Several studies have surveyed action recognition [177,247,274] but to the best of our knowledge we are the first to study it in this level of depth.

We have analyzed multiple attributes of activities, several modern activity recognition algorithms, and the latest activity datasets. We demonstrated that even though human disagreement and ambiguity are an inevitable part of activity annotation, they do not present significant roadblocks to progress in activity understanding. We showed that more detailed understanding of scenes depicted in videos, at the level of individual objects and human poses, holds promise for the next iteration of algorithms. We showed that this generation of rich, multi-label, fine-grained activity benchmarks provides ample opportunities for complex joint high-level reasoning about human activities. We hope the community learns from our analysis, and builds upon our work.

Appendix.. We look at the performance among different subsets of data. Without any consideration, random chance performance would be different on different subsets. For mAP we use normalized precision $P(c)$ similar to [88]: $P(c)=\frac{R(c)\cdot N_{\text{pos}}}{R(c)\cdot N_{\text{pos}}+F(c)\cdot N_{\text{neg}}}$, where $R(c)$ is the recall and $F(c)$ is the false positive rate. N are set to the average numbers on the Charades test set.

Part II

Algorithms for Video Understanding

Video data is an unique source of information, with challenges fundamental to intelligence. For example, reasoning over time, credit assignment, concept learning, and attention. One of the primary problems video understanding has recently faced is converting successful image architectures to handle video input. This problem has seen significant progress for video lengths up to few seconds, and our analysis indicates low-level temporal reasoning beyond a few seconds is likely not needed (Chapter 7). This suggests that specific high-level temporal reasoning is needed for large time-scales, and detailed low-level temporal reasoning is needed for few second time-scales. To improve the few second time-scale modeling, we explore techniques that better model people, such that their intentions, and better attend to the available data, to enable synthetic eye movement with neural networks in world coordinates. This is covered in Chapter 5 and Chapter 7. In Chapter 5 and Chapter 6 we then explore ideas for extending temporal reasoning from seconds to minutes and weeks.

Chapter 5

Modeling Long Activities in Video

Our first algorithmic contribution tackles how to model activities that are approximately 30 seconds long, such as the ones in our Charades dataset. Actions are more than just movements and trajectories: we cook to eat and we hold a cup to drink from it. A thorough understanding of videos requires going beyond appearance modeling and necessitates reasoning about the sequence of activities, as well as the higher-level constructs such as intentions. But how do we model and reason about these? We propose a fully-connected temporal CRF model for reasoning over various aspects of activities that includes objects, actions, and intentions, where the potentials are predicted by a deep network. End-to-end training of such structured models is a challenging endeavor: For inference and learning we need to construct mini-batches consisting of whole videos, leading to mini-batches with only a few videos. This causes high-correlation between data points leading to breakdown of the backprop algorithm. To address this challenge, we present an asynchronous variational inference method that allows efficient end-to-end training. Our method achieves a classification mAP of 22.4% on the *Charades* [223] benchmark, outperforming the state-of-the-art (17.2% mAP), and offers equal gains on the task of temporal localization.

Consider the video shown in Figure 5.1: A man walks through a doorway, stands at a table, holds a cup, pours something into it, drinks it, puts the cup on the table, and finally walks away. Despite depicting a simple activity, the video involves a rich interplay of a sequence of actions with underlying goals and intentions. For example, the man stands at the table 'to take a cup', he holds the cup 'to drink from it', etc. Thorough understanding of videos requires us to model such interplay between activities as well as to reason over extensive time scales and multiple aspects of actions (objects, scenes, etc).

Most contemporary deep learning based methods have treated the problem of video understanding as that of only appearance and motion (trajectory) modeling [66,152,226,244]. While this has fostered interesting progress in this domain, these methods still struggle to outperform models based on hand-crafted features, such as Dense Trajectories [264]. Why such a disconnect? We argue that video understanding requires going beyond appearance modeling, and necessitates reasoning about the activity sequence as well as higher-level constructs such as intentions. The recent emergence of large-scale datasets containing rich sequences of realistic activities [223,275,288] comes at a perfect time facilitating us to explore such complex reasoning.

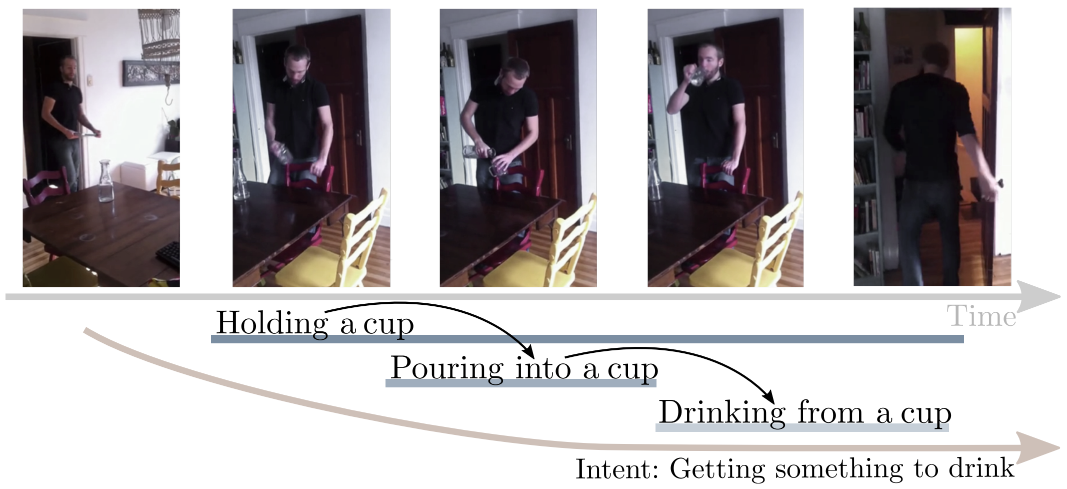


Figure 5.1: Understanding human activities in videos requires jointly reasoning about multiple aspects of activities, such as 'what is happening', 'how', and 'why'. We present an end-to-end deep structured model over time trained in a stochastic fashion. The model captures rich semantic aspects of activities, including *Intent* (why), *Category* (what), *Object* (how). The figure shows video frames and annotations used in training from the *Charades* [223] dataset.

But what is the right way to model and reason about temporal relations and goal-driven behaviour? Over the last couple of decades, graphical models such as Conditional Random Fields (CRFs) have been the prime vehicles for structured reasoning. Therefore, one possible alternative is to use ConvNet-based approaches [122] to provide features for a CRF training algorithm. Alternatively, it has been shown that integrating CRFs with ConvNet architectures and training them in an end-to-end manner provides substantial improvements in tasks such as segmentation and situation recognition [23,286,296].

Inspired by these advances, we present a deep-structured model that can reason temporally about multiple aspects of activities. For each frame, our model infers the activity category, object, action, progress, and scene using a CRF, where the potentials are predicted by a jointly end-to-end trained ConvNet over all predictions in all frames. This CRF has a latent node for the intent of the actor in the video and pair-wise relationships between all individual frame predictions.

While our model is intuitive, training it in an end-to-end manner is a non-trivial task. Particularly, end-to-end learning requires computing likelihoods for individual frames and doing joint inference about all connected frames with a CRF training algorithm. This is in stark contrast with the standard stochastic gradient descent (SGD) training algorithm (backprop) for deep networks, where we require mini-batches with a large number of independent and uncorrelated samples, not just a few whole videos. In order to handle this effectively: (1) we relax the Markov assumption and choose a fully-connected temporal model, such that each frame's prediction is influenced by all other frames, and (2) we propose an asynchronous method for training fully-connected structured models for videos. Specifically, this structure allows for an implementation where the influence (messages) from other frames are approximated by emphasizing influence from frames computed in recent iterations. They are more accurate, and show advantage over being limited to only neighboring frames. In addition to being more suitable for stochastic training, fully-connected models have shown increased performance on various tasks [118, 296].

In summary, our key contributions are: (a) a deep CRF based model for structured understanding and comprehensive reasoning of videos in terms of multiple aspects, such as action sequences, objects, and even intentions; (b) an asynchronous training framework for expressive temporal CRFs that is suitable for end-to-end training of deep networks; and, (c) substantial improvements over state-of-the-art, increasing performance from 17.2% mAP to 22.4% mAP on the challenging Charades [223] benchmark.

5.1 Background

Understanding activities and actions has an extensive history [31, 115, 126–128, 150, 171, 177, 264, 274]. Interestingly, analyzing actions by their appearance has gone through multiple iterations. Early success was with hand-crafted representations such as Space Time Interest Points (STIP) [127], 3D Histogram of Gradient (HOG3D) [115], Histogram of Optical Flow (HOF) [128], and Motion Boundary Histogram [31]. These methods capture and analyze local properties of the visual-temporal datastream. In the past years, the most prominent hand-crafted representations have been from the family of trajectory based approaches [126, 150, 171, 264], where the Improved Dense Trajectories (IDT) [264] representation is in fact on par with state-of-the-art on multiple recent datasets [68, 223].

Recently there has been a push towards mid-level representations of video [97, 125, 198, 231], that capture beyond local properties. However, these approaches still used hand-crafted features. With the advent of deep learning, learning representations from data has been extensively studied [33, 66, 100, 106, 131, 213, 227, 239, 244, 259, 266, 284]. Of these, one of the most popular frameworks has been the approach of Simonyan et al. [227], who introduced the idea of training separate color and optical flow networks to capture local properties of the video.

Many of those approaches were designed for short clips of individual activities and hence do not generalize well to realistic sequences of activities. Capturing the whole information of the video in terms of temporal evolution of the video stream has been the focus of some recent approaches [57, 95, 174, 190, 235, 238]. Moving towards more expressive deep networks such as LSTM has become a popular method for encoding such temporal information [42, 215, 233, 236, 269, 289, 292]. Interestingly, while those models move towards more complete understanding of the full video stream, they have yet to significantly outperform local methods [227] on standard benchmarks.

A different direction in understanding comes from reasoning about the complete video stream in a complementary direction — Structure. Understanding activities in a human-centric fashion encodes our particular experiences with the visual world. Understanding activities with emphasis on objects has been a particularly fruitful direction [73, 133, 179, 197, 257]. In a similar vein, some works have also tried modeling activities as transformations [269] or state changes [53]. Recently, there has been significant progress in modelling the complete human-centric aspect, where image recognition is phrased in terms of objects and their roles [75, 286]. Moving beyond appearance and reasoning about the state of *agents* in the images requires understanding human intentions [114, 175]. This ability to understand people in terms of beliefs and intents has been traditionally studied in psychology as the

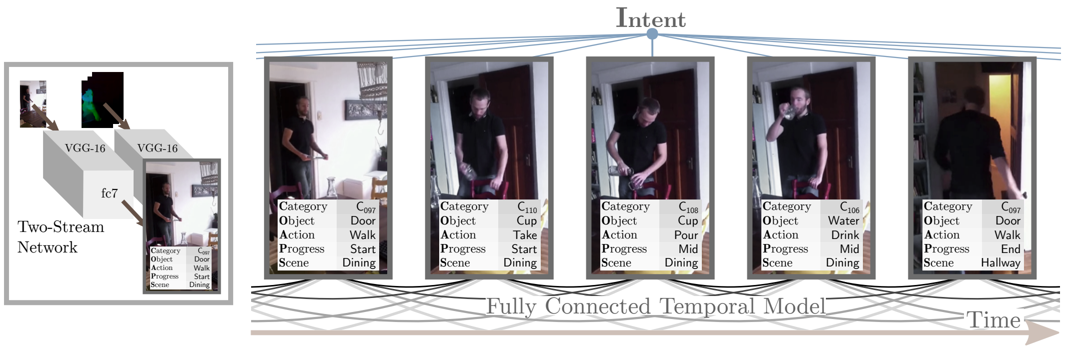


Figure 5.2: An overview of our structured model. The semantic part captures *object*, *action*, etc. at each frame, and temporal aspects captures those over time. On the left side, we show how for each timepoint in the video, a Two-Stream Network predicts the potentials. Our model jointly reasons about multiple aspects of activities in all video frames. The *Intent* captures groups of activities of the person throughout the whole sequence of activities, and fine-grained temporal reasoning is through fully-connected temporal connections.

Theory of mind [178].

How to exactly model structure of the visual and temporal world has been the pursuit of numerous fields. Of particular interest is work that combines the representative power of deep networks with structured modelling. Training such models is often cumbersome due to the differences in jointly training deep networks (stochastic sampling) and sequential models (consecutive samples) [159, 296]. Here we focus on fully-connected random fields, that have been popular in image segmentation [118], where image filtering was used for efficient message passing, and later extended to use CNN potentials [206].

5.2 Proposed Method

Given a video with multiple activities, our goal is to understand the video in terms of activities. Understanding activities requires reasoning about objects being interacted with, the place where the interaction is happening, what happened before and what happens after this current action and even the intent of the actor in the video. We incorporate all these by formulating a deep Conditional Random Field (CRF) over different aspects of the activity over time. That is, a video can be interpreted as a graphical model, where the components of the activity in each frame are nodes in the graph, and the model potentials are the edges in the graph.

In particular, we create a CRF which predicts activity, object, etc., for every frame in the video. For reasoning about time, we create a *fully-connected temporal CRF*, referred as Asynchronous Temporal Field in the text. That is, unlike a linear-chain CRF for temporal modelling (the discriminative counterpart to Hidden Markov Models), each node depends on the state of every other node in the graph. We incorporate intention as another latent variable which is connected to all the action nodes. This is an unobserved variable that influences the sequence of activities. This variable is the common underlying factor that guides and better explains the sequence of actions an agent takes. Analysis of what structure this latent variable learns is presented in the experiments. Our model has three advantages: (1) it addresses the problem of long-term interactions; (2) it incorporates reasoning about multiple parts of the activity, such as objects and intent; and (3) more interestingly, as we will see, it allows for efficient end-to-end training in an asynchronous stochastic fashion.

5.2.1 Architecture

We encode multiple components of an activity. Each video with T frames is represented as $\{X_1, \dots, X_T, I\}$ where X_t is a set of frame-level random variables for time step t and I is an unobserved random variable that represent global intent in the entire video. We can further write $X_t = \{C_t, O_t, A_t, P_t, S_t\}$, where C is the activity category (e.g., 'drinking from cup'), O corresponds to the object (e.g., 'cup'), A represents the action (e.g., 'drink'), P represents the progress of the activity {start, middle, end}, and S represents the scene (e.g. 'Dining Room'). For clarity in the following derivation we will refer to all the associated variables of X_t as a single random variable X_t. A more detailed description of the CRF is presented in the appendix.

Mathematically we consider a random field $\{X, I\}$ over all the random variables in our model ($\{X_1, \dots, X_T, I\}$). Given an input video $V{=}\{V_1, \dots, V_T\}$, where V_t is a video frame, our goal is to estimate the maximum a posteriori labeling of the random field by marginalizing over the intent I. This can be written as:

$$\mathbf{x}^* = \arg\max_x \sum_I P(x, I|V). \tag{5.1}$$

For clarity in notation, we will drop the conditioning on V and write $P(X, I)$. We can define $P(X, I)$ using Gibbs distribution as: $P(X, I){=}\frac{1}{Z(\mathbf{V})}\exp\left(-E(x, I)\right)$ where $E(x, I)$ is the Gibbs energy over x. In our CRF, we model all unary and pairwise cliques between all frames $\{X_1, \dots, X_T\}$ and the intent I. The Gibbs energy is:

$$E(\mathbf{x}, I) = \underbrace{\sum_i \phi_{\mathcal{X}}(x_i)}_{\text{Semantic}} + \underbrace{\sum_i \phi_{\mathcal{XI}}(x_i, I) + \sum_{\substack{i,j\\ i\neq j}} \phi_{\mathcal{XX}}(x_i, x_j)}_{\text{Temporal}}, \tag{5.2}$$

where $\phi_{\mathcal{XX}}(x_i, x_j)$ is the potential between frame i and frame j, and $\phi_{\mathcal{XI}}(x_i, I)$ is the potential between frame i and the intent. For notational clarity $\phi_{\mathcal{X}}(x_i)$ incorporates all unary and pairwise potentials for C_t, O_t, A_t, P_t, S_t. The model is best understood in terms of two aspects: Semantic aspect, which incorporates the local variables in each frame (C_t, O_t, A_t, P_t, S_t); and Temporal aspect, which incorporates interactions among frames and the intent I. This is visualized in Figure 5.2. We will now explain the semantic, and temporal potentials.

Semantic aspect The frame potential $\phi_{\mathcal{X}}(x_i)$ incorporates the interplay between activity category, object, action, progress and scene, and could be written explicitly as $\phi_{\mathcal{X}}(C_t, O_t, A_t, P_t, S_t)$. In practice this potential is composed of unary, pairwise, and tertiary potentials directly predicted by a CNN. We found predicting only the following terms to be sufficient without introducing too many additional parameters: $\phi_{\mathcal{X}}(C_t, O_t, A_t, P_t, S_t){=}\phi(O_t, P_t){+}\phi(A_t, P_t){+}\phi(O_t, S_t){+}\phi(C_t, O_t, A_t, P_t)$ where we only model the assignments seen in the training set, and assume others are not possible.

Temporal aspect The temporal aspect of the model is both in terms of the frame-intent potentials $\phi_{\mathcal{XI}}(x_i, I)$ and frame-frame potentials $\phi_{\mathcal{XX}}(x_i, x_j)$. The frame-intent potentials

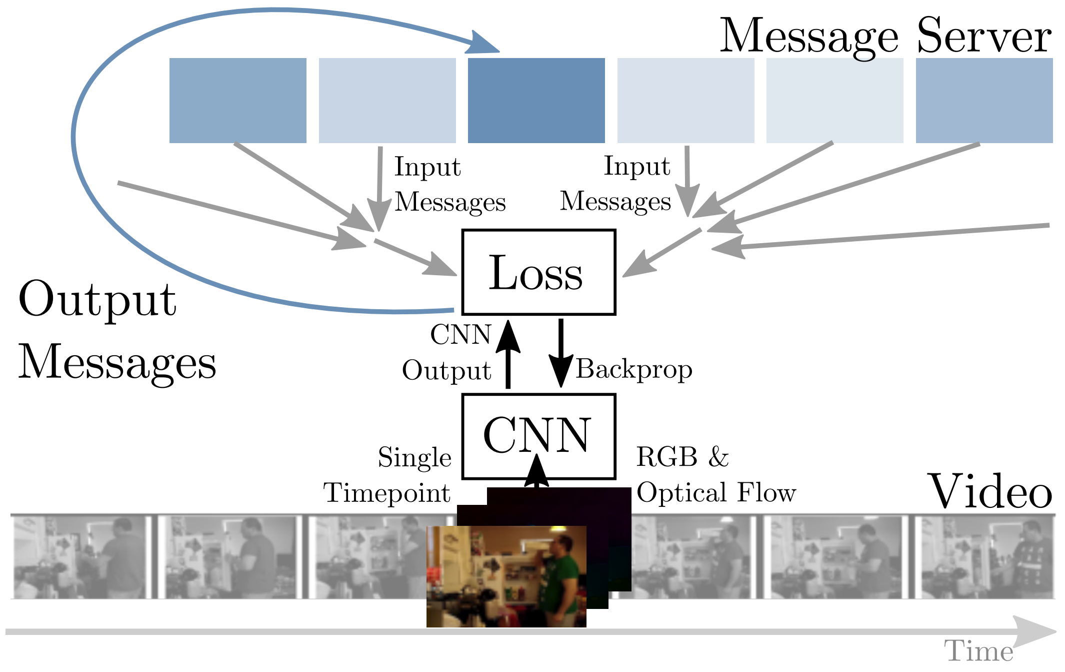


Figure 5.3: Illustration of the learning algorithm, and the message passing structure. Each timepoint that has been processed has a message (Blue highlights messages that have recently been computed). The loss receives a combination of those messages, uses those to construct new messages, and updates the network.

are predicted with a CNN from video frames (pixels and motion). The pairwise potentials $\phi_{\mathcal{X}\mathcal{X}}(x_i, x_j)$ for two time points i and j in our model have the form:

$$\phi_{\mathcal{X}\mathcal{X}}(x_i, x_j) = \mu(x_i, x_j) \sum_m w^{(m)} k^{(m)}(v_i, v_j), \tag{5.3}$$

where μ models the asymmetric affinity between frames, w are kernel weights, and each $k^{(m)}$ is a Gaussian kernel that depends on the videoframes v_i and v_j. We use a single kernel that prioritises short-term interactions:

$$k(v_i, v_j) = \exp\left(-\frac{(j-i)^2}{2\sigma^2}\right) \tag{5.4}$$

The parameters of the general asymmetric compatibility function $\mu(x_i, x_j)$ are learned from the data, and σ is a hyper-parameter chosen by cross-validation.

5.2.2 Inference

While it is possible to enumerate all variable configurations in a single frame, doing so for multiple frames and their interactions is intractable. Our algorithm uses a structured variational approximation to approximate the full probability distribution. In particular, we use a mean-field approximation to make inference and learning tractable. With this approximation, we can do inference by keeping track of message between frames, and asynchronously train one frame at a time (in a mini-batch fashion).

More formally, instead of computing the exact distribution $P(X, I)$ presented above, the structured variational approximation finds the distribution $Q(X, I)$ among a given family of distributions that best fits the exact distribution in terms of KL-divergence. By choosing a family of tractable distributions, it is possible to make inference involving the ideal distribution tractable. Here we use $Q(X, I) = Q_{\mathcal{I}}(I) \prod_i Q_i(x_i)$, the structured mean-field approximation. Minimizing the KL-divergence between those two distributions yields the following iterative update equation:

$$\begin{aligned} Q_i(x_i) \propto \exp\Big\{ & \phi_{\mathcal{X}}(x_i) + \mathrm{E}_{U \sim Q_{\mathcal{I}}}\left[\phi_{\mathcal{X}\mathcal{I}}(x_i, U)\right] \\ & + \sum_{j>i} \mathrm{E}_{U_j \sim Q_j}\left[\phi_{\mathcal{X}\mathcal{X}}(x_i, U_j)\right] \Big\} \\ & + \sum_{j<i} \mathrm{E}_{U_j \sim Q_j}\left[\phi_{\mathcal{X}\mathcal{X}}(U_j, x_i)\right] \Big\} \end{aligned} \tag{5.5}$$

$$Q_{\mathcal{I}}(I) \propto \exp\Big\{ \sum_j \mathrm{E}_{U_j \sim Q_j}\left[\phi_{\mathcal{X}\mathcal{I}}(U_j, I)\right] \Big\} \tag{5.6}$$

where Q_i is marginal distribution with respect to each of the frames, and $Q_{\mathcal{I}}$ is the marginal with respect to the intent. An algorithmic implementation of this equation is as presented in Algorithm 1.

Algorithm 1 Inference for Asynchronous Temporal Fields

```
 1: Initialize Q                                          ▷ Uniform distribution
 2: while not converged do
 3:     Visit frame $i$
 4:     Get $\sum_{j>i} \mathrm{E}_{U_j \sim Q_j}[\phi_{\mathcal{X}\mathcal{X}}(x_i, U_j)]$
 5:     Get $\sum_{j<i} \mathrm{E}_{U_j \sim Q_j}[\phi_{\mathcal{X}\mathcal{X}}(U_j, x_i)]$
 6:     Get $\sum_{j} \mathrm{E}_{U_j \sim Q_j}[\phi_{\mathcal{X}\mathcal{I}}(U_j, I)]$
 7:     while not converged do
 8:         Update $Q_i$ and $Q_{\mathcal{I}}$ using Eq. 5.6
 9:     Send $\mathrm{E}_{U \sim Q_i}[\phi_{\mathcal{X}\mathcal{X}}(x, U)]$
10:     Send $\mathrm{E}_{U \sim Q_i}[\phi_{\mathcal{X}\mathcal{X}}(U, x)]$
11:     Send $\mathrm{E}_{U \sim Q_i}[\phi_{\mathcal{X}\mathcal{I}}(U, I)]$
```

Here 'Get' and 'Send' refer to the message server, and $f(x)$ is a message used later by frames in the same video. The term message server is used for a central process that keeps track of what node in what video sent what message, and distributes them accordingly when requested. In practice, this could be implemented in a multi-machine setup.

5.2.3 Learning

Training a deep CRF model requires calculating derivatives of the objective in terms of each of the potentials in the model, which in turn requires inference of $P(X, I|V)$. The network is trained to maximize the log-likelihood of the data $l(X) = \log \sum_I P(x, I|V)$. The goal is to update the parameters of the model, for which we need gradients with respect to the parameters. Similar to SGD, we find the gradient with respect to one part of the parameters

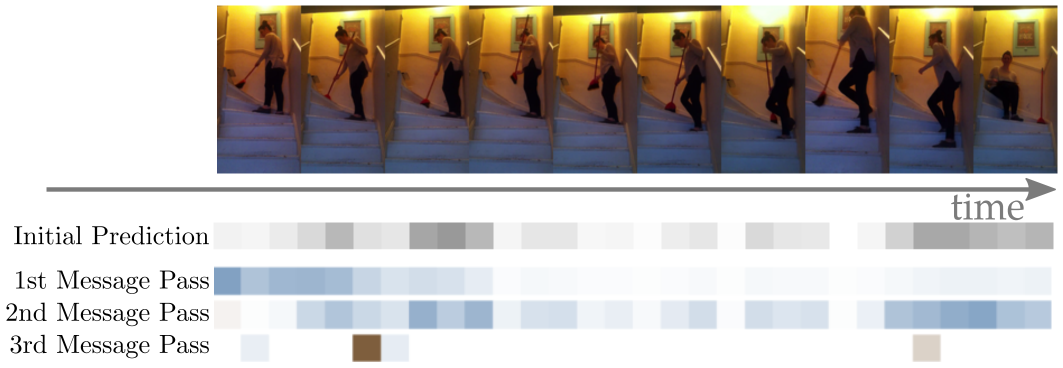


Figure 5.4: Evolution of prediction with increasing messages passes. The first row shows the initial prediction for the category *tidying with a broom* without any message passing, where darker colors correspond to higher likelihood, blue is then an increase in likelihood, and brown decrease. In the first message pass, the confidence of high predictions gets spread around, and eventually increases the confidence of the whole prediction.

at a time, specifically with respect to one potential in one frame. That is, $\phi^i_{\mathcal{X}}(\hat{x})$ instead of $\phi_{\mathcal{X}}(\hat{x})$. The partial derivatives of this loss with respect to each of the potentials are as follows:

$$\frac{\partial l(X)}{\partial \phi^i_{\mathcal{X}}(\hat{x})} = \mathbf{1}_{x=\hat{x}} - Q_i(\hat{x}) \tag{5.7}$$

$$\frac{\partial l(X)}{\partial \phi^i_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})} = \frac{\exp\sum_j \phi_{\mathcal{X}\mathcal{I}}(x_j,\hat{I})}{\sum_I \exp\sum_j \phi_{\mathcal{X}\mathcal{I}}(x_j,I)}\mathbf{1}_{x=\hat{x}} - Q_i(\hat{x})Q_{\mathcal{I}}(\hat{I}) \tag{5.8}$$

$$\begin{aligned}\frac{\partial l(X)}{\partial \mu^i(a,b)} &= \sum_{j>i}\mathbf{1}_{x=a}k(v_i,v_j) - Q_i(\hat{x})\sum_{j>i}Q_{\mathcal{I}}(b)k(v_i,v_j)\\ &+\sum_{j<i}\mathbf{1}_{x=b}k(v_j,v_i) - Q_i(\hat{x})\sum_{j<i}Q_{\mathcal{I}}(a)k(v_i,v_j)\end{aligned} \tag{5.9}$$

where $\phi^i_{\mathcal{X}}(\hat{x})$ and $\phi^i_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})$ is the frame and frame-intent potentials of frame i, and we use $\hat{x}$ to distinguish between the labels and variables the derivative is taken with respect to. $\mu^i(a,b)$ are the parameters of the asymmetric affinity kernel with respect to frame i, and $\mathbf{1}_{x=\hat{x}}$ is a indicator variable that has the value one if the ground truth label corresponds to the variable. Complete derivation is presented in the appendix. These gradients are used to update the underlying CNN model. These update equations lead to the learning procedure presented in Algorithm 2.

Algorithm 2 Learning for Asynchronous Temporal Fields

```
1: Given videos $\mathcal{V}$
2: while not converged do
3:     for each example in mini-batch do
4:         Sample frame $v \in \mathbf{V} \subseteq \mathcal{V}$
5:         Get incoming messages
6:         Update $Q_i$ and $Q_{\mathcal{I}}$
7:         Find gradients with Eq. 5.7-5.9
8:         Backprop gradients through CNN
9:         Send outgoing messages
```

Figure 5.3 graphically illustrates the learning procedure. Since the videos are repeatedly visited throughout the training process, we do not have to run multiple message passes to calculate each partial gradient. This shares ideas with contrastive divergence [85,201]. Given a single video at test time, we visualize in Figure 5.4 how the predictions changes as the distribution converges with multiple messages passes.

Message Passing The key thing to note is all the incoming messages are of the form $M(z)=\sum_j f_j(z)$ where f_j is some function from node j; for e.g., $M(z) = \sum_j \mathrm{E}_{U_j \sim Q_j}[\phi_{\mathcal{XI}}(U_j, z)] = \sum_j f_j(z)$ from Algorithm 1. We use the following approximation during training:

$$M(z) \approx \frac{h}{\sum_j d^j} \sum_j d^j f_{J(j)}(z), \tag{5.10}$$

where $d \in [0,1]$ is a discount factor, h is a hyperparameter, and $J(\cdot)$ is an ordering of the messages in that video based on the iteration in which the message was computed. The messages are a weighted combination of stored messages.

5.3 Experimental Results and Analysis

We analyzed the efficacy of our model on the challenging tasks of video activity classification and temporal localization. In addition, we investigated the different parts of the model, and will demonstrate how they operate together.

Dataset Recent years have witnessed an emergence of large-scale datasets containing sequences of common daily activities [223,275,288]. For our evaluation, we chose the *Charades* dataset [223]. This dataset is a challenging benchmark containing 9,848 videos across 157 action classes with 66,500 annotated activities, including nouns (objects), verbs (actions), and scenes. A unique feature of this dataset is the presence of complex co-occurrences of realistic human-generated activities making it a perfect test-bed for our analysis. We evaluate video classification using the evaluation criteria and code from [223]. Temporal localization is evaluated in terms of per-frame classification using the provided temporal annotations.

Implementation details We use a VGG16 network [225] with additional layers to predict the model potentials (Figure 5.5). We train both a network on RGB frames, and stacks of optical flow images, following the two-stream architecture [227]. The main challenge in training the network is the increase in the output layer size. For the larger potentials, we used the following structure to go from fc7 to $\phi_{\mathcal{XI}}$: Linear layer (4096 to 100), ReLU, Dropout, Linear layer (100 to the potential values).

The input to the RGB network is an image of size $224{\times}224{\times}3$ where we crop random location, size, and aspect ratio. We use data augmentation with color jitter and PCA lighting noise. The RGB network was pretrained on ImageNet. The input to the Flow network is a stack of 10 consecutive optical flow frames at 24 FPS starting with the current frame. Since each optical flow has two channels, the input size is $224{\times}224{\times}20$ as in [227]. The Flow network was pretrained on UCF101 [232] as in Sigurdsson et al. [223], and random cropped in the same way as RGB.

We follow the training setup in Charades [223] and consider a frame to have one activity label at a time. Even so, our method is still able to reason about other activities in the video. Convergence of the model is evaluated using the approximate distribution $Q_i(X)$ at

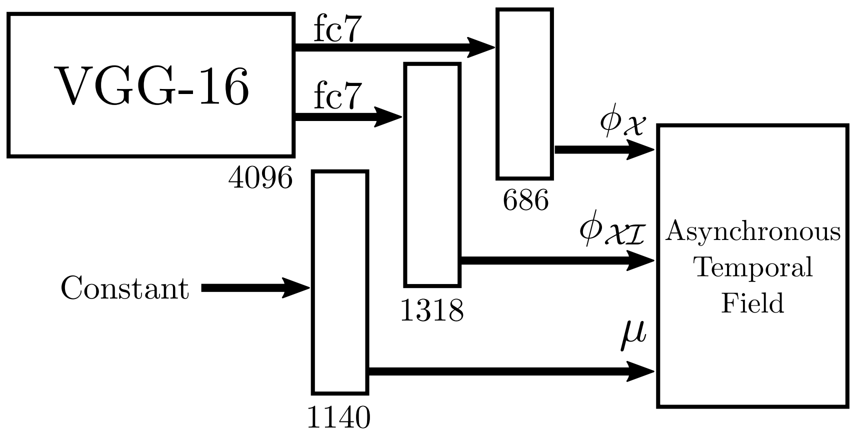


Figure 5.5: The VGG-16 variant predicts the potentials for both RGB and Flow. The network predicts the values of all potentials except one (in this figure we group the frame potentials $\phi_{\mathcal{X}}$ into one layer for clarity). The model is trained end-to-end by passing gradients from the Asynchronous Temporal Field through the network.

each frame. The Charades dataset has the property that scenes were chosen at random for each sequence of activities. For this reason, we found reasoning about scenes to reduce the performance, and the weight of that term was lowered in the model.

To obtain annotations for action progress p_t, we split each activity annotation into three equally sized parts. All layers of the network are trained with a batch size of 240 and a learning rate of 10^{-3} (RGB), 10^{-5} (Flow). Learning rate was reduced by a factor of 10 every 30k iterations for RGB, and every 140k iterations for Flow. The value of the message decay parameter d was set to $d = 0.9$, and the standard deviation σ in (5.4) was set to 6.25 sec (150 frames).

For testing, we sampled 25 equally spaced frames from the video and synchronously pass messages between the frames until convergence (10 message passes). The predictions of the RGB and Flow networks are combined in a probabilistic fashion by multiplying their probabilistic predictions for each class. More implementation details may be found in the appendix. The networks were implemented in Torch, and the code is available on project page.

Diverse batches As highlighted in Section 5, the standard way of sampling batches for temporal models results in high correlation between data points leading to a breakdown of the SGD. To understand the importance of having many diverse examples from multiple videos, we compare the convergence of our method to two alternatives using homogeneous batches: CNN+LSTM from Ng et al. [292], and a synchronous version of our method, where each batch contains full videos (only three videos fit into each mini-batch). We do synchronous message passing until convergence before calculating gradients for backprop. Figure 5.6 shows that our asynchronous training method, containing more diverse training batches, has faster and more stable convergence.

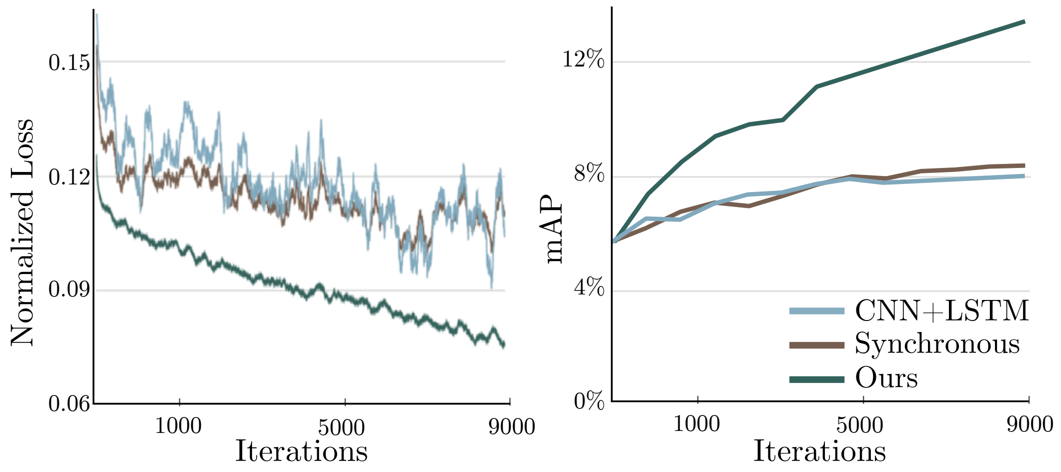


Figure 5.6: Convergence of our method compared to other methods that capture temporal structure. Our asynchronous training method contains more diverse batches, has faster and more stable convergence, and reaches higher accuracy on the test set.

5.3.1 Video Classification

Given a video, the task here is to verify whether it contains one or several of the 157 activity categories. Classification accuracy is measured with the standard mean average precision (mAP) criterion, where a prediction is given for each video. This task has been shown to be highly challenging, with the state-of-the-art non-ensemble methods reaching an mAP of only 17.2%, particularly as each video in this dataset has a sequence of multiple fine-grained activities with a real-world long-tailed activity distribution.

We trained our models using the provided training split following the procedure outlined in Section 5.2. To make predictions for the whole video, we marginalize out everything except the activity category for 25 equidistant frames in the video. The score for each activity category is the maximum across all frames following the setup from [223]. In our analysis, we include the provided non-ensemble baselines from [223] as well as the following additional baselines:

Two-Stream++. We reimplemented the network described in [223], which follows Simonyan et al. [227], with the same parameters. We added data augmentation and fine-tuned all layers of the network. The performance of only the RGB stream is included (*RGB++*). We also consider *Two-Stream Extended* which is the same network, but the Flow network was trained for 25 times more iterations than the RGB network (two weeks of computation on a Titan X GPU). Combined with the augmentation, we found this to non-trivially increase the accuracy.

Two-Stream+LSTM. We followed the method outlined in [292] to jointly train a LSTM on top of the two-stream network. We trained both an RGB and an Optical Flow network using the same setup from [223]. The trained networks from Two-Stream++ were used to initialize the models.

Table 5.1 displays the accuracy obtained by our method along with the baselines. Our proposed approach obtains an mAP of 22.4% substantially outperforming the Two-stream

Approach	mAP	Approach	mAP
Random [223]	5.9	RGB++	15.6
C3D [244]	10.9	Two-Stream++	16.8
AlexNet [122]	11.3	Two-Stream+LSTM	17.8
IDT [264]	17.2	Two-Stream Extended	18.6
Two-Stream [226]	14.3	Ours (RGB Only)	18.3
		Ours	**22.4**

Table 5.1: Video classification results on Charades [223]. The left shows the published baselines from [223] and the right show additional new baselines. Our proposed approach outperforms all competing methods on this dataset.

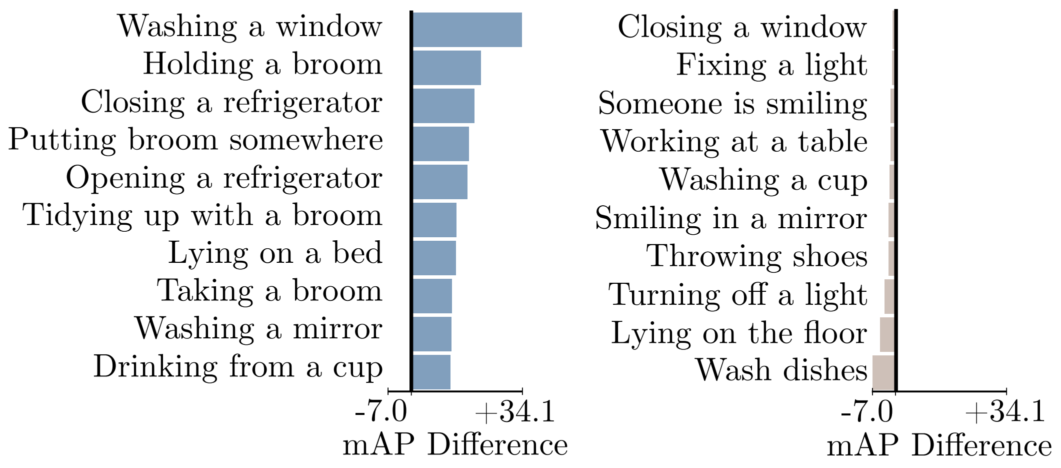


Figure 5.7: The classes with the highest positive and negative difference between our method and Two-Stream (no structure). Our method does better on many classes, without doing much worse on any. In particular, activities that have temporal structure, such as Opening/Closing a refrigerator have significantly higher performance, since our model can reason jointly about those.

Extended baseline at 18.6% mAP, and the IDT baseline at 17.2%. Our method reasons over significantly larger timescales and multiple aspects of the activities. To ascertain this, we highlight in Figure 5.7, the activity classes with the highest positive and negative difference between our method and the Two-Stream network. It is interesting to note that two of those activities are *opening* and *closing* a refrigerator, that arguably have a significant causal structure (an *open* refrigerator was *opened* at some point), which our model harnesses to significantly increase the accuracy.

Ablation studies To study the contribution of different model parts, we also train ablated versions of our model separately choosing the best hyperparameters for each version. In addition to our model with only RGB or Flow, we also consider dropping $\phi_{\mathcal{X}\mathcal{X}}$ (i.e., no sequential information), $\phi_{\mathcal{X}\mathcal{I}}$ (i.e., no intent), both (i.e., only semantic information), and further dropping $\phi_{\mathcal{X}}$ (i.e., dropping all structure). Figure 5.8 shows that semantic reasoning improves over the baseline. Further, while both $\phi_{\mathcal{X}\mathcal{I}}$ and $\phi_{\mathcal{X}\mathcal{X}}$ capture temporal information, they are complementary.

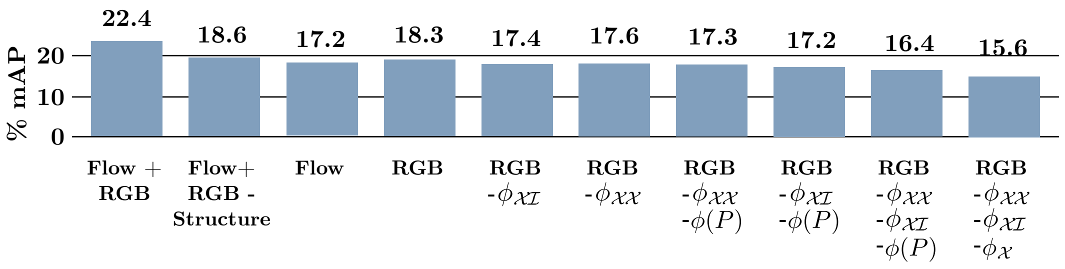


Figure 5.8: Ablation analysis for our proposed model. Y-axis is video classification mAP %. Each factor helps in improving the overall model performance. $\phi(P)$ indicates dropping the 'progress' term within the semantic factor $\phi_{\mathcal{X}}$.

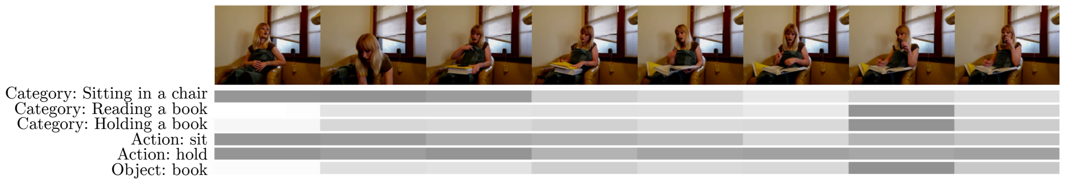


Figure 5.9: Model predictions for a sample video. We see the interplay between categories, objects and actions over time. For example, model becomes confident about the action *sit* early, which aids the understanding of *Sitting in a chair* once the chair becomes visible, and helps predicting *Reading a book*. Darker colors represent higher likelihood, and we average predictions to correspond to each frame.

5.3.2 Temporal Localization

To measure the ability of the methods to temporally localize and understand when exactly activities happen, we adapt the benchmark of [223] to evaluate with the same mAP metric but on individual frames. That is, instead of having a single prediction per video, evaluation is now split into 25 equidistant timepoints having zero or more activities, and the models make a prediction for each of those*. We find this way of evaluating localization robust to annotation ambiguity, and informative for challenging datasets. All hyperparameters were kept equal between localization and classification experiments. All baselines are run on 75 frames across the video, and then every third frame selected for a total of 25 frames. We also considered methods with *post-processing* where the model predictions for the 75 frames are averaged across 30 frames to obtain more spatial consistency, and then 25 frames selected as before.

Table 5.2 shows that our method outperforms the alternatives, including the LSTM model which has been shown to be a powerful temporal modeling tool, but challenging to train on top of a two-stream network due to correlations between consecutive samples. These results demonstrate the our method is tractable way of training end-to-end structured models to understand activities. Interestingly, our method still benefits from adding post-processing, significantly more than the LSTM baseline, likely since our method is reasoning on larger time-scales. This suggests that our model could further benefit from joint training with additional kernels in the temporal term.

*This evaluation code has been included as a part of the Charades dataset (`allenai.org/plato/charades/`).

	Random	RGB	Two-Stream++	Two-Stream +LSTM	Two-Stream Extended	Ours
Standard	2.42	7.89	8.94	9.60	9.37	**9.69**
Post-processing	2.42	9.05	10.9	10.4	11.6	**12.8**

Table 5.2: Temporal localization results (mAP %) on the Charades [223] dataset. Our proposed method outperforms the LSTM model, and is also more tractable to train at a large-scale.

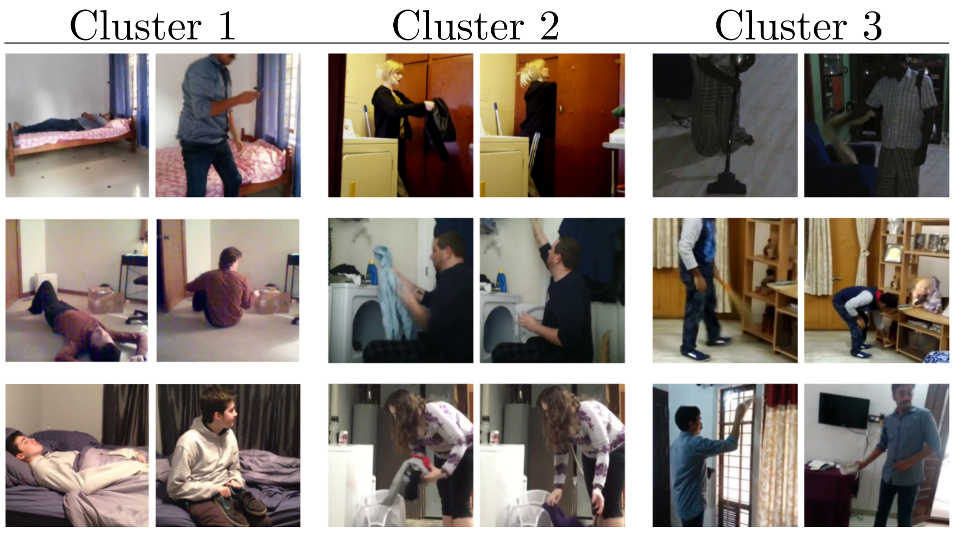


Figure 5.10: To visualize the learned intent, we cluster videos based on intent. In *Cluster 1*, the model captures the intent of get up from lying down. In *Cluster 2*, folding clothes is followed by putting them away, and *Cluster 3* shows cleaning with a broom/vacuum/towel, followed by picking up things.

Qualitative visualization A key advantage of our model is the structured understanding of videos in terms of multiple aspects, such as action sequences, objects, and even intentions. To visualize this, we display predictions over time in Figure 5.9 for the three most confident activity categories, two most confident actions, and the most confident object. More examples are presented in the Appendix.

Interpretation of Intent In our model, the intent I is a continuous distribution over the latent variables. To get an insight into how our model learns the intent, we ran a simple experiment that clustered videos in the dataset that have the most similar inferred intent distributions. The first cluster in Figure 5.10 shows the model captures the simple intent that the person intends to get up from lying down. In the videos, these actions are 10-20 seconds apart, demonstrating that the intent helps reason over large time scales.

In order to further analyze the 'intent' variable, we plot the t-SNE embedding of the intent variable for the videos in the test set. We see that there is clear clustering of similar videos in Fig. 5.11a. We also annotated 10 types of intent (100 videos total). More details

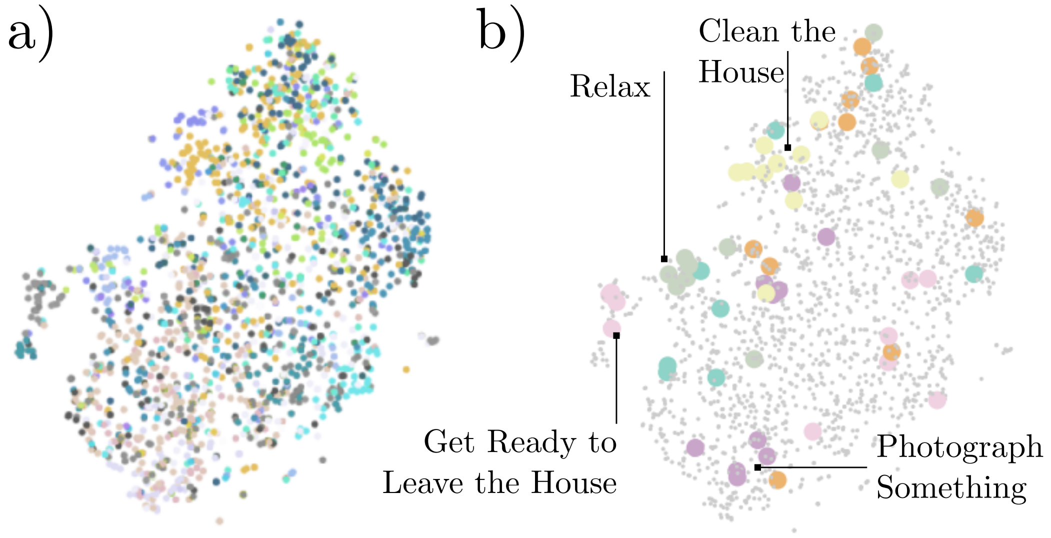


Figure 5.11: t-SNE visualization for the learned intent. Each point corresponds to a video. In a) it is colored based on its activity shared by the most of the 10 nearest neighbors (each video has multiple actions). In b) videos with 6 annotated intent types are emphasized with larger points colored by the type.

are presented in the Appendix. We observe that the intent representation preserves some of the intent types in Fig. 5.11b. Quantitatively, even without mitigating outliers, the average distance (in 10^{-3}) between pairs of videos within an intent type was 6.02 compared to 7.25 (σ=1.06) for any points, and the difference is significant for 5 of 10 intent types (p=0.1). This tentatively suggest that the intent captures interesting structure in the data, and we hope this will encourage future work.

5.4 Discussion

We have presented a deep-structured model using a fully-connected temporal CRF that not only models semantic aspects of activities but also reasons about long-term temporal relations. We also presented an asynchronous stochastic inference algorithm that circumvents a key bottleneck in the large-scale end-to-end model learning. Using our proposed method, we have demonstrated impressive activity classification and temporal localization results on a challenging dataset of realistic activities.

5.5 Appendix

This appendix contains the following additional content:

1. Description of the CRF.
2. Derivation of the update equations.
3. Details of the learning algorithm.
4. Additional implementation details.
5. Details about intent analysis.
6. Additional visualizations of output predictions.

5.5.1 Description of the CRF

We create a CRF which predicts activity, object, etc., for every frame in the video. For reasoning about time, we create a *fully-connected temporal CRF*, referred to as Asynchronous Temporal Field in the text. That is, unlike a linear-chain CRF for temporal modelling (the discriminative counterpart to Hidden Markov Models), each node depends on the state of every other node in the graph. We incorporate intention as another latent variable which is connected to all the action nodes.

We encode multiple components of an activity. Each video with T frames is represented as $\{X_1, \dots, X_T, I\}$ where X_t is a set of frame-level random variables for time step t and I is a random variable that represent global intent in the entire video. For clarity of derivation X_t includes all frame level variables $(C_t, O_t, A_t, P_t, S_t)$

Mathematically we consider a random field $\{X, I\}$ over all the random variables in our model ($\{X_1, \dots, X_T, I\}$). We now list the complete description of the CRF.

CRF Variables:

- Random field $\{X, I\} = \{X_1, \dots, X_T, I\}$
- Frame $X_t = \{C_t, O_t, A_t, P_t, S_t\}$, $X_t \in \mathcal{X}, \mathcal{X} = \mathcal{C} \times \mathcal{O} \times \mathcal{A} \times \mathcal{P} \times \mathcal{S}$
  - Category $C_t \in \mathcal{C}, \mathcal{C} = \{1, 2, ..., 157\}$ (For each category in the dataset)
  - Object $O_t \in \mathcal{O}, \mathcal{O} = \{1, 2, ..., 38\}$ (Includes "No object")
  - Action $A_t \in \mathcal{A}, \mathcal{A} = \{1, 2, ..., 33\}$
  - Progress $P_t \in \mathcal{P}, \mathcal{P} = \{1, 2, 3\}$ (Before, Middle, End)
  - Scene $S_t \in \mathcal{S}, \mathcal{S} = \{1, 2, ..., 15\}$
- Intent $I \in \mathcal{I}, \mathcal{I} = \{1, 2, ..., N_I\}$ ($N_I = 30$ is used here)

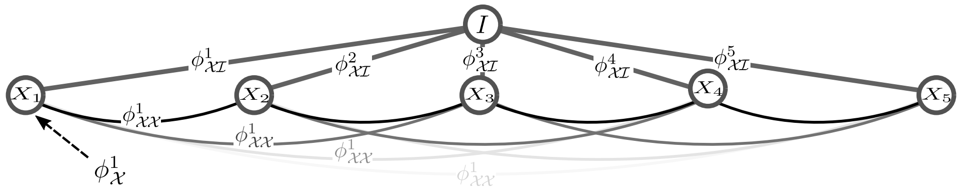


Figure 5.12: The model captures interactions between all frames X_t and the intent I, that is, a fully-connected model. Here shown for $T = 5$. We visualize some of the potentials of the model, and where they fit into the graph. All $\phi^i_{\mathcal{XI}}$ share the same parameters, but we calculate the gradients with respect for each of them separately below. For efficient inference, we use a mean-field approximation presented below. A mean-field approximation is a simpler distribution that is fit to the original distribution when needed.

CRF Potentials:

- $\phi_{\mathcal{X}} : \mathcal{X} \mapsto \mathcal{R}$, equivalently: $\phi_{\mathcal{X}} : \mathcal{C}{\times}\mathcal{O}{\times}\mathcal{A}{\times}\mathcal{P}{\times}\mathcal{S} \mapsto \mathcal{R}$
- $\phi_{\mathcal{X}}$ decomposes as follows: $\phi_{\mathcal{X}}(C_t, O_t, A_t, P_t, S_t){=}\phi(O_t, P_t){+}\phi(A_t, P_t){+}\phi(O_t, S_t){+}\phi(C_t, O_t, A_t, P_t)$
  - $\phi(O_t, P_t) : \mathcal{O}{\times}\mathcal{P} \mapsto \mathcal{R}$
  - $\phi(A_t, P_t) : \mathcal{A}{\times}\mathcal{P} \mapsto \mathcal{R}$
  - $\phi(O_t, S_t) : \mathcal{O}{\times}\mathcal{S} \mapsto \mathcal{R}$
  - $\phi(C_t, O_t, A_t, P_t) : \mathcal{B} \mapsto \mathcal{R}$, here $\mathcal{B}$ is all configurations of C_t, O_t, A_t, P_t that exist in the training data.
- $\phi_{\mathcal{XI}} : \mathcal{X}{\times}\mathcal{I} \mapsto \mathcal{R}$ (specifically we parametrize this as $\phi_{\mathcal{XI}} : \mathcal{O}{\times}\mathcal{I} \mapsto \mathcal{R}$)
- $\phi_{\mathcal{XX}} : \mathcal{X}{\times}\mathcal{X} \mapsto \mathcal{R}$ (specifically we parametrize this as $\phi_{\mathcal{XI}} : \mathcal{O}{\times}\mathcal{O} \mapsto \mathcal{R}$)

The complete distribution of the model is:

$$P(X, I) = \frac{1}{Z} \exp \left\{ \sum_i \phi^i_{\mathcal{X}}(x_i) + \sum_i \phi^i_{\mathcal{XI}}(x_i, I) + \sum_i \sum_{j \neq i} \phi^i_{\mathcal{XX}}(x_i, x_j) \right\} \tag{5.11}$$

where $\phi_{\mathcal{XX}}(x_i, x_j)$ is the potential between frame i and frame j, and $\phi_{\mathcal{XI}}(x_i, I)$ is the potential between frame i and the intent. For notational clarity $\phi_{\mathcal{X}}(x_i)$ incorporates all potentials for C_t, O_t, A_t, P_t, S_t. The model is presented in Figure 5.12.

5.5.2 Derivation of the Update Equations

Given an input video $V{=}\{V_1, \ldots, V_T\}$, our goal is to estimate the maximum a posteriori labeling of the random field by marginalizing over the intent I, $\sum_I P(X, I|V)$ as discussed above. In the following derivations we omit the conditioning on V and write $P(X, I)$ and $\phi(X, I)$.

Before we present the update equations and gradients, we define the following messages which will be used in the final version of the following equations for clarity in their presentation. Messages are a term used for cached computations sent between different functions in a dynamic programming fashion. In the following derivations, X^* is used to explicitly denote the ground truth used for training. Plain X is used to refer to the variable.

Outgoing Messages (Messages that are calculated from a single frame)

$$\begin{aligned} FA_j(x_j) &= E_{U \sim Q_j}\left[\mu(x_j, U)\right] && (5.12)\\ FB_j(x_j) &= E_{U \sim Q_j}\left[\mu(U, x_j)\right] && (5.13)\\ H_j(I) &= E_{U \sim Q_j}\left[\phi_{\mathcal{XI}}(U, I)\right] && (5.14)\\ H_j^*(I) &= \phi_{\mathcal{XI}}(x_j^*, I) && (5.15)\\ K_j(x_j) &= Q_j(x_j) && (5.16)\\ K_j^*(x_j) &= \mathbf{1}_{x_j = x_j^*} && (5.17) \end{aligned}$$

Incoming Messages (Messages that are calculated from messages from multiple frames and used for the computation of a single frame)

$$\begin{aligned} \mathbb{FA}_i(x_i) &= \sum_{j>i} E_{U_j \sim Q_j}[\mu(x_i, U_j)] K(v_i, v_j) = \sum_{j>i} FA_j(x_i) K(v_i, v_j) && (5.18)\\ \mathbb{FB}_i(x_i) &= \sum_{j<i} E_{U_j \sim Q_j}[\mu(U_j, x_i)] K(v_j, v_i) = \sum_{j<i} FB_j(x_i) K(v_j, v_i) && (5.19)\\ \mathbb{H}_i(I) &= \sum_{j \neq i} E_{U_j \sim Q_j}\left[\phi_{\mathcal{XI}}(U_j, I)\right] = \sum_{j \neq i} H_j(I) && (5.20)\\ \mathbb{H}_i^*(I) &= \sum_{j \neq i} \phi_{\mathcal{XI}}(x_j^*, I) = \sum_{j \neq i} H_j^*(I) && (5.21)\\ \mathbb{KA}_i(x_i) &= \sum_{j>i} Q_j(x_j) K(x_i, x_j) = \sum_{j>i} K_j(x_i) && (5.22)\\ \mathbb{KA}_i^*(x_i) &= \sum_{j>i} \mathbf{1}_{x_j = x_j^*} K(x_i, x_j^*) = \sum_{j>i} K_j^*(x_i) && (5.23)\\ \mathbb{KB}_i(x_i) &= \sum_{j<i} Q_j(x_j) K(x_j, x_i) = \sum_{j<i} K_j(x_i) && (5.24)\\ \mathbb{KB}_i^*(x_i) &= \sum_{j<i} \mathbf{1}_{x_j = x_j^*} K(x_j^*, x_i) = \sum_{j<i} K_j^*(x_i) && (5.25) \end{aligned}$$

Instead of computing the exact distribution $P(X, I)$ presented above, the structured variational approximation finds the distribution $Q(X, I)$ among a given family of distributions that best fits the exact distribution in terms of KL-divergence. By choosing a family of tractable distributions, it is possible to make inference involving the ideal distribution tractable. Here we use $Q(X, I) = Q_{\mathcal{I}}(I) \prod_i Q_i(x_i)$, the structured mean-field approximation. More details on mean-field approximation are presented section 11.5 generic update equation for Q (Equation 11.54 in [116]) is:

$$Q(x_i) \propto \exp\left\{E_{X_{-i} \sim Q}\left[\log P(x_i | X_{-i})\right]\right\} \qquad (5.26)$$

where X_{-i} refers to all variables except x_i. Using Eq. 5.11 along with Eq. 5.26 we get the following update equations:

$$Q_i(x_i) \propto \exp\left\{\phi_{\mathcal{X}}(x_i) + \mathrm{E}_{U\sim Q_{\mathcal{I}}}\left[\phi_{\mathcal{X}\mathcal{I}}(x_i, U)\right] + \sum_{j>i} \mathrm{E}_{U_j\sim Q_j}\left[\phi_{\mathcal{X}\mathcal{X}}(x_i, U_j)\right] + \sum_{j<i} \mathrm{E}_{U_j\sim Q_j}\left[\phi_{\mathcal{X}\mathcal{X}}(U_j, x_i)\right]\right\}$$

$$\propto \exp\left\{\phi_{\mathcal{X}}(x_i) + \mathrm{E}_{U\sim Q_{\mathcal{I}}}\left[\phi_{\mathcal{X}\mathcal{I}}(x_i, U)\right] + \mathbb{FA}_i(x_i) + \mathbb{FB}_i(x_i)\right\} \tag{5.27}$$

$$Q_{\mathcal{I}}(I) \propto \exp\left\{\sum_{j} \mathrm{E}_{U_j\sim Q_j}\left[\phi_{\mathcal{X}\mathcal{I}}(U_j, I)\right]\right\} \tag{5.28}$$

$$\propto \exp\left\{\mathbb{H}_i(I) + H_i(I)\right\} \qquad \text{(Here } i \text{ refers to the frame of interest, but any choice of } i \text{ holds)} \tag{5.29}$$

where Q_i is marginal distribution with respect to each of the frames, and $Q_{\mathcal{I}}$ is the marginal with respect to the intent.

5.5.3 Details of the learning algorithm

Training a deep CRF model requires calculating derivatives of the objective in terms of each of the potentials in the model, which in turn requires inference of $P(X, I|V)$. The network is trained to maximize the log-likelihood of the data:

$$l(X^*) = \log \sum_{I} P(X^*, I|V) \tag{5.30}$$

$$= \log \sum_{I} \frac{\tilde{P}(X^*, I|V)}{Z(V)} \tag{5.31}$$

$$= \log \sum_{I} \tilde{P}(X^*, I|V) - \log Z(V) \tag{5.32}$$

$$Z(V) = \sum_{I} \sum_{X} \tilde{P}(X, I|V) \tag{5.33}$$

where we explicitly write out the partition function Z(V), and $\tilde{P}()$ is the unnormalized version of $P()$. Again, we use X^* to explicitly refer to the ground truth labels. As before, V is omitted from the following derivations. The goal is to update the parameters of the model, for which we need gradients with respect to the parameters. Similar to SGD, we find the gradient with respect to one part of the parameters at a time, specifically with respect to one potential in one frame. That is, $\phi^i_{\mathcal{X}}(x)$ instead of $\phi_{\mathcal{X}}(x)$. The partial derivatives of this loss with respect to each of the potentials are as follows.

Updating the frame potential $\phi_{\mathcal{X}}$

The frame potential $\phi_{\mathcal{X}}(x_i)$ incorporates the interplay between activity category, object, action, progress and scene, and could be written explicitly as $\phi_{\mathcal{X}}(C_t, O_t, A_t, P_t, S_t)$. In practice this potential is composed of unary, pairwise, and tertiary potentials directly predicted by a CNN. We found predicting only the following terms to be sufficient without introducing too many additional parameters: $\phi_{\mathcal{X}}(C_t, O_t, A_t, P_t, S_t) = \phi(O_t, P_t) + \phi(A_t, P_t) + \phi(O_t, S_t) +$

$\phi(C_t, O_t, A_t, P_t)$ where we only model the assignments seen in the training set, and assume others are not possible.

Let us first derive the update equation for $\phi_{\mathcal{X}}$ as a whole, and then demonstrate how to update each of the individual potentials. In the following derivation, we simply take the partial derivative where appropriate and iteratively use the chain rule.

$$\frac{\partial l(X^*)}{\partial \phi^{\hat{i}}_{\mathcal{X}}(\hat{x})} = \frac{1}{\sum_I \tilde{P}(X^*, I)} \left(\sum_I \tilde{P}(X^*, I) \right) \frac{\partial \left(\sum_i \phi^i_{\mathcal{X}}(x^*_i) \right)}{\partial \phi^{\hat{i}}_{\mathcal{X}}(\hat{x})} - \frac{\partial \log Z}{\partial \phi^{\hat{i}}_{\mathcal{X}}(\hat{x})} \tag{5.34}$$

$$= \mathbf{1}_{\hat{x}=x^*} - \frac{1}{Z} \sum_X \sum_I \frac{\partial \tilde{P}(X, I)}{\partial \phi^{\hat{i}}_{\mathcal{X}}(\hat{x})} \qquad \text{(Denominator and numerator cancel)} \tag{5.35}$$

$$= \mathbf{1}_{\hat{x}=x^*} - \frac{1}{Z} \sum_X \sum_I \mathbf{1}_{\hat{x}=x} \tilde{P}(X, I) \tag{5.36}$$

$$= \mathbf{1}_{\hat{x}=x^*} - \sum_X \sum_I \mathbf{1}_{\hat{x}=x} P(X, I) \tag{5.37}$$

$$\approx \mathbf{1}_{\hat{x}=x^*} - \sum_X \sum_I \mathbf{1}_{\hat{x}=x} Q(X, I) \qquad \text{(Using the mean-field)} \tag{5.38}$$

$$= \mathbf{1}_{\hat{x}=x^*} - \sum_X \sum_I \mathbf{1}_{\hat{x}=x} Q_{\mathcal{I}}(I) \prod_i Q_i(x_i) \tag{5.39}$$

$$= \mathbf{1}_{\hat{x}=x^*} - Q_{\hat{i}}(\hat{x}) \qquad \text{(Since } \sum_{x_i} Q_i(x_i) = 1\text{)} \tag{5.40}$$

where we use X^* to refer to the ground truth labels, and $\hat{X}$ to refer to the variables we are taking the partial derivative with respect to. We note that $\frac{\partial \left(\sum_i \phi^i_{\mathcal{X}}(x^*_i) \right)}{\partial \phi^{\hat{i}}_{\mathcal{X}}(\hat{x})} = \mathbf{1}_{\hat{x}=x^*}$. Intuitively this implies the partial gradient is the difference between the ground truth and the model prediction. This equation is easily extended to update each of the individual potentials as follows:

$$\frac{\partial l(X^*)}{\partial \phi^{\hat{i}}(\hat{O}_t, \hat{P}_t)} = \mathbf{1}_{(\hat{O}_t, \hat{P}_t) = (O^*_t, P^*_t)} - \sum_{C_t} \sum_{A_t} \sum_{S_t} Q_{\hat{i}}(X^*_t) \tag{5.41}$$

$$\frac{\partial l(X^*)}{\partial \phi^{\hat{i}}(\hat{A}_t, \hat{P}_t)} = \mathbf{1}_{(\hat{A}_t, \hat{P}_t) = (A^*_t, P^*_t)} - \sum_{C_t} \sum_{O_t} \sum_{S_t} Q_{\hat{i}}(X^*_t) \tag{5.42}$$

$$\frac{\partial l(X^*)}{\partial \phi^{\hat{i}}(\hat{O}_t, \hat{S}_t)} = \mathbf{1}_{(\hat{O}_t, \hat{S}_t) = (O^*_t, S^*_t)} - \sum_{C_t} \sum_{A_t} \sum_{P_t} Q_{\hat{i}}(X^*_t) \tag{5.43}$$

$$\frac{\partial l(X^*)}{\partial \phi^{\hat{i}}(\hat{C}_t, \hat{O}_t, \hat{A}_t, \hat{P}_t)} = \mathbf{1}_{(\hat{C}_t, \hat{O}_t, \hat{A}_t, \hat{P}_t) = (C^*_t, O^*_t, A^*_t, P^*_t)} - \sum_{S_t} Q_{\hat{i}}(X^*_t) \tag{5.44}$$

where we marginalize out the variables that are not a part of each potential. Again, X_t incorporates all the frame variables $\{C_t, O_t, A_t, P_t, S_t\}$. These partial derivatives are passed down the CNN (backprop) to update the parameters of the network.

Updating the frame-intent potential $\phi_{\mathcal{X}\mathcal{I}}$

Similarly to $\phi_{\mathcal{X}}$ we proceed as follows:

$$\frac{\partial l(X^*)}{\partial \phi^{\hat{i}}_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})} = \frac{1}{\sum_I \tilde{P}(X^*,I)} \left(\sum_I \tilde{P}(X^*,I)\mathbf{1}_{\hat{x}=x^*}\mathbf{1}_{\hat{I}=I} \right) - \frac{\partial \log Z}{\partial \phi^{\hat{i}}_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})} \tag{5.45}$$

$$= \frac{\tilde{P}(X^*,\hat{I})}{\sum_I \tilde{P}(X^*,I)}\mathbf{1}_{\hat{x}=x^*} - \frac{\partial \log Z}{\partial \phi^{\hat{i}}_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})} \tag{5.46}$$

$$= \frac{\exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,\hat{I})\right\}}{\sum_I \exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,I)\right\}}\mathbf{1}_{\hat{x}=x^*} - \frac{\partial \log Z}{\partial \phi^{\hat{i}}_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})} \qquad \text{(Terms without } I \text{ cancel)} \tag{5.47}$$

$$= \frac{\exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,\hat{I})\right\}}{\sum_I \exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,I)\right\}}\mathbf{1}_{\hat{x}=x^*} - \frac{1}{Z}\sum_X\sum_I \frac{\partial \tilde{P}(X,I)}{\partial \phi^{\hat{i}}_{\mathcal{X}\mathcal{I}}(\hat{x},\hat{I})} \tag{5.48}$$

$$= \frac{\exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,\hat{I})\right\}}{\sum_I \exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,I)\right\}}\mathbf{1}_{\hat{x}=x^*} - \frac{1}{Z}\sum_X\sum_I \tilde{P}(X,I)\mathbf{1}_{\hat{x}=x}\mathbf{1}_{\hat{I}=I} \tag{5.49}$$

$$= \frac{\exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,\hat{I})\right\}}{\sum_I \exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,I)\right\}}\mathbf{1}_{\hat{x}=x^*} - \sum_X\sum_I P(X,I)\mathbf{1}_{\hat{x}=x}\mathbf{1}_{\hat{I}=I} \tag{5.50}$$

$$\approx \frac{\exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,\hat{I})\right\}}{\sum_I \exp\left\{\sum_i \phi^i_{\mathcal{X}\mathcal{I}}(x^*_i,I)\right\}}\mathbf{1}_{\hat{x}=x^*} - \sum_X\sum_I Q(X,I)\mathbf{1}_{\hat{x}=x}\mathbf{1}_{\hat{I}=I} \qquad \text{(Mean-field approximation)} \tag{5.51}$$

$$= \frac{\exp \sum_i \phi_{\mathcal{X}\mathcal{I}}(x^*_i,\hat{I})}{\sum_I \exp \sum_i \phi_{\mathcal{X}\mathcal{I}}(x^*_i,I)}\mathbf{1}_{\hat{x}=x^*} - Q_{\hat{i}}(\hat{x})Q_{\mathcal{I}}(\hat{I}) \tag{5.52}$$

$$= \frac{\exp\left\{\mathbb{H}^*_{\hat{i}}(\hat{I}) + H^*_{\hat{i}}(\hat{I})\right\}}{\sum_I \exp\left\{\mathbb{H}^*_i(I) + H^*_i(I)\right\}}\mathbf{1}_{\hat{x}=x^*} - Q_{\hat{i}}(\hat{x})Q_{\mathcal{I}}(\hat{I}) \tag{5.53}$$

This equation can be interpreted in that it captures the difference between the distribution of the intent given the ground truth, and the predicted distribution of the intent.

Updating the frame-frame potential $\phi_{\mathcal{X}\mathcal{X}}$

The pairwise potentials $\phi_{\mathcal{X}\mathcal{X}}(x_i,x_j)$ for two time points i and j in our model have the form:

$$\phi_{\mathcal{X}\mathcal{X}}(x_i,x_j) = \mu(x_i,x_j)\sum_m w^{(m)}k^{(m)}(v_i,v_j) \tag{5.54}$$

$$= \mu(x_i,x_j)k(v_i,v_j) \tag{5.55}$$

where μ models the asymmetric affinity between frames, w are kernel weights, and each $k^{(m)}$ is a Gaussian kernel that depends on the videoframes v_i and v_j which are omitted from this notation for convenience, but the probability and the potentials are conditioned on V.

Here we use a single kernel that prioritises short-term interactions:

$$k(v_i, v_j) = \exp\left(-\frac{(j-i)^2}{2\sigma^2}\right) \tag{5.56}$$

The parameters of the general asymmetric compatibility function $\mu(x_i, x_j)$ are learned from the data, and σ is a hyper-parameter chosen by cross-validation. The parameters of μ are learned as follows, and this could be extended to a more general form of $\phi_{\mathcal{X}\mathcal{X}}$:

$$\frac{\partial l(X^*)}{\partial \mu^{\hat{i}}(\hat{x},\hat{b})} = \frac{1}{\sum_I \tilde{P}(X^*, I)} \left(\sum_I \tilde{P}(X^*, I)\right) \frac{\partial}{\partial \mu^{\hat{i}}(\hat{x},\hat{b})} \left(\sum_{j>\hat{i}} \phi^i_{\mathcal{X}\mathcal{X}}(x^*_i, x^*_j) + \sum_{j<\hat{i}} \phi^i_{\mathcal{X}\mathcal{X}}(x^*_j, x^*_i)\right) - \frac{\partial \log Z}{\partial \mu^{\hat{i}}(\hat{x},\hat{b})} \tag{5.57}$$

$$= \sum_{j>\hat{i}} \mathbf{1}_{\hat{x}=x^*} \mathbf{1}_{\hat{b}=x^*_j} k(v_{\hat{i}}, v_j) + \sum_{j<\hat{i}} \mathbf{1}_{\hat{x}=x^*} \mathbf{1}_{\hat{b}=x^*_j} k(v_j, v_{\hat{i}}) - \frac{1}{Z} \sum_X \sum_I \frac{\partial \tilde{P}(X, I)}{\partial \mu^{\hat{i}}(\hat{x},\hat{b})} \tag{5.58}$$

$$= \sum_{j>\hat{i}} \mathbf{1}_{\hat{x}=x^*} \mathbf{1}_{\hat{b}=x^*_j} k(v_{\hat{i}}, v_j) + \sum_{j<\hat{i}} \mathbf{1}_{\hat{x}=x^*} \mathbf{1}_{\hat{b}=x^*_j} k(v_j, v_{\hat{i}})$$
$$- \frac{1}{Z} \sum_X \sum_I \tilde{P}(X, I) \sum_i \left(\sum_{j>i} \mathbf{1}_{\hat{x}=x} \mathbf{1}_{\hat{b}=x_j} k(v_i, v_j) + \sum_{j<i} \mathbf{1}_{\hat{x}=x} \mathbf{1}_{\hat{b}=x_j} k(v_j, v_i)\right) \tag{5.59}$$

$$= \sum_{j>\hat{i}} \mathbf{1}_{\hat{x}=x^*} \mathbf{1}_{\hat{b}=x^*_j} k(v_{\hat{i}}, v_j) + \sum_{j<\hat{i}} \mathbf{1}_{\hat{x}=x^*} \mathbf{1}_{\hat{b}=x^*_j} k(v_j, v_{\hat{i}})$$
$$- \sum_X \sum_I Q_{\mathcal{I}}(I) \prod_i Q_i(x_i) \sum_i \left(\sum_{j>i} \mathbf{1}_{\hat{x}=x} \mathbf{1}_{\hat{b}=x_j} k(v_i, v_j) + \sum_{j<i} \mathbf{1}_{\hat{x}=x} \mathbf{1}_{\hat{b}=x_j} k(v_j, v_i)\right) \quad \text{(Mean-field)} \tag{5.60}$$

$$\frac{\partial l(X^*)}{\partial \mu^{\hat{i}}(a,b)} = \sum_{j>\hat{i}} \mathbf{1}_{a=x^*_{\hat{i}}} \mathbf{1}_{b=x^*_j} k(v_{\hat{i}}, v_j) - Q_{\hat{i}}(a) \sum_{j>\hat{i}} Q_j(b) k(v_{\hat{i}}, v_j) + \sum_{j<\hat{i}} \mathbf{1}_{b=x^*_{\hat{i}}} \mathbf{1}_{a=x^*_j} k(v_j, v_{\hat{i}}) - Q_{\hat{i}}(b) \sum_{j<\hat{i}} Q_j(a) k(v_j, v_{\hat{i}}) \tag{5.61}$$

$$= \mathbf{1}_{a=x^*_{\hat{i}}} \mathbb{KA}^*_{\hat{i}}(b) - Q_{\hat{i}}(a) \mathbb{KA}_{\hat{i}}(b) + \mathbf{1}_{b=x^*_{\hat{i}}} \mathbb{KB}^*_{\hat{i}}(a) - Q_{\hat{i}}(b) \mathbb{KB}_{\hat{i}}(a) \tag{5.62}$$

This update equation consists of two symmetric parts, one for influence from frames before, and one for influence from frames after. Intuitively, this captures the difference in the true affinity between frame i and all frames j on the one hand, and on the other hand the predicted affinity, where the affinity is weighted by the kernel.

5.5.4 Additional implementation details

A more detailed algorithmic description of the model is presented in Algorithm 3. More details can be found on the project page `https://github.com/gsig/temporal-fields/`.

Training time Training the models took a while: The RGB stream of the Two-Stream model converged after only 0.2 epochs (20% of the total data, randomly selected) of the training

Algorithm 3 Learning for Asynchronous Temporal Fields (Detailed)

```
1: Given videos $\mathcal{V}$
2: **while** not converged **do**
3:     **for each** example in mini-batch **do**
4:         Sample frame $v \in \mathbf{V} \subseteq \mathcal{V}$ that has index $i$
5:         Calculate messages with Eq. 5.18-5.25, approximated by Eq. 9 (above)
6:         Alternate updating $Q_i$ and $Q_\mathcal{I}$ until convergence
7:         Find gradients with Eqs. 5.40,5.53,5.62
8:         Backprop gradients through CNN
9:         Store computations of Eq. 5.12-5.17 for later use
10:    Update CNN using accumulated gradients
```

data, but training the Flow stream needed 4.0 epochs to reach the best performance. Our model needed 0.7 epochs for the RGB stream and 8.3 epochs for the Flow stream. Each 0.1 epoch is approximately 1450 batches of size 256 (all labelled frames at 8 FPS), and takes between 3-8 hours depending on hardware and model. Our learning rate schedule was chosen by finding the largest learning rate that did not cause divergence, and then making sure the learning rate was decayed by a factor of 100 over the course of training. Investigations into training these kinds of models faster are likely to yield substantial benefits.

Training Deep Models with Latent Variables One of the pursuits of this work was introducing latent variables into a deep framework, the intent. The gradient for the frame-intent potential, contains predictions of the model on both sides, which is a common problem in deep reinforcement learning, where a variety of tricks such as target fixing, double Q-learning, and gradient clipping, are used to combat the instability caused by this. We found that simply severing the dependency of the frame-intent variable on the input data got rid of the instability, and still gave acceptable performance on the RGB stream, however we found that this did not give good performance on the Flow stream.

In order to train the network with the frame-intent potential depending on the input data, we experimented with a variety of techniques from the reinforcement learning literature. Only two methods were found to help: Alternating target and prediction networks, and regularization. For alternating target and prediction networks, the network predicts two frame-intent potentials, and then the network randomly chooses which to use as the target, and which to use as the source, and backprop only through one of them. For regularization, we enforce the frame-intent potential to be close to zero, similar to weight decay (set to $4 \cdot 10^{-4}$). Regularization was found to be give slightly better performance, and easy to implement/tune, and was used here.

5.5.5 Details about intent analysis

To analyze the learned intent variable, we defined 10 types of intent: *getting something to eat*, *clean the living space*, *getting dressed*, *getting something from storage*, *get informed*, *get out of bed*, *leave the house*, *photograph something*, *relaxing*, *working*. To identify videos corresponding to the intent, we used keyword related to the intent (such as `closet` and `clothes` for *getting dressed*) and manually verified that the content of the video matched the intent. The analysis demonstrates that the latent intent variables captures non-trivial structure of the label space, but precisely understanding goal-oriented behavior compared to simple activity analysis remains important future work.

5.5.6 Additional Visualizations of Output Predictions

We present here additional visualizations from the model. In Figure 5.13 we present in the same way as Figure 9 (above). That is, we present the 3 most confident categories, 2 most confident actions, and 1 most confident object. For example, in the first row we can see that once the light turns on in the room and the couch becomes visible the category *Sitting on a sofa/couch* fires, which in turn increases the likelihood of *sitting* in the next few frames. Furthermore, in Figure 5.14 we present similar visualizations, but only the 6 most confident categories, to further understand the interplay between the activity categories. In the first row, we can see a video of a person walking towards the camera, and we can see how one after the other the model recognizes cup, phone, and sandwich, and reasons about these connected activities. Finally, in Figure 5.15 we present a breakdown of the mean average precision (mAP) by our model for each class of the dataset, sorted by the mAP of our model.

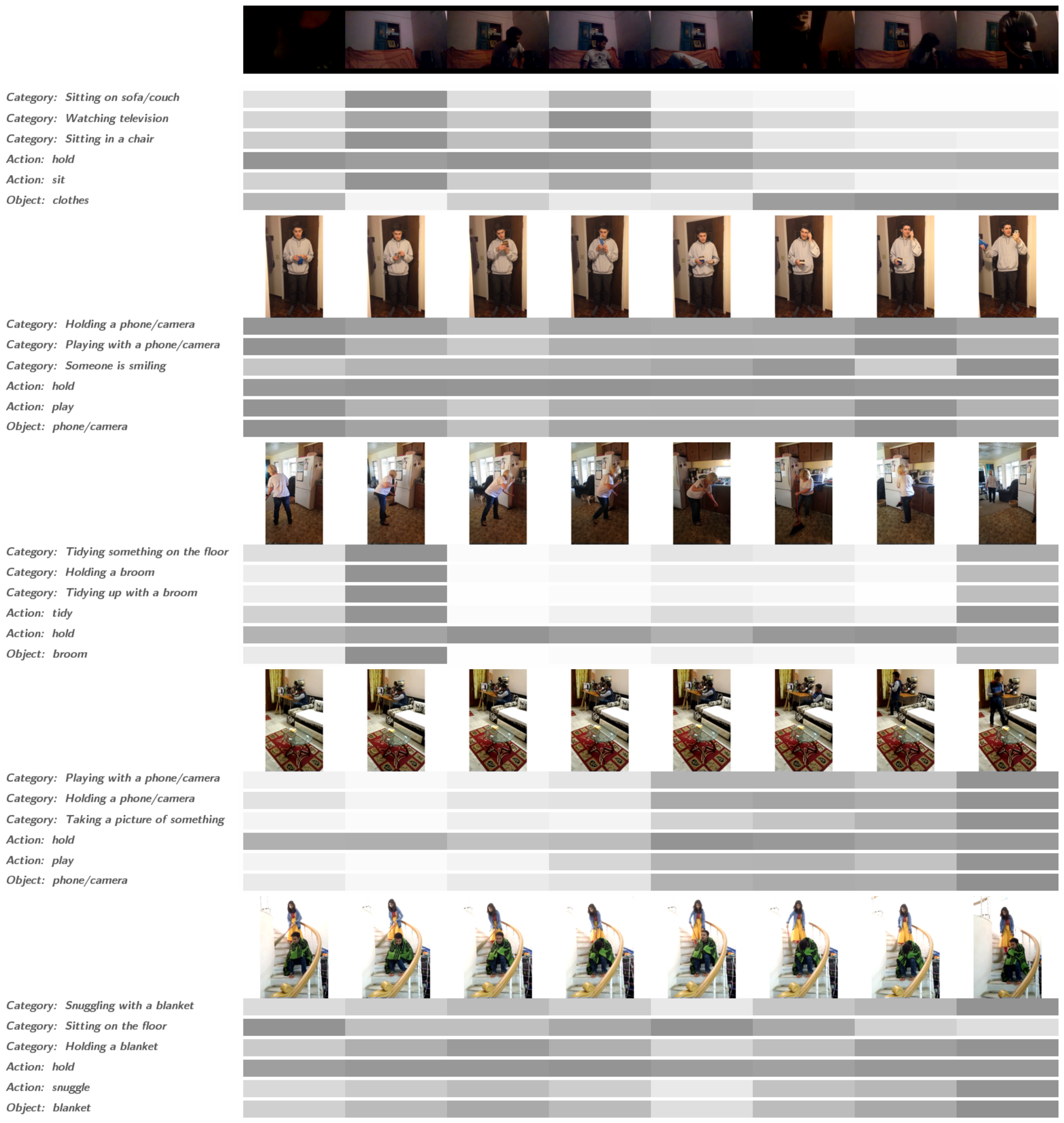


Figure 5.13: Visualizations of the model predictions for the 3 most confident categories, 2 most confident actions, and 1 most confident object. Darker colors indicate higher likelihood.

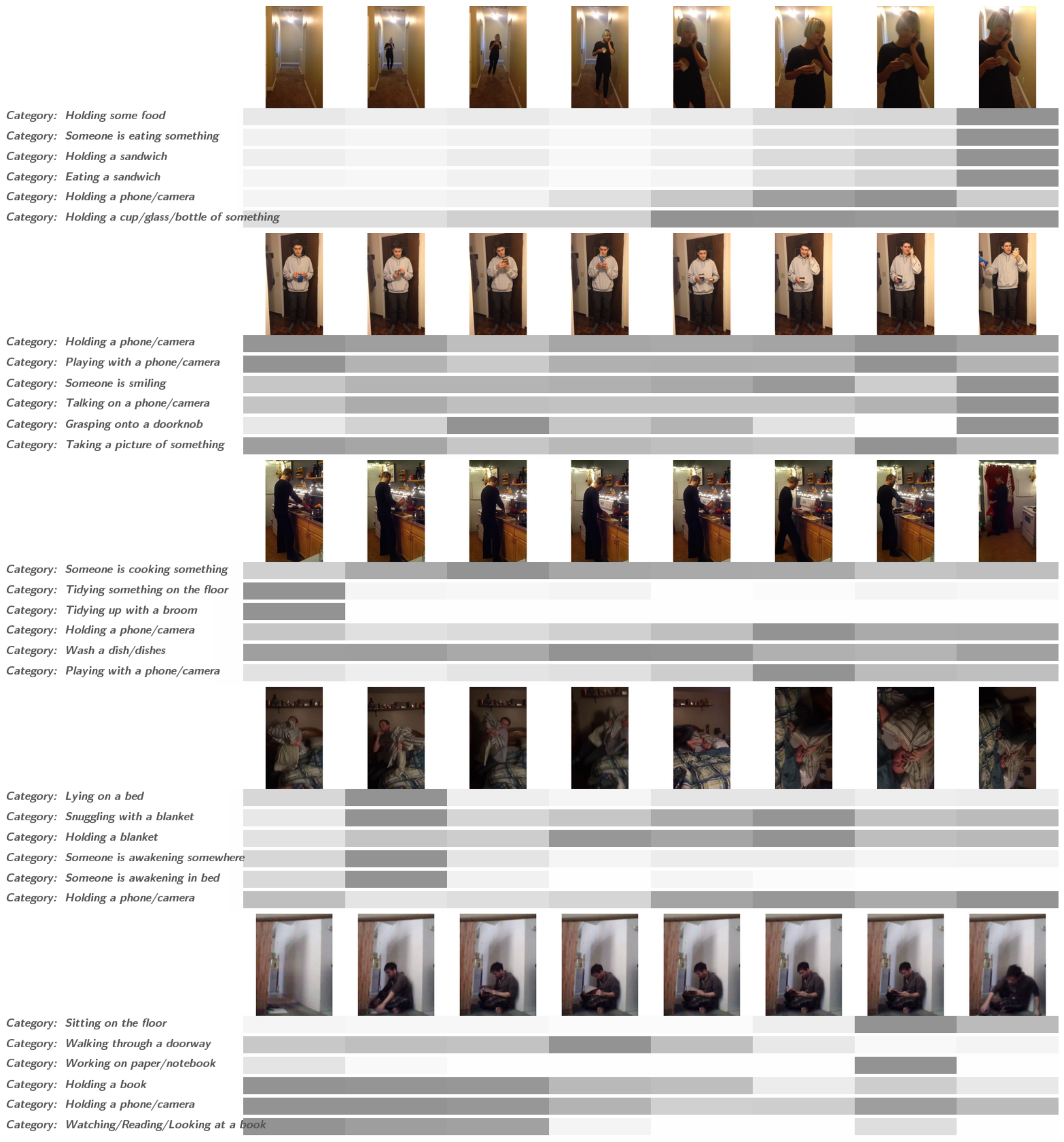


Figure 5.14: Visualizations of the model predictions for the 6 most confident categories. Darker colors indicate higher likelihood.

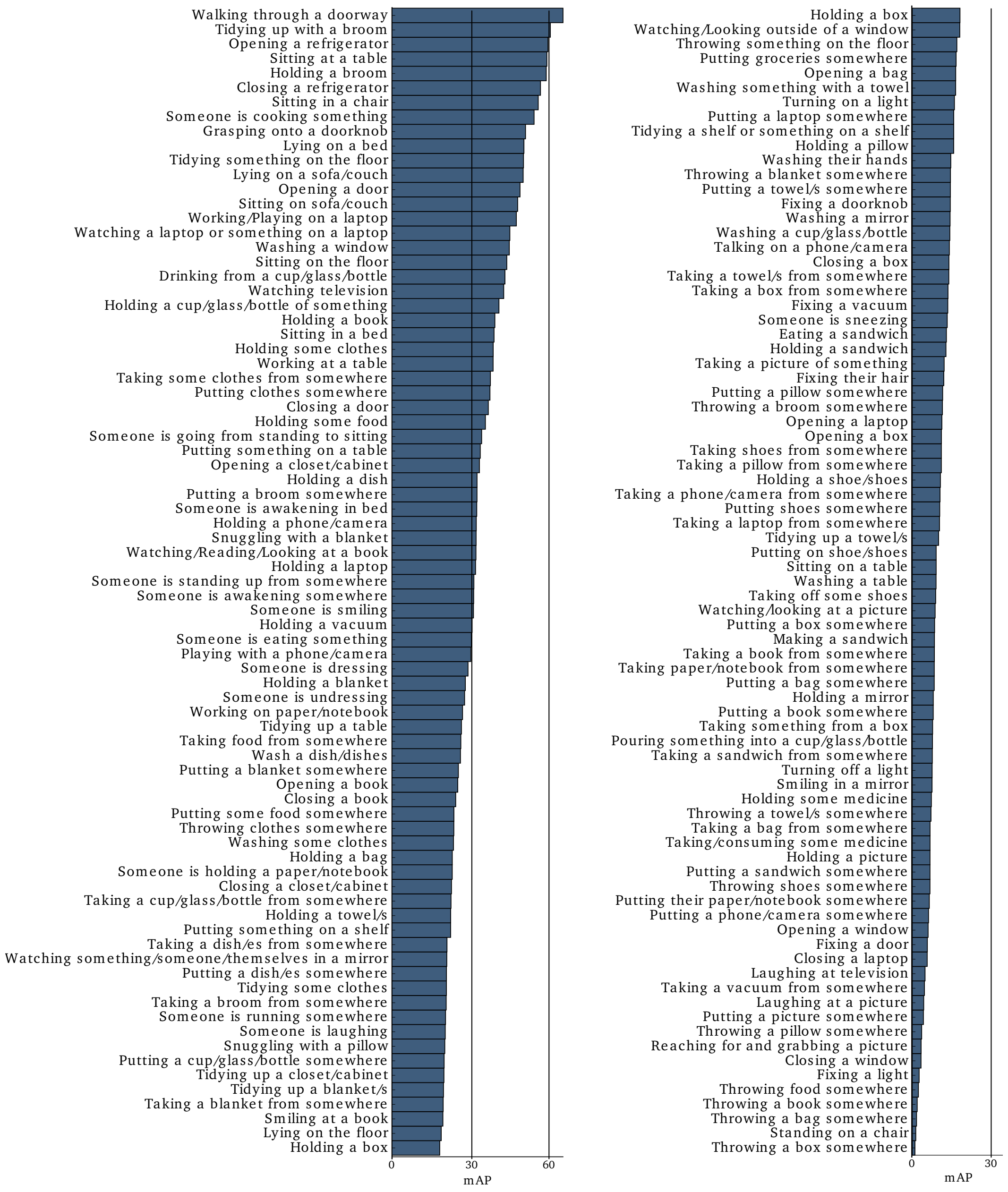


Figure 5.15: mAP for our model for all classes, sorted by mAP. The column on the right is the continuation of the left column.

Chapter 6

Modeling Week-long Activities

If we want to model activities that are much longer than the 30 second activities addressed in the previous chapter, we need to change our approach. For example, what does a typical visit to Paris look like? Do people first take photos of the Louvre and then the Eiffel Tower? Can we visually model a temporal event like "Paris Vacation" using current frameworks? This activity, or event, can be measured in weeks, not seconds. We explore how we can automatically learn the temporal aspects, or storylines of visual concepts from web data. Previous attempts focus on consecutive image-to-image transitions and are unsuccessful at recovering the long-term underlying story. Our novel Skipping Recurrent Neural Network (S-RNN) model does not attempt to predict each and every data point in the sequence, like classic RNNs. Rather, S-RNN uses a framework that skips through the images in the photo stream to explore the space of all ordered subsets of the albums via an efficient sampling procedure. This approach reduces the negative impact of strong short-term correlations, and recovers the latent story more accurately. We show how our learned storylines can be used to analyze, predict, and summarize photo albums from Flickr. Our experimental results provide strong qualitative and quantitative evidence that S-RNN is significantly better than other candidate methods such as LSTMs on learning long-term correlations and recovering latent storylines. Moreover, we show how storylines can help machines better understand and summarize photo streams by inferring a brief personalized story of each individual album.

In the past few years, there has been a remarkable success in learning visual concepts [21, 39] and relationships [21, 199] from images and text on the web. In theory, this allows the creation of systems that, given enough time and resources, can grow to know everything there is to learn. However, most of these approaches are still largely centered around single images and focus on learning static semantic relationships such as *is-part-of* [21], *is-eaten-by* [199] *etc*. Moreover, many semantic concepts have not only a visual aspect but also a temporal aspect or even storylines associated with them. For example, a visual representation of *Wedding* would involve guests entering the venue, followed by exchange of rings and finally celebrations in the wedding reception. How can we learn such visual storylines from the web as well?

There are two aspects to these storylines: the visual aspect, often represented by modes in visual appearances, and the temporal aspect, which is the temporal order in which these modes appear. How do we capture both of these aspects from the web data? User photo

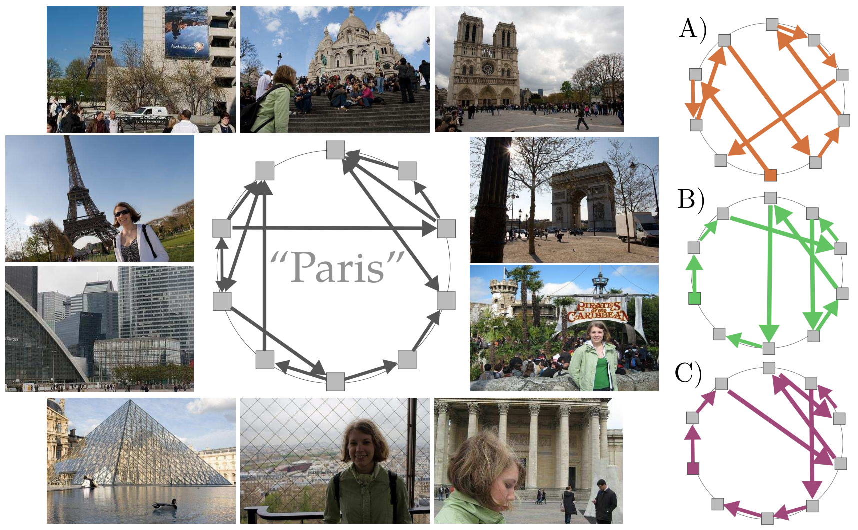


Figure 6.1: Given a concept, our algorithm can automatically learn both its visual and temporal aspect (storylines) from the web. To do this, we retrieve related albums from Flickr and apply our S-RNN model to automatically discover long-term temporal patterns. Here is a visualization of the storylines the model learned for the concept *Paris*. For visualization, we distill the top images that a trained S-RNN model prefers by sampling storylines from a *Paris* photo album. Denoting the images as nodes in a graph, we visualize the most common pairwise transitions using arrowed lines. On the right, we sample three probable storylines (A,B,C) that include these 10 images. We can see that the *Eiffel Tower* is prominent early in the story followed by sightseeing of common landmarks (*Arc de Triomphe* and others) and finally visiting the *Lourve*. On a map of Paris, the *Eiffel Tower* and the *Arc de Triomphe* are indeed in close proximity

albums in Flickr are a perfect example of web data that capture both aspects. First, most Flickr images are supplied with sufficiently informative tags, like *Paris* [94]. Second, meta-information like time is usually available. In particular, the photos in each album are taken in ordered sequences, which hypothetically embed common storylines for concepts such as *Paris*. Therefore, we propose to utilize Flickr photo streams across thousands of users and learn underlying visual storylines associated with these concepts. What is the right representation for these storylines and how do we learn it?

Recently, there has been momentous success in using CNN [122] features along with Recurrent Neural Networks [46,145,212,282,285,300] (RNNs) to represent those temporal dynamics in data [22,42,70,105,251,254,283,301]. We aim to extend that idea to modeling the dynamics in storylines. In theory, RNN can model any sequence, but has limited memory in practice, and can only learn short-term relationships due to vanishing gradients [13].

Our Skipping Recurrent Neural Network (S-RNN) skips through the photo sequences to extract the common latent stories, instead of trying to predict each and every item in the sequence. This effectively alleviates the artifacts of short-term correlations* (*e.g.* repetition) between consecutive photos in the stream, and focuses the learning effort towards the

*In our Flickr dataset, 71.1% of consecutive images are above average (cosine) similarity.

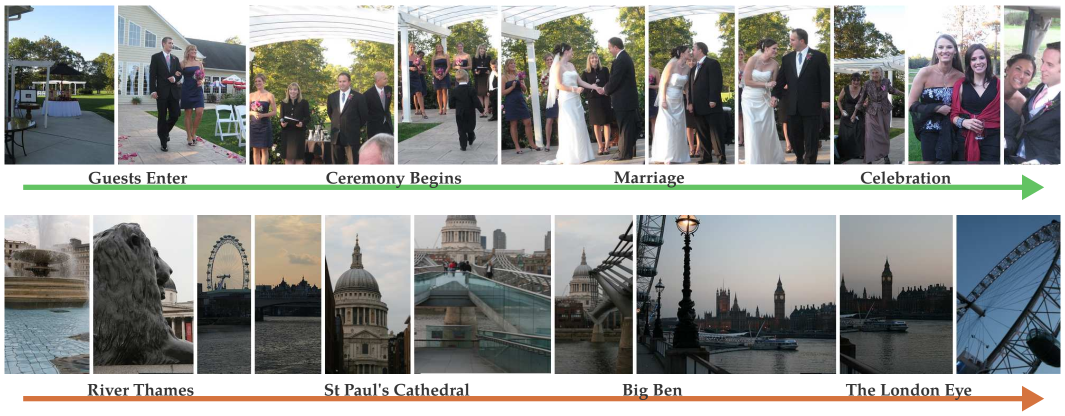


Figure 6.2: Given a concept, such as *Wedding*, our algorithm can retrieve an ordered collection of images to describe that concept (Sec. 6.3). In this figure we show the collections discovered by our model for two concepts. For example, for *Wedding* (first row), it picks images that represent four steps: guests enter; ceremony begins, marriage and celebration. For travel-related concepts like *London*, it prefers iconic landmarks for the story. The subtitles are manually provided for visualization. This is distilled from 1000 photo albums. More examples are provided in the appendix.

underlying story. This solution is complementary to, and different from, more complex RNN architectures such as LSTMs [86] that still focus on learning transitions between consecutive images. Similar to clustering, the S-RNN model can be efficiently trained in an *unsupervised* manner to learn a global storyline and infer a private story for each album. Different from most clustering techniques, S-RNN inherits the power of RNNs that can capture the temporal dynamics in the data.

We evaluate the effectiveness of our storyline model by comparing the storylines with baselines. In addition we evaluate the storyline model on two applications: a) image prediction [112, 113]; and b) photo album summarization [35, 109, 148, 160]. Constructing a convincing storyline for a concept of interest requires both visual and temporal aspects. Therefore, algorithms need to retrieve a diverse collection of images, with the right ordering among them. For *image prediction*, we show that our model is particularly suited for discovering the long-term correlations buried under the short-term repetitions in Flickr albums, while other approaches do not. Finally in the *summarization* task, the goal is to take images in a single photo album and select a small summary of those. A typical example is a series of photos, taken by a family on their visit to *Paris*, visiting all the iconic landmarks, such as the Eiffel Tower. Classically, summarization is approached by collecting a dataset of videos/albums and their associated summaries generated by people [26, 109, 111, 200, 281], in order to learn how to make a summary in a supervised way. This process is, however, considerably laborious. In this chapter, we specifically experiment with the hypothesis that a quality summary of an album can be constructed by exploiting the similarities across thousands of similar albums (*e.g. Paris*). Then a summary of the album is inferred by telling a personalized version of the story.

Contributions. a) We present a new way of approaching sequence modeling with RNNs, by exploring all ordered subsets of the data to avoid short-term correlations between consecutive

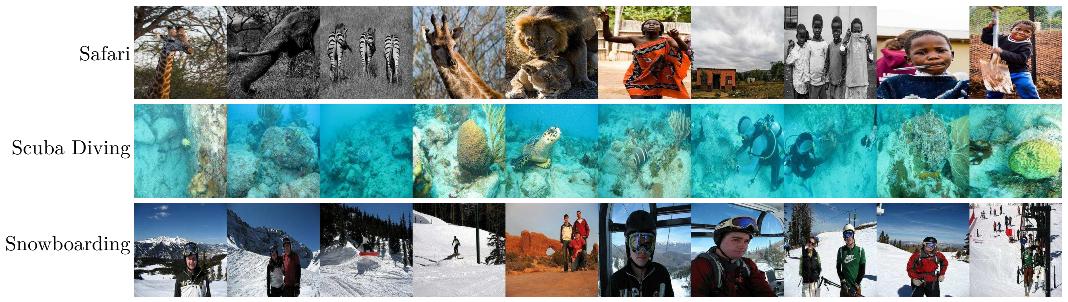


Figure 6.3: Given an individual photo album, our algorithm can summarize the photo album with a ordered collection of images that capture the album in terms of its underlying concept, by first learning about the concept from thousands of albums. (Sec. 6.3). In this figure we show the summaries generated for three photo albums. One about a *Safari*, the second about *Scuba Diving*, and the third *Snowboarding*. More examples are provided in the appendix.

elements in the sequence. b) We present the novel S-RNN architecture that efficiently implements this idea on web-scale datasets. c) We demonstrate that this method can learn visual storylines for a concept (*e.g*. *Paris*) from the web, by showing state-of-the-art results on selecting representative images, long-term image prediction, and summarizing photo albums.

6.1 Background

Learning storylines. The earliest form of storyline can be traced back to the 1970-80s, where *scripts* [208] (structured representations of events, causation relationships, participants, *etc*.) are used as knowledge backbones for tasks like text summarization and question answering. Unfortunately, these rich knowledge structures require hand construction by the experts, which fundamentally limits their usage in an open domain. This motivates the recent developments of *unsupervised* approaches that can learn underlying storylines automatically [18,151] from text. Inspired by this idea, our work aims to acquire the temporal aspect of a concept automatically from images. Similar work in vision is limited by either the scale of the data [262,281] or the domain to which the approach is applied [74]. Perhaps the most similar work is [112,113], where the storyline graphs are learned for Flickr albums. However, our work differs in several important aspects. First, while [113] is an important step in learning storylines, it focuses its learning effort on each and every pairwise transition, but our method learns the long-term latent story. In fact, [113] could be extended using this framework, but here we extend a standard RNN model. Second, our method requires no a-priori clustering, feature independence, nor a Markov assumption, and does parameters sharing like RNNs.

Temporal visual summarization. Summarizing video clips is an active area of research [245]. Many approaches have been developed seeking cues ranging from low-level motion and appearances [17,35,160] to high level concepts [109,132] and attentions [143]. This line of research has been recently extended to photo albums, and more external factors are considered for summarization besides the narrative structure. For example, in [228] the authors put forward three criteria: quality, diversity, and coverage. Later, in [164] a system is proposed that considers the social context (*e.g.* characters, aesthetics) into the summarization framework. Sadeghi *et al.* [200] also consider if a photo is memorable or iconic. Moreover, most of these approaches are *supervised*, namely the associated summaries for videos/albums are first collected by crowd-sourcing, then a model is learned to generate good summaries. While performance-wise it may seem best to leverage human supervision and external factors when available, practically it suffers serious issues like scalability and inconsistency in the ground-truth collection process, and generalizablility when applied to other domains. On the other hand, the task of summarization will be less ambiguous if the concept is given, which is exactly what we want to explore here.

Sequential learning with RNNs. Recurrent neural networks [46] are a subset of neural networks that can carry information across time steps. Compared to other models for sequential modeling (*e.g.* hidden Markov models, linear dynamic systems), they are better at capturing the long-range and high-order time-dependencies, and have shown superior performance on tasks like language modeling [155] and text generation [237]. We extend the network to model high dimensional trajectories in videos and user albums through the space of continuous visual features. Interestingly, since our network is trained to predict images several steps away, it can be viewed as a simple and effective way to learn long term memories [86] and predict context [157] as well. Fundamentally, LSTM still looks at only the next image and decides if it should be stored it in memory, but S-RNN reasons over all future images, and decides which it should store in memory (greedy vs. global). We outperform multiple LSTM baselines in our results. Furthermore, running LSTMs directly on high-dimensional continuous features is non-trivial, and we present a network that accomplishes that.

6.2 Learning Visual Storylines

Given hundreds of albums for a concept, our goal is to learn the underlying visual appearances and temporal dynamics simultaneously. Once we have learned this by building upon state-of-the-art tools, we can use it for multiple storyline tasks, and distill the explicit knowledge as needed, such as in Fig. 6.1. In this section, we explain our novel S-RNN architecture that is trained over all ordered subsets of the data, and show that this can be accomplished with update equations equally efficient to original RNN. The full derivation of these update equations by using the EM-method is presented in the appendix. We formulate the storyline learning problem as learning an S-RNN. To understand S-RNN, we start by introducing the basic RNN model.

Recurrent Neural Networks The basic form of RNN [46] models a time sequence by decomposing the probability of a complete sequence into sequentially predicting the next item given the history (in our application, this sequence is images in a temporal order). Given a sequence of T images $\mathbf{x}_{1:T} = \{x_1, \dots, x_T\}$,[†] the network is trained to maximize the

[†]For simplicity in notation, we assume a single training sequence, but in our experiments we use multiple albums for one concept to discover *common* latent storylines.

log-likelihood:

$$\begin{aligned} \mathcal{M}^* &= \arg\max_{\mathcal{M}} \log \mathrm{P}(\mathbf{x}_{1:T}; \mathcal{M}) - \lambda \mathcal{R}(\mathcal{M}) \\ &\text{where } \log \mathrm{P}(\mathbf{x}_{1:T}; \mathcal{M}) = \sum_t \log \mathrm{P}\left(x_{t+1} | \mathbf{x}_{1:t}; \mathcal{M}\right). \end{aligned} \tag{6.1}$$

Here $\mathcal{M}$ is the set of all model parameters, and $\mathcal{R}(\cdot)$ is the regularizer (*e.g.* ℓ_2). The probability $\mathrm{P}(\cdot|\cdot,\cdot)$ is task dependent, *e.g.* for language models it directly compares the soft-max output y_t with the next word x_{t+1} [155]. The standard optimization algorithm for RNNs is Back Propagation Through Time [276, 277] (BPTT), a variation of gradient ascent where the gradient is aggregated through time sequences.

The model consists of three layers: input, recurrent, and output. The input layer uses the input x_t to update the hidden recurrent layer h_t using weights $\mathbf{W}_I$. The recurrent layer h_t updates itself via $\mathbf{W}_R$ and predicts the output $\mathbf{y}_t$ via weights $\mathbf{W}_O$. The update function at step t writes as follows:

$$h_t = \sigma\left(\mathbf{W}_I x_t + \mathbf{W}_R h_{t-1}\right); \quad y_t = \zeta(\mathbf{W}_O h_t). \tag{6.2}$$

Here $\sigma(\cdot)$ and $\zeta(\cdot)$ are non-linear activation functions, *e.g.* sigmoid, soft-max, rectified linear units [122], *etc.* All the history in RNN is stored in the memory h_t. This assumes conditional independence of x_{t+1} and $\mathbf{x}_{1:t}$ given h_t.

In practice, the recurrent layer h_t has limited capacity and the error cannot be back propagated effectively (due to vanishing gradients [13]). This can be a critical issue for modeling sequences like photo streams—due to the high correlation between consecutive images, where the dominant pattern in the short term is *repetition*. For example, people can take multiple pictures of the same object (*e.g.* the Eiffel Tower or family members), or the entire album is about things that are visually similar (*e.g.* artwork in the Louvre or fireworks). This pattern is so salient that if an RNN is directly trained on these albums, the signals of underlying storylines are largely suppressed. How to resolve this issue of learning long-term patterns? One way is to regularize RNN with a diversity term [228]. However, note that if an album is indeed single-themed, we still want visually similar images in the storyline. Furthermore, Flickr tags are not perfect and noise in the album set can easily distract the model.

Skipping Recurrent Neural Networks We now build upon the RNN framework to propose a skipping recurrent neural network model. Instead of learning each consecutive transition, S-RNN chooses to learn a "higher-level" version of the story, and focuses its learning effort accordingly. The key underlying idea is to select the storyline nodes by skipping a lot of images in the album and then modeling the transitions between the images selected as nodes.

Formally, let us suppose $\mathbf{x}_{1:T}$ represents the T images in the album, $\mathbf{z}_{1:N}$ is the set of indexes that represent the selected images for the storyline and the constant N is the number of nodes in the storyline. Note that $N \ll T$, $z_n \in \{1, 2, \ldots, T\}$, and $z_n < z_{n+1}$ since $\mathbf{z}$ defines an ordered subset. Our goal is to learn the maximum likelihood model parameters ($\mathcal{M}$) by maximizing the marginal likelihood of the observed data. Therefore, our objective function is:

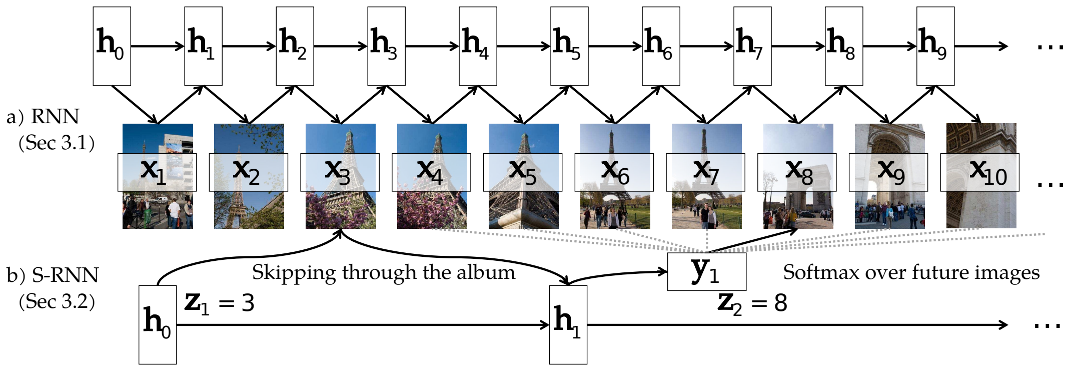


Figure 6.4: Our S-RNN model (unrolled in time). Instead of trying to predicting each and every photo in the sequence (as in the basic RNN model), latent variables $\mathbf{z}_n$ are introduced into our model to skip through the photo sequences, which is an effective strategy to address the local repetition issue (multiple pictures are taken for a single object like the Eiffel Tower) and can help extract common latent stories in the entire set of albums related to a concept (*e.g. Paris*). To overcome the high-dimensional regression problem, the loss is an softmax loss over future images

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} \log \sum_{\mathbf{z}_{1:N}} \mathrm{P}(\mathbf{x}_{1:T}, \mathbf{z}_{1:N}; \mathcal{M}) - \lambda \mathcal{R}(\mathcal{M}). \tag{6.3}$$

We can factorize $\mathrm{P}(\mathbf{x}_{1:T}, \mathbf{z}_{1:N}; \mathcal{M})$ as $\mathrm{P}(\mathbf{x}_{1:T}|\mathbf{z}_{1:N}; \mathcal{M})\mathrm{P}(\mathbf{z}_{1:N})$ where $\mathrm{P}(\mathbf{z}_{1:N})$ is a prior on $\mathbf{z}$. As described above, we use a simple prior on $\mathbf{z}$ that it is an ordered subset. Here we make an assumption that the likelihood of a whole album is proportional to the likelihood of the selected sub-sequence of images $\mathbf{x}_\mathbf{z}$ (that is, we assume $\mathrm{P}(\mathbf{x}_{1:T}|\mathbf{z}; \mathcal{M}) \propto \mathrm{P}(\mathbf{x}_\mathbf{z}; \mathcal{M})$). Factorizing, and inserting this assumption into Eq. 6.3 we have:

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} \log \sum_{\mathbf{z}_{1:N}} \left(\prod_n \mathrm{P}\left(\mathbf{x}_{z_{n+1}} | \mathbf{x}_{\mathbf{z}_{1:n}}; \mathcal{M}\right) \right) \mathrm{P}(\mathbf{z}_{1:N}) - \lambda \mathcal{R}(\mathcal{M}). \tag{6.4}$$

We observe that this equation is starting to look similar to standard RNN (Eq. 6.1).

Maximizing the S-RNN Objective Maximizing the marginal likelihood over all possible subsets of $\mathbf{z}$ is computationally intractable. Therefore, we make use of the Expectation Maximization (EM) algorithm, and then sequentially factor the update equations. More details of the EM derivation are given in the appendix. During the E-step, we sample $\mathbf{z}$ given the current model, and use that to train the model in the M-step, as we would an RNN. We initialize the EM-algorithm by setting $\mathbf{z}$ based on a randomly ordered subsets of images.

S-RNN Implementation Details Now that we know how to optimize the objective, the only design choice left is the loss $\mathrm{P}\left(\mathbf{x}_{z_{n+1}} | \mathbf{x}_{z_{1:n}}; \mathcal{M}\right)$ (the data likelihood in Eq. 6.4). While Gaussian likelihood is often used for real-valued regression, we recognize that the space of allowed future images is not infinite, but simply images after x_{z_n}, defined as $\mathcal{X}_n$. Thus the likelihood is defined as a *softmax likelihood over the future images*:

$$\mathrm{P}\left(x_{z_{n+1}}|\mathbf{x}_{\mathbf{z}_{1:n}};\mathcal{M}\right)=\frac{\exp(\mathbf{y}_n^T x_{z_{n+1}})}{\sum_{x\in\mathcal{X}_n}\exp(\mathbf{y}_n^T x)} \tag{6.5}$$

where $\mathbf{y}_n$ is the output of the network after step n. Effectively, this avoids modeling the negative world as "everything except the ground truth" and instead models the negative world as "other possible choices". This significantly helps with high-dimensional data ($fc7$ features), since the possible image choices in an album are usually only few hundred, but visual features few thousand.

In summary, during training and testing, **z** is sequentially sampled using the current model (which skips through the sequence), and during training those samples used to sequentially update the network with BPTT to maximize the objective. A visualization of the idea can be found in Fig. 6.4. A full implementation of is available.

6.3 Experiments

Since there has been so little done in the area of learning storyline models and their applications, there are no established datasets, evaluation methodologies, or even much in terms of relevant previous work to compare against. Therefore, we will present our evaluation in two parts: (a) first, in Section 6.3, we directly evaluate how "good" our learned storyline model is. Specifically, we ask the Amazon Mechanical Turk (AMT) users how good our storyline model is compared to a baseline in terms of the representativeness and the diversity of image nodes in the storyline model; (b) next, in Section 6.3 and Section 6.3, we evaluate our storyline model for two applications: long-term prediction and album summarization. For these tasks, we show qualitative, quantitative, and user studies to demonstrate the effectiveness of S-RNN based storyline model. We begin by describing our data collection process and the baselines.

Flickr Albums Dataset We gather collections of photo albums by querying Flickr through the YFCC100M dataset [240], a recently released public subset of the Flickr corpus containing 99.3 million images with all the meta-information like tags and time stamps. This dataset is an unrefined subset of images on Flickr, making it a reproducible way of working with web data. The selection process gathers at most 1000 photo albums for a single concept (*e.g. Paris*), with an average size of 150 images. Each album is sorted based on a photo's date taken. We experimented with seven concepts: *Christmas*, *London*, *Paris*, *Wedding*, *Safari*, *Scuba-diving*, and *Snowboarding* with a total number of 700k images. Examples from the dataset are provided in the appendix. This subset will be made available.

Implementation Details We compare our S-RNN model with several approaches to demonstrate its effectiveness in learning visual storylines. For fairness, all the methods used the same *fc7* features from AlexNet [122] pre-trained on ImageNet.

For **S-RNN**, the *fc7* features are directly fed into the model. The network is trained with BPTT, which unrolls the network, and uses gradient ascent with a momentum of 0.9. We set the starting learning rate as 0.05, and gradually reduce it when the likelihood on the validation set no longer increases. The input size of the layer is set to 4096 (size of $fc7$), and the hidden recurrent layer size 50. We keep $N = 10$ for all the concepts as a good compromise between content and brevity (The appendix contains analysis of different sizes of N). We choose ℓ_2 regularization and set weight decay λ to be 10^{-7}. Training takes approximately 2-3 hours on a single CPU. Each story was generated by sampling from the model 500 times, and picking the sampled sequence with the highest likelihood. The code is available at `github.com/gsig/srnn`.

Below we list the main baselines, and note that additional baselines will be added for individual experiments when necessary.

Sample. We uniformly sample from the data distribution.

K-Means. To take advantage of the global storylines shared in a concept, we apply K-Means to all the albums (similar to the first step of [113] except with different features).

Graph. We adapted the original code for [113] to use $fc7$ features. Then a storyline is generated with the forward-backward algorithm as described in [113].

RNN. This architecture is similar to a language model [155] except it predicts the cluster (as in *K-Means*) of the next image. We sample without replacement to generate the story. This is a standard application of RNN to the problem.

LSTM. We train an LSTM network [104], similar to the RNN baseline.

LSTMsub. LSTM trained as before, but when generating the summary, we first generate a longer sequence ($N = 100$) and then sub-sample that sequence to the desired summary length 10. Intuitively, if LSTM was indeed able to learn the long-term correlations regardless of the repetitions, this should perform well.

S-RNN-. For ablation analysis, we also provide a baseline where we use the network without skipping, but with the softmax loss over future images. All the hyper-parameters for training are kept identical to our model except the network predicts each and every item in the sequence. This is similar to RNN, but benefits from our improved loss.

D-RNN Similar to S-RNN-, except trained on a diverse subset of each album using the k-means++ algorithm [9]. This was significantly better than other variants, including training on random subsets, fixed interval subsets, or a random diverse subset.

Evaluating Storylines In the first experiment, we directly evaluate how "good" the learned storyline model is for a given concept. We define the goodness of a storyline model in terms of how representative and diverse the selected images are for a given concept. Two qualitative examples for *Wedding* and *London* are shown in Fig. 6.2. Fig. 6.5 shows more examples of learned storylines for different concepts. Our storyline model captures the essence of scuba-diving, snowboarding, etc., by capturing representative and diverse images (e.g., beer, fun and snowboarding during day).

Setup. For each concept, we have each method select only 10 images from 50 photo albums (thousands of photos) that best describe the concept, and AMT workers select which one they prefer. Each algorithm has access to the full training data to train the model. For Graph and RNN-based baselines, we sample multiple times from each album and use the highest ranked collection in terms of likelihood. Sample and K-Means are simply applied on all images, and in K-Means we assign the closest image to each cluster center. The appendix contains more qualitative examples.

Results. Table 6.1 summarizes the results. Each comparison was given to 15 separate AMT

	K-Means	**Sample**	**Graph**	**LSTM**	**LSTMsub**	**D-RNN**	**RNN**	**S-RNN-**
S-RNN	71.2%	68.3%	79.8%	84.3%	70.9%	60.0%	85.1%	75.5%

Table 6.1: *Evaluating Storylines*. Fraction of the time our S-RNN storylines are preferred against competing baselines. 50% is equal preference. Our method significantly outperforms the baselines, being preferred 60**%** of the time against the strongest baseline. See Section 6.3 for details

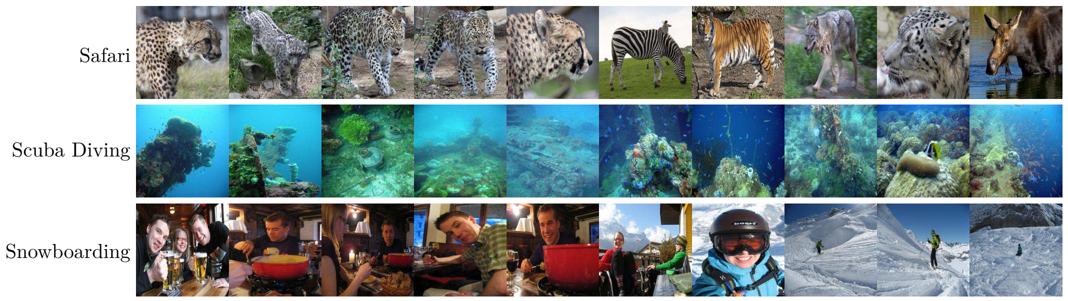


Figure 6.5: *Evaluating Storylines*. Images selected by S-RNN for three storylines from thousands of images for the concepts *Safari*, *Scuba Diving*, and *Snowboarding*

workers. We can see that S-RNN is preferred **60% of the time against the strongest baseline** across all the concepts. Different baselines fail in different ways. For example, Sample and K-Means can capture a diverse set of images to represent the concept, but are prone to the inherent noise in the Flickr albums. On the other hand, Graph and LSTM overfit to the short-term correlations in the data and select repetitive images. Finally, S-RNN outperforms D-RNN since S-RNN is not restricted to a single specific diversity method as in D-RNN.

Task1: Prediction Next, we evaluate our storyline models for two applications. The first application we consider is the *prediction* task. There are two possible prediction goals: short-term prediction and long-term prediction. Short-term prediction can be considered as prediction of the next image in the album. This was the task used in [112,113]. In the case of long-term prediction, we predict the next representative event. In the case of *Paris* vacation, if the current event is Eiffel tower, the next likely event would be visiting the Trocadero. In the case of *Wedding*, if the current event is the ring ceremony, then the next representative event is the kiss of the newlyweds.

Setup. For the short-term prediction, the ground-truth is the next image in the album. But how do we collect ground-truth for long-term prediction? We ask experts to summarize the albums (hoping that album summaries will suppress short-term correlations and capture only representative events). Now we can reformulate long-term prediction as predicting the next image in the human-generated *summary* of the album. We collected 10 ground truth summaries on average for each concept from volunteers familiar with the concepts (such as *Paris*, and *London*). Each summary consists of 10 images from a photo album that capture what the album was about. This was used as ground truth only for evaluation.

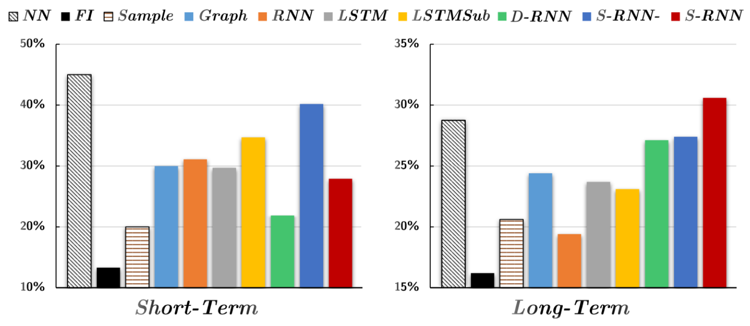


Figure 6.6: *Predicting the next image.* S-RNN is best at capturing long-term correlations, and nearest-neighbors is best at capturing short-term correlations, as expected

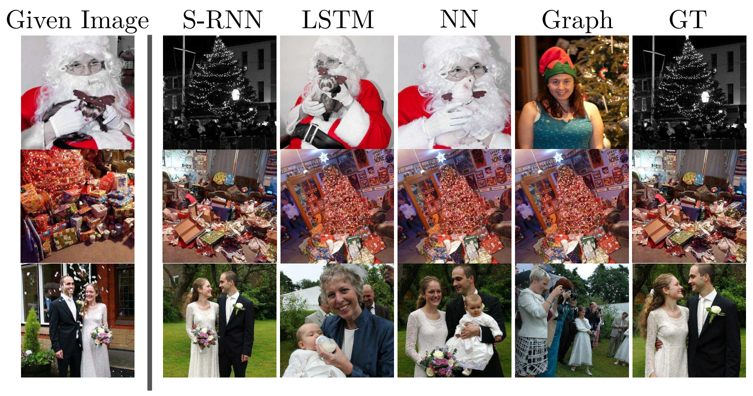


Figure 6.7: *Long-term prediction.* Examples of the images predicted by our method compared to baselines. The image is chosen from a line-up of five images from the same album (generated by experts as summaries). We see our method captures *Santa→Tree* and *Closed Presents→Open Presents* while the baselines focus on similar images

Two settings are compared, the first one (labeled "*long-term*") predicts the next image in a *summary* (N=198 over 10 folds each); and the second one (labeled "*short-term*") predicts the next image in the original photo album (N=1742). The problem is posed as a classification task choosing from the true image, and four other images selected uniformly at random from the same album. Here we also consider **NN** that simply picks the nearest neighbor, and **FI** that picks the furthest image from the given image, both in cosine distance of $fc7$ features. K-Means is not suitable for this task since it does not include temporal information. All methods were trained in an unsupervised manner for each concept as before.

Results. In Fig. 6.6 we present results for the prediction of the next image. When we consider long-term interactions between images, S-RNN successfully predicts the next image in the storyline **31**% of the time, significantly higher than baselines. On the other hand, we can see that when we simply want to predict consecutive images, NN is the best. To further visualize the results for "*long-term*" correlations, we also give example comparisons with baseline methods in Fig. 6.7.

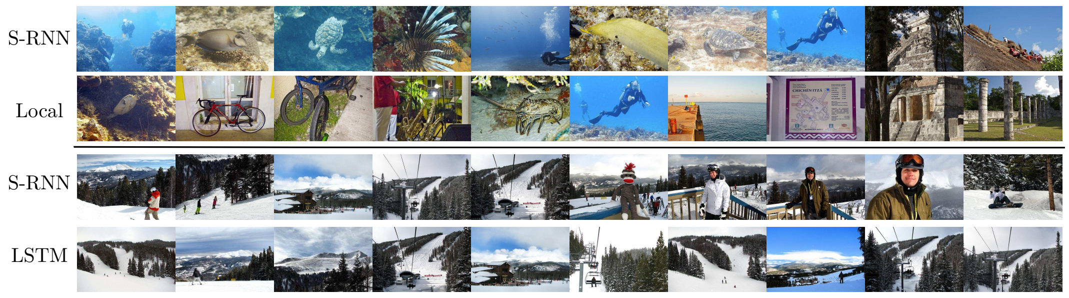


Figure 6.8: Examples of summaries generated by our method and two representative baselines for *Scuba-diving* and *Snowboarding*. In the *Scuba-diving* example Local aims to capture diversity, and thus our method is more relevant. In *Snowboarding*, LSTM focuses on short-term correlations, and chooses many similar images, while our method effectively captures the album

Task2: Photo Album Summarization In the final experiment, we evaluate on the task of album summarization. In particular, we focus on summarizing an individual album based on the concept (*e.g.* a *Paris* album), rather than heuristics such as image quality or presence of faces [164, 200, 228]. This experiment addresses the question whether storylines can help to summarize an album.

Human Generated Summaries. Photo album summarization is inherently a subjective and difficult task. To get a sense of the difficulty, we first compared the human summaries (used in Sec. 6.3) to baselines with a separate AMT preference study. We had two findings. First, for some concepts, such as *Wedding*, the albums are frequently already summaries by professional photographers, and thus generating summaries is trivial. Specifically, there is no significant difference between human generated summaries and uniformly sampling from the data distribution (Sample). We thus only evaluate on concepts where there is significant difference between human generated summaries and ones generated by baselines. Second, we found human generated summaries are only preferred $\mathbf{59.5}\%$ of the time against the strongest baseline.

Setup. The photo albums for a given concept are randomly divided into a training set and a validation set with a ratio of 9:1, and no ground truth summaries were provided. We additionally consider the baseline **Local** where K-Means clustering is used for summarization by applying clustering on *fc7* features for each individual album. As before, we assign the closest image to the cluster center for clustering-based methods. While it is not required for S-RNN, we sort the selected photos in temporal order as a post-processing step for all the baselines when necessary for fair comparison.

Qualitative Results. The results for a few concepts are presented in Fig. 6.8. We can see that S-RNN captures a set of relevant images without losing diversity. In contrast, Local captures only diversity, and LSTM that tries to learn short-term correlations between consecutive images, and as result often prefers similar images in a row. Additional summaries by S-RNN are presented in Fig. 6.3.

Quantitative Evaluation. To directly compare the quality of the generated summaries, another AMT preference study was conducted. For S-RNN and each baseline, 200 random pairwise comparisons were generated. Each question was given to 5 separate workers for

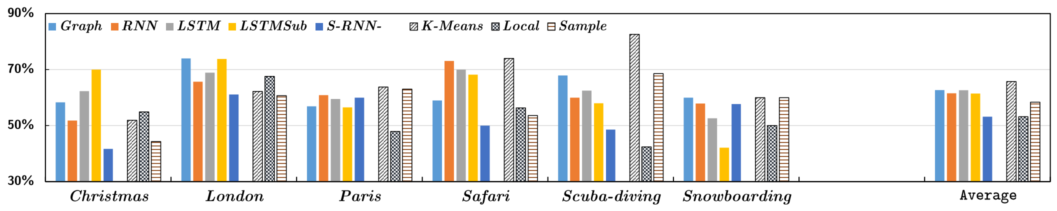


Figure 6.9: Photo album summarization. AMT pairwise preference between our method and multiple baselines. 70% means that summaries by our method were preferred 70% of the time. It is important to keep in mind that compared to the strongest baseline, a human generated summary was on average only preferred 59.5% of the time. Sec. 6.3 contains a detailed explanation of the experiment setup and analysis of the results

consistency. We used a consensus approach where a comparison gets a score of 1 if there is a tie, or a score of 2 if there is consensus.

In Fig. 6.9 we present comparison with the baselines. We can see that on average our method is preferred over all the baselines. To provide a more detailed analysis, we divide the baseline methods into two groups: the *Storyline* group (filled with pure colors) that captures the latent temporal information in the data, and the Non-Storyline group (filled with patterns) that do not. The *Storyline* group includes Graph, RNN, LSTM, LSTMSub, and S-RNN- (Our method also falls into this group), while the *Non-Storyline* group has K-Means, Local and Sample. There are few interesting points:

1. S-RNN performs relatively better on travel-related albums (*Paris*, *London*) suggesting it is easier to latch onto landmarks than high-level concepts like in *Christmas*.

2. For concepts like *Christmas*, methods that learn short-term correlations from the data distribution are still preferred by the users. The fact that S-RNN- outperforms LSTMs and RNNs, can be interpreted as follows. RNNs suffer from the curse of dimensionality if naively applied to storyline learning, but the S-RNN loss reduces the dimensionality of the output space by an order of magnitude (4096 to 100s).

3. While simple as they seem, Local and Sample are very competitive baselines. We believe the reason is that Local aims to provide a diverse set of images from each album, and Sample is representative of the underlying data. Therefore, with the post-processing step that re-arranges the selected images in temporal order, these methods can do well on good albums. However, they do poorly when the album is noisy, as illustrated in Fig. 6.8 first example.

Does Time Information Help Summarization? For further analysis, we compared the described S-RNN with S-RNN trained on shuffled data (ordering discarded) with a preference study on AMT. S-RNN using the time information was preferred 68.4% over S-RNN without time information, demonstrating that the time information significantly helps to generate a summary liked by people.

Transferring storyline knowledge. Each album can have different stories and themes. In Figure 6.10 we present two different summaries of two photo albums. The first album is a *Scuba Diving* album, and the first summary from that album is generated with the model

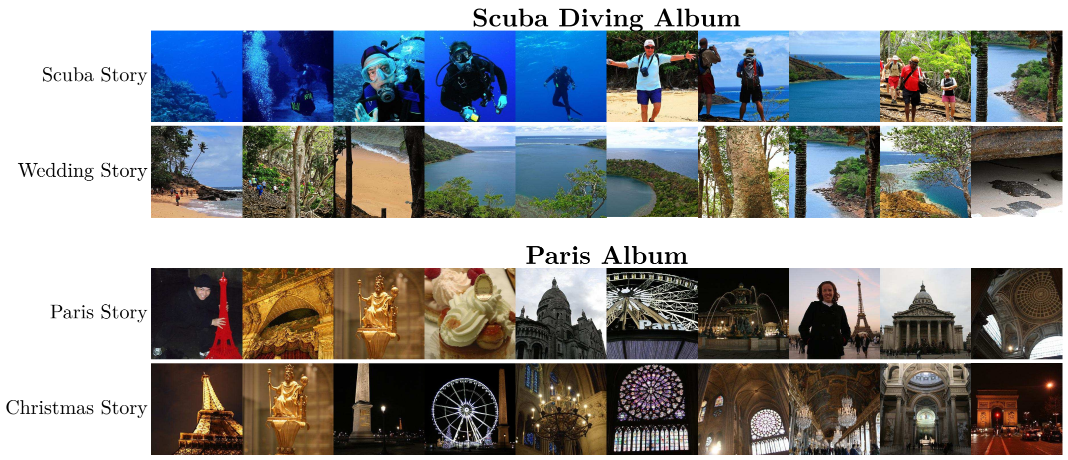


Figure 6.10: The first two rows show a *Scuba Diving* album summarized with a *Scuba* model and a *Wedding* model, and the last two show a *Paris* album summarized with a *Paris* model and a *Christmas* model. The *Wedding* story emphasizes the beach resort images of the *Scuba* album, and the *Christmas* story emphasizes the churches and sparkling lights images in the *Paris* album

trained on *Scuba Diving* albums. In the second row, the same album is summarized using a model trained on *Wedding* albums. We can see that this emphasizes scenic beach pictures reminiscent of a beach resort wedding. The second album is a *Paris* album, and the first summary is generated using *Paris* model. The second summary however, is generated using a *Christmas* model, and we can see that this emphasizes pictures of churches and sparkling lights at night.

6.4 Discussion

We have presented an approach to learn visual storylines for concepts automatically from the web. Specifically, we use Flickr albums and train an S-RNN model to capture the long-term temporal dynamics for a concept of interest. The model is designed to overcome the challenges posed by high correlations between consecutive photos in the album if sequence predictors are directly applied. We evaluate our model on learning storylines, image prediction and album summarization, and show both qualitatively and quantitatively that our method excels at both extracting salient visual signals for the concept, and learning long-term storylines to capture the temporal dynamics.

6.5 Appendix

6.5.1 EM-Derivation of the Update Equations

We begin with a standard RNN. Given T data points $\mathbf{x}_{1:T} = \{x_1, \dots, x_T\}$ (images) and the model parameters $\mathcal{M}$, RNN maximizes $\mathrm{P}(x_t|\mathbf{x}_{1:t-1}; \mathcal{M})$ at step t, which is an exact decomposition of $\mathrm{P}(\mathbf{x}_{1:T}; \mathcal{M})$, no independence assumed.

The key idea of S-RNN is to train over all ordered subsets (of size N) in $\mathbf{x}_{1:T}$, selected by latent variables $\mathbf{z}_{1:N}=\{z_1, \dots, z_N\}$. Here we assume the likelihood of $\mathbf{x}_{1:T}$ selected by $\mathbf{z}_{1:N}$, is only related to the selected subset $\hat{\mathbf{z}}_{1:N}=\{\hat{z}_1, \dots, \hat{z}_N\}$, namely (here $\mathbf{x}_{\hat{\mathbf{z}}_{1:N}}=\{x_{\hat{z}_1}, \dots, x_{\hat{z}_N}\}$):

$$\mathrm{P}(\mathbf{x}_{1:T}|\mathbf{z}_{1:N}=\hat{\mathbf{z}}_{1:N}; \mathcal{M}) \propto \mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}_{1:N}}|\mathbf{z}_{1:N}=\hat{\mathbf{z}}_{1:N}; \mathcal{M}). \tag{6.6}$$

S-RNN maximizes the likelihood of $\mathbf{x}_{1:T}$ over all $\mathbf{z}_{1:N}$:

$$\begin{aligned}
\max_{\mathcal{M}} \mathrm{P}(\mathbf{x}_{1:T}; \mathcal{M}) &= \max_{\mathcal{M}} \sum_{\mathbf{z}_{1:N}} \mathrm{P}(\mathbf{x}_{1:T}|\mathbf{z}_{1:N}; \mathcal{M})\mathrm{P}(\mathbf{z}_{1:N}) && (6.7)\\
&= \max_{\mathcal{M}} \sum_{\mathbf{z}_{1:N}} \mathrm{P}(\mathbf{x}_{\mathbf{z}_{1:N}}|\mathbf{z}_{1:N}; \mathcal{M})\mathrm{P}(\mathbf{z}_{1:N}) && (6.8)\\
&= \max_{\mathcal{M}} \sum_{\mathbf{z}_{1:N}} \left(\prod_n \mathrm{P}(x_{z_{n+1}}|\mathbf{x}_{\mathbf{z}_{1:n}}, \mathbf{z}_{1:N}; \mathcal{M})\right) \mathrm{P}(\mathbf{z}_{1:N}) && (6.9)
\end{aligned}$$

Here we assume the prior $\mathrm{P}(\mathbf{z}_{1:N})$ does not depend on $\mathcal{M}$, and use Eq. 6.6. In Eq. 6.9 we also used the chain rule to make it more similar to RNN (this is Eq. 4 above), but here Eq. 6.8 is directly solved with the EM-algorithm and factorized later. In the **E-Step**, we sample $\mathbf{z}$ to approximate the expectation: (for simplicity we remove the subscripts in $\mathbf{z}_{1:N}$ and $\hat{\mathbf{z}}_{1:N}$)

$$Q(\mathcal{M}; \mathcal{M}_0) := \mathbb{E}_{\hat{\mathbf{z}} \sim q_0}\left[\log\left(\mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}}|\mathbf{z}=\hat{\mathbf{z}}; \mathcal{M})\mathrm{P}(\mathbf{z}=\hat{\mathbf{z}})\right)\right], \tag{6.10}$$

where q_0 is $\mathrm{P}(\mathbf{z}|\mathbf{x}; \mathcal{M}_0)$. For a single sample $\hat{\mathbf{z}}_{1:N}$, the **M-Step** follows:
(for simplicity of notation $\mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}}|\mathbf{z}=\hat{\mathbf{z}}; \mathcal{M})=\mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}}; \mathcal{M})$)

$$\begin{aligned}
\max_{\mathcal{M}} Q(\mathcal{M}; \mathcal{M}_0) &= \max_{\mathcal{M}} \log \mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}_{1:N}}; \mathcal{M}). && (6.11)\\
&= \max_{\mathcal{M}} \sum_n \log \mathrm{P}(x_{\hat{z}_{n+1}}|\mathbf{x}_{\hat{\mathbf{z}}_{1:n}}; \mathcal{M}). && (6.12)
\end{aligned}$$

This is the standard RNN objective except over a subset. The final implementation detail, is that we can rewrite $\mathrm{P}(\mathbf{z}|\mathbf{x}; \mathcal{M}_0)$ in a simpler form:

$$\begin{aligned}
\mathrm{P}(\mathbf{z}=\hat{\mathbf{z}}|\mathbf{x}; \mathcal{M}_0) &\propto \mathrm{P}(\mathbf{x}|\mathbf{z}=\hat{\mathbf{z}}; \mathcal{M}_0)\mathrm{P}(\mathbf{z}=\hat{\mathbf{z}}; \mathcal{M}_0) && (6.13)\\
&\propto \mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}}|\mathbf{z}=\hat{\mathbf{z}}; \mathcal{M}_0)\mathrm{P}(\mathbf{z}=\hat{\mathbf{z}}) && (6.14)\\
&\propto \mathrm{P}(\mathbf{x}_{\hat{\mathbf{z}}}; \mathcal{M}_0)\mathrm{P}(\mathbf{z}=\hat{\mathbf{z}}) && (6.15)\\
&= \prod_n \mathrm{P}(x_{\hat{z}_{n+1}}|\mathbf{x}_{\hat{\mathbf{z}}_{1:n}}; \mathcal{M}_0)\mathrm{P}(z_{n+1}=\hat{z}_{n+1}|\mathbf{z}_{1:n}=\hat{\mathbf{z}}_{1:n}) && (6.16)
\end{aligned}$$

where we used Bayes rule, Eq. 6.6, and the chain rule of probability. This equation allows us to sample z sequentially, and train the RNN as before. $\mathrm{P}(z_{n+1}|\mathbf{z}_{1:n})$ is just the sequential version of the prior $\mathrm{P}(\mathbf{z}_{1:N})$ and captures the fact that $\mathbf{z}$ defines an ordered subset. ($z_n \in \{1, 2, \ldots, T\}$, and $z_n < z_{n+1}$)

In summary, the training method simply alternates sampling from $\mathrm{P}(\mathbf{z}|\mathbf{x}; \mathcal{M}_0)$ (E-Step in Eq. 6.10) and updating $\mathcal{M}$ using Eq. 6.12 (M-Step). This falls neatly into the RNN pipeline, with only a simple sampling step before feeding new samples to the network. Pseudo-code for training S-RNN is presented in Algorithm 4, for a single training sequence. Note that $\mathrm{P}(x{=}x_j|\mathbf{x}_{\hat{\mathbf{z}}_{1:n}}; \mathcal{M})$ is the prediction of the RNN at step n, and Cat is the categorical distribution. The RNN is updated by using the prediction of the model at step n ($\mathbf{y}_n$), $x_{\hat{z}_{n+1}}$, and $\mathcal{X}_n$, with the loss function (described in Eq. 5 above). This provides a gradient which is back-propagated through the network.

Note that the sampling depends on the model prediction. Intuitively, this is the key that allows exploring all subsets of the data, since the model is used to intelligently guide the exploration.

Algorithm 4

```
1:  function TRAIN S-RNN($\{x_1, x_2, \ldots\}$)                          ▷ Images
2:      $\hat{\mathbf{z}}_{1:N}, \mathcal{M} \leftarrow$ random                        ▷ Randomly initialize
3:      $n \leftarrow 1$
4:      while not converged do
5:          $\mathcal{X}_n \leftarrow \{x_{z_n+1}, x_{z_n+2}, \ldots\}$                ▷ Set of future images
6:          $p_j \leftarrow 0$
7:          for $x_j \in \mathcal{X}_n$ do
8:              $p_j \leftarrow \mathrm{P}(x{=}x_j|\mathbf{x}_{\hat{\mathbf{z}}_{1:n}}; \mathcal{M})\mathrm{P}(z_{n+1}{=}j|\mathbf{z}_{1:n}{=}\hat{\mathbf{z}}_{1:n})$
9:          $\hat{z}_{n+1} \sim \mathrm{Cat}(p_1, p_2, \ldots)$         ▷ Sample $\hat{z}_{n+1} = j$ with probability $p_j$
10:         Update RNN ($\mathcal{M}$) using $x_{\hat{z}_{n+1}}, \mathcal{X}_n$                ▷ See text
11:         $n \leftarrow n + 1$
```

6.5.2 Choosing the size of the summary

We fixed the size of the storyline and summaries to be $N = 10$ for all the concepts as a good compromise between content and brevity, i.e. intuitively allows for a compact but informative summary of a concept. In Fig. 6.11 we follow the same setup as in Section 4.4. We plot the performance for S-RNN with three values of $N = 5, 10, 20$ on three tasks: Short-term prediction (Short-term), Long-term prediction as before (Long-term10), and Long-term prediction using a ground truth with storylines of length 5 (Long-term5). We observe that the model trained with $N = 10$ has the highest performance on all three tasks. This implies that S-RNN is not overfitting to only the case of 10 image storylines, since then we would expect the $N = 5$ method to do significantly better for Long-term5. Our interpretation is that for small N, the sequences are short and easy to learn, but not very informative. For large N, the sequence approaches the full album, and more data and model capacity is needed to learn the long-term correlations.

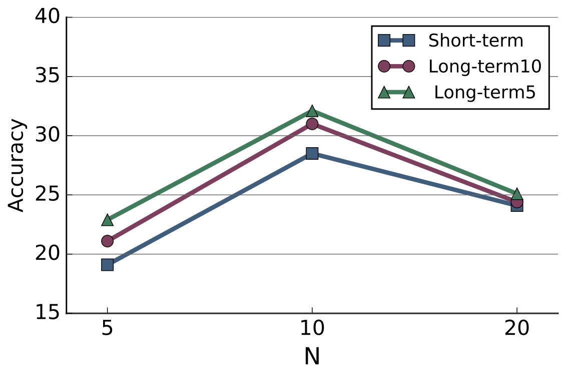


Figure 6.11: The performance for S-RNN with three values of $N = 5, 10, 20$ on three tasks: Short-term prediction (Short-term), Long-term prediction as before (Long-term10), and Long-term prediction using a ground truth of size 5 instead of 10 (Long-term5).

6.5.3 Dataset

In Fig. 6.12 we sample images from two randomly selected albums for each concept. As we can see, some albums are quite relevant to the concept, such as the first *Wedding* album. However for this simple sampling, many albums, such as the second *Christmas* and first *Safari* albums, have little relevance. This is typical of the dataset, since a simple query for *Christmas* returns, for example, Christmas parades, Christmas plays, and community events, in addition to the expected family celebrations. Moreover, it can be seen that many images are not related to the concept. What this means for our tasks, is that retrieving relevant images for a concept is difficult. However, in any given album, if the album is good, then uniformly sampling images does quite well at summarizing that album. In addition, in Fig. 6.13 we show few human-generated summaries for the albums.

6.5.4 Storylines for Different Concepts

In Fig. 6.14, we motivate the problem of selecting images for the storyline by presenting, for each concept, K-means clusters using $fc7$ features. We see that while some images are relevant in each group, some concepts are very different from what we expect, for example it is difficult to recognize *Paris* or *London*. We argue that perhaps more importantly, all the images in the collection do not complement each other, that is, there is no sequence of events, or relationship between the images.

However, in Fig. 6.15 we present the images retrieved by our method (**S-RNN**). We can see that each collection (each row), captures a coherent story of the concept. If we compare that with our method without skipping (**S-RNN-**, Fig. 6.16), we see that the coherence in those collections is focused on short-term correlations, and those baselines choose highly correlated images.

6.5.5 Example Summaries

We present many randomly selected summaries from the dataset. Photo album summarization depends heavily on the quality of the given album, and the album's relevance. Summarization is therefore a more difficult task than selecting relevant images for evaluation, since on some albums, simple baselines do remarkably well. In Fig. 6.17 we present summaries by our method (**S-RNN**) on the same randomly selected albums as in Fig. 6.12.

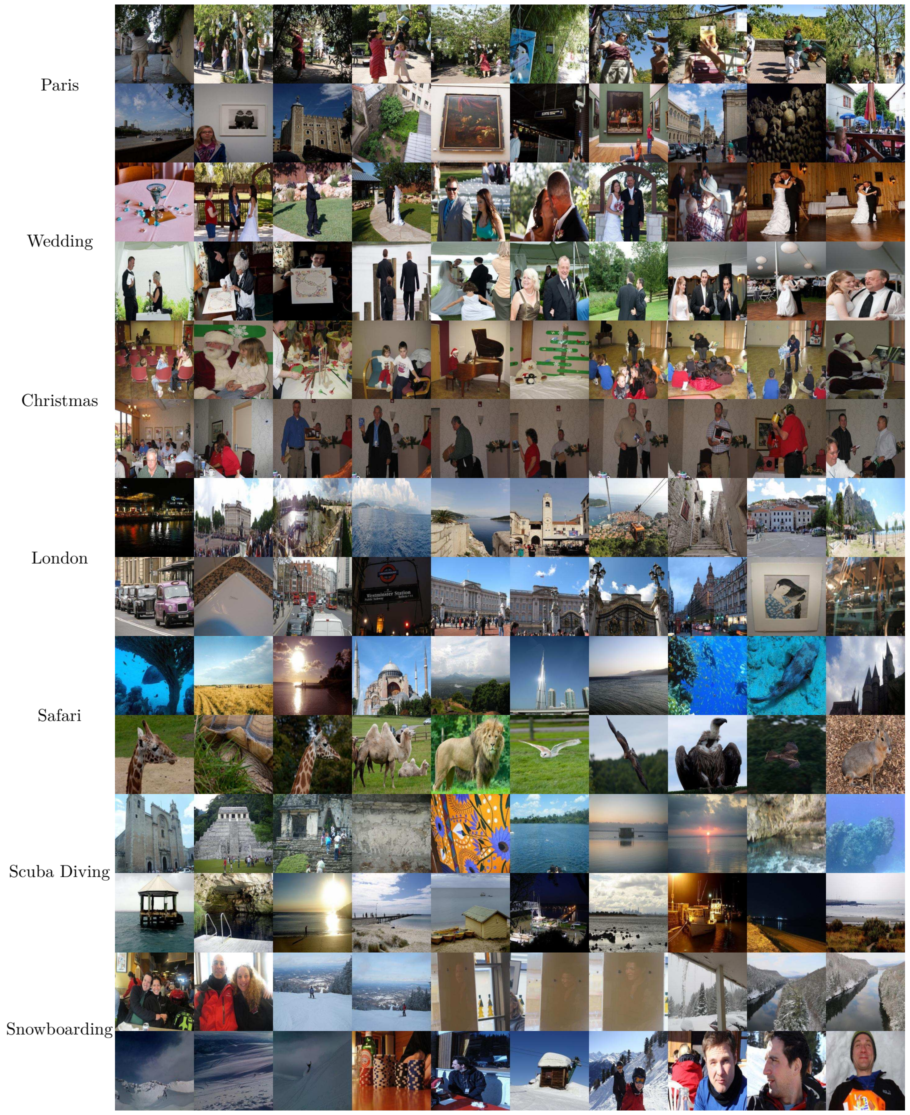


Figure 6.12: Each row consists of a randomly selected album from the dataset, two for each topic. Each album is visualized using 10 images uniformly sampled from the album.

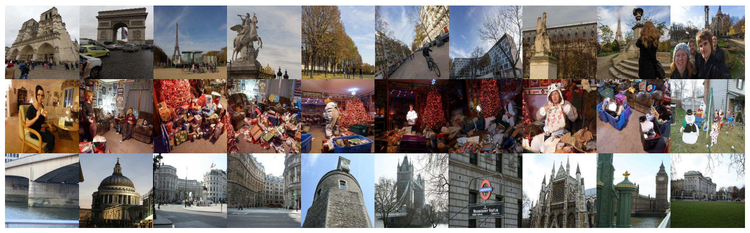


Figure 6.13: Example human-generated summaries for *Paris*, *Christmas*, and *London* albums.

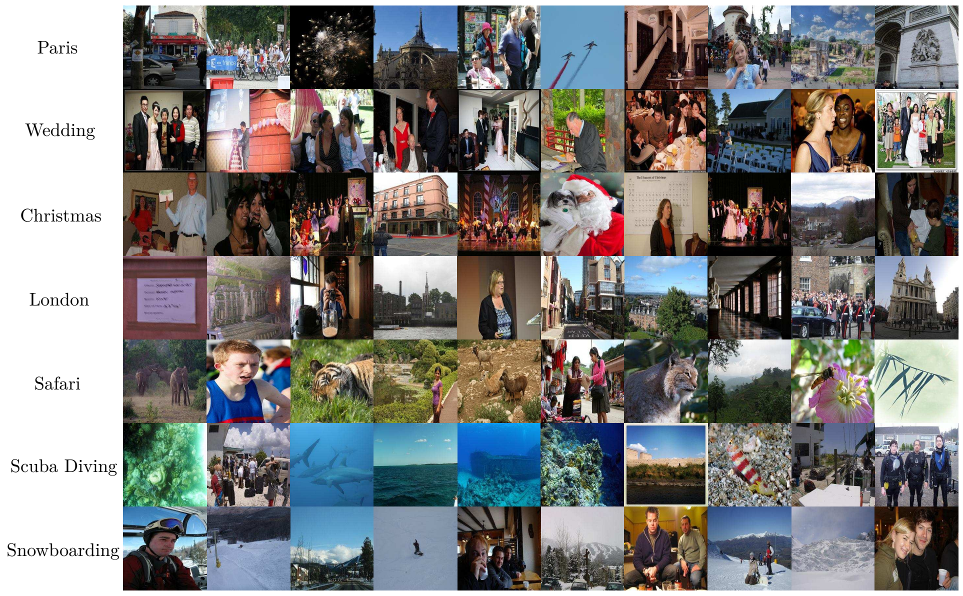


Figure 6.14: (**K-Means**). Storylines by K-Means for different concepts. Each collection contains some relevant images, but many of the images are not relevant, since K-Means emphasizes diversity. Furthermore, each collection is not very coherent.

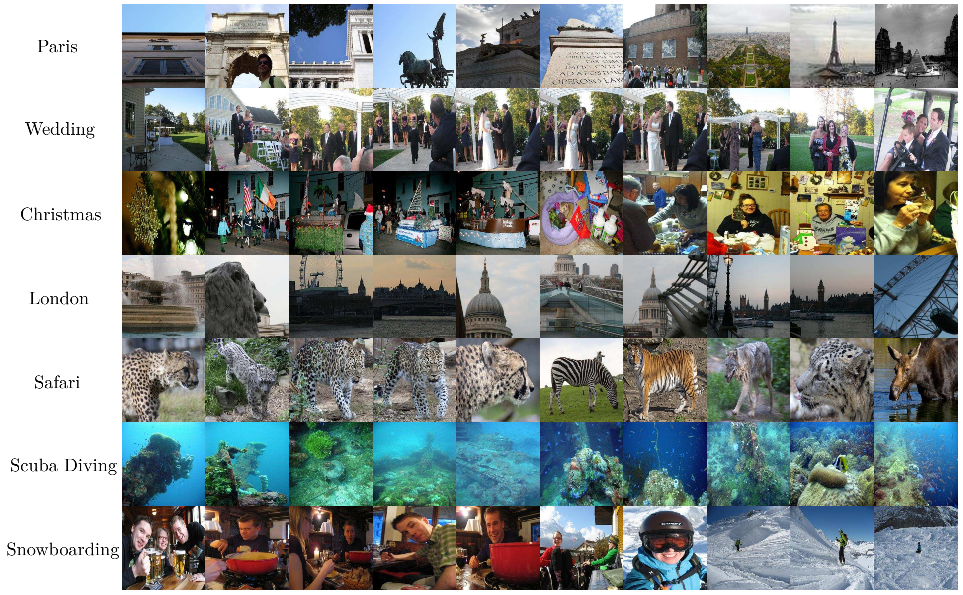


Figure 6.15: (**S-RNN**) Storylines by our method for different concepts. Each collection of images is more related to the concept, and each collection captures a coherent story of the concept.

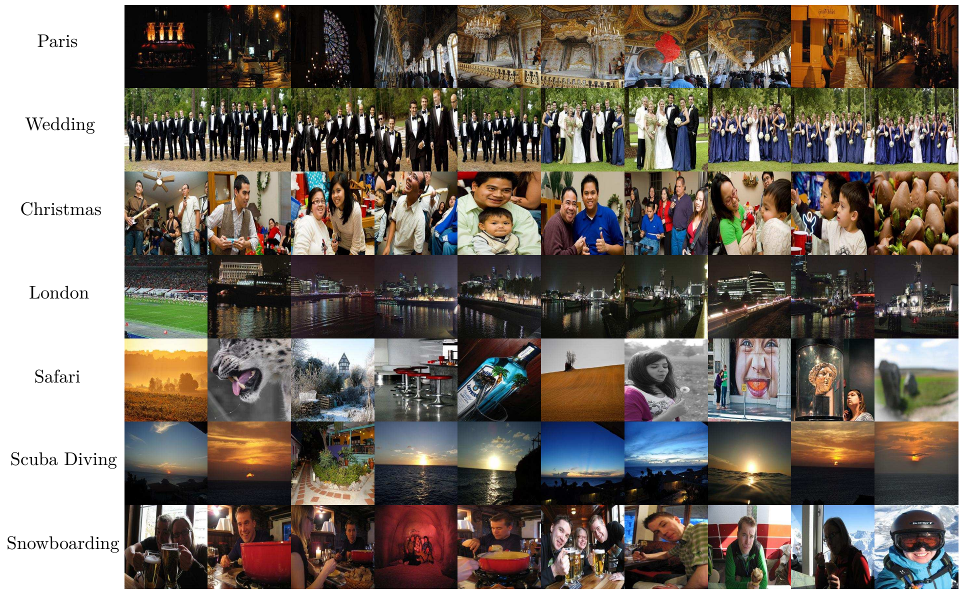


Figure 6.16: (**S-RNN-**). Storylines by our method without skipping for different concepts. While these collections are relevant to the topic, and the collections are coherent, it appears that the model is focusing on short-term correlations, and the images are not very diverse.

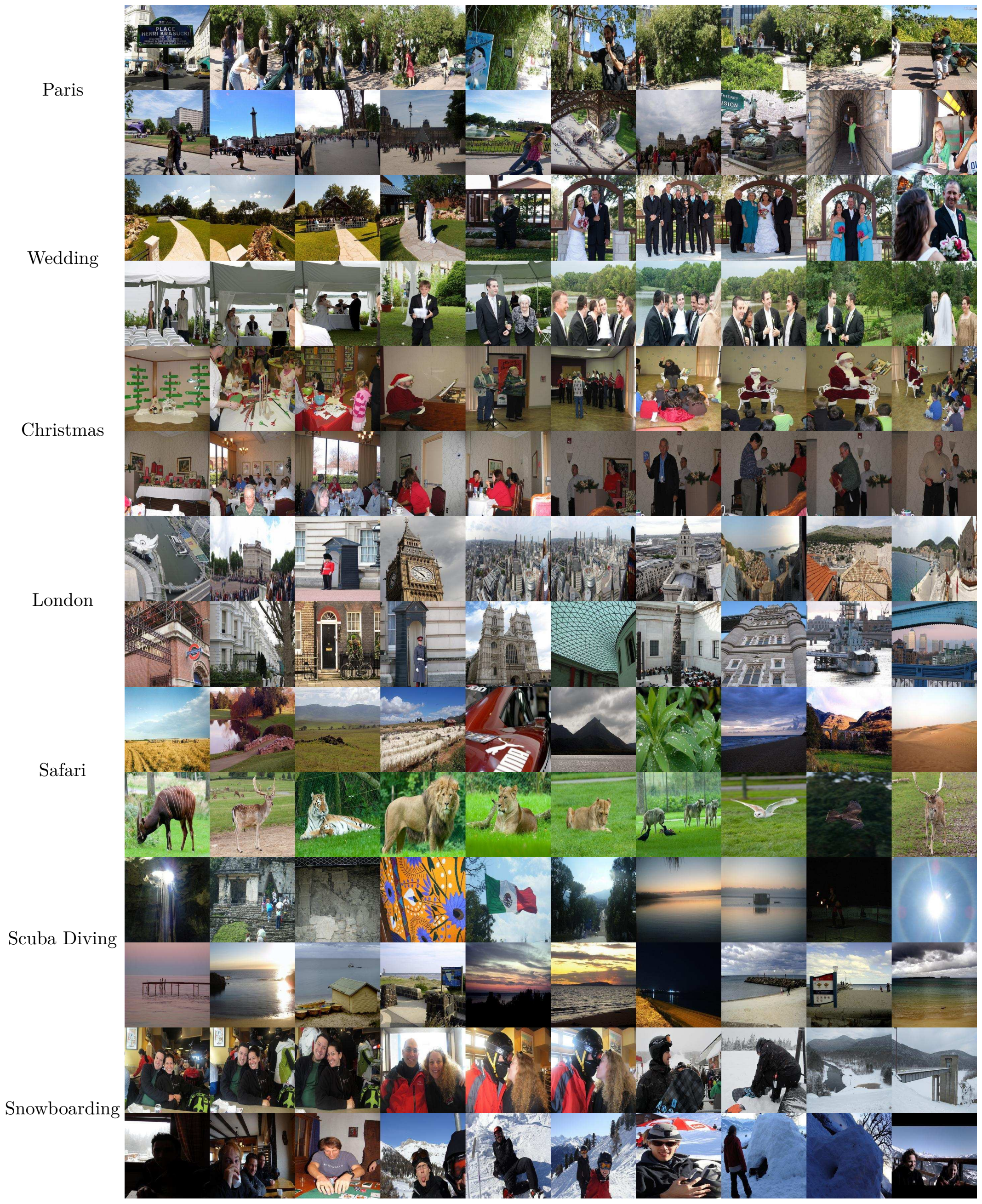


Figure 6.17: Randomly selected summaries using our method. These albums correspond to the albums presented in Fig. 6.12. It is difficult to judge the quality of the summaries without seeing the hundreds of images behind each summary, but hopefully provide insight.

Chapter 7

Neural Networks in World Coordinates

Our final algorithmic contribution explores a different axis than time—space. Eye movement and strategic placement of the visual field onto the retina, gives animals increased resolution of the scene and suppresses distracting information. This fundamental system has been missing from video understanding with deep networks, typically limited to 224 by 224 pixel content locked to the camera frame. We propose a simple idea, WorldFeatures, where each feature at every layer has a spatial transformation, and the feature map is only transformed as needed. We show that a network built with these WorldFeatures, can be used to model eye movements, such as saccades, fixation, and smooth pursuit, even in a batch setting on pre-recorded video. That is, the network can for example use all 224 by 224 pixels to look at a small detail one moment, and the whole scene the next. We show that typical building blocks, such as convolutions and pooling, can be adapted to support WorldFeatures using available tools. Experiments are presented on the Charades, Olympic Sports, and Caltech-UCSD Birds-200-2011 datasets, exploring action recognition, fine-grained recognition, and video stabilization.

The success of recent vision systems is in a way surprising, since the input is typically a 224×224 pixel image, or in the case of videos, a temporal stack of such images [16,55]. This is both low resolution, and does not give much flexibility to investigate important signals in the image. The system is constrained to what the camera recorded, and has typically no active role in collecting the data. In particular, video recognition architectures have been shown to be vulnerable to camera motion, subject size, and temporal scale [222]. In comparison, the human visual system is not a passive receiver—our eyes are constantly moving to increase the effective resolution of the scene and suppress irrelevant signals [59].

These eye movements exist in humans and animals with foveal vision, and can be categorized as: *Stabilization*, the vestibulo-ocular reflex describes a control system the human visual system uses to stabilize images on the retina given proprioceptive information, like head rotation [29]. *Smooth Pursuit*, where the gaze is voluntarily shifted to track a moving object, effectively stabilizing the object of interest on the retina [119]. *Fixation*, where gaze is fixed towards a single location, to enhance resolution of that area [193]. *Saccades*, where the eyes quickly jerk between phases of fixation to explore the scene [98]. In Fig. 9.1 we illustrate

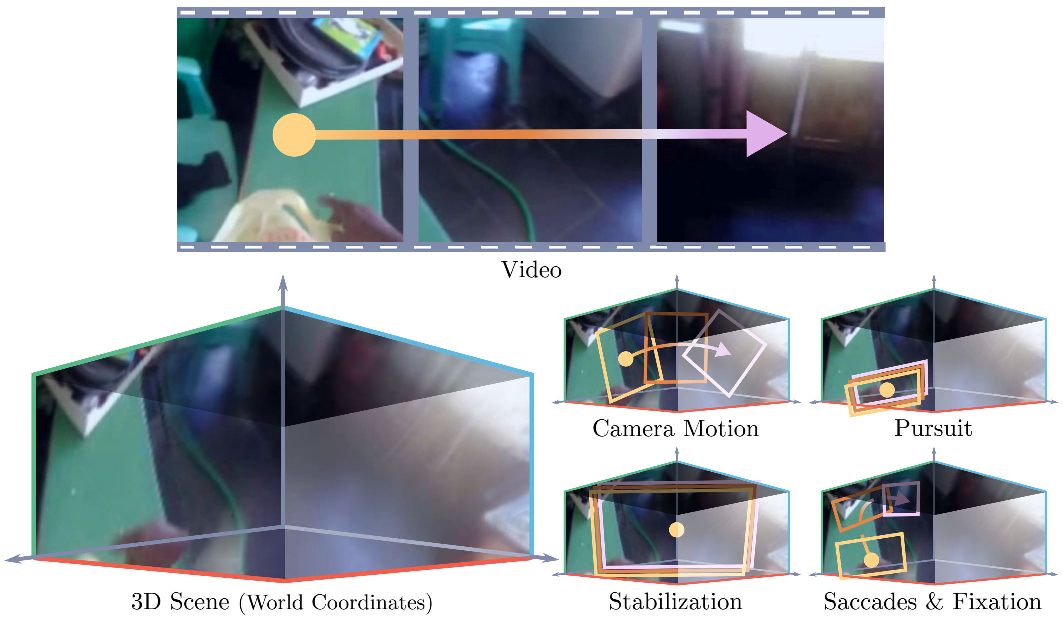


Figure 7.1: We explore how to model the variety of eye motions, even on pre-recorded data. We generalize the concept of feature maps, to WorldFeatures, that include a relative location in space and time, which can encode any type of "eye motion" we want apply to the recorded data, or undo. Orange to pink is used to denote features at time $t-1$, t, and $t+1$. We show a video from a first-person view, along with the motion of the camera in the reconstructed world frame (*Camera Motion*). The original video follows a particular sequence of viewpoints, however, crucial information about the activities in the video requires different views—*Stabilization* transforms the views to the world frame, *Pursuit* transforms the views to follow the subject of the video, and *Saccades & Fixation* scans the scene to extract fine-grained information from the video.

how a video is constrained to what the camera recorded, but that also the video may be refocused. How can we build a vision system that has this flexibility to freely explore the data, and move beyond a fixed 224×224 window?

We propose a simple and effective idea—each feature, has a location in real-world coordinates. This combination of `(feature, transformation)` pairs is referred to as a WorldFeature. An example transformation might be image coordinates to real-world coordinates (camera matrix). It turns out preprocessing the video data to, for example, stabilize it, introduces new problems, as demonstrated in Fig. 7.2. Instead, we use these WorldFeatures in all layers in the network and index the data according to the transformation or transform it as needed. That is, *implicit* transformation instead of *explicit*. With such network, we can utilize any type of "eye motion" we want apply to the recorded data, or undo.

To illustrate the idea, in Fig. 7.2 we present a simple two frame featuremap with three different transformations $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ to showcase how different transformations can highlight different signals in the data. For example, in the third it is easy to understand the global layout of the scene, and in the second it is easier to understand the difference in the pose of the person. In Sec. 7.2 we show more details on WorldFeatures, and how to keep track of the

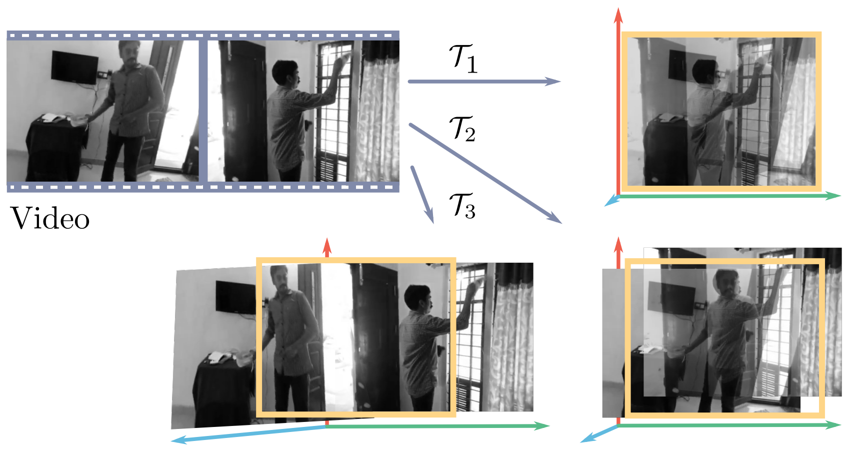


Figure 7.2: Illustration of WorldFeatures, attaching transformations to features. Top left is the original video shown as two frame featuremap. We show the temporal average of an aligned feature map, with three different transformations $\mathcal{T}_1$ (identity), $\mathcal{T}_2$ (pursuit), $\mathcal{T}_3$ (stabilization). Observe how different transforms highlight different aspects of the data. We show a potential coordinate frame, that could be used to fit the data into a network, highlighting a problem with naively stabilizing the data, for example, when 64 frames in a row need to be aligned. WorldFeatures get around this by keeping the transformation and only using it as needed.

transformations after every layer to build networks with these operations.

Contributions. We propose the idea of WorldFeatures, where each feature has a transformation, on multiple vision tasks: Fine-grained Image Recognition (Sec. 7.3.1), Video Stabilization (Sec. 7.3.2), and Video Activity Recognition (Sec. 7.3.3). We formalize WorldFeatures, and show how different eye movements can be modeled in this framework. Further, we propose a simple implementation that can adapt optimized neural networks tools to operate on WorldFeatures. This implementation can extend many state-of-the-art building blocks that operate over space and time, such as 3D Convolutions, and MaxPooling. The output of each building block, is also WorldFeatures, such that it can be processed hierarchically (e.g. 3D-ResNet [270]).

7.1 Background

Many image and video understanding architectures have been proposed in the last few decades and we refer the reader to [177, 274] for a detailed survey, and focus here on recent video architectures related to our approach.

Video Architectures Recently, networks have become more evolved for separating content and motion [16, 55, 195, 227]. This direction is analogous to the different specialization of the ventral and dorsal streams in the mammalian brain. Similarly, separating camera motion and subject motion has shown great promise in video segmentation [253]. As the field moves towards utilizing longer temporal context [278], equipping these networks with the capability to separate camera motion, object permanence, scene dynamics, and subject motion can greatly improve those efforts. We aim to provide the building blocks necessary to construct these systems.

Video Stabilization Related to our work on understanding eye motion, is the task of video stabilization that has been pursued directly [71,141,267], or with additional supervision [140]. Of particular relevance is recent work that has incorporated stabilization and tracking of points over time to improve accuracy of video recognition [263, 266]. Inspired by these successes, we go a step further, and build a framework that can operate given an arbitrary transformation signal, and use this to improve the vision system.

Improving Visual Input On the other hand, recent work has also explored how to use the input data more effectively. This has been done through spatially or temporally modifiable connections [30], non-local connections where feature similarity guides the connections [270], or transformations of the input [96]. Furthermore, [181] investigated the idea that not all input features are equally useful, and explored how to learn how to highlight and "zoom in" on the important content in each image. As we will see, eye movement is primarily useful to allow for efficient allocation of resources, and since our system can handle arbitrary transformations, we can explicitly utilize various "zoom in" of the data. Furthermore, our WorldFeatures is a system that allows a network to generalize beyond the camera, and is complementary to frameworks that learn attention to emphasize parts of the data.

7.2 WorldFeatures: Feature Maps with Camera Movements

We start by introducing the ideas behind WorldFeatures, that is, `(feature, transformation)` pairs. Then we demonstrate how different eye movements can be modeled in this framework. A video is a specific window into the world at the time it was recorded, and a system analyzing this video might have a completely different objective for watching the scene than the camera operator. To undo the camera bias or highlight signals in the video, we allow a transformations that are analogous to eye movements, after the data has already been recorded.

To undo camera transformations, could we pre-process the data? For example, applying affine transformations to each frame to undo camera movement, for stabilization. In fact, even this simple case has problems: How to fit this transformed data into the model? At what scale should we process the data? How to use imperfect transforms? Consider eyes scanning a scene, fixating on a variety of points in the scene at multiple resolutions. Creating data that follows a similar pattern without an understanding how they relate to each other creates a noisy and disjointed view of the scene. Instead, by keeping track of the relative transformations between features, we can crop, scale, and zoom in on the data as required, while still maintaining correct relationships between features.

7.2.1 WorldFeatures Definition

The key to our idea is the concept of WorldFeatures $\mathtt{x}^{\mathcal{T}}$. That is, a feature map $\mathtt{x}$ of size $T \times H \times W$ (time, height, width) and an arbitrary transform $\mathcal{T}$ which can encode any type of

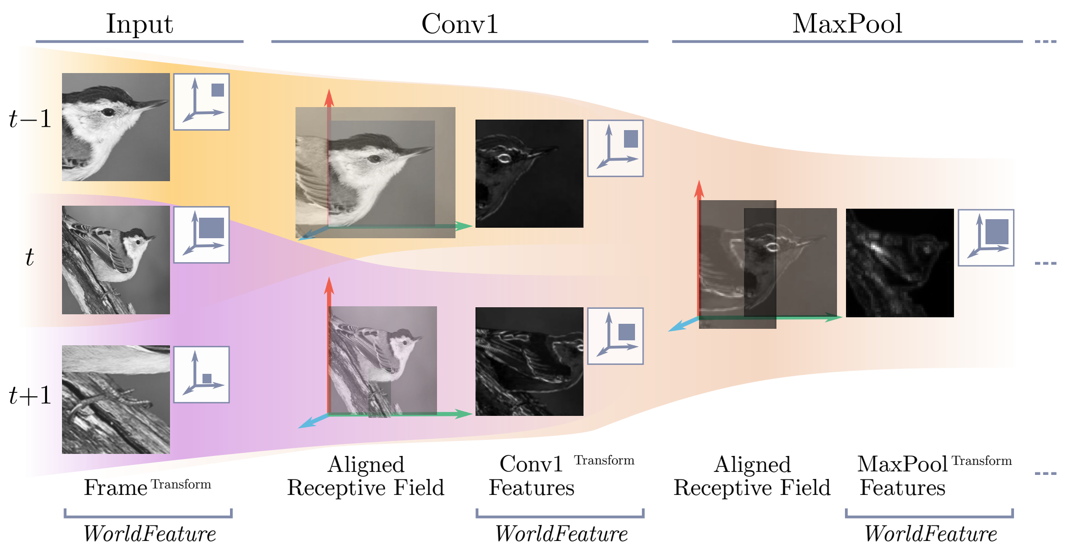


Figure 7.3: Demonstration of a model using WorldFeatures at each layer. Here we look at an illustration of the outputs of conv1 and maxpool layers, when applied to an "eye movement" task. The input and output of each layer are WorldFeatures. The receptive field of each output featuremap is aligned, explicitly or implicitly depending on implementation, before computing the output. Since each output feature map has a different alignment, each output has a different transformation, and we repeat the process at every layer. This shows that WorldFeatures allow the network to combine information across multiple locations and scales in the image.

"eye motion" we want to apply to the recorded data—or undo.[*] We refer to a single $H\times W$ timepoint as a frame, and note that $\mathbf{x}=\mathbf{x}^{\mathcal{I}}$, where $\mathcal{I}$ is the identity transform.

The feature of $\mathbf{x}^{\mathcal{T}}$ at (t,h,w) is written $\mathbf{x}\,(\mathcal{T}(t,h,w))$. Therefore, extending a convolution operation to work on $\mathbf{x}^{\mathcal{T}}$ looks as follows:

$$\mathbf{y}^{\mathcal{T}'}(t,h,w) = \sum_i \sum_j \sum_k \mathbf{x}(\mathcal{T}(i,j,k)) \cdot \mathbf{f}(t-i, h-j, w-k)\,, \tag{7.1}$$

where $\mathbf{y}^{\mathcal{T}'}$ is also a WorldFeature, that can then be passed to the next convolution operation and so on. Here $\mathbf{f}$ is a filter, and we omit batch and channel dimensions for clarity. Compared to regular convolutions, the transformation $\mathcal{T}$ is necessary to transform the features as needed. For example, the valid values of each feature map are only defined for $t\in[0,T)$, $h\in[0,H)$, and $w\in[0,W)$, and transforming the video before processing would move content out of the frame, etc. This convolution layer could use a Spatial Transformer Network [96] independently transforming the receptive field for each output frame, at every layer, exponentially increasing computation. This implementation of WorldFeatures is illustrated in Fig. 7.3. Fortunately, in Sec. 7.3 we outline an efficient implementation of this for CNN building blocks, which allows us to explore this type of networks.

[*]For example, the affine transform, $\mathcal{T}(t,h,w) = (t,\ a_t^0h+a_t^1w+a_t^2,\ a_t^3h+a_t^4w+a_t^5)$, where a_t^n are the parameters at frame t.

7.2.2 Transformation Types

We now outline how this framework can be used to model eye movements: Stabilization, Smooth Pursuit, Fixation and Saccades. Fig. 9.1 contains illustrative examples for each, and we refer to this figure in the paragraphs below to explain each of these transformations.

Stabilization The first and most intuitive transformation $\mathcal{T}$ we encounter is the stabilizing transform. We obtain the camera matrix for every frame, and invert it to obtain the stabilizing transform. That is, the transformed feature map $\mathbf{x}\left(\mathcal{T}(t,h,w)\right)$, will contain the same point in the world at location (h,w) for any t. The *Stabilization* example in Fig. 9.1 visualizes how the frames of the given video are arranged in the world frame.

Smooth Pursuit Since the transform $\mathcal{T}$ is arbitrary we can also choose it to stabilize with respect to a different point of reference. That is, instead of (h,w) for any t pointing to the same point in the world, we can make it point to the same point on a moving object of interest, such as a person. The *Pursuit* example in Fig. 9.1 visualizes how the frames of the given video can be arranged, and combined with attention to the hands of the person.

Fixation To model fixation, where the high-resolution part of the retina (the fovea) is used, we use a transform that enhances part of the frame. That is, if $\mathbf{x}^{\mathcal{T}}(t,h,w)$ is the original WorldFeature, then we add a "fixation transform" $\mathcal{F}$ to get the WorldFeature $\mathbf{f}^{\mathcal{T}\circ\mathcal{F}}$ where $\mathbf{f} = \mathbf{x}\left(\mathcal{F}(t,h,w)\right)$. This fixation transform "zooms in" on a part of the input, increasing the resolution.

Saccades / Visual Search To model saccades, we proceed similarly to fixation, and define a transform $\mathcal{S}$ that gazes at a particular part of the frame, allowing the model to attend to different parts of the image. The similarity with the implementation of fixation is because saccades are in fact defined as the jump between fixations. In the *Saccades & Fixation* example in Fig. 9.1 we demonstrate how the transformations can be chosen to pay close attention to important aspects of the scene, such as the hands, objects and context.

In conclusion, we now have a series of eye movements that are possible within our framework, allowing a variety of augmentations without sacrificing relative location between features. In the following experiments we use these definitions to provide the model with a transformation for each video, but a variety of transformations that are learned from data could be used as well.

7.3 Experiments

In this section we explore WorldFeatures for diverse applications. We use a 3DResNet-style architecture for videos built from WorldFeatures. This architecture can be used to incorporate arbitrary eye movements in videos and images, where images are just treated as an uneventful video (replicating images in time). The network can then utilize the transformations to "zoom in" on various parts of the image. First, we demonstrate that a 3D network, can outperform 2D networks on fine-grained recognition of nature categories (image task). Second, we look at video stabilization, and show how implicit transformations can bypass the problems with explicitly stabilizing the input data with preprocessing. Third, we utilize enhancing transformations, smooth pursuit and fixation, to improve video recognition accuracy on the Charades dataset [223].

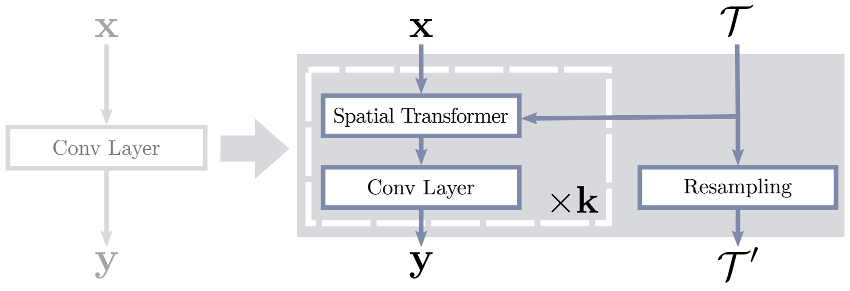


Figure 7.4: Our simple implementation of WorldFeatures "wraps" a layer (e.g. conv layer) and applies a spatial transformer to the data to align the receptive field of each output, k times for a layer with temporal extent of k (e.g. 3 in ResNet), and combines the outputs.† More details are provided in the Appendix.

Implementation Our implementation extends a 3DResNet50 model, following [270] (similar to I3D [16]). All convolution, max pool, and average pool layers are converted to WorldFeature layers, which implicitly operate on a transformed video. All models start with parameters from a regular 3DResNet50 pretrained on the Kinetics dataset [16]. In Fig. 7.4 we illustrate a simple method to convert a layer, such as a conv layer, to use WorldFeatures, without having to engineer special low-level GPU programs for each. More details are provided in the Appendix. We use the grid sampler from Spatial Transformer Networks [96], with arbitrary transformation grids. After fitting the transformer, we use the grid sampler in the nearest (pixel) setting. This avoids blurring of the feature maps as they are repeatedly transformed. To address missing values after transforming, we add a channel denoting missing data at a point, similar to [56]. For comparison, all methods use the same batch-size of 2 unless otherwise noted, and follow the same learning rate schedule tuned for the baseline. The models are implemented in PyTorch, and will be made publicly available.

7.3.1 Saccades - Fine-Grained Recognition

Motivated by human gaze studies in image recognition, we explore using 3D video networks for fine-grained recognition on the Caltech-UCSD Birds-200-2011 dataset [261]. To provide a gaze trajectory for the network, we use saliency [90] and objectness [24] to generate 64 bounding boxes. We replicate each 2D image 64 times in time to form a 3D input. The 64 bounding boxes form the basis for the fixation in each frame, and are ordered to maximize the overlap between consecutive boxes, allowing the network to learn filters that combine information between various fixations and scales. See Fig. 7.5 for an example. We compare with regular *3DResNet50*, and *ResNet50*. We start with a 3DResNet50 and fine-tune with the saccades/fixation setup, and combine with ResNet50.

Results The results are presented in Table 7.1. With a input size of 224 pixels, our method outperforms architectures specialized for fine-grained recognition, even with recent methods such as [181] that learn the function of what to zoom to, whereas we use off-the-shelf saliency.

†For k=3 the first transformer aligns frames {0,1,2} (the receptive field for output frame 1), and frames {3,4,5} (receptive field for output frame 4), etc. The second transformer aligns {1,2,3} (receptive field for output frame 2), etc.

	Top-1 (%)	Resolution (px.)
3DResNet50	79.8	224
DT-RAM [137]	82.8	227
ResNet50	83.4	224
Learning to Zoom [181]	84.5	227
3DResNet50 w/Saccades	**85.3**	224
DT-RAM [137]	86.0	448

Table 7.1: Fine-grained Recognition on the Caltech-UCSD Birds-200-2011 dataset.

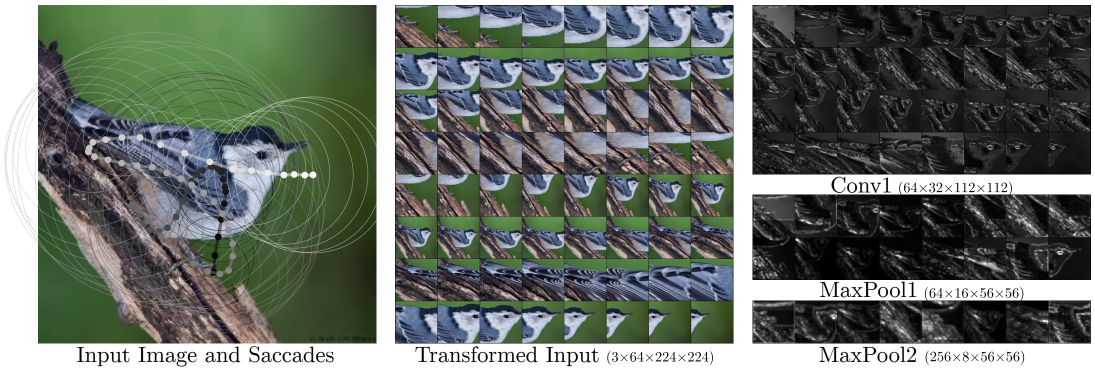


Figure 7.5: In the left column we visualize the path (the saccades) that the model chooses, given static saliency and objectness, where dark to light denotes frames 1 through 64, and the large circles denote the scale of the fixaton in that frame. In the the middle column, we show the pixel data that the model uses (the model also receives the transformations between those). Finally in the right column, we show featuremaps from Conv1, MaxPool1, and MaxPool2, where high values denote high variance over time in the features at that pixel.

Combining our framework and learning saliency is likely to yield additional gains. The method from [137] only outperforms other methods when provided with 448×448 image requiring a special network beyond the scope of this work. These results demonstrate an exciting new avenue for video architectures applied to the conventionally 2D task of fine-grained recognition. In Fig. 7.5 we visualize the points of interests and scales suggested by the saliency algorithm, and how that is translated into pixel data that the network sees (along with the transformations between those). We also show example of featuremaps from 3 layers in the network, where we can see how the network combines information across the different saccades, and builds detailed fine-grained understanding of the image.

7.3.2 Stabilization - Video Activity Recognition

Stabilization is a fundamental vision task, and has been explored in various contexts. Here we specifically explore stabilization for improving video recognition.

To evaluate the stabilization performance, we start with videos with synthetic camera motion. We use 50% of the Charades [223] videos with the least motion[‡], and add synthetic camera movement by linearly moving between 3 locations at randomly chosen scale levels (30%–300%, *Synthetic*). Next we consider Olympic-Sports [162], that was used to showcase the advantage of stabilization in the seminal IDT work [263]. Charades is evaluated with a video-level mean average precision over 157 classes, and Olympic-Sports uses a top1 accuracy for each video over 20 classes. Our baselines are *3DResNet50* [270], and 3DResNet50 with all frames are stabilized to the center frame (*3DResNet50*++). All models use a 16 temporal frame stack as input to make stabilization feasible for the baseline, and are trained with the same learning rate schedule and hyperparameters tuned to the baseline. For each video we construct the camera motion from the video using direct optimization of affine transformations parameters between consecutive frames. The details are provided in the Appendix.

Results In Table 7.2 we see that using the stabilization transformation allows our method to improve over the baselines. That is, modelling features and transformations together instead of preprocessing allows our method to avoid compounding errors do to stabilization over many frames, and loss of resolution.

	3DResNet50	3DResNet50++	Pursuit
Synthetic (Video mAP)	13.1	13.5	**14.1**
Olympic-Sports (% Top-1)	96.4	97.2	**97.2**

Table 7.2: Results on stabilization on two video activity recognition datasets. Stabilizing the video before inputting it to the network (*3DResNet50*++) helps in some cases, but has drawbacks, whereas *implicit* stabilization can better utilize the input.

Analysis We also looked at an egocentric video dataset, Charades-Ego. With 64 frame inputs, preprocessing (*3DResNet50*++) actually reduces performance in the baseline ($23.6 \rightarrow 22.8$ mAP), and with 16 frame inputs (only 0.67 sec video clip), preprocessing only slightly improves performance ($17.5 \rightarrow 18.5$ mAP). This demonstrates the challenge of stabilizing a video before passing it to the network, and interestingly, stabilization does not seem to help activity recognition much. Perhaps, since what the camera operator is looking at is highly informative in first-person video. In any case, WorldFeatures have performance at par with the better method in each case (23.4 and 18.4 mAP).

In Fig. 7.6 we demonstrate how camera motion makes learning from video challenging. We show point in time from 4 feature maps. We observe that WorldFeatures can utilize a transformation to suppress irrelevant signals. Although a deep network might, with sufficient data, be able to learn all possible variations, this adds complexity the model. We will see in the next section how we can go even further and use the transformations to focus on particular parts of the data.

7.3.3 Smooth Pursuit & Fixation - Video Activity Recognition

To demonstrate how WorldFeatures can utilize any transformations, we explored *Smooth Pursuit*, where we use a transformation to stabilize the content with respect to a human

[‡]We used variance of optical flow, yielding 4396 videos of average 30 sec.

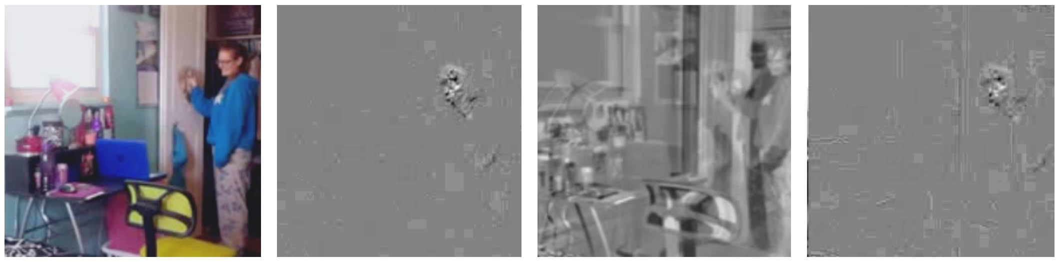


Figure 7.6: A single frame from 4 feature maps. First feature map is original video, second is the output of a temporal difference filter applied to an ideal (stable) video, third is output of the filter applied to the actual unstable video, and finally the output of our method given the estimated stabilizing transform.

bounding box (*Pursuit*). We use an RCNN person detector [209], and construct a the transform such that the person is always centered and fills the input to the network. Going further, we explored how to utilize the transformations to increase the effective resolution of particular parts of the scene or the subject in a video. Concretely, we used simple video saliency (temporal difference), and fit a bounding box around 80% of the variance in the saliency, and the fixation transforms the input such that the box fills the input (*Fixation*).

Our experiments are on the Charades dataset [223], evaluated with a video-level mean average precision over 157 classes. We start with a standard pre-trained baseline network from the literature (*3DResNet50*), fine-tune with the new framework, and combine with the old network.

Results Our results are presented in Table 7.3. Interestingly, adding smooth pursuit on its own does not significantly improve over the baseline, which stabilizes the video with respect to the person. However, this is expected since many of the videos in Charades have camera motion that already follows the video subject. Fortunately, we do see that using smooth pursuit followed by fixation (*FixationPursuit*) we can improve performance. This suggests that whereas stabilization on its own is not particularly helpful to current neural networks, when it is used to locate an area to increase the resolution it can offer significant benefits, since 224×224 pixels is not provide much detail, and learn filters that combine information across multiple locations and resolutions.

	3DResNet50	Pursuit	Fixation	FixationPursuit
Charades [223] (Video mAP)	31.3	31.5	32.3	**32.6**

Table 7.3: Activity Recognition on the Charades dataset using smooth pursuit and fixation transformations.

In Fig. 7.7 we visualize frames from a Conv1 feature map applied to a video on a person sitting at their desk, since each feature has a transformation associated with it, even though some of the feature maps are at very small scales, the network can move them into a common coordinate frame as needed.

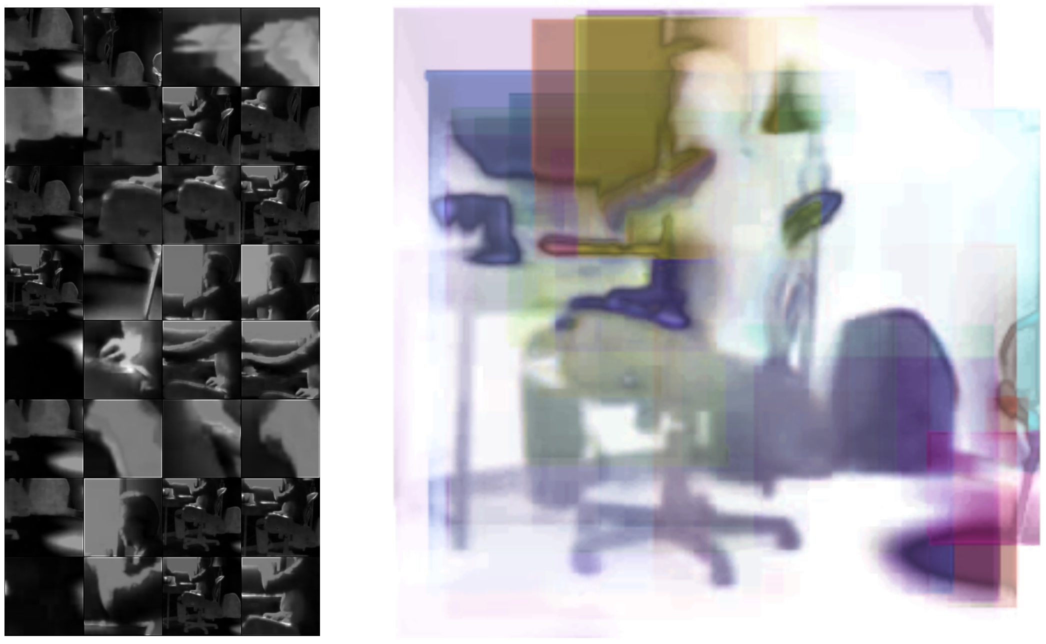


Figure 7.7: Composite of 32 Conv1 feature maps from the fixation model. Different color indicates different frames.

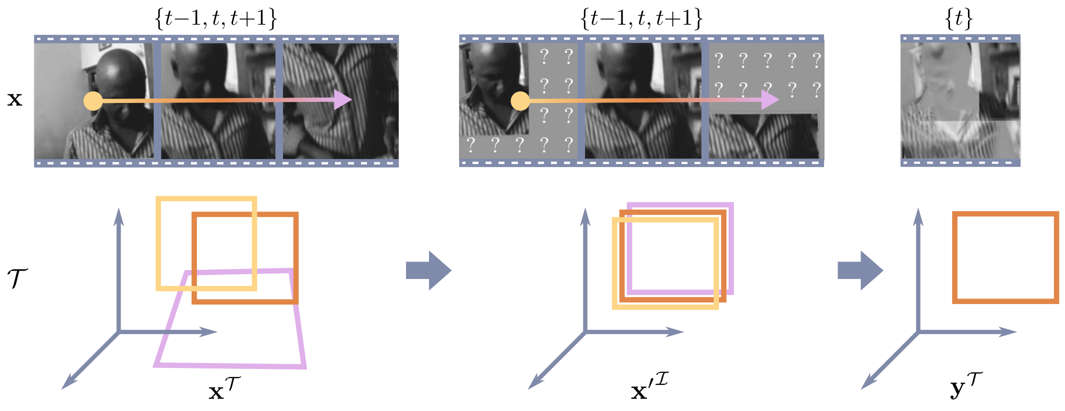

$\{t{-}1, t, t{+}1\}$
$\{t{-}1, t, t{+}1\}$
$\{t\}$
$\mathbf{x}$
$\mathcal{T}$
$\mathbf{x}^{\mathcal{T}}$
$\mathbf{x}'^{\mathcal{I}}$
$\mathbf{y}^{\mathcal{T}}$

Figure 7.8: Illustration of our implementation of WorldFeatures. We start with a world feature map $\mathbf{x}^{\mathcal{T}}$ and and in order to calculate the output of a convolution applied to the feature map at time t, that is $\mathbf{y}^{\mathcal{T}}(t, h, w)$, we first transform the needed features into the coordinate system at time t, and then apply the convolution as normal. Black, gray and white, are used to denote $t{-}1$, t, and $t{+}1$, respectively.

7.4 Appendix

Implementation Details First, we note that layers using WorldFeatures could be implemented with similar efficiency to their regular counterparts, which would require a dedicated engineering effort. To quickly iterate the idea, we implemented WorldFeatures in high-level PyTorch code. We observe that current neural network architectures have filter extent of 3 frames [16,80,270]. Thus, we simply transform each chunk of consecutive frames into a common coordinates as needed for each filter computation. Concretely, mirroring Eq. (7.1) above, suppose we only need to calculate $\mathbf{y}^{\mathcal{T}}(t, h, w)$ for some particular frame: $t{=}t_0$, $h \in \{0,1,\ldots,H{-}1\}$, and $w \in \{0,1,\ldots,W{-}1\}$. Then we can rewrite as follows:

$$\mathbf{x}'_t(t',h',w') = \mathbf{x}(\mathcal{T}(t',h',w')) \tag{7.2}$$

$$\mathbf{y}^{\mathcal{T}}(t, h, w) = \sum_i \sum_j \sum_k \mathbf{x}'_t(i,j,k) \cdot \mathbf{f}(t{-}i, h{-}j, w{-}k) \tag{7.3}$$

where the last equation describes regular convolution on $\mathbf{x}'$. Here $\mathbf{x}'$ is an explicitly transformed version of $\mathbf{x}$. Since $\mathbf{f}$ only has non-zero values at $t = \{-1, 0, +1\}$ (if filter extent is 3) $\mathbf{x}'$ only needs to be computed for $t{-}1$, t, and $t{+}1$. Thus if we need to compute this for all values T of t, we need to create T such versions of $\mathbf{x}'$, which corresponds to 3 versions of the size of the input $(T{\times}H{\times}W)$, and then we apply the regular convolution to these 3 copies and merge the results to create $\mathbf{y}^{\mathcal{T}}(t, h, w)$. This process is illustrated in Fig. 7.8, where 3 frames are temporarily transformed into a common coordinate frame (the coordinates of frame t), and that used to compute the output of the convolution.

This has numerous advantages: 1) Since 3 or less frames are being transformed at a time, both cumulative errors in transformation and missing value problems are minimal. 2) Transforming across long timescales is only done in higher layers, where spatial resolution is lower, meaning less accuracy is needed. 3) Any layer that operates on the time dimension, such as maxpool, average pool, and 3d convolution, can be adapted in the same way using

layer		output size	transformer output size
conv$_1$	7×7, 64, stride 2, 2, 2	32×112×112	32
pool$_1$	3×3×3 max, stride 2, 2, 2	16×56×56	16
res$_2$	$\begin{bmatrix}1\times1, 64\\ 3\times3, 64\\ 1\times1, 256\end{bmatrix}\times3$	16×56×56	16
pool$_2$	3×1×1 max, stride 2, 1, 1	8×56×56	8
res$_3$	$\begin{bmatrix}1\times1, 128\\ 3\times3, 128\\ 1\times1, 512\end{bmatrix}\times4$	8×28×28	8
res$_4$	$\begin{bmatrix}1\times1, 256\\ 3\times3, 256\\ 1\times1, 1024\end{bmatrix}\times6$	8×14×14	8
res$_5$	$\begin{bmatrix}1\times1, 512\\ 3\times3, 512\\ 1\times1, 2048\end{bmatrix}\times3$	8×7×7	8
global average pool, fc		1×1×1	1

Table 7.4: Our 3DResNet50 model for video. The dimensions of 3D output maps and filter kernels are in T×H×W, with the number of channels following. The input is 64×224×224, and the pre-computed transformer corresponding to the input is of size 64. Residual blocks are shown in brackets. The table is adapted from [270].

this method.

In Table 7.4 we illustrate 3DResNet50 extended with WorldFeatures, and how the size of the transformer changes to correspond to each layer. In our setup, all consecutive layers receive a (potentially resampled) transformer that contains the transformations to align consecutive timepoints of the feature map.

Computing Stabilization Before stabilization takes place, we need to compute the transformations between consecutive frames. Traditionally, optical flow or feature matching have been utilized for this purpose, but with increase in computation, recent work has moved towards direct methods for estimating the transformation. This optimizes the transformation directly on the image intensities, which enables utilizing all of the information in the image, making the estimation higher accuracy, and particularly robust to frames with few keypoints [47]. In particular, we directly optimize the parameters of a transformation (affine or homography), but in a batch fashion on each pair of consecutive frames in each mini batch. The objective from LSD-SLAM [47] operates directly on images intensities (Huber norm) and downweighs large values. We simplify it to:

$$\min_{\theta} \sum_{h,w} \min\left(\|\mathbf{x}(t,h,w) - \mathbf{x}(\mathcal{T}_\theta(t+1,h,w))\|^2, \delta \right) , \tag{7.4}$$

where we align the frame at t and t+1. We scale the source $(t+1)$ and target (t) image equally such that target image has unit variance, and $\delta = 0.01$. We then use gradient-based optimization with adaptive learning rate to minimize this objective with respect to the transformation parameters θ.

7.5 Discussion

We presented a framework that can use transformations to better use the available data, moving beyond the implicit biases in the camera that recorded a given video.

Video Stabilization One of the hypotheses that we explore is stabilization in terms of a world frame, and in terms of a target (smooth pursuit). We discovered that stabilization on its own is not very helpful for current vision systems. However, when combined with fixation and saccades that highlight useful aspects of the input, it has significant advantages.

Fixation in Animals Similarly, eye movement and fixation in animals is hypothesized to be primarily in order to fully utilize the center of the retina, the fovea, that has high-resolution and only covers about 1-2 degrees of vision [180]. This is, it is about efficient usage of resources. Furthermore, average fixation duration in humans is only 330 ms (8 frames at 24 fps) [82], suggesting long-term stabilization might be unnecessary for some visual systems. Thus eye movement primarily plays a role in allowing efficient visual search of the scene.

Stabilization in First-Person Videos Stabilization in the first-person vision case is complicated by the fact that the camera is commonly doing smooth pursuit of the objects/hands of interest already, and zooming in is unnecessary because the field of view is already narrow. In the Charades third-person videos the camera often follows the person, that is, smooth pursuit is already present.

We addressed several challenges that come with moving beyond the camera, and highlighted problems that have to be considered. Such as, variability due to scale and aspect ratio changes, how to include data augmentation, and working with pretrained networks that only consider particular types of input data. We hope this work opens the door for systems that learn policies for visual search and efficient allocation of visual resources.

Part III

Understanding Humans from Their Perspective

Understanding humans, is as elusive of a task as it is to define. Even for the complexities of human intelligence, even relatively minor damage can significantly impair the ability to understand human behavior. This suggests that simple observations of humans may not be sufficient for full understanding, and if we really want our systems to understand humans we need to teach them to put themselves in our shoes. These ideas are explored in Chapter 8. To fully understand human culture and humans themselves, it may be needed to go beyond building a complete dataset that captures all possible concepts. Thus, venturing forth away from prepared datasets is eventually needed to expand the knowledge of the model. Chapter 9 explores this idea of learning concepts in an unsupervised manner from videos uploaded by people to the web.

Chapter 8

Modeling both First and Third-Person

Several theories in cognitive neuroscience suggest that when people interact with the world, or simulate interactions, they do so from a first-person egocentric perspective, and seamlessly transfer knowledge between third-person (observer) and first-person (actor). Despite this, learning such models for human action recognition has not been achievable due to the lack of data. We take a step in this direction, with the introduction of Charades-Ego, a large-scale dataset of paired first-person and third-person videos, involving 112 people, with 4000 paired videos. This enables learning the link between the two, actor and observer perspectives. Thereby, we address one of the biggest bottlenecks facing egocentric vision research, providing a link from first-person to the abundant third-person data on the web. We use this data to learn a joint representation of first and third-person videos, with only weak supervision, and show its effectiveness for transferring knowledge from the third-person to the first-person domain.

What is an action? How do we represent and recognize actions? Most of the current research has focused on a data-driven approach using abundantly available third-person (observer's perspective) videos. But can we really learn how to represent an action without understanding goals and intentions? Can we learn goals and intentions without simulating actions in our own mind? A popular theory in cognitive psychology, the Theory of Mind [178], suggests that humans have the ability to put themselves in each others' shoes, and this is a fundamental attribute of human intelligence. In cognitive neuroscience, the presence of activations in mirror neurons and motor regions even for passive observations suggests the same [184].

When people interact with the world (or simulate these interactions), they do so from a first-person egocentric perspective [103]. Therefore, making strides towards human-like activity understanding might require creating a link between the two worlds of data: first-person and third-person. In recent years, the field of egocentric action understanding [99,132,135,173,183,195] has bloomed due to a variety of practical applications, such as augmented/virtual reality. While first-person and third-person data represent the two sides of the same coin, these two worlds are hardly connected. Apart from philosophical reasons, there are practical reasons for establishing this connection. If we can create a link, then we can use billions of easily available third-person videos to improve egocentric video

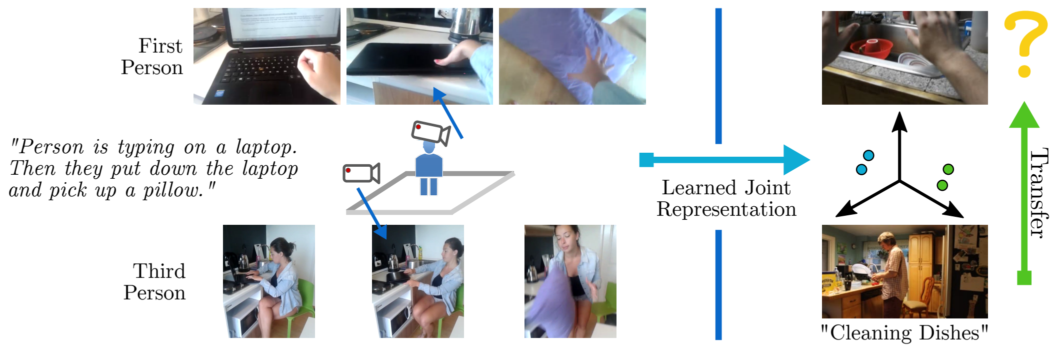

First Person
"Person is typing on a laptop. Then they put down the laptop and pick up a pillow."
Third Person
Learned Joint Representation
?
Transfer
"Cleaning Dishes"

Figure 8.1: We explore how to reason jointly about first and third-person for understanding human actions. We collect paired data of first and third-person actions sharing the same script. Our model learns a representation from the relationship between these two modalities. We demonstrate multiple applications of this research direction, for example, transferring knowledge from the observer's to the actor's perspective.

understanding. Yet, there is no connection: why is that?

The reason for the lack of link is the lack of data! In order to establish the link between the first and third-person worlds, we need aligned first and third-person videos. In addition to this, we need a rich and diverse set of actors and actions in these aligned videos to generalize. As it turns out, aligned data is much harder to get. In fact, in the egocentric world, getting diverse actors and, thus, a diverse action dataset is itself a challenge that has not yet been solved. Most datasets are lab-collected and lack diversity as they contain only a few subjects [51, 54, 173].

Here, we address one of the biggest bottlenecks facing egocentric vision research. We introduce a large-scale and diverse egocentric dataset, Charades-Ego, collected using the Hollywood in Homes [223] methodology. We demonstrate an overview of the data collection and the learning process in Figure 8.1, and present examples from the dataset in Figure 8.2. Our new dataset has 112 actors performing 157 different types of actions. More importantly, we have the same actors perform the same sequence of actions from both first and third-person perspective. Thus, our dataset has semantically similar first and third-person videos. These "aligned" videos allow us to take the first steps in jointly modeling actions from first and third-person's perspective. Specifically, our model, ActorObserverNet, aligns the two domains by learning a joint embedding in a weakly-supervised setting. We show a practical application of joint modeling: transferring knowledge from the third-person domain to the first-person domain for the task of zero-shot egocentric action recognition.

8.1 Background

Action recognition from third-person perspective has been extensively studied in computer vision. The most common thread is to use hand-crafted features [115, 127, 128] or learn features for recognition using large-scale datasets [16, 227]. We refer the reader to [177, 274] for a detailed survey of these approaches, and in the following we focus on the work most relevant to our approach. Our work is inspired by methods that attempt to go beyond

modeling appearances [99,271]. Our core hypothesis is that modeling goals and intentions requires looking beyond the third-person perspective.
Egocentric understanding of activities. Given recent availability of head-mounted cameras of various types, there has been a significant amount of work in understanding first-person egocentric data [52,132,135,142,173,195]. This unique insight into people's behaviour gives rise to interesting applications such as predicting where people will look [135], and how they will interact with the environment [182]. Furthermore, it has recently been shown that egocentric training data provides strong features for tasks such as object detection [99].
Datasets for egocentric understanding. Egocentric video understanding has unique challenges as datasets [51,54,132,173] are smaller by an order of magnitude than their third-person equivalents [50,223]. This is due to numerous difficulties in collecting such data, e.g., availability, complexity, and privacy. Recent datasets have targeted this issue by using micro-videos from the internet, which include both third and first-person videos [161]. While they contain both first and third-person videos, there are no paired videos that can be used to learn the connection between these two domains. In contrast, our dataset contains corresponding first and third-person data, enabling a joint study.
Unsupervised and self-supervised representation learning. In this chapter, we use the multi-modal nature of the data to learn a robust representation across those modalities. It allows us to learn a representation from the data alone, without any explicit supervision. This draws inspiration from recent work on using other cues for representation learning, such as visual invariance for self-supervised learning of features [1,99,136,149,168,233,268,271]. For example, this visual invariance can be obtained by tracking how objects change in videos [271] or from consecutive video frames [149]. Typically, this kind of invariance is harnessed via deep metric learning with Siamese (triplet) architectures [25,67,78,87,265,294].
Data for joint modeling of first and third person. To learn to seamlessly transfer between the first and third-person perspectives we require paired data of these two domains. Some recent work has explored data collected from multiple viewpoints for a fine-grained understanding human actions [102]. Due to the difficulty of acquiring such data, this is generally done in a small-scale lab setting [51,102], with reconstruction using structure-from-motion techniques [102], or matching camera and head motion of the exact same event [176,290]. Most related to our work is that of Fan et al. [51] which collects 7 pairs of videos in a lab setting, and learns to match camera wearers between third and first-person. In contrast, we look at thousands of diverse videos collected by people in their homes.

8.2 Charades-Ego

In order to link first-person and third-person data, we need to build a dataset that has videos shot in first and third-person views. We also need the videos to be semantically aligned, i.e., the same set of actions should appear in each video pair. Collection in a controlled lab setting is difficult to scale, and very few pairs of videos of this type are available on the web. In fact, collection of diverse egocentric data is a big issue due to privacy concerns. So how do we scale such a collection?

We introduce the Charades-Ego dataset. The dataset is collected by following the methodology outlined by the "Hollywood in Homes" approach [223], originally used to collect the Charades dataset [216,223], where workers on Amazon Mechanical Turk (AMT) are incentivized to record and upload their own videos. This in theory allows for the creation of any desired data.

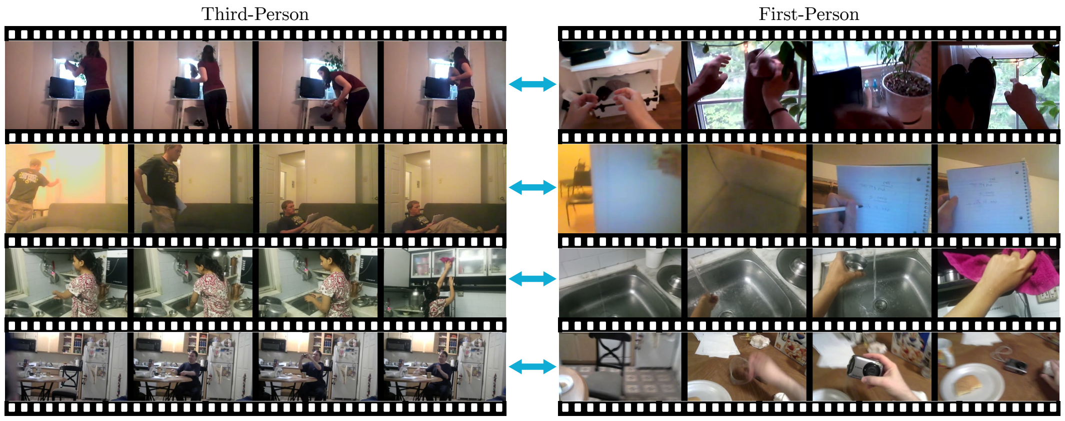


Figure 8.2: Examples from Charades-Ego, showing third-person (left) and the corresponding first-person (right) video frames.

In particular, to get data that is both in first and third-person we use publicly available scripts from the Charades dataset [223], and ask users to record two videos: (1) one with them acting out the script from the third-person; and (2) another one with them acting out the same script in the same way, with a camera fixed to their forehead. We ensure that all the 157 activity classes from Charades occur sufficiently often in our data. The users are given the choice to hold the camera to their foreheads, and do the activities with one hand, or create their own head mount and use two hands. We encouraged the latter option by incentivizing the users with an additional bonus for doing so.* This strategy worked well, with 59.4% of the submitted videos containing activities featuring both hands, courtesy of a home-made head mount holding the camera.

Specifically, we collected 4000† pairs of third and first-person videos (8000 videos in total), with over 364 pairs involving more than one person in the video. The videos are 31.2 seconds long on average. This data contains videos that follow the same structure semantically, i.e., instead of being identical, each video pair depicts activities performed by the same actor(s) in the same environment, and with the same style. This forces a model to latch onto the semantics of the scene, and not only landmarks. We evaluated the alignment of videos by asking workers to identify moments that are shared across the two videos, similar to the algorithmic task in Section 8.4.3, and found the median alignment error to be 1.3s (2.1s average). This offers a compromise between a synchronized lab setting to record both views simultaneously, and scalability. In fact, our dataset is one of the largest first-person datasets available [51, 54, 132, 173], has significantly more diversity (112 actors in many rooms), and most importantly, is the only large-scale dataset to offer pairs of first and third-person views that we can learn from. Examples from the dataset are presented in Figure 8.2. Our data is publicly available at `github.com/gsig/actor-observer`.

8.3 Jointly Modeling First and Third-Person

As shown in Figure 8.1, our aim is to learn a shared representation, i.e., a common embedding for data, from the corresponding frames of the first and the third-person domains. In the

*We compensated AMT workers $1.5 for each video pair, and $0.5 in additional bonus.

†Since the scripts are from the Charades dataset, each video pair has another third-person video from a different actor. We use this video also in our work.

example in the figure, we have a full view of a person working on a laptop in third-person. We want to learn a representation where the corresponding first-person view, with a close-up of the laptop screen and a hand typing, has a similar representation. We can use the correspondence between first and third-person as supervision to learn this representation that can be effective for multiple tasks. The challenges in achieving this are: the views are very visually different, and many frames are uninformative, such as walls, doors, empty frames, and blurry frames. We now describe a model that tackles these challenges by learning how to select training data for learning a joint representation.

8.3.1 Formulation

The problem of modeling the two domains is a multi-modal learning problem, in that, data in the first-person view is distinct from data in the third-person view. Following the taxonomy of Baltrusaitis et al. [10] we formulate this as learning a *coordinated representation* such that corresponding samples in both the first and third-person modalities are close-by in the joint representation. The next question is how to find the alignment or corresponding frames between the two domains. We define ground-truth alignment as frames from first and third-person being within Δ-seconds of each other, and non-alignment as frames being further than Δ'-seconds, to allow for a margin of error.

If a third-person frame x and a first-person frame z map to representations $f(x)$ and $g(z)$ respectively, we want to encourage similarity between $f(x){\sim}g(z)$ if their timestamps t_x and t_z satisfy $|t_x - t_z| < \Delta$. If the two frames do not correspond, then we maximize the distance between their learned representations $f(x)$ and $g(z)$. One possible way to now learn a joint representation is to sample all the corresponding pairs of (x, z), along with a non-corresponding first-person frame z' and use a triplet loss. However, this is not ideal for three reasons: (1) It is inefficient to sample all triplets of frames; (2) Our ground truth (correspondence criteria) is weak as videos are not perfectly synchronized. (3) We need to introduce a mechanism which selects samples that are informative (e.g., hand touching the laptop in Figure 8.1) and conclusive. These informative samples can also be non-corresponding pairs (negative).

We define the problem of learning the joint representation formally with our loss function l_θ. The loss is defined over triplets from the two modalities (x,z,z'). The overall objective function is given by:

$$L = \mathop{\mathbb{E}}_{(x,z,z')\sim P_\theta} \left[l_\theta(x,z,z') \right], \tag{8.1}$$

where l_θ is a triplet loss on top of ConvNet outputs, and θ is set of all the model parameters. The loss is computed over a *selector* P_θ. We also learn P_θ, a parameterized discrete distribution over data, that represents how to sample more informative data triplets (x,z,z'). Intuitively, this helps us find what samples are likely too hard to learn from. To avoid the degenerate solution where P_θ emphasizes only one sample, we constrain P_θ by reducing the complexity of the function approximator, as discussed in Section 8.3.2.

The joint model from optimizing the loss and the selector can be used to generate the other view, given either first or third-person view. We illustrate this in Figure 8.3, where we find the closest first-person frames in the training set, given a third-person query frame. We see that the model is able to connect the two views from the two individual frames, and hallucinate what the person is seeing.

Our setup is related to previous formulations in self-supervised and unsupervised learning, where the pairs (x,z) are often chosen with domain-specific heuristics, e.g., tempo-

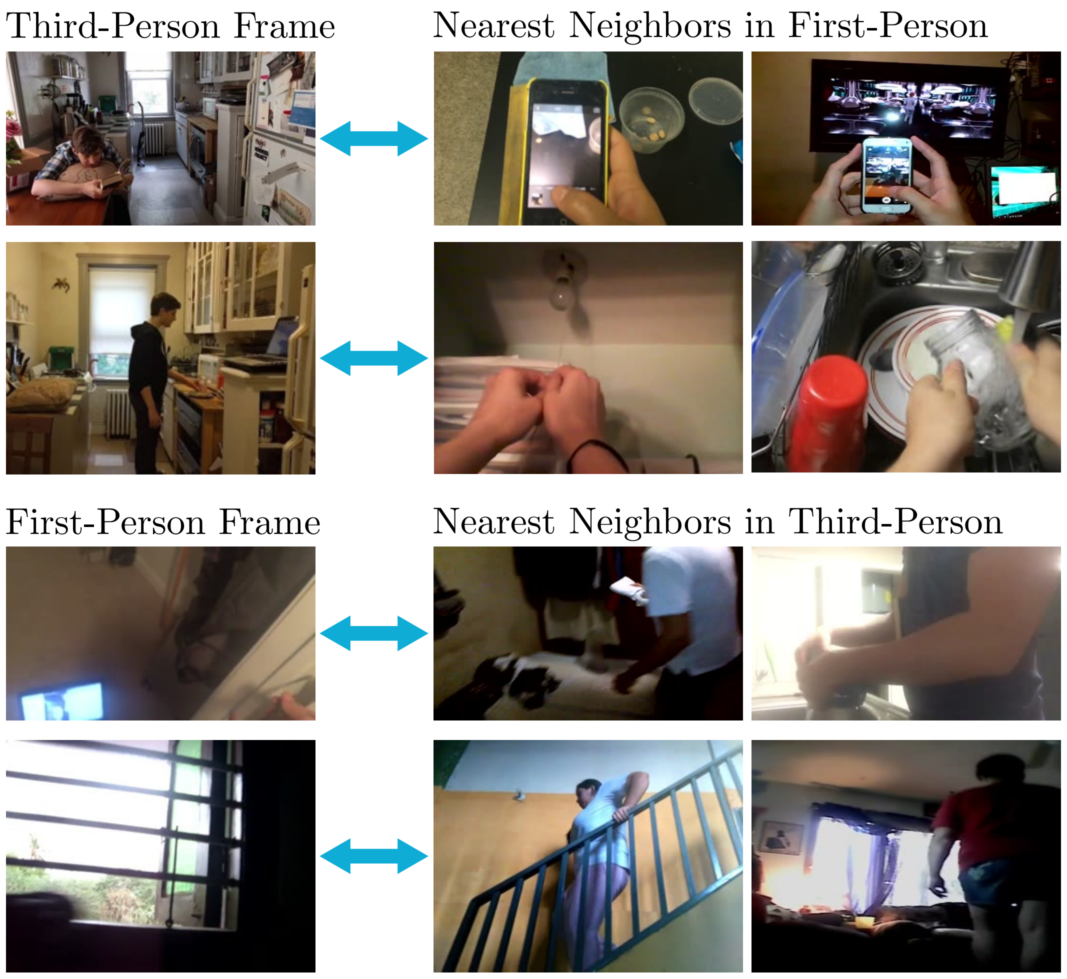

Third-Person Frame
Nearest Neighbors in First-Person
First-Person Frame
Nearest Neighbors in Third-Person

Figure 8.3: Using our joint first and third-person model we can hallucinate how a scene might look through the eyes of the actor in the scene. The top two rows show nearest neighbours (on the right) from first-person videos. The bottom two rows show the observer's perspective, given a first-person video frame.

ral [99,271] and spatial [41] proximity. Triplet loss is a common choice for the loss l_θ for these tasks [41,87,99,271]. We will now address how we model our loss function with a ConvNet, and optimize it with stochastic gradient descent.

8.3.2 Optimizing the objective

Optimizing the objective involves learning parameters of both the triplet loss l_θ, as well as the selector P_θ. This correlated training can diverge. We address this by using importance sampling to rewrite the objective L (8.1) to an equivalent form. We move the distribution of interest P_θ to the objective and sample from a different fixed distribution Q as follows:

$$L = \mathop{\mathbb{E}}_{(x,z,z')\sim Q}\left[\frac{p_\theta(x,z,z')}{q(x,z,z')} l_\theta(x,z,z')\right]. \tag{8.2}$$

We choose Q to be a uniform distribution over the domain of possible triplets: $\{(x, z, z') \mid |t_x - t_z| < \Delta, |t_x - t'_z| > \Delta'\}$. We uniformly sample frames from first and third-person videos, but re-weight the loss based on the informativeness of the triplet. Here, $p_\theta(x, z, z')$ is the value of the selector for the triplet choice (x, z, z').

Instead of modeling the informativeness of the *whole* triplet, we make a simplifying assumption. We assume the selector P_θ factorizes as $p_\theta(x,z,z') = p_\theta(x) p_\theta(z) p_\theta(z')$. Further, we constrain P_θ such the probability of selecting any given frame in that video sums to one for a given video. This has similarities with the concept of "bags" in multiple instance learning [4], where we only know whether a given set (bag) of examples contains positive examples, but not if all the examples in the set are positive. Similarly, here we learn a distribution that determines how to select the useful examples from a set, where our sets are videos. We use a ConvNet architecture to realize our objective.

8.3.3 Architecture of ActorObserverNet

The ConvNet implementation of our model is presented in Figure 8.4. It consists of three streams: one for third-person, and two for first-person with some shared parameters. The streams are combined with a L2-based distance metric [87] that enforces small distance between corresponding samples, and large distance between non-corresponding ones:

$$l_\theta(x,z,z') = \frac{e^{\|x-z\|_2}}{e^{\|x-z\|_2} + e^{\|x-z'\|_2}}. \tag{8.3}$$

The computation of the selector value, $p_\theta(x,z,z')$, for a triplet (x,z,z') is also done by the three streams. The selector values are the result of a 4096×1 fully-connected layer, followed by a scaled tanh nonlinearity[‡] for each stream. We then define a novel non-linearity, VideoSoftmax, to compute the per-video normalized distribution over frames in different batches, which are then multiplied together to form $p_\theta(x) p_\theta(z) p_\theta(z')$. Once we have the different components of the loss in (8.2) we add a loss layer ("Final loss" in the figure). This layer combines the triplet loss l_θ with the selector output p_θ and implements the loss in (8.2). All the layers are implemented to be compatible with SGD [217].

VideoSoftmax layer. The distribution P_θ is modeled with a novel layer which computes a probability distribution across multiple samples corresponding to the same video, even if they occur in different batches. The selector value for a frame x is given by:

$$p_\theta(x) = \frac{e^{f_\theta(x)}}{\sum\limits_{x' \in \mathcal{V}} e^{f_\theta(x')}}, \tag{8.4}$$

where $f_\theta(x)$ is the input to the layer and denominator is the sum of $e^{f_\theta(x')}$ computed over all frames x' in the same video $\mathcal{V}$. This intuitively works like a softmax function, but across frames in the same video.

Since triplet loss l_θ is weighted by the output of the selector, the gradient updates with respect to the triplet loss are simply a weighted version of the original gradient. The gradient for optimizing the loss in (8.2) with respect to the selector in (8.4) is (with slight abuse of notation for simplicity):

$$\frac{\partial L}{\partial f} \propto p_\theta(x,z,z')(l_\theta(x,z,z') - L), \tag{8.5}$$

[‡]The choice of Tanh nonlinearity makes the network more stable than unbounded alternatives like ReLU.

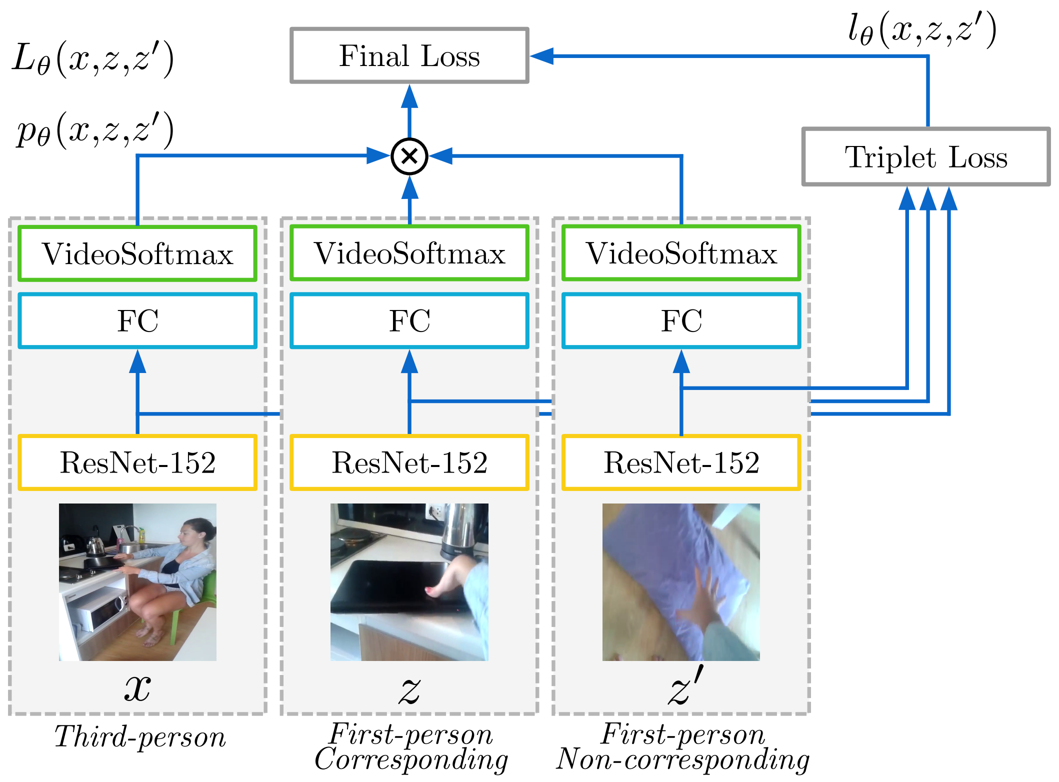


Figure 8.4: Illustration of our ActorObserverNet. The model has separate streams for first and third-person. Given a triplet of frames from these two modalities, the model computes their fc7 features, which are used to compare and learn their similarity. The FC and the VideoSoftmax layers also compute the likelihood of this sample with respect to the selector P_θ.

where the gradient is with respect to the input of the VideoSoftmax layer f, so we can account for the other samples in the denominator of (8.4). Q is defined as a constant over the domain, and can be ignored in the derivation. The intuition is that this decreases the weight of the samples that are above the loss L (8.1), and increases it otherwise. Our method is related to mining easy examples. The selector learns to predict the relative weight of each triplet, i.e., instead of using the loss directly to select triplets (as in mining hard examples). The gradient is then scaled by the magnitude of the weight. The average loss L is computed across all the frames.

8.4 Experiments

We demonstrate the effectiveness of our joint modeling of first and third-person data through several applications, and also analyze what the model is learning. In Section 8.4.2 we evaluate the ability of the joint model to discriminate correct first and third-person pairs from the incorrect ones. We investigate how well the model localizes a given first-person moment in a third-person video, from the same as well as users, by temporally aligning a one-second moment between the two videos (Section 8.4.3). Finally, in Section 8.4.4 we present results for

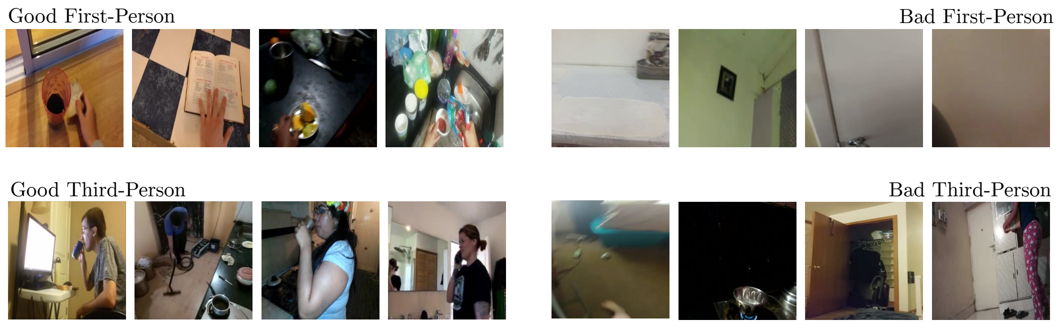


Figure 8.5: A selection of frames, from third and first-person videos, the model assigns the highest and the lowest weights, i.e., $p_\theta(x)$ and $p_\theta(z)$ from (8.2) respectively. This provides intuition into what the model is confident to learn from.

transferring third-person knowledge into the first-person modality, by evaluating zero-shot first-person action recognition. We split the 8000 videos into 80% train/validation, and 20% test for our experiments.

8.4.1 Implementation details

Our model uses a ResNet-152 video frame classification architecture, pretrained on the Charades dataset [223], and shares parameters between both the first and third-person streams. This is inspired by the two-stream model [227], which is a common baseline architecture even in ego-centric videos [51, 142]. The scale of random crops for data augmentation in training was set to $80-100\%$ for first-person frames, compared to the default $8-100\%$ for third-person frames. We set the parameter Δ for the maximum distance to determine a positive pair as one second (average alignment error in the dataset), and the parameter Δ' for the negative pair as 10 seconds.

We sample the training data triplets, in the form of a positive pair with first and third-person frames, which correspond to each other, and a negative pair with the same third-person frame and an unrelated first-person frame from the same video. This sampling is done randomly following the uniform distribution Q in (8.2). The scales of tanh are constrained to be positive. For the experiments in Sections 8.4.3 and 8.4.4, the parameters of the fully connected layers for the two first-person streams are shared. Our code is implemented in the PyTorch machine learning framework and is available at `github.com/gsig/actor-observer`.

8.4.2 Mapping third-person to first-person

The first problem we analyze is learning to model first and third-person data jointly, which is our underlying core problem. We evaluate the joint model by finding a corresponding first-person frame, given a third-person frame, under two settings: (1) using the whole test set ('All test data'); and (2) when the model assigns weights to each sample ('Choose $X\%$ of test data'). In the second case, the triplets with the top 5%, 10%, or 50% highest weights are evaluated. Each triplet contains a given third-person frame, and a positive and negative first-person frames. This allows the model to choose examples from the test set to evaluate. NEW

From Table 8.1 we see that the original problem ('All test data') is extremely challenging,

	Random	ImageNet ResNet-152	Charades Two-Stream	ActorObserverNet
Same person				
All test data	50.0	53.6	55.5	51.7
Choose 50% of test data	50.0	55.7	60.2	**73.9**
Choose 10% of test data	50.0	57.9	68.8	**97.2**
Choose 5% of test data	50.0	56.5	71.9	**96.8**
Different persons				
All test data	50.0	50.6	51.7	50.4
Choose 50% of test data	50.0	50.4	51.6	**76.3**
Choose 10% of test data	50.0	49.6	50.8	**98.8**
Choose 5% of test data	50.0	45.6	51.4	**98.3**

Table 8.1: Given a third-person frame, we determine whether a first-person frame corresponds to it. Results are shown as correspondence classification accuracy (in %). Higher is better. See Section 8.4.2 for details.

even for state-of-the-art representations. The baseline results are obtained with models using fc7 features from either ResNet-152 trained on ImageNet or a two-stream network (RGB stream using ResNet-152 from [223]) trained on Charades to compute the loss. The baselines use the difference in distance between positive and negative pairs as the weight used to pick what samples to evaluate on in the second setting.

The results of the two-Stream network ('Charades Two-Stream' in the table) and our ActorObserverNet using all test data ('All test data') are similar, but still only slightly better than random chance. This is expected, since many of the frames correspond to occluded human actions, people looking at walls, blurry frames, etc., as seen in Figure 8.5. On the other hand, our full model, which learns to weight the frames ('Choose $X\%$ of test data' in the table), outperforms all the other methods significantly. Note that our model assigns a weight for each image frame independently, and in essence, learns if it is a good candidate for mapping. We observe similar behavior when we do the mapping with third and first-person videos containing the same action performed by different people ('Different persons' in the table).

Figure 8.5 shows a qualitative analysis to understand what the model is learning. Here, we illustrate the good and the bad frames chosen by the model, according to the learned weights, both in the third and first-person cases. We observe that the model learns to ignore frames without objects and people, and blurry, feature-less frames, such as the ones seen in the bottom row in the figure. Furthermore, our model prefers first-person frames that include hands, and third-person frames with the person performing an action, such as answering a phone or drinking; see frames in the top row in the figure.

Quantitatively, we found that 68% of high-ranked and only 15% of low-ranked frames contained hands. This is further highlighted in Figures 8.6 and 8.7 where we visualize conv5 activations, and gradients at the image layer, respectively. We observe the network attending to hands, objects, and the field of view. Figure 8.8 illustrates the selection over a video sequence. Here, we include the selector value of $p_\theta(z)$ for each frame in a first-person video. The images highlight points in the graph with particularly useful/useless frames.

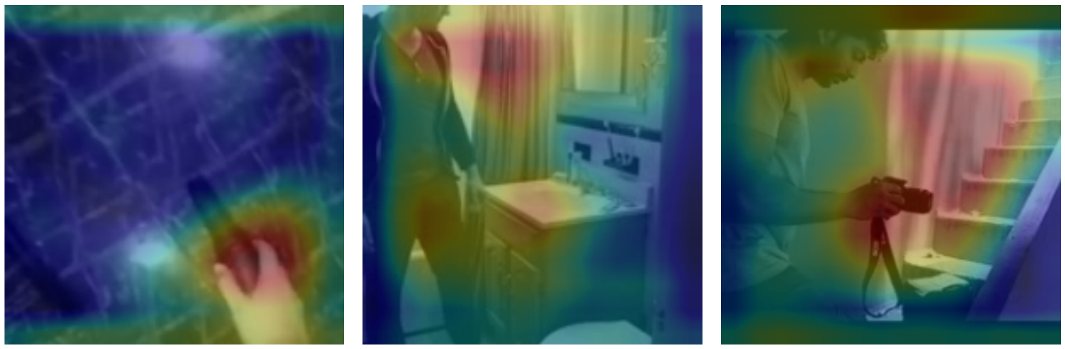


Figure 8.6: Conv5 activations of ActorObserverNet. The colors range from blue to red, denoting low to high activations. We observe the network attending to hands, objects, and the field of view.

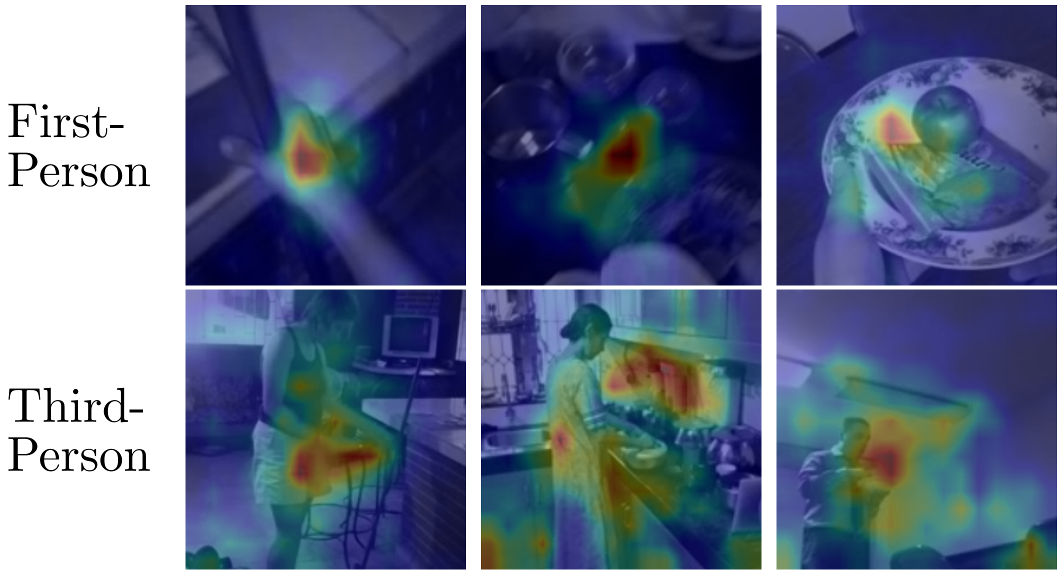


Figure 8.7: By backpropagating the similarity loss to the image layer, we can visualize what regions the model is learning from. The colors range from blue to red, denoting low to high importance.

In general, we see that the weights vary across the video, but the high points correspond to useful moments in the first-person video (top row of images), for example, with a clear view of hands manipulating objects.

8.4.3 Alignment and localization

In the second experiment we align a given first-person moment in time, i.e., a set of frames in a one-second time interval, with a third-person video, and evaluate this temporal localization. In other words, our task is to find any one-second moment that is shared between those

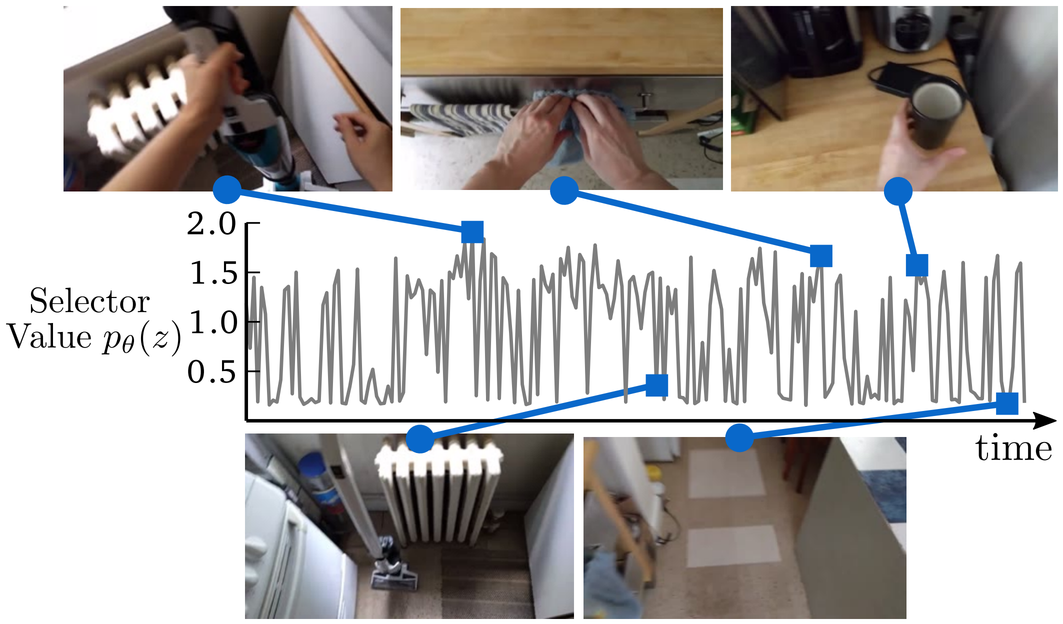


Figure 8.8: Our model learns to assign weights to all the frames in both third and first-person videos. Here we show the selector value $p_\theta(z)$ (the importance of each frame) for a sample first-person video, and highlight frames assigned with high and low values. See Section 8.4.2 for details.

first and third-person perspectives, thus capturing their *semantic similarity*. This allows for evaluation despite uninformative frames and approximate alignment. For evaluation, we assume that the ground truth alignment can be approximated by temporally scaling the first-person video to have the same length as the third-person video.

If m denotes all the possible one-second moments in a first-person and n in a third-person video, there are $m \times n$ ways to pick a pair of potentially aligned moments. Our goal is to pick the pair that has the best alignment from this set. The moments are shuffled so there is no temporal context. We evaluate this chosen pair by measuring how close these moments are temporally, in seconds, as shown in Table 8.2. To this end, we use our learned model, and find one-second intervals in both videos that have the lowest sum of distances between the frames within this moment. We use L2 distance between fc7 features in these experiments.

We present our alignment results in Table 8.2, and compare with other methods. These results are reported as median alignment error in seconds. The performance of fc7 features from the ImageNet ResNet-152 network is close to that of a random metric (11.0s). 'Two-Stream', which refers to the performance of RGB features from the two-stream network trained on the Charades dataset, performs better. Our 'ActorObservetNet' outperforms all these methods.

We visualize the temporal alignment between a pair of videos in Figure 8.9. We highlight in green the best moment in the video chosen by the model: the person looking at their cell phone in the third-person view, and a close-up of the cell phone in the first-person view.

	Random Chance	Human	ImageNet ResNet-152	Charades Two-Stream	ActorObserverNet
Same person	11.0	1.3	8.3	6.5	**5.2**
Different persons	11.0	1.3	8.7	7.0	**6.1**

Table 8.2: Alignment error in seconds for our method 'ActorObserverNet' and baselines. Lower is better. See Section 8.4.3 for details.

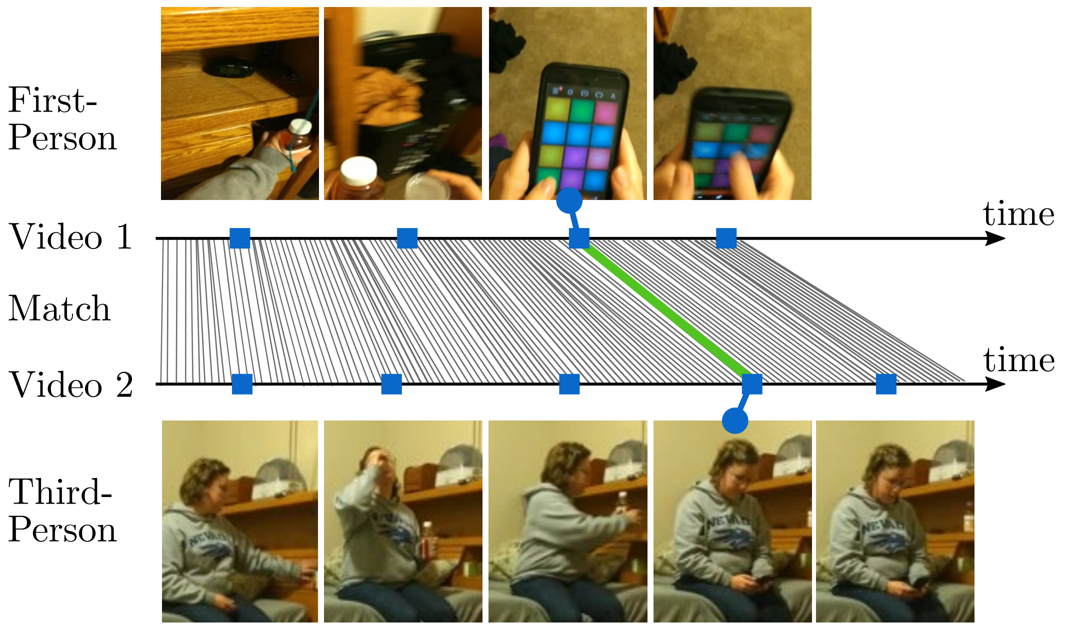


Figure 8.9: Our model matches corresponding moments between two videos. We find the moment in the third-person video (bottom row) that best matches (shown in green) our one second first-person moment (top row), along with other possible matches (gray). (*Best viewed in pdf.*)

8.4.4 Zero-shot first-person action recognition

Since our ActorObserverNet model learns to map between third and first-person videos, we use it to transfer knowledge acquired from a dataset of third-person videos, annotated with action labels, to the first-person perspective. In essence, we evaluate first-person action recognition in a zero-shot setting. We annotated first-person videos in the test set with the 157 categories from Charades [223] to evaluate this setup. Following the evaluation setup from Charades, we use the video-level multi-class mean average precision (mAP) measure.

In order to transfer knowledge from the third-person to the first-person perspective, we add a classification loss to the third-person model after the fc7 layer. To train this framework, we use third-person training examples from the Charades dataset, in addition to the training set from our Charades-Ego dataset. Note that the third-person videos from Charades are annotated with action labels, while our data only has unlabelled first/third person pairs. Thus, we use the mapping loss in (8.2) when updating the network parameters due to first/third person pair, and the RGB component of the two-stream classification loss for an

	Random	Charades VGG-16	Charades ResNet-152	ActorObserverNet
Accuracy	8.9	17.8	22.7	**25.9**

Table 8.3: Egocentric action recognition in the zero-shot learning setup. We show the video-level mAP on our Charades-Ego dataset. Higher is better. See Section 8.4.4 for details.

update due to a Charades third-person example.

Our model now learns to not only map both first and third-person frames to a shared representation, but also a third-person activity classifier on top of that shared representation. At test time, we make a prediction for each frame in a first-person test video, and then combine predictions over all the video frames with mean pooling. We present the results in Table 8.3.

Baseline results. The performance of random chance is 8.9% on the Charades-Ego dataset. We also compare to the RGB two-stream model trained on Charades (third-person videos), using both VGG-16 and ResNet-152 architectures, which achieve 18.6% and 22.8% mAP respectively, on the Charades test set. Both are publicly available [223], and show a 8.9% and 13.8% improvement respectively, over random chance on our first-person videos.

Our results. Our ActorObserverNet further improves over the state-of-the-art two-stream network by 3.2%. This shows that our model can transfer knowledge effectively from the third-person to the first-person domain.

To further analyze whether the gain in performance is due to a better network, or third to first-person transfer, we evaluated our network on the Charades test set. It achieves 23.5% on third-person videos, which is only 0.7% higher than the original model, which suggests that the performance gain is mainly due to the new understanding of how third-person relates to first-person view.

8.5 Discussion

We proposed a framework towards linking the first and third-person worlds, through our novel Charades-Ego dataset, containing pairs of first and third-person videos. This type of data is a first big step in bringing the fields of third-person and first-person activity recognition together. Our model learns how to jointly represent those two domains by learning a robust triplet loss. Semantic equivalence in data allows it to relate the two perspectives from different people. Our results on mapping third-person to first-person, alignment of videos from the two domains, and zero-shot first-person action recognition clearly demonstrate the benefits of linking the two perspectives.

Chapter 9

Semantic Grounding in Video

In this final chapter, we explore how to expand the semantic knowledge of a video model. Whereas our model might now perfectly understand the *information* in the video, we need to connect that information to concepts that we understand. For example, there are thousands of actively spoken languages on Earth, but a single visual world. Grounding in this visual world has the potential to bridge the gap between all these languages. Our goal is to use visual grounding to improve unsupervised word mapping between languages. The key idea is to establish a common visual representation between two languages by learning embeddings from *unpaired* instructional videos narrated in the native language. Given this shared embedding we demonstrate that (i) we can map words between the languages, particularly the 'visual' words; (ii) that the shared embedding provides a good initialization for existing unsupervised text-based word translation techniques, forming the basis for our proposed hybrid visual-text mapping algorithm, MUVE; and (iii) our approach achieves superior performance by addressing the shortcomings of text-based methods – it is more robust, handles datasets with less commonality, and is applicable to low-resource languages. We apply these methods to translate words from English to French, Korean, and Japanese – all without any parallel corpora and simply by watching many videos of people speaking while doing things.

Children can learn multiple languages by merely observing their environment and interacting with other people, without any explicit supervision or instruction; multilingual children do not hear a sentence and its translation simultaneously, and they do not hear a sentence in multiple languages while observing the same situation [64]. Instead, they can leverage visual similarity across situations: what they observe while hearing "the dog is eating" on Monday is very similar to what they see as they hear "le chien mange" on Friday.

We take a first step towards building an unsupervised multimodal translation system by relating the machine translation task to the way children learn multiple languages: we expose the system to videos of people from different countries performing a task while explaining what they are doing in their native languages. There are many such videos in YouTube: for example, we can learn how to squeeze orange juice by watching Korean or English videos. Instructional videos tend to look visually similar and the underlying concepts being spoken are often the same. We obtained a large number of such videos and the corresponding subtitles using automatic speech recognition, extending the recent procedure of [154] to multiple languages.

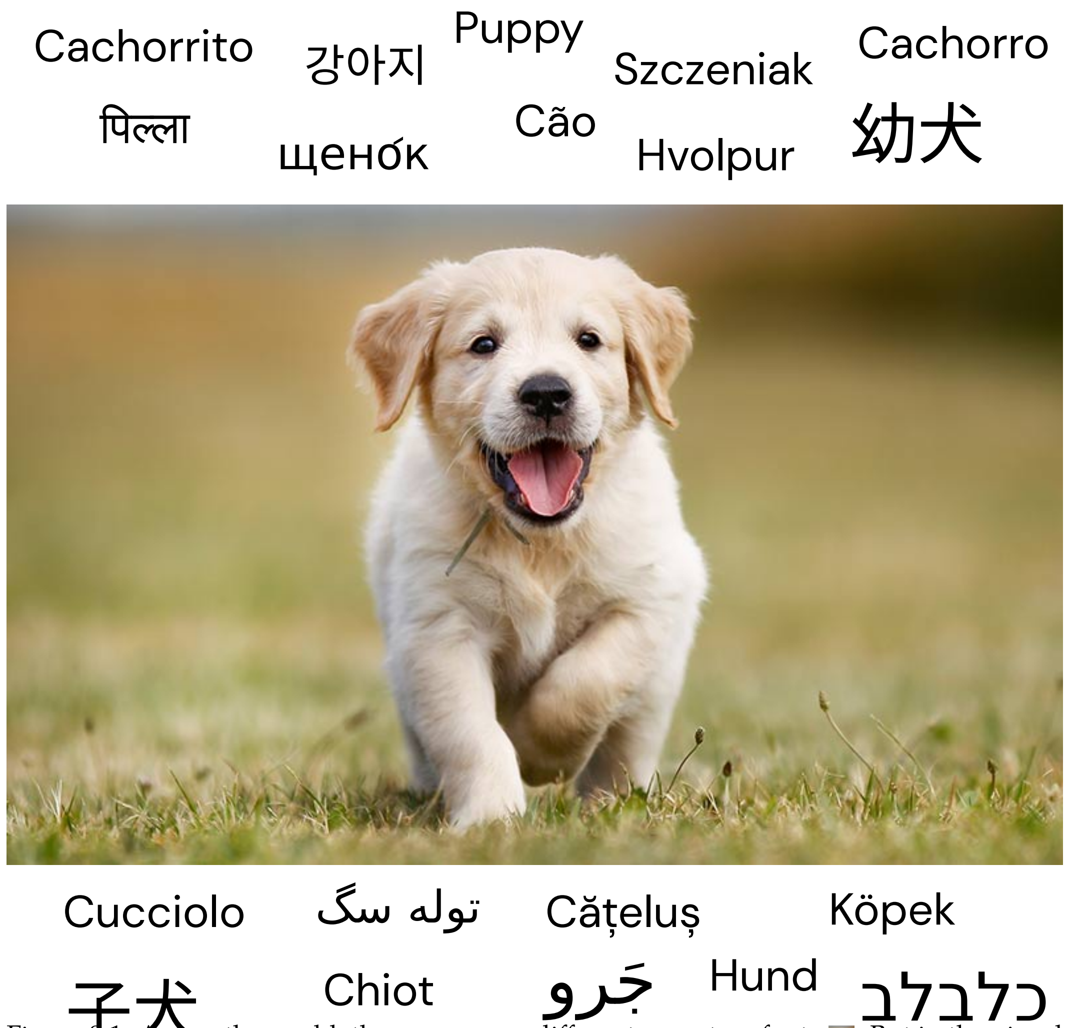


Figure 9.1: Across the world, there are many different ways to refer to . But in the visual domain, a is simply a everywhere on Earth. We leverage this observation to learn to translate words in different languages without *any* paired bilingual data.

Working with this data introduces various challenges. First of all, despite significant recent progress, visual understanding in videos is far from solved – even with the state-of-the-art models, clustering similar activities is not easy. Additionally, and in contrast to manually-captioned datasets where words tend to describe the scene, in instructional videos the words correspond to what the instructors are saying. While performing a task, the instructors often talk about random topics (such as subscriber counts and audience interaction) that do not have any visual relevance.

We demonstrate, that despite these challenges, a shared visual representation can facilitate the mapping of different languages at the word level. As illustrated in Fig. 9.2, we propose a model that maps two languages through the visual domain (videos). For English and French, the model correctly translates 28.0% and 45.3% of common words and visual

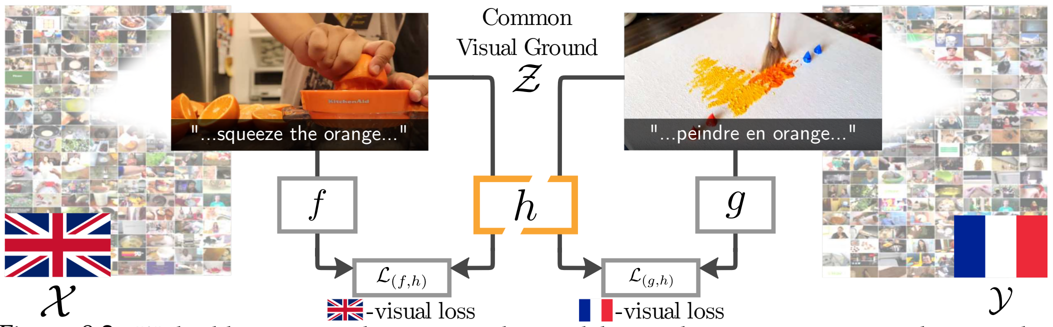


Figure 9.2: We build on recent advances in video modeling and train an unsupervised system that learns to translate words in multiple languages by grounding the language in video and without any paired data. ("peindre en orange"="painting in orange".)

words, all by only watching videos. For comparison, a retrieval-based baseline (without sharing the visual representation) achieves 12.5% and 18.6% for common words and visual words.

Moreover, we show that our model is more robust than the state-of-the-art unsupervised *text-based* word mapping models which exploit co-occurrence statistics [6, 28], in terms of sensitivity to (a) the degree to which the two languages differ (*e.g.*, English is more similar to French than Korean), (b) the dissimilarity of the training corpora of the two languages (*e.g.*, English and French Wikipedia are highly similar), and (c) the amount of training data. Finally, we show that the combination approach (with text-based approaches) is reliable for a large variety of tasks. For example, when the training corpora in French and English are dissimilar (instructional videos in French and Wikipedia in English), our method achieves a 32.6% recall while that of the text-based ones is less than 0.5%.

Contributions. The contributions are threefold. **(i)** We propose a method to map languages through the visual domain using *only* unpaired instructional videos, **(ii)** we demonstrate that our method is effective at connecting words in different languages through vision in an unsupervised manner, and finally **(iii)** we show that our method can serve as a good initialization for existing word mapping techniques addressing many shortcomings of *text-based* methods.

9.1 Background

Bilingual child language acquisition. An open question in the field of bilingual language acquisition is to what extent the systems and representations learned for each language are shared. This sharing can happen for different aspects of language such as grammar, morphology, or the conceptual representations [32, 63]. For example, bilingual children eventually learn that both "chien" and "dog" refer to the actual animal dog, but whether and when this representation is shared is a matter of debate. We explore whether sharing the conceptual (visual) representation improve the quality of word translation for different languages.

Unsupervised text-based word alignment. Words often occur in the same context in different languages – in both English and French, "dog", "catch", and "ball" co-occur together. Previous work has used this insight to align the embedding space of different languages and use the aligned space to translate words from one language to another language [130,156]. Earlier work used various degrees of supervision through ground-truth dictionaries or heuristics [6,89,229]; recently, fully unsupervised approaches achieved a similar performance on word alignment for different language pairs without any supervision [8,28]. However, because these methods take advantage of the similarity between both the language pairs and their training corpora, they are not robust when the languages (or their training corpora) are very different [7,230].

Vision and language. There is a growing interest in combining methods developed in computer vision and natural language processing to solve more challenging problems at the intersection of these fields [3,42,84,91,107,146,210,254]. Grounding language is at the core of the interest of these two communities. It also has a long tradition in symbolic artificial intelligence, where "meaningless symbols cannot be grounded in anything but other meaningless symbols" [79]. The same problem of assigning meaning to symbols has been a fruitful research direction in computer vision. Early work explored weak supervision and the correspondence problem between text annotations and image regions [11,43], with more modern approaches exploring joint image-text word embeddings [60], or building a language conditioned attention map over the images in caption generation, visual question answering and text-based retrieval [5,34,83,144,188,191,254,284,300]. Of particular interest, recent work has focused on multimodal and multilingual settings such as producing captions in many languages, visual-guided translations [12,45,234,272], or bilingual visual question answering [61]. However, these use a paired corpora, *i.e.* same video or images are associated with captions in multiple languages [272]. Obtaining paired corpora in several languages is expensive, and does not scale.

Instructional videos. We rely on instructional videos [2,207,291] since they can be obtained at scale *without any manual annotation* [154]: they consist of YouTube videos and their associated narrations which is generated using automatic speech recognition (ASR). We propose to use instructional videos in different languages to show that we can translate words by only watching and listening to people performing various tasks.

9.2 Unsupervised Multilingual Learning

We describe our approach for unsupervised multilingual word alignment through grounding in the visual domain $\mathcal{Z}$. Our method is *unsupervised* in that it learns the correspondences between two languages $\mathcal{X}$ and $\mathcal{Y}$ (*e.g.* English and French) without *any* parallel (paired) corpora. Instead, we are given two distinct collections of instructional videos, *i.e.* n videos narrated with language $\mathcal{X}$ and another m *different* videos with language $\mathcal{Y}$. Equipped with this, our goal is to learn to map languages $\mathcal{X}$ and $\mathcal{Y}$ by leveraging the shared visual modality $\mathcal{Z}$ – the videos. We evaluate this ability in terms of the accuracy of word translation, *i.e.* how well the vocabulary in one language can be mapped to the other one.

Mapping languages through instructional videos is challenging: first, learning video-text embeddings from instructional videos is difficult as the speech in these videos is only *loosely* related to the scene.* Second, in multilingual setting, such errors compound since both languages have this low video-text relevance; moreover, visually similar videos may not be semantically similar.

*For example, only 50% of the captions and videos in HowTo100M are semantically related [154].

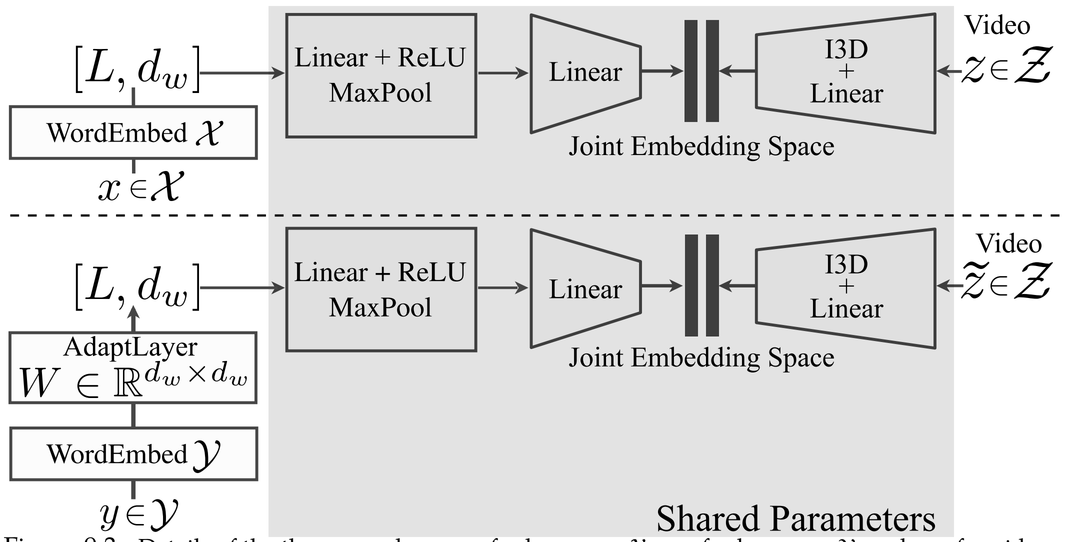


Figure 9.3: Details of the three encoders: one for language $\mathcal{X}$, one for language $\mathcal{Y}$, and one for videos $\mathcal{Z}$. Coupling of the two languages is obtained by sharing relevant parts of the model (shaded region).

This challenge *cannot* be addressed by using the similarity of videos to construct a parallel text corpora (see Fig. 9.4). Instead, following Miech *et al.* [154], we learn a joint (monolingual) video-text embedding space from instructional videos. We extend the training strategy to the multilingual case by defining the following objective:

$$\min_{f,g,h} \underbrace{\mathcal{L}_{(f,h)}(\mathcal{X} \times \mathcal{Z})}_{\text{Language } \mathcal{X} \text{ and vision}} + \underbrace{\mathcal{L}_{(g,h)}(\mathcal{Y} \times \mathcal{Z})}_{\text{Language } \mathcal{Y} \text{ and vision}}, \tag{9.1}$$

where $\mathcal{L}$ is a metric-learning loss between text and video embeddings [153]. The parameters f, g, and h define the embedding functions of the language $\mathcal{X}$, language $\mathcal{Y}$, and the video domain $\mathcal{Z}$, respectively. The idea is that *sharing* the visual encoder h across the two languages is crucial to align the two languages $\mathcal{X}$ and $\mathcal{Y}$.

Next, we describe the proposed approach (Eq. (9.1)) in detail. Sec. 9.2.1 explains our choice of embedding models f, g and h. Sec. 9.2.2 defines the loss function $\mathcal{L}$. Finally, in Sec. 9.2.3, we explain how our initial model can be used to improve text-based word mapping techniques.

9.2.1 Multilingual Visual Embedding: Architecture

An illustration of our architecture is given in Fig. 9.3.

Input to the model. We represent sentences as a fixed-length sequence of integers, *i.e.* $\mathcal{X}$ and $\mathcal{Y}$ are of the form $\{1, \dots, K\}^L$ where K and L are the vocabulary size and sentence length, respectively. On average, sentences consist of 10 words. Videos are in pixel space: $\mathcal{Z} = \mathbb{R}^{T \times H \times W \times 3}$, where T is the number of frames in the video clips (here 32 frames at 10 FPS); H and W are the height and width of the video respectively, with 3 RGB channels.

Text encoders. The text encoder f in language $\mathcal{X}$, following [154], consists of: (i) a word embedding layer that takes as input a sequence composed of L tokens and outputs L vectors of dimension d_w, (ii) a position-wise fully connected feed-forward layer followed by max pooling over the words to generate a single d_i-dimensional vector for the whole sequence, and finally (iii) a linear layer to map the intermediate representation to the joint embedding space $\mathbb{R}^d$.

For the text encoder g in language $\mathcal{Y}$, we share model weights across languages, following [77, 101]. More specifically, we share the weights of the feed forward layers and the last linear layer between f and g. To input different languages to the shared layers, we simply add a linear layer, referred to as the *AdaptLayer*, after the word embedding layer in language $\mathcal{Y}$.

Intuitively, the role of the *AdaptLayer* is to transform the word embedding space of language $\mathcal{Y}$ such that word embeddings in language $\mathcal{Y}$ become as similar as possible to the word embeddings in language $\mathcal{X}$. Then, the rest of the network can be shared, and yet preserve the monolingual properties of the word embeddings if needed. As this layer is only linear, the model does not impose any asymmetry between languages $\mathcal{X}$ and $\mathcal{Y}$.

Video encoder. For the video encoder, we use the standard I3D [16] model followed by a linear layer that maps the output into the joint embedding space.

9.2.2 The Base Model: Training and Inference

Training data. We are given a set of n videos narrated in language $\mathcal{X}$: $\{(x_i, z_i)\}_{i=1}^n$ and a set of m *different* videos narrated in language $\mathcal{Y}$: $\{(y_j, \tilde{z}_j)\}_{j=1}^m$. Note that there is *no* overlap in videos in the first and second set, *i.e.* we do not have access to *paired* bilingual data.

Training objective. The first term $\mathcal{L}_{(f,h)}$ in our objective function Eq. (9.1) is defined as follows:

$$\mathcal{L}_{(f,h)}\left(\{(x_i, z_i)\}_{i=1}^n\right) = \sum_i -\log \mathrm{NCE}\left(f(x_i), h(z_i)\right), \tag{9.2}$$

where NCE corresponds to the noise contrastive estimation [76] discriminative operator:

$$\mathrm{NCE}(x, z) = \frac{e^{f(x)^\top h(z)}}{e^{f(x)^\top h(z)} + \sum\limits_{(x',z')\sim\mathcal{N}} e^{f(x')^\top h(z')}}, \tag{9.3}$$

where $\mathcal{N}$ is a set of negative pairs used to enforce that video and narration that co-occur in the data are close in the space and those that do not are far. Here the negatives are x and z paired with other x' and z' chosen uniformly at random from the training set $\mathcal{X}$, following [153]. In practice, each training batch includes clips from either language, and the negatives for each element in the NCE loss are the other elements from the batch in the same language. $\mathcal{L}_{(g,h)}$ in Eq. (9.1) has the same form, except with g and $\{(y_j, \tilde{z}_j)\}_{j=1}^m$.

Inference. Because we use the same visual encoder h for the two languages, we can assume that the outputs of the language encoders f and g are in the same space. After training our model with the joint loss in Eq. (9.1), we can directly map the first language to the second one; for a given $x \in \mathcal{X}$, we find $y \in \mathcal{Y}$ for which the embedding $g(y)$ has the smallest cosine distance to $f(x)$.

9.2.3 MUVE: Improving Unsupervised Translation

In this section, we explain how the Base Model can be used to improve a state-of-the-art *text-based* word translation technique.

Text-based word translation. It has been shown that distributed representations of words (*e.g*. Word2Vec [157]) share similarities across languages. In particular, Mikolov *et al*. [156] show that a word embedding matrix in a target language can be approximated by simply applying a *linear* mapping on a word embedding matrix in a different source language. To recover that linear mapping, Mikolov *et al*. [156] employ a supervised method where, given a subset of 5,000 pairs of words in the two languages, the mapping is learned by minimizing a L_2 distance between the word embeddings of the source language and the linearly mapped word embeddings of the target language. Xing et al. [280] show that the results can be further improved by adding an orthogonality constraint to the linear mapping. This can be done in closed form with the Procrustes algorithm (see [28] for implementation details).

The unsupervised *MUSE* method. Conneau et al. [28] propose the *MUSE* approach that, in contrast to the method of Mikolov *et al*. [156], does not require any supervised pairs of words. *MUSE* has three main steps (**i**) finding an initial linear mapping via an adversarial approach, then (**ii**) refining the mapping with the Procrustes algorithm, and finally (**iii**) normalizing the distances using the local neighborhood.

MUVE: aligning words through vision. As explained in Section 9.2.1, the intuition behind the linear *AdaptLayer* (see Fig. 9.3) is to map word embeddings from language $\mathcal{Y}$ to a similar vector space as word embeddings from language $\mathcal{X}$ before being fed to the shared layers. Given this, we propose to *replace* the step (**i**) (adversarial initialization) of the *MUSE* algorithm by the *AdaptLayer* of our Base Model, after training it on videos. We call that method MUVE for *Multilingual Unsupervised Visual Embeddings*. To further improve the performance, we follow the observation of [280] by adding to the objective (9.1) an orthogonal penalty $\|WW^\top - I\|_F^2$ on the weights $W \in \mathbb{R}^{d_w \times d_w}$ of the *AdaptLayer*, where I is the d_w-dimensional identity matrix. In Sec. 9.4.3, we demonstrate that MUVE is more robust than its text-based counterparts in multiple aspects.

9.3 Multimodal and Multilingual Datasets

This section explains the training and evaluation datasets used in our experiments (which are discussed in Sec. 9.4). All the datasets will be made publicily available.

9.3.1 The HowToWorld Dataset

Existing instructional video datasets curated from YouTube (*e.g*., the HowTo100M dataset) are in English. We follow the approach of [154] to obtain data in three new languages: French (fr), Japanese (ja) and Korean (ko). We use their list of 23,000 tasks (*e.g*., making a latte) and translate them to French, Japanese and Korean. We obtain 31M, 30M and 34M unique clips with narration from automatic speech recognition for the French, Japanese and Korean datasets, respectively. We use HowTo100M [154] as the English (en) dataset. To ensure that our datasets are stricly unpaired we removed any videos present in more than one of the datasets.

9.3.2 Text Corpora for Training Embeddings

To compare MUVE to the state-of-the-art unsupervised text-based word alignment methods, we use three text corpora: **(i) English and French Wikipedia**: we use the publicly available release of Wikipedia in English and French. We filter the structured output to extract the sentences before processing the text as described in Sec. 9.4.1, **(ii) HowToW-Text-{En,Fr,Ko,Ja}**: we use the narration extracted from the videos of HowToWorld in multiple languages and **(iii) WMT Fr-En corpus**: we use the publicly available WMT French-English corpus, that consists of English to French translations for a variety of news articles.

9.3.3 Evaluation benchmarks

Our goal is to translate words from one language to another (*e.g.*, en to fr, en to ko, en to ja). In this section, we describe the datasets used to analyse the quality of translations.

The *Dictionary* en-{fr,ko,ja}. We use the test split of the ground-truth bilingual dictionaries used in the MUSE paper [28] to compare our method to text-based word mapping methods. Each dictionary provides the translation of 1500 English words in another language (*e.g.*, fr) and list multiple translations for each English word. There are 2943 en-fr, 1922 en-ko, and 1799 en-ja pairs. As we focus on vision and to understand how different methods compare on visual versus non visual words, we also manually annotate the bilingual dictionary for en-fr to select words that can be visually observed (*Dictionary* (*Visual*)). This results in 637 English words and 1430 en-fr pairs. Among example words in the *Dictionary* dataset are: {*torpedo, giovanni, chat, catholics, herald, chuck, ...*} whereas the *Dictionary* (*Visual*) contains {*torpedo, chuck, pit, garrison, sprint, ...*}.

***Simple Words* en-{fr,ko,ja}.** To examine the role of word frequency in the performance of each method, we create a list of the 1000 most common English words from the Simple English Wikipedia. We translate the words in this list to French, Korean, and Japanese using the Google Translate interface. We also manually filter these words to create a list of visual words (*Simple Words* (*Visual*)). Example words in the *Simple Words* dataset include {*correct, touch, hit, either, regard, carry, with, three, ...*} and *Simple Words* (*Visual*) contains {do, fall, police, carry, make, station, afternoon, money, club...}

Human Queries en-{fr,ko,ja}. In order to also qualitatively assess the performance of our proposed model in Sec. 9.4.5, we create a text dataset containing expressions similar to narrations contained in instructional videos. To that end, we manually defined a set of 444 visual queries along with their translations in English, French, Korean, and Japanese. We call this dataset *Human Queries*. Typical examples include {*oil painting, make snowman, glue wood, cut tomato, play violin, open car door, paint shirt, tennis service, brew coffee, dribbling basketball, ...*}.

9.4 Experiments

In this section, we first provide our implementation details (Sec. 9.4.1); in Sec. 9.4.2, we demonstrate the effectiveness of our Base Model in word translation benchmarks. In Sec. 9.4.3, we show that the representations learned by our model can be used to improve the quality of text-based word translation methods. We also show that our method (MUVE) is more robust than the text-based methods (Sec. 9.4.4). Finally, in Sec. 9.4.5, we showcase various qualitative results that give further insight into our method.

9.4.1 Implementation Details

We tokenise the extracted transcripts of all the videos, lower case all the words, and count their occurrences. We generate a vocabulary of the 65,536 most common words for each language independently. All remaining words are mapped to the UNK symbol. After preprocessing the text, for each language, we train monolingual word embeddings using Word2Vec [157] (Skip-Gram, 300 dimensions, window of 5 words, 5 negatives). We use these pretrained embeddings in MUVE, MUSE, and VecMap models.

At training, we sample a video clip (32 frames at 10 FPS) with its corresponding narration from the given datasets (*e.g.*, HowToW-En or the relevant HowToW-{fr-ko-ja}). Each training batch includes clips from either language, and the negatives for each element in the NCE loss are the other elements from the batch in the same language. For the video encoder, we finetune an I3D model [16] pretrained on the Kinetics-400 dataset [16]. For the language models, we use the model described in Sec. 9.2.1, where the word embedding layers are pretrained on the corresponding HowToW-Text datasets to incorporate distributional semantics. We use the Adam optimizer with an initial learning rate of 10^{-3} with batch size of 128 and train the model for 200k iterations on 2 Cloud TPUs.

Evaluation metrics. We report Recall@n in all our experiments: given a query (*e.g.* 'Dog'), we retrieve n results (*e.g.* 'Chien', 'Chienne', 'Chiot', ...), and the retrieval considered a success if *any* of the n results are listed as a correct translation in the ground-truth dictionary. If not specified otherwise, we report Recall@1 . However, we have observed a similar trend with Recall@10 .

9.4.2 The Base Model Evaluation

We investigate whether sharing the visual encoder across languages improves the quality of word translations; to do so, we compare the results of our Base Model with two baselines which we explain below.

Baselines. Our first baseline method (*Random Chance*) retrieves a random hypothesis translation without using videos. The second baseline – *Video Retrieval* – uses videos to create a parallel corpus between the two languages. We first extract I3D features pretrained on Kinetics [16] for all video clips in HowToW-En and HowToW-Fr. We then, for each of the English video clips (100M), find the three closest French video clips (in terms of the L2 distance). Finally, we take the narrations associated with these video pairs to create a parallel text corpus. Given the parallel corpus, we can find alignments between English and French words based on their co-occurrence. More specifically, we calculate the joint probability between the English and French word pairs. For each English word, we can then rank the French words using this joint probability.

Results. We report the results of our models and the baselines on the Dictionary and Simple Words benchmarks in Table 9.1. We observe that our Base Model outperforms the baseline by a significant margin in both benchmarks. Moreover, not surprisingly, the performance of all methods is better on the *Visual* portion of these benchmarks. In Fig. 9.4, we provide two examples of the two types of failures of the *Video Retrieval* model: In the first row, the retrieved video is correct (visually related to the query) but the narrations in English and

English-French		Dictionary		Simple Words	
		All	Visual	All	Visual
1)	Random Chance	0.1	0.2	0.1	0.2
2)	Video Retrieval	6.3	7.6	12.5	18.6
3)	Base Model	9.1	15.2	28.0	45.3
4)	MUVE	**28.9**	**39.5**	**58.3**	**67.5**

Table 9.1: The performance of our models and the baselines measured as Recall@1 on the English-French *Dictionary* and *Simple Words* benchmarks.

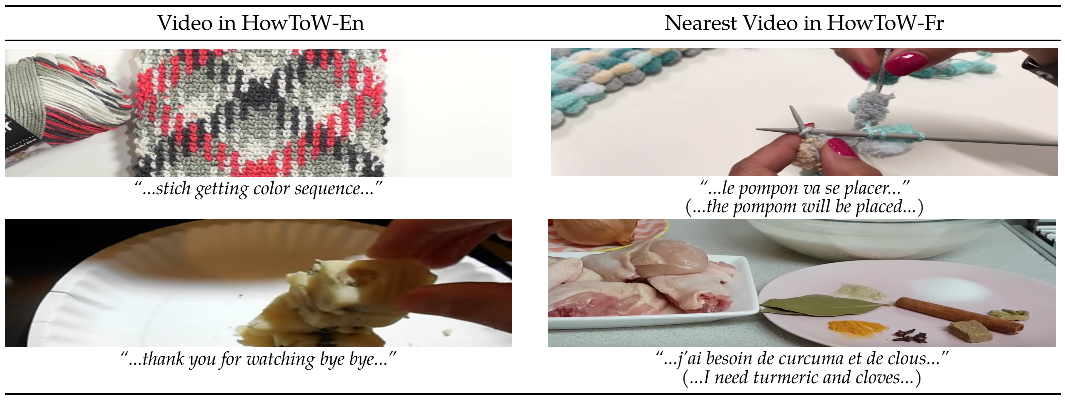

Video in HowToW-En
Nearest Video in HowToW-Fr
"...stich getting color sequence..."
"...le pompon va se placer..."
(...the pompom will be placed...)
"...thank you for watching bye bye..."
"...j'ai besoin de curcuma et de clous..."
(...I need turmeric and cloves...)

Figure 9.4: Examples of two types of failures of the *Video Retrieval* baseline.

French do not convey the same meaning. In the second row, the frame from the retrieved video is somewhat visually similar to the query (both contain food) but does not depict the same concept. This example shows how visual understanding poses a challenge for this task.

9.4.3 MUVE: Improving Text-Based Alignment

We evaluate the proposed MUVE approach, how much the representations learned by our Base Model can improve the text-based word translation methods. We first describe text-based methods that use large scale corpora for word translation, then show how using representations from our model (Sec. 9.2.1) improves the performance of a text-based approach. More specifically, we compare MUVE to three unsupervised and one supervised methods described below. All methods use word embeddings trained on *HowToW-Text* for their respective languages.

Iterative Procrustes iteratively maps word embeddings of two languages using a distance-based heuristic; then it finds the orthogonal matrix that best maps the chosen pairs. We choose the best solution from 25 different initializations (either the identity matrix or random matrices).

MUSE [28] uses adversarial training to map the word embeddings to a space where they are indistinguishable, which provides better starting point for the *Iterative Procrustes* method. The results obtained from *MUSE* [28] have been found to be sensitive to the initialization [6].

VecMap [6] is more robust to initialization and differences across languages when compared to *MUSE*; it obtains better linear transformation by careful normalization, whitening, and

Dictionary		En-Fr		En-Ko	En-Ja
		All	Visual	All	All
1)	Iterative Procrustes	0.2	0.3	0.3	0.3
2)	MUSE [28]	26.3	36.2	11.8	11.6
3)	VecMap [6]	28.4	**40.8**	13.0	13.7
4)	MUVE	**28.9**	39.5	**17.7**	**15.1**
5)	Supervised	57.9	60.3	41.8	41.1

Table 9.2: Performance of our and text-based methods across different language pairs. We report Recall@1 on the *Dictionary* dataset. All method use word embeddings trained on *HowToW-Text* for their respective languages.

dimensionality reduction.

Supervised alignment method provides an upper bound on the unsupervised ones: it uses 5,000 words and their translations to find an optimal orthonormal matrix that best aligns the embeddings of those words.

Results. In Table 9.2, we present the word translation results between English and French, Korean, and Japanese. Our method, MUVE, outperforms all the text-based methods. We observe a bigger improvement over the text-based methods for English-Korean and English-Japanese pairs. These results confirm previous findings that suggest text-based methods are more suited for similar languages (*e.g.*, English and French) [6, 230] and shows that grounding in visual domain for word translation is especially effective in that regime. Finally, we also observe in Table 9.1 a significant improvement of MUVE (row 4) over our Base model alone (row 3) (+19.8% and +30.3% absolute improvement on the Dictionary and Simple Words benchmarks, respectively). Overall, this experiment validates our intuition that the information contained in the visual domain is *complementary* to the word co-occurence statistics used by the text-based methods for the task of unsupervised word translation.

Importance of the orthogonal constraint. As explained in Sec. 9.2.3, we add an orthogonal constraint to the *AdaptLayer* when applying MUVE. We observe that this penalty was a *key* component for MUVE. Precisely, there is a 43.0% relative drop of performance for Recall@1 on the Dictionnary En-Fr (going from 28.9 in Table 9.2 to 16.6) benchmark when removing the orthogonal constraint. This further corroborates the findings described in [280].

9.4.4 Robustness of Unsupervised Word Translation

Sec. 9.4.3 shows that MUVE is more robust to the difference between language pairs when compared to the text-based methods (*i.e.* performance degrades less when going from French to Japanese and Korean in Table 9.2). Here we examine two other axes of robustness: the dissimilarity of the training corpora of the two languages and the amount of training data. All results reported in this section are on English and French languages because text-based models perform better for this pair.

Model selection. We observe that MUSE [28] and VecMap [6] are both sensitive to initialization. To address this, we select the optimal hyperparameters for the text-based method on the *test set*: we perform an extensive search over hyperparameters and random initialisations, *e.g.* 213 runs for the *MUSE* method, and compute the performance of these runs. We then select the *best* performing run on the test set, and hence reporting an upper bound of the

	HowToW-Fr				WMT-Fr				Wiki-Fr			
	∼	[28]	[6]	MUVE	∼	[28]	[6]	MUVE	∼	[28]	[6]	MUVE
HTW-En	.62	45.8	45.4	**47.3**	.67	0.3	0.7	**35.1**	.65	0.3	0.1	**41.2**
WMT-En	.54	0.3	0.2	**26.4**	.40	**88.0**	87.2	85.0	.44	45.9	1.3	**54.9**
Wiki-En	.54	0.3	0.1	**32.6**	.46	**56.7**	52.3	55.9	.39	86.2	**86.7**	82.4

Table 9.3: Robustness of different methods to the dissimilarity of training corpora. We report Recall@10 on *English-French Dictionary* dataset for *MUSE* [28], *VecMap* [6], and *MUVE*, as well as the dissimilarity ($\sim$) of the training corpora expressed with the Jensen Shannon Distance.

true performance of these baselines. Note that when reporting numbers for MUVE we *only* use the monolingual validation loss for model selection, and all numbers for MUVE use the same hyperparameters.

Dissimilarity of the training corpora. We examine how the dissimilarity of the training corpora affects the models. We follow the approach of [44] and measure the dissimilarity of two given corpora by comparing their word co-occurrence statistics. More specifically, we count the frequency of co-occurrences of each pair of words in the same sentence for each corpus, normalize the numbers in order to get a distribution per word, align pair of words in English and French using the Google Translate API, and average the Jensen Shannon distance between distributions for each corresponding words.

We report the results in Table 9.3; all methods are evaluated on the Dictionary dataset with the Recall@10 metric. Looking at the diagonal of the table, we observe when the corpora are *similar* (*e.g.*, Wiki-En and Wiki-Fr), all methods perform well. However when the corpora are *less similar* (off-diagonal elements), we observe that MUVE significantly outperforms its text-based counterparts. We note that methods trained on Wiki-En and WMT-Fr perform better compared to Wiki-Fr and WMT-En. This is likely due to the combination of Wiki-Fr and WMT-En being a smaller corpora: Wiki-En is much larger than Wiki-Fr while the WMT corpora in both languages are of the same size. In conclusion, our method by using visual grounding is more robust to the dissimilarity of the corpora in two languages.

Amount of training data. Unsupervised word translation is especially appealing for low-resource languages where there is no large corpora available. We investigate to what extent MUVE and the text-based methods are robust to the varying size of training data. More specifically, we use 100%, 10%, and 1% of the target training corpora (Wiki-Fr or HowToW-Fr) and report Recall@10. For MUVE, when reducing HowToW-Fr, we also reduce the amount of videos processed. Our results are shown in Fig. 9.5. MUVE is more robust to conditions where the training corpora is small when compared to the text-based methods, revealing another advantage of visual grounding for the task of unsupervised word translation.

Vocabulary size. The text-based methods rely on words' context to align the space of two languages; consequently, the size of vocabulary (and the number of words' neighbors) can play a role in their performance. For low-resource languages, we do not have access to a large corpus and as a result words might not have many neighbors. We explore to what extent the vocabulary size influences the performance of different methods. Fig. 9.6 shows Recall@10 for different methods and vocabulary sizes. We keep the full English vocabulary and vary the size of French vocabulary. We only evaluate on words that are seen in both English and French vocabularies. We observe that MUVE is the only method whose performance does not deteriorate when vocabulary size decreases (even when it is as small as 500).

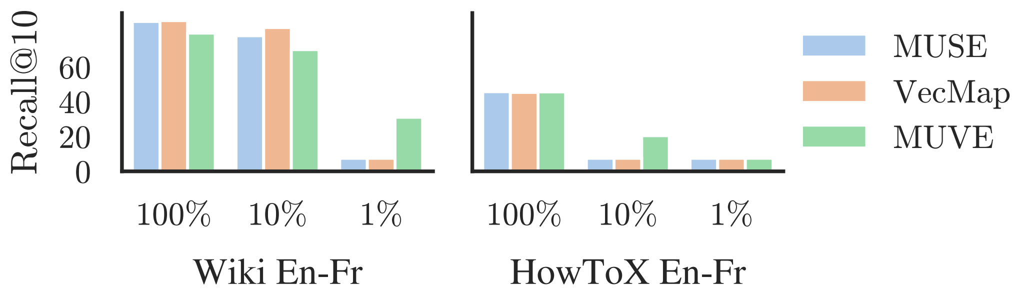


Figure 9.5: Recall@10 on *English-French Dictionary* for *MUSE*, *VecMap*, and *MUVE* using varying amounts of data for each corpus.

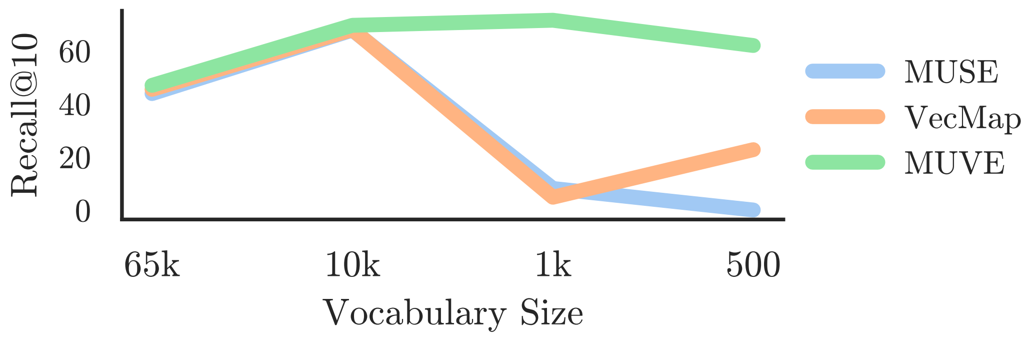


Figure 9.6: Recall@10 on *English-French Dictionary* for *MUSE*, *VecMap*, and *MUVE* for English and French pretrained word embeddings with various vocabulary sizes in French (65k, 10k, 1k, or 500 most common French words). All methods use HowToW-En and HowToW-Fr.

9.4.5 Qualitative Results

In Fig. 9.7, we visualize a 2-stage inference process: (**1**) given an English query (from the *Human Queries* dataset), using our Base Model, we retrieve the video from the training set that is most similar to that query. (**2**) Given that video, we retrieve the closest text from the French Human Queries dataset. The model is able to retrieve relevant videos. However, we also observe that such 2-stage approach can be problematic for translation (*e.g*. the second row of Fig. 9.7 where both individual steps makes sense but the overall result is incorrect due to model drift).

In Table 9.4, we visualize the 1-stage inference process described in Sec. 9.2.2. The model is often accurate, and errors often result in semantically similar words, such as translating *"a man with a dog"* as *"walk dog"* and *"feed dog"*.

9.5 Discussion

Learning multiple languages is a challenging problem that multilingual children tackle with ease. The shared visual domain can help as it allows children to relate words in different

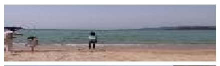
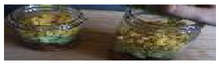
English	Retrieved Video	Top French Hypotheses Given Video	
Beach		Plage (*Beach*)	Courir sur la plage (*Running on the beach*)
Point at the sky		Des Nuages (*Clouds*)	Le coucher du soleil (*Sunset*)
Christmas tree		Sapin de noël (*Christmas Tree*)	Faire un bonhomme de neige (*Make snowman*)
Cut carrot		Couper la carotte (*Cut carrot*)	Carotte (*Carrot*)
Add pickle		Ajoutez des cornichons (*Add pickle*)	Mélanger les legumes verts (*Mix greens*)
Add water		Verser de l'eau (*Pour water*)	Bien mélanger (*Mix thoroughly*)

Figure 9.7: Left: a frame from the video that the model chose as most related to the english query. Right: top 2 french predictions conditioned on the video. The visual grounding provides signal for unsupervised translation, but introduce errors at inference time.

English Text	1st Model Retrieval (*English Meaning*)	2nd Model Retrieval (*English Meaning*)
Boy Playing	Balle qui rebondit par le chat (*Ball Bouncing by the Cat*)	Homme jouant au foot (*Man Playing Football*)
Girl Eats Ice Cream	Chocolat (*Chocolate*)	Sucrer les pancakes (*Top Pancake Sugar*)
Man Driving Red Car	Homme conduit voiture rouge (*Man Driving Red Car*)	Voiture rouge (*Red Car*)
A Man with a Dog	Promener un chien (*Walk Dog*)	Nourrir un chien (*Feed Dog*)
Air Conditioning	Voler dans les airs (*Fly Air*)	Air conditionné (*Air Conditioning*)

Table 9.4: Top 2 retrieved results in French on the *Human Queries* dataset given an English query.

languages through the similarity of their visual experience. Inspired by this idea, we propose an unsupervised multimodal model for word translation that learns from instructional YouTube videos. We show that this approach is beneficial over text-based methods, in that it allows for significantly more robust translation when faced with a variety of realistic perturbations across the corpora.

Chapter 10

Discussion and Conclusion

We proposed a new approach for building datasets. Our Hollywood in Homes approach allows not only the labeling, but the data gathering process to be crowdsourced. We demonstrated that asking multiple questions simultaneously about a video provides the most effective tradeoff between annotation time and accuracy. In addition, Charades offers a novel large-scale dataset with diversity and relevance to the real world. We hope that Charades and Hollywood in Homes will have the following benefits for our community. This kind of realistic bias, may provide new insights that aid robots equipped with our computer vision models operating in the real world. Using this data we have analyzed multiple attributes of activities, several modern activity recognition algorithms, and the latest activity datasets. We demonstrated that even though human disagreement and ambiguity are an inevitable part of activity annotation, they do not present significant roadblocks to progress in activity understanding. We showed that more detailed understanding of scenes depicted in videos, at the level of individual objects and human poses, holds promise for the next iteration of algorithms. We showed that this generation of rich, multi-label, fine-grained activity benchmarks provides ample opportunities for complex joint high-level reasoning about human activities. We hope the community learns from our analysis, and builds upon our work.

We have presented a deep-structured model using a fully-connected temporal CRF that not only models semantic aspects of activities but also reasons about long-term temporal relations. We also presented an asynchronous stochastic inference algorithm that circumvents a key bottleneck in the large-scale end-to-end model learning. We have presented an approach to learn visual storylines for concepts automatically from the web. We show that our method excels at both extracting salient visual signals for the concept, and learning long-term storylines to capture the temporal dynamics. Our framework can use transformations to better use the available data, moving beyond the implicit biases in the camera that recorded a given video. We hope this work opens the door for systems that learn policies for visual search and efficient allocation of visual resources.

We proposed a framework towards linking the first and third-person worlds, through our novel Charades-Ego dataset, containing pairs of first and third-person videos. This type of data is a first big step in bringing the fields of third-person and first-person activity recognition together. Our model learns how to jointly represent those two domains by learning a robust triplet loss. We proposed an unsupervised multimodal model for word translation that learns from instructional YouTube videos. We show that it allows for significantly more robust translation when faced with a variety of realistic perturbations across the corpora.

Future Work in Video Understanding. Autonomy requires training, and training requires supervision. Videos of humans contain oceans of usable data for imitation, and understanding humans. Our effort on multi-modal learning from video [214] and video modeling [217] demonstrates a direction of utilizing videos for better perception of the environment. Utilizing multiple modalities to learn better models, such as third-person video, first-person video, audio, text, tactile, and biometrics, has the potential to significantly help this direction.

Future Work on Improving autonomy with humans-in-the-loop. Generalizing autonomous systems to unseen and noisy scenarios requires going beyond what is currently possible with simulators and engineering. Our work on incorporating humans-in-the-loop to create any data we require through crowdsourcing has the potential to bridge this gap [220,223]. By incorporating crowdsourcing in the pipeline, we can for example gather data for unlikely scenarios and new concepts. These ideas are effective for learning robust model-based control [192], and similar ideas could be applied to improve generalization of high-level concepts using humans-in-the-loop.

Towards Embodied Vision. To build social robots requires building systems that understand both the environment and the humans in the environment. Our work on investigating theory of mind for AI systems [219], and grounding intelligence in a shared experience [214], has the potential to aid this effort. Further investigation is needed on systems that ground their reasoning in a modality that is shared with humans (such as the visual world and the first-person viewpoint), and use this to improve social interaction with autonomous systems.

Bibliography

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *ICCV*, 2015. 125

[2] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Ivan Laptev, Josef Sivic, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *CVPR*, 2016. 140

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 140

[4] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *NeurIPS*, 2003. 129

[5] Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney, Kate Saenko, and Trevor Darrell. Deep compositional captioning: Describing novel object categories without paired training data. In *CVPR*, 2016. 140

[6] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *ACL*, 2017. 139, 140, 146, 147, 148

[7] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *AAAI*, 2018. 140

[8] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *ACL*, 2018. 140

[9] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007. 93

[10] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *PAMI*, 2018. 127

[11] K. Barnard, P. Duygulu, N. de Freitas, D. Forsyth, D. Blei, and M. Jordan. Matching words and pictures. *JMLR*, 2003. 140

[12] Loïc Barrault, Fethi Bougares, Lucia Specia, Chiraag Lala, Desmond Elliott, and Stella Frank. Findings of the third shared task on multimodal machine translation. 2018. 140

[13] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *TNN*, 1994. 86, 90

[14] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, et al. VizWiz: nearly real-time answers to visual questions. In *User Interface Software and Technology* (*UIST*), 2010. 24

[15] Jonathan Bragg, Daniel S Weld, et al. Crowdsourcing multi-label classification for taxonomy creation. In *HCOMP*, 2013. 22, 24

[16] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 107, 109, 113, 118, 124, 142, 145

[17] Zuzana Cernekova, Ioannis Pitas, and Christophoros Nikou. Information theory-based shot cut/fade detection and video summarization. *TCSVT*, 2006. 88

[18] Nathanael Chambers and Daniel Jurafsky. Unsupervised learning of narrative event chains. In *ACL*, 2008. 88

[19] David L Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. 2011. 9

[20] X. Chen, H. Fang, TY Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv:1504.00325*, 2015. 17

[21] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. NEIL: Extracting visual knowledge from web data. In *ICCV*, 2013. 85

[22] Xinlei Chen and C Lawrence Zitnick. Learning a recurrent visual representation for image caption generation. *CVPR*, 2015. 86

[23] L.-C. Chen*, A. G. Schwing*, A. L. Yuille, and R. Urtasun. Learning Deep Structured Models. In *Proc. ICML*, 2015. * equal contribution. 59

[24] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, 2014. 113

[25] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005. 125

[26] Wen-Sheng Chu, Yale Song, and Alejandro Jaimes. Video co-summarization: Video summarization by visual co-occurrence. In *CVPR*, 2015. 87

[27] James A. Coan and John J. B. Allen. *Handbook of Emotion Elicitation and Assessment*. New York, 2007. 23, 24

[28] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv*, 2017. 139, 140, 143, 144, 146, 147, 148

[29] JD Crawford and T Vilis. Axes of eye rotation and listing's law during rotations of the head. *Journal of neurophysiology*, 1991. 107

[30] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 110

[31] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006. 43, 60

[32] Annick De Houwer. Bilingual language acquisition. *The handbook of child language*, 2017. 139

[33] César Roberto de Souza, Adrien Gaidon, Eleonora Vig, and Antonio Manuel López. Sympathy for the details: Dense trajectories and hybrid classification architectures for action recognition. In *ECCV*, 2016. 60

[34] Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In *CVPR*, 2017. 140

[35] Daniel DeMenthon, Vikrant Kobla, and David Doermann. Video summarization by curve simplification. In *ACM MM*, 1998. 87, 88

[36] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 7, 21, 22, 23, 27, 34, 41

[37] Jia Deng, Olga Russakovsky, Jonathan Krause, Michael S Bernstein, Alex Berg, and Li Fei-Fei. Scalable multi-label annotation. In *SIGCHI Conference on Human Factors in Computing Systems*, 2014. 21, 24, 25

[38] Jacob Devlin, Saurabh Gupta, Ross Girshick, Margaret Mitchell, and C Lawrence Zitnick. Exploring nearest neighbor approaches for image captioning. *arXiv:1505.04467*, 2015. 18

[39] Santosh K Divvala, Ali Farhadi, and Carlos Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *CVPR*, 2014. 85

[40] Santosh K Divvala, Derek Hoiem, James H Hays, Alexei A Efros, and Martial Hebert. An empirical study of context in object detection. In *CVPR*, 2009. 55

[41] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 128

[42] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CVPR*, 2015. 60, 86, 140

[43] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, 2002. 140

[44] Julian Eisenschlos, Sebastian Ruder, Piotr Czapla, Marcin Kardas, Sylvain Gugger, and Jeremy Howard. Multifit: Efficient multi-lingual language model fine-tuning. *EMNLP*, 2019. 148

[45] Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. Multi30k: Multilingual english-german image descriptions. *ACL*, 2016. 140

[46] Jeffrey L Elman. Finding structure in time. *Cognitive science*, 1990. 86, 89

[47] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, 2014. 119

[48] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 41

[49] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. Assessing the significance of performance differences on the pascal voc challenges via bootstrapping. Technical report, 2013. 46

[50] Bernard Ghanem Fabian Caba Heilbron, Victor Escorcia and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015. 8, 10, 23, 27, 42, 43, 53, 55, 125

[51] Chenyou Fan, Jangwon Lee, Mingze Xu, Krishna Kumar Singh, Yong Jae Lee, David J. Crandall, and Michael S. Ryoo. Identifying first-person camera wearers in third-person videos. In *CVPR*, 2017. 124, 125, 126, 131

[52] A. Fathi, A. Farhadi, and J. M. Rehg. Understanding egocentric activities. In *ICCV*, 2011. 125

[53] Alireza Fathi and James M. Rehg. Modeling actions through state changes. In *ICCV*, 2013. 60

[54] Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in egocentric activities. In *CVPR*, 2011. 23, 124, 125, 126

[55] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *arXiv*, 2018. 107, 109

[56] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 2009. 113

[57] Basura Fernando, Efstratios Gavves, M. Jose Oramas, Amir Ghodrati, and Tinne Tuytelaars. Modeling video evolution for action recognition. In *CVPR*, 2015. 60

[58] Vittorio Ferrari, Manuel Marín-Jiménez, and Andrew Zisserman. 2d human pose estimation in tv shows. In *Statistical and Geometrical Approaches to Visual Motion Analysis*. 2009. 9

[59] John M Findlay. Saccadic eye movement programming: Sensory and attentional factors. *Psychological Research*, 2009. 107

[60] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NeurIPS*, 2013. 140

[61] Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine? dataset and methods for multilingual image question. In *NeurIPS*, 2015. 140

[62] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012. 23

[63] Fred Genesee. Early bilingual development: One language or two? *Journal of child language*, 1989. 139

[64] Fred Genesee, Johanne Paradis, and Martha B. Crago. *Dual language development & disorders: A handbook on bilingualism & second language learning*. 2004. 137

[65] Rohit Girdhar, Deva Ramanan, Abhinav Gupta, Josef Sivic, and Bryan Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *CVPR*, 2017. 44, 54

[66] Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *CVPR*, 2015. 58, 60

[67] Yunchao Gong, Y. Jia, T. K. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. *ICLR*, 2014. 125

[68] A. Gorban, H. Idrees, Y.-G. Jiang, A. Roshan Zamir, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://www.thumos.info/`, 2015. 8, 9, 10, 23, 42, 43, 53, 55, 60

[69] Lena Gorelick, Moshe Blank, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *PAMI*, 2007. 8

[70] Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv:1502.04623*, 2015. 86

[71] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *CVPR*, 2011. 110

[72] Abhinav Gupta and Larry S Davis. Objects in action: An approach for combining action understanding and object perception. In *CVPR*, 2007. 9

[73] Abhinav Gupta, Aniruddha Kembhavi, and Larry S Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *TPAMI*, 2009. 60

[74] Abhinav Gupta, Praveen Srinivasan, Jianbo Shi, and Larry S Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *CVPR*, 2009. 88

[75] Saurabh Gupta and Jitendra Malik. Visual semantic role labeling. *CoRR*, 2015. 60

[76] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010. 142

[77] Thanh-Le Ha, Jan Niehues, and Alexander Waibel. Toward multilingual neural machine translation with universal encoder and decoder. *arXiv*, 2016. 142

[78] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 125

[79] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 1990. 140

[80] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 118

[81] Fabian Caba Heilbron and Juan Carlos Niebles. Collecting and annotating human activities in web videos. In *ICMR*, 2014. 23

[82] John M Henderson. Human gaze control during real-world scene perception. *Trends in cognitive sciences*, 2003. 120

[83] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. *ICCV*, 2017. 140

[84] Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, and Raia Hadsell. Learning to follow directions in street view. *arXiv*, 2019. 140

[85] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 2002. 66

[86] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997. 87, 89

[87] Elad Hoffer, Itay Hubara, and Nir Ailon. Spatial contrasting for deep unsupervised learning. *NeurIPS Workshop*, 2016. 125, 128, 129

[88] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *ECCV*, 2012. 43, 46, 48, 55

[89] Yedid Hoshen and Lior Wolf. Non-adversarial unsupervised word translation. *arXiv*, 2018. 140

[90] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *CVPR*, 2007. 113

[91] Ronghang Hu, Anna Rohrbach, Trevor Darrell, and Kate Saenko. Language-conditioned graph networks for relational reasoning. *arXiv*, 2019. 140

[92] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *arXiv:1608.08614*, 2016. 55

[93] Y. Iwashita, A. Takamine, R. Kurazume, and M. S. Ryoo. First-person animal activity recognition from egocentric videos. In *ICPR*, 2014. 9

[94] Hamid Izadinia, Ali Farhadi, Aaron Hertzmann, and Matthew D Hoffman. Image classification and retrieval from user-supplied tags. *arXiv:1411.6909*, 2014. 86

[95] Hamid Izadinia and Mubarak Shah. Recognizing complex events using large margin joint low-level event model. *ECCV*, 2012. 60

[96] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NeurIPS*, 2015. 110, 111, 113

[97] Arpit Jain, Abhinav Gupta, Mikel Rodriguez, and Larry S. Davis. Representing videos using mid-level discriminative patches. In *CVPR*, 2013. 60

[98] Émile Javal. Essai sur la physiologie de la lecture. *Annales d'Oculistique*, 1878. 107

[99] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *ICCV*, 2015. 123, 125, 128

[100] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *TPAMI*, 2013. 43, 60

[101] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google's multilingual neural machine translation system: Enabling zero-shot translation. *ACL*, 2017. 142

[102] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social motion capture. In *ICCV*, 2015. 125

[103] Takeo Kanade and Martial Hebert. First-person vision. *Proc. IEEE*, 2012. 123

[104] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and Understanding Recurrent Networks. *Iclr*, 2016. 55, 93

[105] Andrej Karpathy and Fei-Fei Li. Deep visual-semantic alignments for generating image descriptions. *arXiv:1412.2306*, 2014. 86

[106] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 8, 9, 10, 16, 23, 42, 43, 60

[107] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014. 140

[108] David G Kendall. A survey of the statistical theory of shape. *Statistical Science*, 1989. 53

[109] Aditya Khosla, Raffay Hamid, Chih-Jen Lin, and Neel Sundaresan. Large-scale video summarization using web-image priors. In *CVPR*, 2013. 87, 88

[110] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei A. Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012. 55

[111] Gunhee Kim, Seungwhan Moon, and Leonid Sigal. Joint photo stream and blog post summarization and exploration. In *CVPR*, 2015. 87

[112] Gunhee Kim, Leonid Sigal, and Eric P Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *CVPR*, 2014. 87, 88, 94

[113] Gunhee Kim and Eric P. Xing. Reconstructing storyline graphs for image recommendation from web community photos. In *CVPR*, 2014. 87, 88, 93, 94

[114] Kris M Kitani, Brian D Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *ECCV*, 2012. 60

[115] Alexander Klaser, Marcin Marszalek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC*, 2008. 43, 60, 124

[116] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. 2009. 75

[117] Adriana Kovashka and Kristen Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *CVPR*, 2010. 43

[118] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NeurIPS*, 2011. 60, 61

[119] Richard J Krauzlis and Steve G Lisberger. Temporal properties of visual motion signals for the initiation of smooth pursuit eye movements in monkeys. *Journal of Neurophysiology*, 1994. 107

[120] Ranjay Krishna, Kenji Hata, Stephanie Chen, Joshua Kravitz, David A Shamma, et al. Embracing error to enable rapid crowdsourcing. In *SIGCHI Conference on Human Factors in Computing Systems*, 2016. 24

[121] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *CoRR*, 2016. 21, 24, 29

[122] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 15, 59, 69, 86, 90, 92

[123] H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *CVPR*, 2014. 8, 9

[124] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. HMDB: a large video database for human motion recognition. In *ICCV*, 2011. 8, 9, 10, 23, 42, 43

[125] Tian Lan, Yuke Zhu, Amir Roshan Zamir, and Silvio Savarese. Action recognition by hierarchical mid-level action elements. In *ICCV*, 2015. 60

[126] Zhenzhong Lan, Ming Lin, Xuanchong Li, Alexander G. Hauptmann, and Bhiksha Raj. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. In *CVPR*, 2015. 60

[127] Ivan Laptev. On space-time interest points. *IJCV*, 2005. 43, 60, 124

[128] Ivan Laptev, Marcin Marszałek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008. 8, 43, 60, 124

[129] Walter S Lasecki, Mitchell Gordon, Danai Koutra, Malte F Jung, et al. Glance: Rapidly coding behavioral video with the crowd. In *User Interface Software and Technology (UIST)*, 2014. 22, 23, 25

[130] Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *ACL*, 2015. 140

[131] Quoc V. Le, Will Y. Zou, Serena Y. Yeung, and Andrew Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011. 60

[132] Y. J. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012. 88, 123, 125, 126

[133] Li-Jia Li and Li Fei-Fei. What, where and who? classifying events by scene and object recognition. In *ICCV*, 2007. 60

[134] Tao Li and Mitsunori Ogihara. Detecting emotion in music. In *Proceedings of the fourth international conference on music information retrieval (ICMIR)*, 2003. 22

[135] Y. Li, Z. Ye, and J. M. Rehg. Delving into egocentric actions. In *CVPR*, 2015. 123, 125

[136] Yin Li, Manohar Paluri, James M. Rehg, and Piotr Dollár. Unsupervised learning of edges. In *CVPR*, 2016. 125

[137] Zhichao Li, Yi Yang, Xiao Liu, Feng Zhou, Shilei Wen, and Wei Xu. Dynamic computational time for visual attention. In *ICCV*, 2017. 114

[138] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 24, 41

[139] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *CVPR*, 2009. 8, 9, 43

[140] Shuaicheng Liu, Yinting Wang, Lu Yuan, Jiajun Bu, Ping Tan, and Jian Sun. Video stabilization with a depth camera. In *CVPR*, 2012. 110

[141] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Steadyflow: Spatially smooth optical flow for video stabilization. In *CVPR*, 2014. 110

[142] Minghuang Ma, Haoqi Fan, and Kris M Kitani. Going deeper into first-person activity recognition. In *CVPR*, 2016. 125, 131

[143] Yu-Fei Ma, Lie Lu, Hong-Jiang Zhang, and Mingjing Li. A user attention model for video summarization. In *ACM MM*, 2002. 88

[144] Mateusz Malinowski, Carl Doersch, Adam Santoro, and Peter Battaglia. Learning visual question answering by bootstrapping hard attention. In *ECCV*, 2018. 140

[145] Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A neural-based approach to answering questions about images. In *ICCV*, 2015. 86

[146] Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask your neurons: A deep learning approach to visual question answering. *IJCV*, 2017. 140

[147] Marcin Marszałek, Ivan Laptev, and Cordelia Schmid. Actions in context. In *CVPR*, 2009. 9

[148] Ricardo Martin-Brualla, Yanling He, Bryan C Russell, and Steven M Seitz. The 3d jigsaw puzzle: Mapping large indoor spaces. In *ECCV*. 2014. 87

[149] Michaël Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. In *ICLR*, 2016. 125

[150] Pyry Matikainen, Martial Hebert, and Rahul Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *ICCV Workshops*, 2009. 60

[151] Neil McIntyre and Mirella Lapata. Learning to tell tales: A data-driven approach to story generation. In *ACL*, 2009. 88

[152] Pascal Mettes, Jan C van Gemert, and Cees GM Snoek. Spot on: Action localization from pointly-supervised proposals. In *ECCV*, 2016. 58

[153] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-End Learning of Visual Representations from Uncurated Instructional Videos. *arXiv*, 2019. 141, 142

[154] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100M: Learning a text-video embedding by watching hundred million narrated video clips. In *ICCV*, 2019. 137, 140, 141, 143

[155] Tomas Mikolov. Recurrent neural network based language model. In *INTERSPEECH*, 2010. 89, 90, 93

[156] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv*, 2013. 140, 143

[157] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, 2013. 89, 143, 145

[158] George A Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 1956. 22, 24, 31

[159] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv:1312.5602*, 2013. 61

[160] Chong-Wah Ngo, Yu-Fei Ma, and Hong-Jiang Zhang. Video summarization and scene detection by graph modeling. *TCSVT*, 2005. 87, 88

[161] Phuc Xuan Nguyen, Gregory Rogez, Charless Fowlkes, and Deva Ramanan. The open world of micro-videos. *arXiv*, 2016. 125

[162] Juan Carlos Niebles, Chih-Wei Chen, and Li Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010. 115

[163] Jon Noronha, Eric Hysen, Haoqi Zhang, and Krzysztof Z. Gajos. Platemate: Crowdsourcing nutritional analysis from food photographs. In *User Interface Software and Technology* (*UIST*), 2011. 24

[164] Pere Obrador, Rodrigo De Oliveira, and Nuria Oliver. Supporting personal photo storytelling for social albums. In *ACM MM*, 2010. 88, 96

[165] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR*, 2011. 8

[166] Ben Packer, Kate Saenko, and Daphne Koller. A combined pose, object, and feature model for action understanding. In *CVPR*, 2012. 43

[167] Devi Parikh and C Zitnick. Human-debugging of machines. *NeurIPS WCSSWC*, 2011. 55

[168] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *CVPR*, 2017. 125

[169] Genevieve Patterson, Grant Van Horn, Serge Belongie, Pietro Perona, and James Hays. Tropel: Crowdsourcing detectors with minimal training. In *HCOMP*, 2015. 21

[170] Genevieve Patterson, Chen Xu, Hang Su, and James Hays. The SUN attribute database: Beyond categories for deeper scene understanding. *IJCV*, 2014. 21

[171] Xiaojiang Peng, Changqing Zou, Yu Qiao, and Qiang Peng. Action recognition with stacked fisher vectors. In *ECCV*, 2014. 60

[172] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010. 15

[173] Hamed Pirsiavash and Deva Ramanan. Detecting activities of daily living in first-person camera views. In *CVPR*, 2012. 9, 10, 43, 123, 124, 125, 126

[174] Hamed Pirsiavash and Deva Ramanan. Parsing videos of actions with segmental grammars. In *CVPR*, 2014. 60

[175] Hamed Pirsiavash, Carl Vondrick, and Antonio Torralba. Inferring the why in images. *arXiv:1406.5472*, 2014. 60

[176] Yair Poleg, Chetan Arora, and Shmuel Peleg. Head motion signatures from egocentric videos. In *ACCV*, 2014. 125

[177] Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 2010. 55, 60, 109, 124

[178] David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and brain sciences*, 1978. 61, 123

[179] Alessandro Prest, Cordelia Schmid, and Vittorio Ferrari. Weakly supervised learning of interactions between humans and objects. *TPAMI*, 2012. 60

[180] Jan M Provis, Adam M Dubis, Ted Maddess, and Joseph Carroll. Adaptation of the central retina for high acuity vision: cones, the fovea and the avascular zone. *Progress in retinal and eye research*, 2013. 120

[181] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: a saliency-based sampling layer for neural networks. In *ECCV*, 2018. 110, 113, 114

[182] N. Rhinehart and K. M. Kitani. Learning action maps of large environments via first-person vision. In *CVPR*, 2016. 125

[183] Nicholas Rhinehart and Kris M. Kitani. First-person activity forecasting with online inverse reinforcement learning. In *ICCV*, 2017. 123

[184] Giacomo Rizzolatti and Laila Craighero. The mirror-neuron system. *Annu. Rev. Neurosci.*, 2004. 123

[185] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008. 8

[186] Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. Coherent multi-sentence video description with variable level of detail. In *Pattern Recognition*. 2014. 8, 9

[187] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In *CVPR*, 2015. 8, 9, 10

[188] Anna Rohrbach, Marcus Rohrbach, Siyu Tang, Seong Joon Oh, and Bernt Schiele. Generating descriptions with grounded and co-referenced people. In *CVPR*, 2017. 140

[189] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *CVPR*, 2012. 8, 9, 10

[190] Marcus Rohrbach, Michaela Regneri, Mykhaylo Andriluka, Sikandar Amin, Manfred Pinkal, and Bernt Schiele. Script data for attribute-based recognition of composite activities. *ECCV*, 2012. 60

[191] Candace Ross, Andrei Barbu, Yevgeni Berzak, Battushig Myanganbayar, and Boris Katz. Grounding language acquisition by training semantic parsers using captioned videos. In *EMNLP*, 2018. 140

[192] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011. 152

[193] Michele Rucci and Martina Poletti. Control and functions of fixational eye movements. *Annual Review of Vision Science*, 2015. 107

[194] Olga Russakovsky, Jia Deng, Zhiheng Huang, Alexander C Berg, and Li Fei-Fei. Detecting avocados to Zucchinis: What have we done, and where are we going? In *ICCV*, 2013. 55

[195] M. S. Ryoo and L. Matthies. First-person activity recognition: What are they doing to me? In *CVPR*, 2013. 109, 123, 125

[196] Michael S Ryoo and Jake K Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *ICCV*, 2009. 9

[197] Michael S Ryoo and JK Aggarwal. Hierarchical recognition of human activities interacting with objects. In *CVPR*, 2007. 60

[198] Sreemananath Sadanand and Jason J. Corso. Action bank: A high-level representation of activity in video. In *CVPR*, 2012. 60

[199] Fereshteh Sadeghi, Santosh K Divvala, and Ali Farhadi. VisKE: Visual knowledge extraction and question answering by visual verification of relation phrases. In *CVPR*, 2015. 85

[200] Fereshteh Sadeghi, J. Rafael Tena, and Sigal Leonid Farhadi Ali. Learning to select and order vacation photographs. In *WACV*, 2015. 87, 89, 96

[201] Ruslan Salakhutdinov and Geoffrey E Hinton. Deep boltzmann machines. In *AISTATS*, 2009. 66

[202] Elliot Salisbury, Sebastian Stein, and Sarvapali Ramchurn. Crowdar: augmenting live video with a real-time crowd. In *HCOMP*, 2015. 23

[203] Gerard Salton and J Michael. Mcgill. *Introduction to modern information retrieval*, 1983. 10

[204] Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 2000. 22

[205] Christian Schüldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *ICPR*, 2004. 8, 9, 42, 43

[206] Alexander G Schwing and Raquel Urtasun. Fully connected deep structured networks. *arXiv:1503.02351*, 2015. 61

[207] Ozan Sener, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collections. In *ICCV*, 2015. 140

[208] Roger Shank and Robert Abelson. Scripts, plans, goals and understanding, 1977. 88

[209] Ren Shaoqing, He Kaiming, Girshick Ross, and Sun Jian. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 51, 116

[210] Kevin Shen, Amlan Kar, and Sanja Fidler. Lifelong learning for image captioning by asking natural language questions. *arXiv*, 2018. 140

[211] Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In *HCOMP*, 2013. 24

[212] Kevin J Shih, Saurabh Singh, and Derek Hoiem. Where to look: Focus regions for visual question answering. *arXiv:1511.07394*, 2015. 86

[213] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, 2016. 60

[214] Gunnar A. Sigurdsson, Jean-Baptiste Alayrac, Aida Nematzadeh, Lucas Smaira, Mateusz Malinowski, João Carreira, Phil Blunsom, and Andrew Zisserman. Visual grounding in video for unsupervised word translation. In *CVPR*, 2020. 1, 3, 4, 152

[215] Gunnar A. Sigurdsson, Xinlei Chen, and Abhinav Gupta. Learning visual storylines with skipping recurrent neural networks. In *ECCV*, 2016. 1, 3, 4, 60

[216] Gunnar A. Sigurdsson, Jonghyun Choi, Ali Farhadi, and Abhinav Gupta. Charades challenge 2017. `http://vuchallenge.org/charades.html`, 2017. 3, 125

[217] Gunnar A. Sigurdsson, Santosh Divvala, Ali Farhadi, and Abhinav Gupta. Asynchronous temporal fields for action recognition. In *CVPR*, 2017. 1, 3, 4, 44, 46, 47, 50, 51, 129, 152

[218] Gunnar A. Sigurdsson, Abhinav Gupta, Cordelia Schmid, and Karteek Alahari. Beyond the camera: Neural networks in world coordinates. In *arXiv:2003.05614*, 2020. 3, 4

[219] Gunnar A. Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. Actor and observer: Joint modeling of first and third-person videos. In *CVPR*, 2018. 1, 3, 4, 152

[220] Gunnar A. Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. Charades-ego: A large-scale dataset of paired third and first person videos. In *arXiv:1804.09626*, 2018. 3, 152

[221] Gunnar A. Sigurdsson, Olga Russakovsky, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Much ado about time: Exhaustive annotation of temporal data. In *HCOMP*, 2016. 3, 4, 11, 12

[222] Gunnar A. Sigurdsson, Olga Russakovsky, and Abhinav Gupta. What actions are needed for understanding human actions in videos? In *ICCV*, 2017. 3, 4, 107

[223] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016. 3, 4, 22, 23, 24, 27, 28, 37, 42, 43, 44, 45, 53, 55, 58, 59, 60, 66, 68, 69, 70, 71, 112, 115, 116, 124, 125, 126, 131, 132, 135, 136, 152

[224] Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 2001. 12

[225] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 15, 16, 66

[226] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, 2013. 58, 69

[227] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014. 16, 43, 44, 45, 47, 48, 51, 52, 54, 60, 66, 68, 109, 124, 131

[228] Pinaki Sinha, Sharad Mehrotra, and Ramesh Jain. Summarization of personal photologs using multidimensional content and context. In *ICMR*, 2011. 88, 90, 96

[229] Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv*, 2017. 140

[230] Anders Søgaard, Sebastian Ruder, and Ivan Vulić. On the limitations of unsupervised bilingual dictionary induction. *ACL*, 2018. 140, 147

[231] Yale Song, Louis-Philippe Morency, and Randall Davis. Action recognition by hierarchical sequence summarization. In *CVPR*, 2013. 60

[232] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In *CRCV-TR-12-01*, 2012. 8, 9, 10, 23, 42, 43, 66

[233] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using LSTMs. *ICML*, 2015. 60, 125

[234] Yuanhang Su, Kai Fan, Nguyen Bach, C-C Jay Kuo, and Fei Huang. Unsupervised multi-modal neural machine translation. In *CVPR*, 2019. 140

[235] Chen Sun and Ram Nevatia. Active: Activity concept transitions in video event classification. *ICCV*, 2013. 60

[236] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E. Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *ICCV*, 2015. 60

[237] Ilya Sutskever, James Martens, and Geoffrey E Hinton. Generating text with recurrent neural networks. In *ICML*, 2011. 89

[238] Kevin Tang, Li Fei-Fei, and Daphne Koller. Learning latent temporal structure for complex event detection. In *CVPR*, 2012. 60

[239] Graham W. Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *ECCV*, 2010. 60

[240] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. The new data and new challenges in multimedia research. *arXiv:1503.01817*, 2015. 92

[241] Simon Thorpe, Denis Fize, Catherine Marlot, et al. Speed of processing in the human visual system. *Nature*, 1996. 21

[242] Atousa Torabi, Christopher Pal, Hugo Larochelle, and Aaron Courville. Using descriptive video services to create a large data source for video annotation research. *arXiv:1503.01070*, 2015. 9

[243] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR*, 2011. 55

[244] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. In *ICCV*, 2015. 16, 58, 60, 69

[245] Ba Tu Truong and Svetha Venkatesh. Video abstraction: A systematic review and classification. *TOMCCAP*, 2007. 88

[246] Kathleen Tuite, Noah Snavely, Dun-yu Hsiao, Nadine Tabing, and Zoran Popovic. Photocity: training experts at large-scale image acquisition through a competitive game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2011. 9

[247] Pavan Turaga, Rama Chellappa, Venkatramana S Subrahmanian, and Octavian Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 2008. 55

[248] Naonori Ueda and Kazumi Saito. Parametric mixture models for multi-labeled text. In *NeurIPS*, 2002. 22

[249] Ranjith Unnikrishnan, Caroline Pantofaru, and Martial Hebert. Toward objective evaluation of image segmentation algorithms. *PAMI*, 2007. 55

[250] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008. 12

[251] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *ICCV*, 2015. 19, 86

[252] Sudheendra Vijayanarasimhan and Kristen Grauman. Active frame selection for label propagation in videos. In *ECCV*, 2012. 23

[253] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv*, 2017. 109

[254] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 86, 140

[255] Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *SIGCHI conference on Human factors in computing systems*, 2004. 21, 24

[256] Luis Von Ahn, Ruoran Liu, and Manuel Blum. Peekaboom: A game for locating objects in images. In *SIGCHI Conference on Human Factors in Computing Systems*, 2006. 21

[257] Carl Vondrick, Deniz Oktay, Hamed Pirsiavash, and Antonio Torralba. Predicting motivations of actions by leveraging text. In *CVPR*, 2016. 60

[258] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *IJCV*, 2013. 23

[259] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Anticipating visual representations from unlabeled video. In *CVPR*, 2016. 60

[260] Carl Vondrick and Deva Ramanan. Video Annotation and Tracking with Active Learning. In *NeurIPS*, 2011. 23

[261] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011. 113

[262] Dingding Wang, Tao Li, and Mitsunori Ogihara. Generating pictorial storylines via minimum-weight connected dominating set approximation in multi-view graphs. In *AAAI*, 2012. 88

[263] Heng Wang, Alexander Kläser, Cordelia Schmid, and Liu Cheng-Lin. Action recognition by dense trajectories. In *CVPR*, 2011. 110, 115

[264] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. 15, 16, 43, 44, 50, 54, 58, 60, 69

[265] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014. 125

[266] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015. 60, 110

[267] Miao Wang, Guo-Ye Yang, Jin-Kun Lin, Ariel Shamir, Song-Hai Zhang, Shao-Ping Lu, and Shi-Min Hu. Deep online video stabilization. *arXiv*, 2018. 110

[268] Weiran Wang, Raman Arora, Karen Livescu, and Jeff Bilmes. On deep multi-view representation learning. In *ICML*, 2015. 125

[269] Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. Actions ~ transformations. In *CVPR*, 2016. 60

[270] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 109, 110, 113, 115, 118, 119

[271] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 125, 128

[272] Xin Wang, Jiawei Wu, Junkun Chen, Lei Li, Yuan-Fang Wang, and William Yang Wang. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. *arXiv*, 2019. 140

[273] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *CVPR*, 2016. 53

[274] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer vision and image understanding*, 2011. 55, 60, 109, 124

[275] Philippe Weinzaepfel, Xavier Martin, and Cordelia Schmid. Towards weakly-supervised action localization. *arXiv:1605.05197*, 2016. 58, 66

[276] Paul J Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1988. 90

[277] Ronald J Williams and David Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. *Back-propagation: Theory, architectures and applications*, 1995. 90

[278] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross B. Girshick. Long-term feature banks for detailed video understanding. *arXiv*, 2018. 109

[279] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva. Sun database: Exploring a large collection of scene categories. *IJCV*, 2014. 25

[280] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *ACL*, 2015. 143, 147

[281] Bo Xiong, Gunhee Kim, and Leonid Sigal. Storyline representation of egocentric videos with an applications to story-based search. In *ICCV*, 2015. 87, 88

[282] Caiming Xiong, Stephen Merity, and Richard Socher. Dynamic memory networks for visual and textual question answering. *arXiv:1603.01417*, 2016. 86

[283] Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv:1502.03044*, 2015. 86

[284] Zhongwen Xu, Yi Yang, and Alexander G. Hauptmann. A discriminative cnn video representation for event detection. *CVPR*, 2015. 60, 140

[285] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. *arXiv:1511.02274*, 2015. 86

[286] Mark Yatskar, Luke Zettlemoyer, and Ali Farhadi. Situation recognition: Visual semantic role labeling for image understanding. *CVPR*, 2016. 59, 60

[287] Guangnan Ye, Yitong Li, Hongliang Xu, Dong Liu, and Shih-Fu Chang. EventNet: A large scale structured concept library for complex event detection in video. In *ACM International Conference on Multimedia*, 2015. 23

[288] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *arXiv:1507.05738*, 2015. 23, 42, 43, 53, 55, 58, 66

[289] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. *arXiv:1511.06984*, 2015. 60

[290] Ryo Yonetani, Kris M Kitani, and Yoichi Sato. Ego-surfing first person videos. In *CVPR*, 2015. 125

[291] Shoou-I Yu, Lu Jiang, and Alexander Hauptmann. Instructional videos for unsupervised harvesting and learning of action examples. In *ACM*, 2014. 140

[292] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015. 44, 54, 60, 67, 68

[293] Jenny Yuen, B. Russell, Ce Liu, and A. Torralba. LabelMe video: Building a video database with human annotations. In *ICCV*, 2009. 23

[294] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. 125

[295] Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. *ECCV*, 2014. 55

[296] Ziyu Zhang, Sanja Fidler, and Raquel Urtasun. Instance-level segmentation with deep densely connected mrfs. In *CVPR*, 2016. 59, 60, 61

[297] Yu Zhong, Walter S Lasecki, et al. RegionSpeak: Quick comprehensive spatial descriptions of complex images for blind users. In *SIGCHI Conference on Human Factors in Computing Systems*, 2015. 24

[298] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *ICLR*, 2015. 51

[299] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NeurIPS*, 2014. 7

[300] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *CVPR*, 2016. 86, 140

[301] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv:1506.06724*, 2015. 86

[302] George Kingsley Zipf. The psycho-biology of language. 1935. 12

[303] C Zitnick and Devi Parikh. Bringing semantics into focus using visual abstraction. In *CVPR*, 2013. 10