VISUAL LEARNING WITH WEAKLY LABELED VIDEO

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Kevin Tang
May 2015

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Fei-Fei Li, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Daphne Koller, Co-Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Silvio Savarese

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

With the rising popularity of Internet photo and video sharing sites like Flickr, Instagram, and YouTube, there is a large amount of visual data uploaded to the Internet on a daily basis. In addition to pixels, these images and videos are often tagged with the visual concepts and activities they contain, leading to a natural source of weakly labeled visual data, in which we aren't told where within the images and videos these concepts or activities occur. By developing methods that can effectively utilize weakly labeled visual data for tasks that have traditionally required clean data with laborious annotations, we can take advantage of the abundance and diversity of visual data on the Internet.

In the first part of this thesis, we consider the problem of complex event recognition in weakly labeled video. In weakly labeled videos, it is often the case that the complex events we are interested in are not temporally localized, and the videos contain varying amounts of contextual or unrelated segments. In addition, the complex events themselves often vary significantly in the actions they consist of, as well as the sequences in which they occur. To address this, we formulate a flexible, discriminative model that is able to learn the latent temporal structure of complex events from weakly labeled videos, resulting in a better understanding of the complex events and improved recognition performance.

The second part of this thesis tackles the problem of object localization in weakly labeled video. Towards this end, we focus on several aspects of the object localization problem. First, using object detectors trained from images, we formulate a method for adapting these detectors to work well in video data by discovering and adapting them to examples automatically extracted from weakly labeled videos. Then, we explore separately the use of large amounts of negative and positive weakly labeled visual data for object localization. With only negative weakly labeled videos that do not contain a particular visual concept, we show how a very simple metric allows us to perform distributed object segmentation in potentially noisy, weakly labeled videos. With only positive weakly labeled images and videos that share a common visual concept, we show how we can leverage correspondence information between images and videos to identify and detect the common object.

Lastly, we consider the problem of learning temporal embeddings from weakly labeled video. Using the implicit weak label that videos are sequences of temporally and semantically coherent images, we learn temporal embeddings for frames of video by associating frames with the temporal context that they appear in. These embeddings are able to capture semantic context, which results in better performance for a wide variety of standard tasks in video.

Acknowledgements

I am truly honored and privileged to have Fei-Fei Li and Daphne Koller as my advisors.

Fei-Fei is one the most inspiring role models I have ever met, both in and out of research. Under her guidance, I learned the most important skills of my PhD and career beyond: working on the right problems, devising effective solutions, and presenting my work. I could not have asked for a better advisor than Fei-Fei, who supported me in everything I wanted to work on, and helped me to strike a balance between my interests and the important problems in computer vision. Her genuine care for her students both in and out of research has made my years during PhD one of the most enjoyable periods of my life.

Daphne is without a doubt one of the smartest people I have ever met, and I am extremely fortunate to have had the opportunity to work with her during my PhD. Taking Daphne's class on graphical models and working with her on my first few papers really helped to shape my thought process when it comes to research, turned me into an independent researcher, and defined the direction for my thesis. Daphne's deep technical knowledge, keen insights and intuitions, along with her meticulous and thorough nature have really set the bar for me and influenced my approach to research.

I am also deeply indebted to my thesis committee members. I would like to thank Silvio Savarese, who has offered insightful discussions and constructive suggestions for my research, which have really helped to refine the direction of my thesis. I would also like to thank Percy Liang, who has taught me to think much more theoretically about my work, and asked many thought-provoking questions that have inspired my research. Finally, I would like to thank Michael Frank for chairing my oral examination and providing a different perspective on my research.

I would not be the person I am today without guidance from some of the most amazing mentors. From my undergraduate years - Dhruv Batra, Tsuhan Chen, Pedro Felzenszwalb, Saikat Guha, Paul Francis, Robert Keller, Adarsh Kowdle, Noah Snavely, Rahul Sukthankar, Marshall Tappen. In particular, I would like to thank Rahul Sukthankar and Marshall Tappen for introducing me to the field of computer vision and giving me invaluable advice throughout my journey. From my graduate years - Lubomir Bourdev, Rob Fergus, Tianshi Gao, Armand Joulin, Jia Li, Greg Mori, Andrew Ng, Juan Carlos Niebles, Manohar Paluri, Rahul Sukthankar, Bangpeng Yao, Will Zou. In particular, I would like to thank Tianshi Gao, who patiently guided me through my first and most difficult few years at Stanford, and Armand Joulin, who completely changed my appreciation and perspective on research.

I would also like to thank the wonderful colleagues I have had the great fortune of interacting with in the Stanford Vision Lab and DAGS - Alexandre Alahi, Chris Baldassano, Lamberto Ballan, Jia Deng, Alireza Fathi, Chuan-Sheng Foo, Tianshi Gao, Timnit Gebru, Michelle Greene, Catalin Iordan, Justin Johnson, Armand Joulin, Irene Kaplow, Andrej Karpathy, Jon Krause, Ranjay Krishna, Jia Li, Juan Carlos Niebles, Guido Pusiol, Vignesh Ramanathan, Olga Russakovsky, Michael Stark, Huayan Wang, Bangpeng Yao, Serena Yeung, Yuke Zhu.

Last but not least, I would like to thank my family. To my father, Jeffrey Tang, for being my role model and giving me the advice and inspiration to always reach higher. To my mother, Haidi Tang, for her unconditional love and support and raising me to become who I am and where I am today. To my sister, Emily Tang, for being my closest friend and always there for me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Following recent advances in computer vision and machine learning, tasks in video understanding have shifted from classifying simple motions and actions [42, 127] to understanding complex events and activities in complex Internet videos [100, 106, 141]. Understanding complex events is a difficult task, requiring probabilistic models and video representations that can reason about and understand the temporal semantics of what is occurring in the video. In addition, because many events are characterized by key objects and their interactions, it is imperative to have robust methods for object localization that can provide accurate spatial localization information as a building block for upstream models.

However, for the most part, the annotations required to train such types of models for difficult tasks such as event recognition [106] and object localization [26, 33] are extremely expensive to obtain, especially in the case of video data. In video, not only are annotators required to provide spatial localization of visual concepts, but they must also provide temporal localization as well. This provides an additional degree of difficulty compared to images, and a significant investment of time. In addition, it has been shown that due to domain differences between image and video data [143], simply applying models trained from images directly to video does not work well.

To circumvent the challenges associated with obtaining expensive spatial and temporal annotations in video data, this thesis focuses on developing methods that are able to learn from weakly labeled video data. In weakly labeled video data, annotations for visual concepts such as events or objects are given at the video level. For example, whereas previous works in object localization required training data with bounding boxes drawn around objects or pixel segmentations of the objects, we only require labels indicating the presence or absence of an object in a video.

The major advantage to this type of data is that it is cheap and easy to obtain. In particular, Internet photo and video sharing sites like Flickr, Instagram, and YouTube contain large amounts of visual data tagged by users with the visual concepts and activities they contain, leading to a natural source of weakly labeled visual data. By developing methods that are able to take advantage of weakly labeled video data, we are able to directly utilize this wealth of data as training data for our algorithms. In this thesis, we address the standard tasks of video classification, object detection, object segmentation, and video representation using only weakly labeled video data.

1.2 Thesis outline

In Chapter 2, we consider the problem of complex event recognition in weakly labeled video. In Chapters 3, 4, 5, and 6, we consider the problem of object localization in weakly labeled images and video. In Chapter 7, we consider the problem of learning temporal embeddings from weakly labeled video. We summarize and highlight the contributions of each chapter below.

Chapter 2 - Learning Latent Temporal Structure for Complex Event Detection. We address the problem of understanding the temporal structure of complex events in highly varying videos obtained from the Internet. Towards this goal, we utilize a conditional model trained in a max-margin framework that is able to automatically discover discriminative and interesting segments of video, while simultaneously achieving competitive accuracies on difficult detection and recognition tasks. We introduce latent variables over the frames of a video, and allow our algorithm to discover and assign sequences of states that are most discriminative for the event. Our model is based on the variable-duration hidden Markov model, and models durations of states in addition to the transitions between states. The simplicity of our model allows us to perform fast, exact inference using dynamic programming, which is extremely important when we set our sights on being able to process a very large number of videos quickly and efficiently.

Chapter 3 - Shifting Weights: Adapting Object Detectors from Image to Video. Typical object detectors trained on images perform poorly on video, as there is a clear distinction in domain between the two types of data. In this chapter, we tackle the problem of adapting object detectors learned from images to work well on videos. We treat the problem as one of unsupervised domain adaptation, in which we are given labeled data from the source domain (image), but only unlabeled data from the target domain (video). Our approach, self-paced domain adaptation, seeks to iteratively adapt the detector by re-training the detector with automatically discovered target domain examples, starting with the easiest first. At each iteration, the algorithm adapts by considering an increased number of target domain examples, and a decreased number of source domain examples. To discover target domain examples from the vast amount of video data, we introduce a simple, robust approach that scores trajectory tracks instead of bounding boxes. We also show how rich and expressive features specific to the target domain can be incorporated under the same framework.

Chapter 4 - Discriminative Segment Annotation in Weakly Labeled Video. The ubiquitous availability of Internet video offers the exciting opportunity to directly learn localized visual concepts from real-world imagery. Unfortunately, most such attempts are doomed because traditional approaches are ill-suited, both in terms of their computational characteristics and their inability to robustly contend with the label noise that plagues uncurated Internet content. We present CRANE, a weakly supervised algorithm that is specifically designed to learn under such conditions. First, we exploit the asymmetric availability of real-world training data, where small numbers of positive videos tagged with the concept are supplemented with large quantities of unreliable negative data. Second, we ensure that CRANE is robust to label noise, both in terms of tagged videos that fail to contain the concept as well as occasional negative videos that do. Finally, CRANE is highly parallelizable, making it practical to deploy at large scale without sacrificing the quality of the learned solution.

Chapter 5 - Co-localization I: Real-World Images. In this chapter, we address the problem of co-localization in real-world images. Co-localization is the problem of simultaneously localizing (with bounding boxes) objects of the same class across a set of distinct images. Although similar problems such as co-segmentation and weakly supervised localization have been previously studied, we focus on being able to perform co-localization in real-world settings, which are typically characterized by large amounts of intraclass variation, inter-class diversity, and annotation noise. To address these issues, we present a joint image-box formulation for solving the co-localization problem, and show how it can be relaxed to a convex quadratic program which can be efficiently solved.

Chapter 6 - Co-localization II: Efficient Image and Video. In this chapter, we address the problem of performing efficient co-localization in images and videos. Building upon our work in the previous chapter, we show how we are able to naturally incorporate temporal terms and constraints for video co-localization into a quadratic programming framework. Furthermore, by leveraging the Frank-Wolfe algorithm (or conditional gradient), we show how our optimization formulations for both images and videos can be reduced to solving a succession of simple integer programs, leading to increased efficiency in both memory and speed.

Chapter 7 - Learning Temporal Embeddings for Complex Video Analysis. In this chapter, we show how to learn temporal embeddings of video frames using large amounts of unlabeled video data, which can be easily obtained from the

Internet. The key idea is to extend the distributed word vector representations commonly used in the language community into the visual space. In the video analogy, sentences are complete videos, and words are frames within each video. We propose three different ways of incorporating contextual information in video data, and comprehensively evaluate various design decisions for learning temporal embeddings. We show improvements on standard video tasks such as retrieval and classification, and also qualitative results to visualize and illustrate various applications of our embeddings.

1.3 Previously published work

Most contributions in this dissertation have first appeared as various publications. These publications are: [141] (Chapter 2), [143] (Chapter 3), [144] (Chapter 4), [142] (Chapter 5), [66] (Chapter 6). I have also worked on several other publications during my PhD [145]. However, they are beyond the scope of this dissertation, and therefore not discussed in detail here.

Chapter 2

Learning Latent Temporal Structure for Complex Event Detection

2.1 Introduction

With the advent of Internet video hosting sites such as YouTube, personal Internet videos are now becoming extremely popular. There are numerous challenges associated with the understanding of these types of videos; we focus on the task of complex event detection. In our problem definition, we are given Internet videos labeled with an event class, where the label specifies the complex event that occurs within the video. This is a weakly-labeled setting, as we are not given *temporally localized* videos. This means that the event can occur anywhere within the video, and we do not have temporal segmentations that indicate the time points at which the event occurs. The *detection* aspect of our problem manifests itself at the video level, where in the testing phase, we are also given large numbers of irrelevant videos, and must *detect* videos that correspond to events of interest. This is in contrast to the typical detection task of localizing the event within the video.

Of the difficulties presented by Internet videos, we focus on two points that have been largely ignored by recent computer vision algorithms. First, there is a large

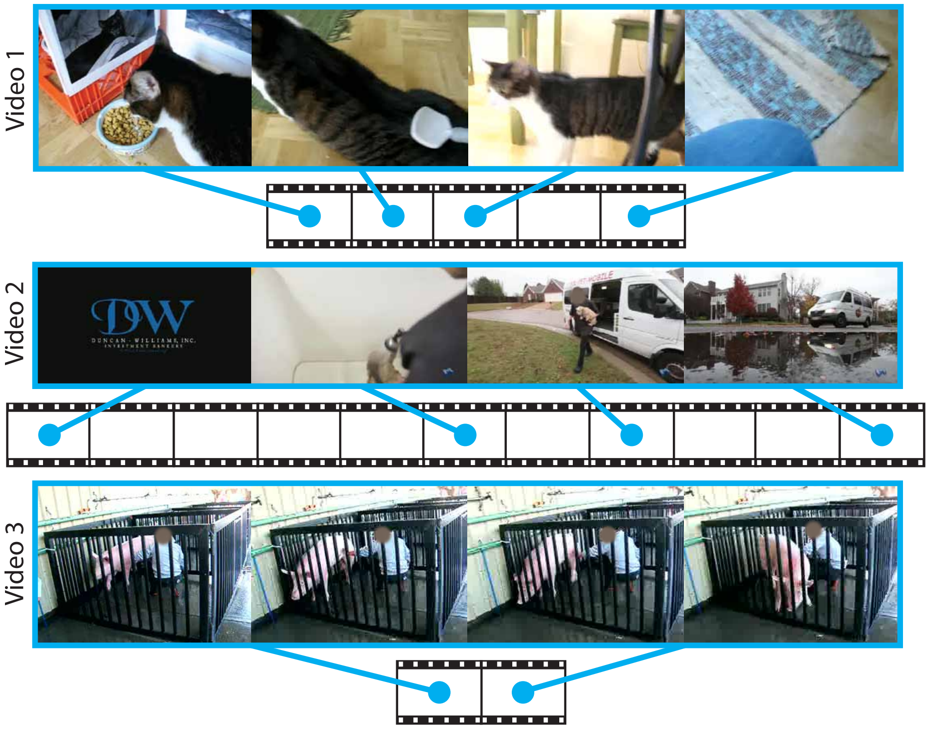


Figure 2.1: Examples of Internet videos for the event of "Grooming an animal" from the TRECVID MED dataset [106] that illustrate the variance in video length and temporal localization of the event.

number of videos available on the Internet, creating the need for algorithms that are able to efficiently index and process this wealth of data. Secondly, there is a large amount of variance in these videos, ranging from differences in low-level processing such as length and resolution, to high-level concepts such as activities, events, and contextual information. In addition, there is high intra-class variance when trying to assign class labels to these types of videos, as more often than not the videos are not temporally localized, and will contain varying amounts of contextual or unrelated segments.

These points have not been addressed by much of the recent research on activity recognition and event detection [42, 127]. Although some of the recent works have considered Internet videos, complex activity recognition tasks are typically already temporally localized [91, 100], and event detection tasks focus only on localizing well-defined primitive events [68]. In addition, few of these works deal with large-scale classification.

In order to successfully classify these types of videos, we formulate a model over the temporal domain that is able to discriminatively learn the transitions between events of interest, as well as the durations of these events. We reiterate the challenges associated with complex event detection in Internet videos and highlight key contributions of our model that address these issues:

Extremely large number of difficult videos. Using dynamic programming, our model is able to perform efficient, exact inference, and our max-margin learning framework is based on the linear kernel Support Vector Machine (SVM), which can be optimized very quickly using LIBLINEAR [34]. Together, the inference and learning procedures allow us to process large numbers of videos very quickly. Also, the discriminative nature of our learning enables us to obtain competitive classification results on difficult datasets.

Large amounts of variation in video length. Several previous methods that attempt to model temporal structure assume a video to be of normalized length [83, 100]. However, this is an unrealistic assumption, as the frame rates of the videos are generally on the same scale. Regardless of the duration of a video, a simple motion should still occupy the same number of frames. Our model is able to account for this by representing videos as sequences of fixed length temporal segments.

Weakly-labeled complex events that are not temporally localized. Our model is flexible and allows for sequenced states of interest to transition and occur anywhere within a video, which is crucial for the weakly-labeled setting. The appearance, transitions, and durations of these states are automatically learned with only a class label for the video. In addition, the states can also correspond to semantically meaningful concepts, such as distinguishing between sequences of frames that are relevant and irrelevant for an event of interest.

In summary, the contributions of this chapter are two-fold. First, we identify several challenges and difficulties associated with complex event detection in Internet videos, a task of growing importance. And secondly, we formulate a discriminative model that is able to address these issues, and show promising results on difficult datasets.

2.2 Related Work

We review related work on Hidden semi-Markov Models (HSMMs), Conditional Random Fields (CRFs), and discriminative temporal segments in the context of video, and refer the reader to a recent survey in the area by Turaga *et al.* [150] for a comprehensive review.

HSMMs [32, 52, 96], CRFs [114, 136], semi-CRFs [125], and similar probabilistic frameworks [1] have been previously used to model the temporal structure of videos and text. However, these works differ from ours in that they are applied to different domains such as surveillance video and gesture recognition, and typically require the states to not be latent in order for the models to work. In addition, many of these models were not formulated with large-scale classification in mind, and have complex inference procedures.

Most similar to our method are recent works in video that learn discriminative models over temporal segments [83, 98, 100, 126]. Satkin & Hebert [126] and Nguyen *et al.* [98] attempt to discover the most discriminative portions or segments of videos. Laptev *et al.* [83] divide videos into rigid spatio-temporal bins and compute separate feature histograms from each bin to capture a rough temporal ordering of features. Niebles *et al.* [100] represent videos as temporal compositions of motion segments, and learn appearance models for each of the segments. Their model is tree structured, and assumes fixed anchors for each motion segment, penalizing segments that occur at a distance from their anchors. Our work is different from these previous methods in that in addition to discovering discriminative segments of video, we also model and learn the transitions between and durations of these segments with a chain structured model. Whereas [100] heuristically fixes the anchor points and durations of their temporal segments before training, our approach is completely model-based, and learns all parameters for our transition and duration distributions. There has also been a separate line of work that seeks to model temporal segments of video with the use of additional annotations [38, 50], which we do not require.

Drawing upon recent successes in the field, our model leverages the Bag-of-Words (BoW) feature representation and max-margin learning. Advances in feature representations have utilized the BoW model with discriminative classifiers to achieve state-of-the-art results on popular video datasets [73, 157]. The representation has also been successfully used with semi-latent topic models [161] and unsupervised generative models [101]. We learn parameters for our model using the max-margin framework, which has recently become very popular for latent variable models through the introduction of general learning frameworks [35, 171].

2.3 Our Model

Our model for videos is the conditional variant of the variable-duration hidden Markov model (HMM), also referred to as an explicit-duration HMM or a hidden semi-Markov model [32, 96]. We start by introducing our representation for videos, then give intuition for our model by briefly describing the variable-duration HMM.

2.3.1 Video representation

Given a video, we first divide it into temporal segments of fixed length l_{seg}, which can be seen in Figure 2.2. By using fixed length segments, we are able to capture the fact that simple motions should occupy similar numbers of frames, and are invariant to the total length of the video. With this division into segments, a video can be represented by n segments, where the number of segments n is proportional to the video length. For each temporal segment i, we then compute BoW histograms $\boldsymbol{x}_i$ over the features in each segment, and treat these histograms as the observed input variables of our temporal model.

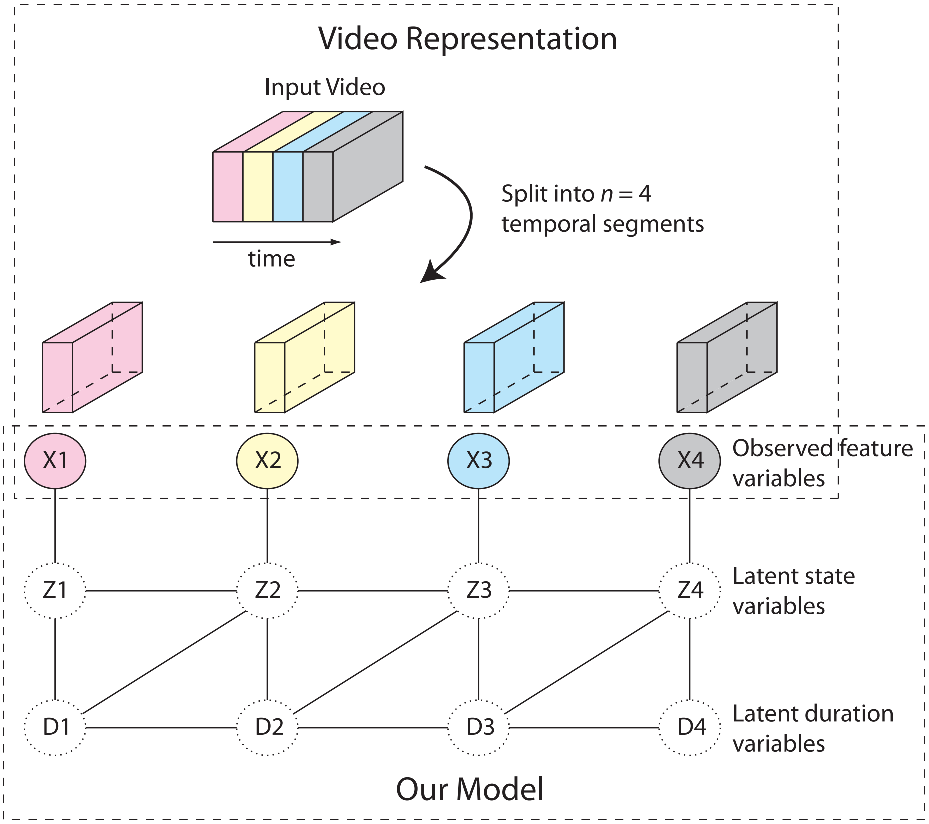


Figure 2.2: Given an input video, our algorithm divides it into temporal segments and builds a structured temporal model on top of the features.

2.3.2 Variable-duration HMM

A traditional approach is to use an HMM to model transitions between states of a video. However, the HMM suffers because it imposes a geometric distribution on the time within a state, which results when a state continuously transitions to itself. To address this, we use the variable-duration HMM, which allows each state to emit a sequence of observations. This means that we must also model the duration of a state, since a state can generate multiple observations before transitioning into another state. We choose to model the duration of a state using a multinomial distribution. The variable-duration HMM is much more appropriate for our application, since we expect a single state to generate several temporal segments of video that are linked together to form a single, coherent action or event. Our hope is that the latent states and their durations will be able to capture semantically meaningful and discriminative concepts that are shared amongst the videos, as in Figure 2.3. Note that by restricting the states to have a duration of one, we obtain the standard HMM as a specific instance of the variable-duration HMM.

The conditional variant of the variable-duration HMM is similar to a hidden chain CRF [114]. The difference is in the duration variables, which form an additional chain structure beneath the hidden chain CRF as seen in Figure 2.2. Since all the v-structures in the conditional variant are moralized, the independencies of the two models are equivalent. Mapping the model onto our video representation, we introduce a latent state for each temporal segment of a video as shown in Figure 2.2. Since these are latent variables, we are not given labels for them during training or testing.

2.3.3 Model representation

In our model, there are three types of potentials that define the energy of a particular sequence assignment to the latent state variables $\boldsymbol{z} = \{z_1, z_2, \dots, z_n\}$ and duration variables $\boldsymbol{d} = \{d_1, d_2, \dots, d_n\}$ as shown in Figure 2.2. Intuitively, the duration variable acts as a counter, and decreases after each consecutive state assignment until it reaches zero, after which a new state transition can be made. While it is counting down, the state assignment is not allowed to change. We assume that we are given the maximum duration $d_{\max}$ for all states and the number of states S for our model. The potentials are defined in terms of parameters $\boldsymbol{w}$ of our model that will be learned.

The first potential is a singleton appearance potential on the latent state variables that measures the similarity of the feature histogram $\boldsymbol{x}_i$ for temporal segment i to its assigned state z_i.

$$\psi^a(Z_i = z_i) = \boldsymbol{w}^a_{z_i} \cdot \boldsymbol{x}_i \tag{2.1}$$

The second potential encompasses both the state and duration variables, and measures the score of transitioning between states, provided we are allowed to transition:

$$\begin{aligned}\psi^t(Z_i =& z_i, Z_{i-1} = z_{i-1}, D_{i-1} = d_{i-1}) = \\ &- \infty \cdot \mathbf{1}[d_{i-1} > 0, z_i \neq z_{i-1}] \\ &+ w^t_{z_{i-1},z_i} \cdot \mathbf{1}[d_{i-1} = 0]\end{aligned} \tag{2.2}$$

The third potential measures the score of a given duration, provided we are entering a new state:

$$\begin{aligned}\psi^d(Z_i =& z_i, D_i = d_i, D_{i-1} = d_{i-1}) = \\ &- \infty \cdot \mathbf{1}[d_{i-1} > 0, d_i \neq d_{i-1} - 1] \\ &+ w^d_{z_i,d_i} \cdot \mathbf{1}[d_{i-1} = 0]\end{aligned} \tag{2.3}$$

Together, these potentials define the energy of a particular sequence assignment of variables $\boldsymbol{z}$ and $\boldsymbol{d}$ to our model:

$$\begin{aligned}E(\boldsymbol{z}, \boldsymbol{d}|\boldsymbol{w}) =& \sum_i (\psi^a(Z_i = z_i) \\ &+\psi^t(Z_i = z_i, Z_{i-1} = z_{i-1}, D_{i-1} = d_{i-1}) \\ &+\psi^d(Z_i = z_i, D_i = d_i, D_{i-1} = d_{i-1}))\end{aligned} \tag{2.4}$$

where we initialize $\psi^t(Z_1, Z_0, D_0) = 0$ and $D_0 = 0$.

2.4 Inference

Exact maximum a posteriori (MAP) inference for our model can be done efficiently using dynamic programming. In MAP inference, we must find the sequence of states $\boldsymbol{z}$ and durations $\boldsymbol{d}$ that maximize the energy function given above in equation 2.4. This can be done using a recurrence relation that computes the best possible score given that temporal segment j is assigned to state i. The score is computed by searching

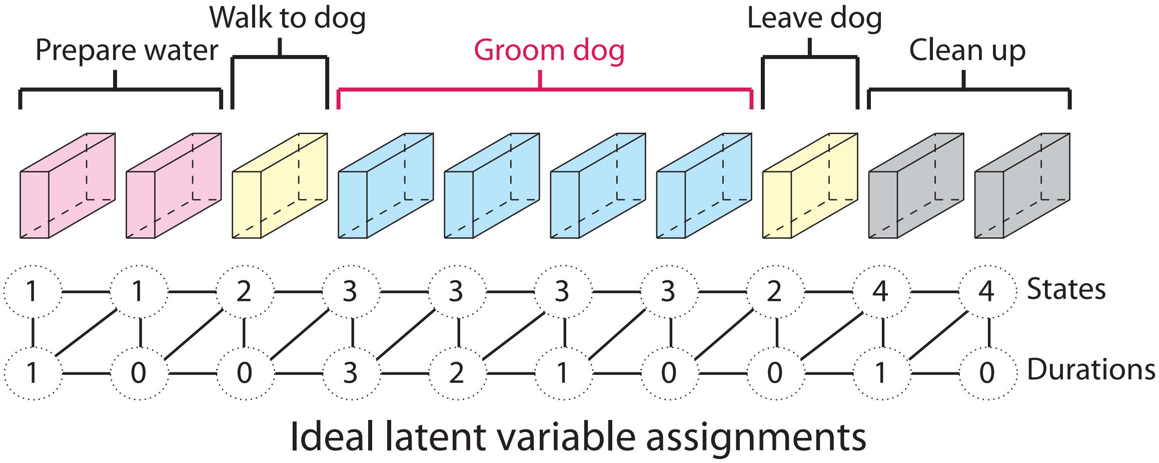


Figure 2.3: Ideal assignments to latent states and durations for a sequence with a known temporal segmentation.

over all possible durations d and previous states s, assuming that segment j is the last segment in the duration of state i. We can use the following recurrence relation for inference:

$$V_{i,j} = \max_{\substack{d \in \{1 \ldots d_{\max}\} \\ s \in \{1 \ldots S\}}} [\boldsymbol{w}_i^a \cdot (\sum_{k=j-d+1}^{j} \boldsymbol{x}_k) \\ + w_{s,i}^t + w_{i,d}^d + V_{s,j-d}] \tag{2.5}$$

After building up the table of scores V, we can then recover the optimal assignments by backtracking through the table. The runtime complexity for this inference algorithm is $O(n_{\max} d_{\max} S^2)$, where $n_{\max}$ is the maximum number of temporal segments in all videos. By utilizing structure in the duration variables, our inference algorithm achieves a complexity that is linear in $d_{\max}$, whereas a naive implementation would have quadratic dependence.

2.5 Learning

There are three sets of parameters that we must learn in our model, the appearance parameters $\boldsymbol{w}^a$, the transition parameters $\boldsymbol{w}^t$, and the duration parameters $\boldsymbol{w}^d$, which we can concatenate into a single weight vector:

$$\boldsymbol{w} = [\boldsymbol{w}^a \quad \boldsymbol{w}^t \quad \boldsymbol{w}^d] \tag{2.6}$$

Given a training set of N videos and their corresponding binary class labels $y_i \in \{-1, 1\}$, we can compute their feature representations to obtain our dataset $(\langle v_1, y_1\rangle, ..., \langle v_N, y_N\rangle)$. To learn our parameters, we adopt the binary Latent SVM framework of Felzenszwalb *et al.* [35], which is a specific instance of the Latent Structural SVM with a hinge loss function [171]. The objective we would like to minimize is given by:

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \max(0, 1 - y_i f_{\boldsymbol{w}}(v_i)) \tag{2.7}$$

where we consider linear classifiers of the form:

$$f_{\boldsymbol{w}}(v) = \max_{\boldsymbol{h}} \boldsymbol{w} \cdot \Phi(v, \boldsymbol{h}) \tag{2.8}$$

The latent variables $\boldsymbol{h}$ in the classifier are solved for by performing MAP inference on the example v to find the state and duration assignments. Using these assignments, we can construct the feature vector $\Phi(v, \boldsymbol{h})$ for an example v as follows. For the $\boldsymbol{w}^a$ parameters we sum the feature histograms that are assigned to each state, and for the $\boldsymbol{w}^t$ and $\boldsymbol{w}^d$ parameters we count the number of times each state transition and duration occurs. We then normalize each of these features and concatenate them together to form the feature vector $\Phi(v, \boldsymbol{h})$.

The objective function is minimized using CCCP [173]. This leads to an iterative algorithm in which we alternate between inferring the latent variables $\boldsymbol{h}$, and optimizing the weight vector $\boldsymbol{w}$. Once the latent variables are inferred and the feature vectors $\Phi(v, \boldsymbol{h})$ are constructed for each example, optimizing the weight vector becomes the standard linear kernel SVM problem, which can be solved very efficiently using LIBLINEAR [34]. This process is repeated for several iterations until convergence or a maximum number of iterations is reached.

2.5.1 Initialization

In our model, we must initialize the latent states of the temporal segments as well as their durations for each of our training examples, subject to the constraint that we have S states we can assign and a maximum duration $d_{\max}$. For each video, we begin by initializing each segment to its own state. Then, we use Hierarchical Agglomerative Clustering to merge adjacent segments. This is done by computing the Euclidean distance between feature histograms of all adjacent segments, and repeatedly merging segments with the shortest distance. The number of merges for a given video is fixed to be half the number of segments in the video.

Then, using all the videos, we run k-means clustering to cluster all the states into S clusters, and assign latent states according to their cluster assignments. This gives us the assignments $\boldsymbol{z}$ for the states. We initialize the duration variables by assuming that all consecutive assignments of the same state are a single state assignment with duration equal to the number of consecutive assignments.

2.6 Experiments

We test our model on two difficult tasks: activity recognition and event detection. In both scenarios, we are only given class labels for the videos. We use the Olympic Sports dataset [100] and the 2011 TRECVID Multimedia Event Detection (MED) dataset [106]. For both datasets, we compare our model to state-of-the-art baselines that consider temporal structure, using the same features for all models.

In our experiments, we use 5-fold cross validation for model selection to select the number of latent states and the C parameter for the SVM. We set the maximum duration to be the average video length, and set the length of temporal segments based on the dataset and density of our sampled features. For the Olympic Sports

Sport Class	Niebles *et al.* [100]	Our Method
high-jump	**27.0%**	18.4%
long-jump	71.7%	**81.8%**
triple-jump	10.1%	**16.1%**
pole-vault	**90.8%**	84.9%
gymnastics-vault	**86.1%**	85.7%
shot-put	37.3%	**43.3%**
snatch	54.2%	**88.6%**
clean-jerk	70.6%	**78.2%**
javelin-throw	**85.0%**	79.5%
hammer-throw	**71.2%**	70.5%
discus-throw	47.3%	**48.9%**
diving-platform	**95.4%**	93.7%
diving-springboard	**84.3%**	79.3%
basketball-layup	82.1%	**85.5%**
bowling	53.0%	**64.3%**
tennis-serve	33.4%	**49.6%**
Mean AP	62.5%	**66.8%**

Table 2.1: Average Precision values for classification on the Olympic Sports dataset [100].

dataset, we used 20 frames per segment, and for the MED dataset, we used 100 frames per segment. We train a model for each class, and report average precision (AP) numbers on the datasets.

2.6.1 Activity recognition

Dataset. The Olympic Sports dataset [100] consists of 16 different sport classes of Olympic Sports activities that contain complex motions going beyond simple punctual or repetitive actions. The sequences are collected from YouTube, and class label annotations obtained using Amazon Mechanical Turk. An important point to note is that the sequences are already temporally localized.

Comparisons. We compare our model to the method of decomposable motion segments [100], which achieves state-of-the-art results using local features. Because much of their performance derives from including a BoW histogram over the entire video in their feature vector, we follow protocol and concatenate the BoW histogram to

Event Class	Chance	Niebles *et al.* [100]	Laptev *et al.* [83]	Our Method, $d_{\max} = 1$	Our Method
Attempting a board trick	1.18%	5.84%	8.22%	6.24%	**15.44%**
Feeding an animal	1.06%	2.28%	2.45%	**5.28%**	3.55%
Landing a fish	0.89%	9.18%	9.77%	7.30%	**14.02%**
Wedding ceremony	0.86%	7.26%	5.52%	9.48%	**15.09%**
Working on a woodworking project	0.93%	4.05%	4.09%	3.42%	**8.17%**
Mean AP	0.98%	5.72%	6.01%	6.34%	**11.25%**

Table 2.2: Average Precision values for detection on the MED DEV-T dataset.

Event Class	Chance	Niebles *et al.* [100]	Laptev *et al.* [83]	Our Method, $d_{\max} = 1$	Our Method
Birthday party	0.54%	2.25%	1.93%	1.97%	**4.38%**
Changing a vehicle tire	0.35%	0.76%	0.98%	**1.01%**	0.92%
Flash mob gathering	0.42%	8.30%	7.60%	7.58%	**15.29%**
Getting a vehicle unstuck	0.26%	1.95%	1.73%	1.82%	**2.04%**
Grooming an animal	0.25%	0.74%	0.72%	0.73%	**0.74%**
Making a sandwich	0.43%	**1.48%**	1.09%	0.80%	0.84%
Parade	0.58%	2.65%	3.77%	**4.17%**	4.03%
Parkour	0.32%	2.05%	1.95%	1.65%	**3.04%**
Repairing an appliance	0.27%	4.39%	1.54%	1.38%	**10.88%**
Working on a sewing project	0.26%	0.61%	1.18%	0.91%	**5.48%**
Mean AP	0.37%	2.52%	2.25%	2.20%	**4.77%**

Table 2.3: Average Precision values for detection on the MED DEV-O dataset.

the end of our feature vector $\Phi(v, \boldsymbol{h})$ before classification. For the feature representation, we use the same features used in [100], which consists of an interest point detector [82] and concatenated Histogram of Gradient (HOG) and Histogram of Flow (HOF) descriptors [83]. In addition, because [100] uses a χ^2-SVM, we use the method of additive kernels [152] to approximate a χ^2 kernel for our BoW features to maintain efficient processing while increasing discriminative power. Because the public release of this dataset is not the full dataset used in the paper [100], we obtained results for their model on the public release through personal communication with the authors. The results are given in Table 2.1.

Results. We obtain better AP numbers for 9 of the 16 classes, as well as better overall mean AP compared to the state-of-the-art baseline model. The promising performance on this dataset shows that, given well-localized videos, our model is able to capture the fine structure between temporal segments that define a complex activity.

Observing the latent states that our model learns, we find that there are three key components that allow us to do better than [100]. First, our model is flexible and allows latent states to appear anywhere within a sequence without penalty. In the "snatch" sequences, the assignment of the first latent state varies approximately equally between two different states. This helps to capture the variability that accompanies the start of a "snatch" sequence, such as differences in preparatory motions of the athletes. The baseline model is unable to easily account for this, as it has a fixed anchor for its segments, and so the beginning of each sequence is almost always modeled by the same segment. The second component is the effect of modeling the duration of the segments. For the same latent state, the durations of the state can vary greatly from sequence to sequence. In some cases, our model is able to realize that the sequence is extremely short and already very discriminative, and assigns the same state to the entire sequence. This is not allowed in the baseline model, as the lengths of the motion segments are pre-specified parameters. Finally, our model is able to discard unnecessary states and represent most of the sport classes with fewer than 3 states. The baseline model is optimally trained with 6 motion segments, and forces sequences into the temporal structure of its segments, causing the optimization to easily overfit.

We note that our model performs poorly in the "high-jump" and "triple-jump" classes. The reason for this can be attributed to the weak discriminative power of the features extracted from these videos. Visualizing the latent states learned for the "high-jump" class, we find that there are a large number of videos that are all assigned to a single state. This occurs because the underlying BoW histograms at the segment level are too similar, and so our model tends to group them together into a single duration. In addition, the number of videos is skewed for several of the classes, and "triple-jump" is one of the classes with fewer examples in both training and testing, which makes it hard for both discriminative models to learn meaningful parameters.

2.6.2 Event detection

Dataset. The 2011 TRECVID MED dataset [106] consists of a collection of Internet videos collected by the Linguistic Data Consortium from various Internet video hosting sites. There are 15 events, and they are split into two sets, the DEV-T set and the DEV-O set. The DEV-T set consists of the 5 events "Attempting a board trick", "Feeding an animal", "Landing a fish", "Wedding Ceremony", and "Working on a woodworking project". The DEV-O set consists of the 10 events "Birthday party", "Changing a vehicle tire", "Flash mob gathering", "Getting a vehicle unstuck", "Grooming an animal", "Making a sandwich", "Parade", "Parkour", "Repairing an appliance", and "Working on a sewing project".

The task, although termed event detection, is more similar to that of a retrieval task. We are given approximately 150 training videos for each event, and in the two testing sets for DEV-T and DEV-O, we are given large databases of videos that consist of both the events in the set as well as null videos that correspond to no event. The null videos significantly decrease the chance AP, causing our resulting numbers to be very low. There are a total of 10,723 videos in the DEV-T test set, and 32,061 videos in the DEV-O test set. In the TRECVID task, the DEV-T set is used for development, while the DEV-O set is used for evaluation. We consider the two sets separately, as it is stated that there may be unidentified positive videos of events from the DEV-T set in the DEV-O test set, and vice versa.

Comparisons. We compare our models to strong baseline methods that can capture temporal structure of local features through decomposable motion segments [100], and rigid spatio-temporal bins [83]. For the feature representation, we extract dense HOG3D features [73, 158], and use a linear kernel SVM for all models. To illustrate the effect of the duration variables, we also train a version of our model with the duration variable set to one, corresponding to a standard hidden chain CRF [114]. Results for the MED datasets are given in Table 2.2 and Table 2.3 for the DEV-T and DEV-O sets, respectively.

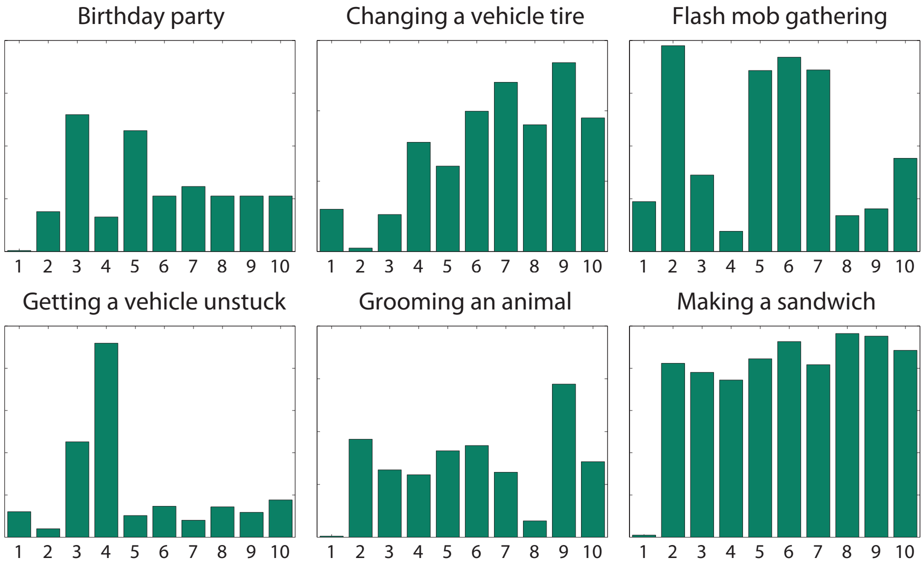


Figure 2.4: Examples of duration parameters learned for events in the MED dataset. The x-axes are values of the duration parameters, and the height of the bars represent the strength of the parameter, which is averaged over all states of the model.

Effect of duration variables. In a few rare cases, the hidden chain CRF is able to outperform our model by a small margin. This can occur because for some events, the videos that contain them vary between different types of motions very quickly, and so the duration variables will sometimes mistakenly merge these variations into a single state. In relation to the bias-variance tradeoff, the low variance and high bias of the hidden chain CRF allow it to generalize better for certain events. In theory, any model learned using the hidden chain CRF can be learned using our duration model as well, by learning large negative parameters for durations greater than one. However, this does not always occur as the duration variables are initialized to different values, and the inference procedures score assignments differently. On the other hand, the increased performance of the hidden chain CRF also speaks well for our model, as it shows that through better initializations and model selection techniques, it is possible to achieve even better accuracies.

Visualizing the parameters learned for the duration variables, we find that the duration variables are commonly utilized for states that correspond to the contextual

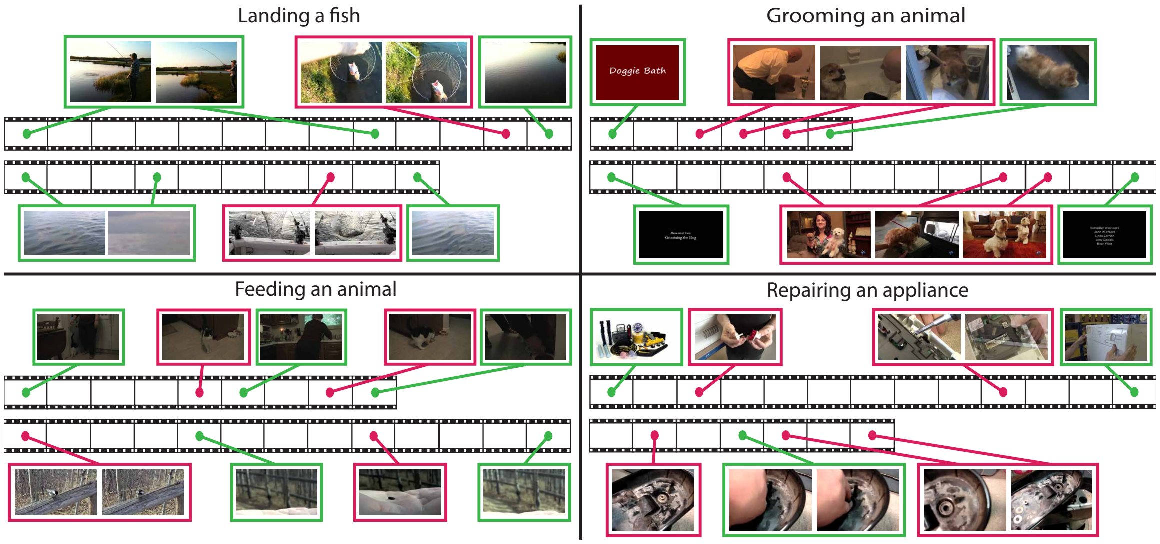


Figure 2.5: Example inference results on two different videos for four of our models learned on the MED dataset. The red and green boxes represent different latent states that are the same across the two videos, but different across models.

and irrelevant portions of videos, as they typically occupy large numbers of consecutive temporal segments. In Figure 2.4, we show examples of the multinomial duration parameters learned for events in the MED dataset. A hidden chain CRF that imposes a geometric distribution would have a large parameter for the duration of 1, and small parameters for all other durations. Our models learn duration parameters in favor of non-geometric distributions, which suggests that the videos are better modeled with state durations.

Results. Our model achieves the best results for both MED datasets, and achieves significant gains in AP for most of the events. Much of the analysis from the previous section on activity recognition holds for these datasets as well. By learning state assignments that can occur at any temporal location and by modeling their durations, our model is able to successfully capture the temporal structure of these highly varying Internet videos, as seen in Figure 2.5. These properties are crucial in MED videos, as events are not temporally localized and there is a large number of contextual segments that we must model. For example, in the "Feeding an animal" visualizations in Figure 2.5, discriminative segments occur at completely different points in time for the two videos. The fixed structure of the baseline models makes it unable for them to capture the varied temporal structure of these videos, as they treat segments at the same relative locations of two videos to be the same.

Latent semantic understanding. In addition to achieving competitive accuracies on difficult datasets, our model is also able to capture semantic concepts in the latent states. We find that in many instances, temporal segments assigned to the same latent state are related in semantic content. This occurs at varying locations across different videos, and is shown in Figure 2.5. The "Landing a fish" class is a particularly nice illustration of this, as we can typically identify a state that corresponds to the actual catching of the fish.

2.7 Summary

In this chapter we have introduced a model for learning the latent temporal structure of complex events in Internet videos. Our model is simple, and lends itself to fast, exact inference, which allows us to process large numbers of videos efficiently. In addition, we train our model in a discriminative, max-margin fashion and are able to achieve competitive accuracies on activity recognition and event detection tasks. We've shown competitive results on difficult datasets, as well as examples of semantic structure that our model is able to automatically extract.

Chapter 3

Shifting Weights: Adapting Object Detectors from Image to Video

3.1 Introduction

Following recent advances in learning algorithms and robust feature representations, tasks in video understanding have shifted from classifying simple motions and actions [42, 127] to detecting complex events and activities in Internet videos [100, 106, 141]. Detecting complex events is a difficult task, requiring probabilistic models that can understand the semantics of what is occuring in the video. Because many events are characterized by key objects and their interactions, it is imperative to have robust object detectors that can provide accurate detections. In this chapter, we focus on the problem of detecting objects in complex Internet videos. It is difficult to obtain labeled objects in these types of videos because of the large number of frames, and the fact that objects may not appear in many of them. Thus, a common approach is to train object detectors from labeled images, which are widely available. However, as seen in Figure 3.1, the domain of images and videos is quite different, as it is often the case that images of objects are taken in controlled settings that differ greatly from where they appear in real-world situations, as seen in video. Thus, we cannot typically expect a detector trained on images to work well in videos.

To adapt object detectors from image to video, we take an incremental, self-paced

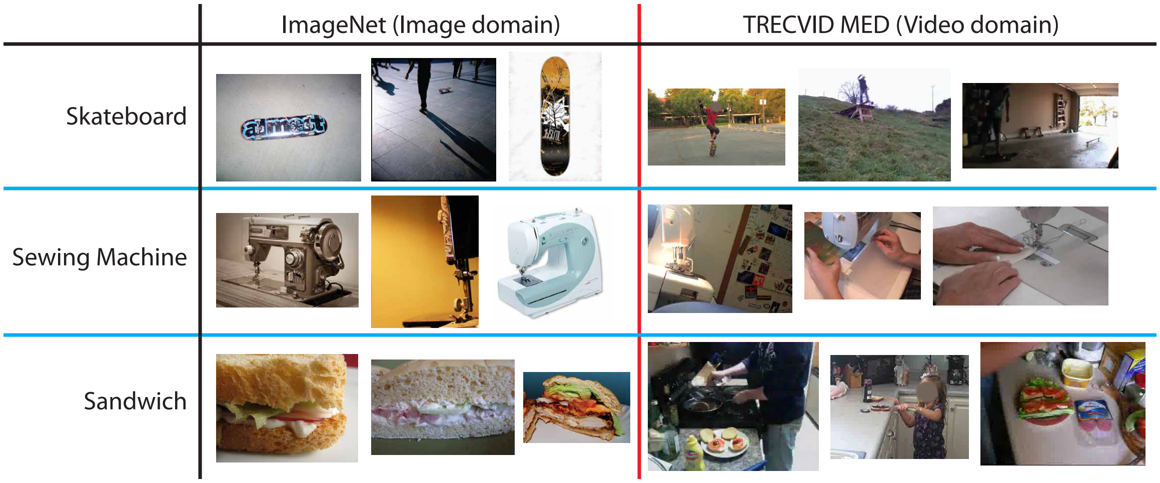


Figure 3.1: Images of the "Skateboard", "Sewing machine", and "Sandwich" classes taken from (left column) ImageNet [26] and (right column) TRECVID MED [106] illustrating differences in domain.

approach to learn from the large amounts of unlabeled video data available. We make the assumption that within our unlabeled video data, there exist instances of our target object. However, we do not assume that every video has an instance of the object, due to the noise present in Internet videos. We start by introducing a simple, robust method for discovering examples in the video data using Kanade-Lucas-Tomasi (KLT) feature tracks [93, 148]. Building on the discovered examples, we introduce a novel formulation for unsupervised domain adaptation that adapts parameters of the detector from image to video. This is done by iteratively including examples from the video data into the training set, while removing examples from the image data based on the *difficulty* of the examples. We define *easy* examples as ones with labels that can be predicted confidently (e.g., high likelihood, large distance from margin), and thus are more likely to be correct. In addition, it is common to have discriminative features that are only available in the target domain, which we term *target features*. For example, in the video domain, there are contextual features in the spatial and temporal vicinity of our detected object that we can take advantage of when performing detection. Our approach is able to incorporate the learning of parameters for these target features into a single objective.

3.2 Related Work

Most relevant are works that also deal with adapting detectors to video [18, 129, 160, 168], but these works typically deal with a constrained set of videos and limited object classes. The work of [113] deals with a similar problem, but they adapt detectors from video to image. Our overall method is also similar to [89], in which we adopt an incremental approach to learn object category models.

Our setting is closely related to the domain adaptation problem, which has been studied extensively in vision settings. Several previous approaches focus on learning feature transformations between domains [41, 76, 123]. More similar to our method are approaches based on optimizing Support Vector Machine (SVM) related objectives [9, 29, 62, 128, 149, 167] or joint cost functions [177], that treat the features as fixed and seek to adapt parameters of the classifier from source to target domain. However, with the exception of [41, 177], previous works deal with *supervised* or *semi-supervised* domain adaptation, which require labeled data in the target domain to generate associations between the source and target domains. In our setting, *unsupervised* domain adaptation, the target domain examples are unlabeled, and we must simultaneously discover and label examples in addition to learning parameters.

The objective we optimize to learn our detector draws inspiration from [39, 77, 90], in which we include and exclude the loss of certain examples using binary-valued indicator variables. Although our formulation is similar to [39, 90], our method is iterative and anneals weights that govern the number of examples to use, which is similar to the idea of self-paced learning [77], where a single weight is decreased to eventually include the loss of all examples in the objective. However, our method is different from [77] in that we have three sets of weights that govern the source examples, target examples, and target features. The weights are annealed in different directions, giving us the flexibility to iteratively include examples from the target domain, exclude examples from the source domain, and include parameters for the target features. In addition, our objective is able to incorporate target features, which is novel and not considered in [39, 77, 90].

Previous works have also considered ideas similar to our target features [20, 53,

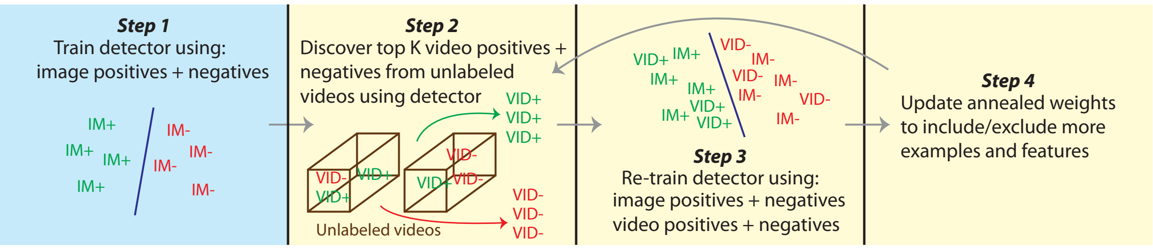


Figure 3.2: Overview of our algorithm for adapting object detectors from image to video.

75, 146]. The work of [53] considers feature augmentation, but only with observed features common to both domains. Unobserved features in the context of clustering are investigated in [75], but in their setting all examples are assumed to have the same unobserved features. In [20, 146], features or modalities unseen in the training data are used to help in testing. However, both works assume there exists relationships between the seen and unseen features, whereas our target features are completely unrestricted.

3.3 Our Approach

We begin by providing an overview of our approach to adapting object detectors, as illustrated in Figure 3.2, and then elaborate on each of the steps. We assume that we are given a large amount of unlabeled video data with positive instances of our object class within some of these videos.

We initialize our detector (Step 1 of Figure 3.2) by training a classifier on the labeled image positives and negatives, which we denote by our dataset $(\langle x_1, y_1\rangle, ..., \langle x_n, y_n\rangle)$ with binary class labels $y_i \in \{-1, 1\}$. We consider a common method of learning weights $\boldsymbol{w}$ of a linear classifier:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \left(r(\boldsymbol{w}) + C \sum_{i=1}^{n} Loss(x_i, y_i; \boldsymbol{w}) \right) \tag{3.1}$$

where $r(\cdot)$ is a regularizer over the weights, $Loss(\cdot)$ is a loss function over the training example, and C controls the tradeoff between the two.

Our goal then is to discover the top K positive and negative examples from the unlabeled videos, and to use these examples to help re-train our detector. We do not attempt to discover all instances, but simply a sufficient quantity to help adapt our detector to the video domain. To discover the top K video positives and negatives (Step 2 of Figure 3.2), we utilize the strong prior of temporal continuity and score trajectory tracks instead of bounding boxes, which we describe in Section 3.3.1. Given the discovered examples, we optimize a novel objective inspired by self-paced learning [77] that simultaneously selects easy examples and trains a new detector (Step 3 of Figure 3.2). Using this new detector, we repeat this process of example discovery and detector training until convergence, as illustrated in Figure 3.2.

3.3.1 Discovering Examples in Video

In this step of the algorithm, we are given weights $\boldsymbol{w}$ of an object detector that can be used to score bounding boxes in video frames. A naive approach would run our detector on frames of video, taking the highest scoring and lowest scoring bounding boxes as the top K video positives and negatives. Although reasonable, this method doesn't take advantage of temporal continuity in videos. An object that appears in one frame of a video is certain to appear close in neighboring frames as well. Previous works have shown this intuition to yield good results [18, 129, 168].

Track-based scoring. Our key idea is to score trajectory tracks, rather than bounding boxes, as illustrated in Figure 3.3. We obtain tracks by running a KLT tracker on our videos, which tracks a sparse set of features over large periods of time. Because of the large number of unlabeled videos we have, we elect to extract KLT tracks rather than computing dense tracks using optical flow. Note that these tracks follow features, and so they may not correspond to centered locations of objects.

For each track, we consider the set of all bounding box placements $\mathcal{B}$ around it that intersect with the track. Each box placement $b_i \in \mathcal{B}$ is associated with a relative coordinate (b_i^x, b_i^y) as well as a score b_i^s. The relative coordinate (b_i^x, b_i^y) is the point within the box (relative to the top-left corner of the box) that intersects the track. Using this coordinate, we can compute the position of b_i at every point in time along

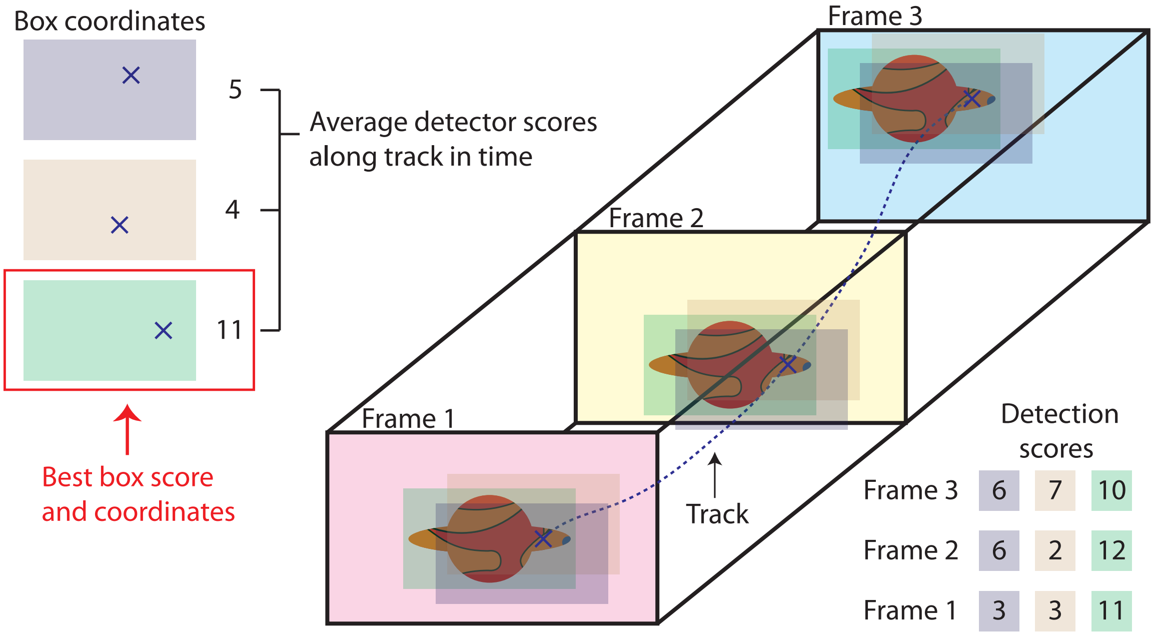


Figure 3.3: For a given KLT track, we consider all bounding box placements that intersect with it, and take the box with the maximum score as the score and associated bounding box coordinates for this track.

the track. Note that the number of bounding boxes in $\mathcal{B}$ is only dependent on the dimensions of the detector and the scales we search over. The score b_i^s is computed by pooling scores of the bounding box along multiple points of the track in time. We use average pooling in our experiments to be robust to noisy scores. Finally, we associate the track with the bounding box $b_{\max}$ with the highest score, and use the score $b_{\max}^s$ as the score of the track.

After scoring each track in our unlabeled videos, we select the top and bottom few scoring tracks, and extract bounding boxes from each using the associated box coordinates $(b_{\max}^x, b_{\max}^y)$ to get our top K video positives and negatives. The boxes are extracted by sampling frames along the track.

Advantages. Compared to the naive approach without tracks, this approach allows us to recover from false detections with high scores, which are common for weak detectors, as it is less likely that there will be multiple false detections with high scores along a KLT track. Similarly, if the detection scores are consistently high along many points of a track, we can be more confident of the object's presence along the track. Hence, we can obtain novel examples of the object from various points of the track that had low scores, since we know the trajectory should correspond to the object. The same intuitions hold for true detections with low scores and obtaining negative examples.

3.3.2 Self-Paced Domain Adaptation

In this step of the algorithm, we are given the discovered top K video positives and negatives, which we denote by the dataset $(\langle z_1, h_1\rangle, ..., \langle z_k, h_k\rangle)$. Together with our original dataset $(\langle x_1, y_1\rangle, ..., \langle x_n, y_n\rangle)$, we would like to learn a new detector.

A simple method would be to re-train our detector with both datasets using Equation 3.1. However, we typically aren't certain that the labels $\boldsymbol{h}$ are correct, especially in the first iteration when our detector is trained solely from the image examples. Ideally, we would like to re-train with a set of easier examples whose labels we are confident of first, and then re-discover video examples with this new detector. We would also like to stop learning from examples we are unsure of in the image domain, as they may be the examples most affected by the differences in domain. By repeating this process, we can avoid bad examples and iteratively refine our set of top K video positives and negatives before having to train with all of them.

Formulating this intuition, our algorithm selects easier examples to learn from in the discovered video examples, and simultaneously selects harder examples in the image examples to stop learning from. An example is *difficult* if it has a large loss, as we are not confident in its correct label. The number of examples selected from the video examples and image examples are governed by weights that will be annealed over iterations (Step 4 of Figure 3.2).

Basic approach. We start by introducing our approach without target features. We introduce binary variables $v_1, ..., v_n$ for the source domain (image) examples, and binary variables $u_1, ..., u_k$ for the target domain (video) examples. A value of 0 indicates that an example is difficult, and so we would like to remove its loss from consideration in the objective function. To prevent the algorithm from assigning all examples to be difficult, we introduce parameters K^{source} and K^{target} that control the number of examples considered from the source and target domain, respectively.

$$(\boldsymbol{w}_{t+1}, \boldsymbol{v}_{t+1}, \boldsymbol{u}_{t+1}) = \underset{\boldsymbol{w},\boldsymbol{v},\boldsymbol{u}}{\arg\min} \Bigg(r(\boldsymbol{w}) + C\Big(\sum_{i=1}^{n} v_i Loss(x_i, y_i; \boldsymbol{w}) + \sum_{j=1}^{k} u_j Loss(z_j, h_j; \boldsymbol{w}) \Big) - \frac{1}{K^{source}} \sum_{i=1}^{n} v_i - \frac{1}{K^{target}} \sum_{j=1}^{k} u_j \Bigg) \tag{3.2}$$

If K^{target} is large, the algorithm prefers to consider only easy target examples with a small $Loss(\cdot)$, and the same is true for K^{source}. In the annealing of the weights for the algorithm (Step 4 of Figure 3.2), we decrease K^{target} and increase K^{source} to iteratively include more examples from the target domain and decrease examples from the source domain.

Similar to self-paced learning [77], we obtain a *tight* relaxation when allowing the binary variables $\boldsymbol{v}$ and $\boldsymbol{u}$ to take on any value in the interval $[0, 1]$. With the choice of $r(\cdot)$ and $Loss(\cdot)$ convex in $\boldsymbol{w}$, the problem becomes a bi-convex problem, and can be solved by alternating between (1) solving for $\boldsymbol{w}$ given $\boldsymbol{v}$ and $\boldsymbol{u}$, and (2) solving for $\boldsymbol{v}$ and $\boldsymbol{u}$ given $\boldsymbol{w}$. We refer the reader to [77] for further intuitions on the binary variables and annealed weights.

Leveraging target features. Often, the target domain we are adapting to has additional features we can take advantage of. At the start, when we've only learned from a few examples in our target domain, we do not wish to rely on these rich and expressive features, as they can easily cause us to overfit. However, as we iteratively adapt to the target domain and build more confidence in our detector, we can start utilizing these target features to help with detection. The inclusion of these features is naturally self-paced as well, and can be easily integrated into our framework.

We assume there are a set of features that are shared between the source and target domains as ϕ_{shared}, and a set of target domain-only features as ϕ_{target}: $\phi = [\phi_{shared} \quad \phi_{target}]$. The weights $\boldsymbol{w}$ we want to learn can now be divided into $\boldsymbol{w}_{shared}$ and

$\boldsymbol{w}_{target}$: $\boldsymbol{w} = [\boldsymbol{w}_{shared} \quad \boldsymbol{w}_{target}]$. Since the source data doesn't have ϕ_{target} features, we initialize those features to be 0 so that $\boldsymbol{w}_{target}$ doesn't affect the loss on the source data. The new objective function is formulated as:

$$(\boldsymbol{w}_{t+1}, \boldsymbol{v}_{t+1}, \boldsymbol{u}_{t+1}) = \underset{\boldsymbol{w},\boldsymbol{v},\boldsymbol{u}}{\arg\min} \Bigg(r(\boldsymbol{w}) + C\Big(\sum_{i=1}^{n} v_i Loss(x_i, y_i; \boldsymbol{w}) + \sum_{j=1}^{k} u_j Loss(z_j, h_j; \boldsymbol{w})\Big) + \frac{1}{K^{feat}}||\boldsymbol{w}_{target}||_1 - \frac{1}{K^{source}}\sum_{i=1}^{n} v_i - \frac{1}{K^{target}}\sum_{j=1}^{k} u_j \Bigg) \tag{3.3}$$

This is similar to Equation 3.2, with the addition of the L_1 norm term written as $\frac{1}{K^{feat}}||\boldsymbol{w}_{target}||_1$. To anneal the weights for target features, we increase K^{feat} to iteratively reduce the L_1 norm on the target features so that $\boldsymbol{w}_{target}$ can become non-zero. Intuitively, we are forcing the weights $\boldsymbol{w}$ to only use shared features first, and to consider more target features when we have a better model of the target domain. The optimization can be solved in the same manner as Equation 3.2. We can also approximate the L_1 norm term for all target features to be effectively binary, forcing K^{feat} to be 0 initially and switching to ∞ at a particular iteration. This amounts to only considering target features after a certain iteration, and is done in our experiments for more tractable learning.

3.4 Experiments

We present experimental results for adapting object detectors on the 2011 TRECVID Multimedia Event Detection (MED) dataset [106] and LabelMe Video [172] dataset. For both, we select a set of objects which are known to appear in the videos. We used images from ImageNet [26] for the labeled image data, as there are a large number of diverse categories on ImageNet that correspond well with the objects that appear in the videos. We evaluate the detection performance of our models with the measure used in the PASCAL Visual Object Classes challenge [33], and report average precision (AP) scores for each class. The detection scores are computed on annotated video frames from the respective video datasets that are disjoint from the unlabeled videos used in the adapting stage.

3.4.1 Implementation Details

In our experiments, we use object detectors that are rectangular filters over Histogram-of-Gradient (HOG) features [22]. We use L_2 regularization for $r(\cdot)$ and hinge loss for $Loss(\cdot)$, which corresponds to the standard linear SVM formulation. For target features, we use contextual spatial features. The spatial features are taken to be HOG features bordering the object with dimensions half the size of the object bounding box. As described previously, we approximate the L_1 norm term to be binary to enable fast training using LIBLINEAR [34] when optimizing for $\boldsymbol{w}$. This also further decreases the number of model parameters needed to be searched over.

To isolate the effects of adaptation and better analyze our method, we restrict our experiments to the setting in which we fix the video negatives, and focus our problem on adapting from the labeled image positives to the unlabeled video positives. This scenario is realistic and commonly seen, as we can easily obtain video negatives by sampling from a set of unlabeled or weakly-labeled videos.

Model parameters. In our experiments, we fix the total number of iterations to 5 for tractable training time. For the K^{target} and K^{source} weights, we set values for the first and final iterations, and linearly interpolate values for the remaining iterations in between. For the K^{target} weight, we estimate the weights so that we start by considering only the video examples that have no loss, and end with all video examples considered. For the K^{source} weight, we vary the ending weight so that differing numbers of source examples are left for training at the final iteration. For the target features, we set the algorithm to allow target features at the midpoint of total iterations. Based on the number of KLT tracks extracted, we set the top K examples to be between 100 and 500.

Model selection. The free model parameters that can be varied are the number of top K examples to discover, the ending K^{source} weight, and whether or not to use

Object	InitialBL	VideoPosBL	Our method(nt)	Our method(full)	Gopalan *et al.* [41] (PLS)	Gopalan *et al.* [41] (SVM)
Skateboard	4.29%	2.89%	10.44%	**10.44%**	0.04%	0.94%
Animal	0.41%	0.40%	0.39%	**3.76%**	0.16%	0.24%
Tire	11.22%	11.04%	15.54%	**15.54%**	0.60%	15.52%
Vehicle	4.03%	**4.08%**	3.57%	3.57%	3.33%	3.16%
Sandwich	10.07%	9.85%	9.45%	**12.49%**	0.21%	6.68%
Sewing machine	9.76%	9.71%	10.35%	**10.35%**	0.12%	3.81%
Mean AP	6.63%	6.33%	8.29%	**9.36%**	0.74%	5.06%

Table 3.1: Average Precision values for object detection on the TRECVID MED dataset

Object	InitialBL	VideoPosBL	Our method(nt)	Our method(full)	Gopalan *et al.* [41] (PLS)	Gopalan *et al.* [41] (SVM)
Car	2.60%	2.13%	2.15%	**9.18%**	0.34%	1.00%
Boat	0.22%	0.22%	0.22%	0.22%	0.05%	**0.32%**
Bicycle	19.85%	19.76%	20.27%	**20.27%**	0.21%	16.32%
Dog	1.74%	2.42%	2.47%	**4.75%**	0.18%	1.48%
Keyboard	0.41%	**0.67%**	0.59%	0.59%	0.13%	0.09%
Mean AP	4.96%	5.04%	5.14%	**7.00%**	0.18%	3.84%

Table 3.2: Average Precision values for object detection on the LabelMe Video dataset

target features. In our results, we perform model selection by comparing the distribution of scores on the discovered video positives. The distributions are compared between the initial models from iteration 1 for different model parameters to select K and K^{source}, and between the final iteration 5 models for different model parameters to determine the use of target features. This allows us to evaluate the strength of the initial model trained on the image positives and video negatives, as well as our final adapted model. We select the model with the distributions indicating the highest confidence in its classification boundary.

3.4.2 Baseline Comparisons

InitialBL. This baseline is the intial detector trained only on image positives and video negatives.

VideoPosBL. This baseline uses the intial detector to discover the top K video positives from the unlabeled video, then trains with all these examples without iterating. Thus, it incorporates our idea of discovering video positives by scoring tracks and re-training, but does not use self-paced domain adaptation for learning weights. It can also be thought of as our method run for one iteration.

Our method(nt). This baseline uses our full method with the exception of target features.

Gopalan *et al.*. This is a state-of-the-art method for unsupervised domain adaptation [41] that models the domain shift in feature space. Since we are not given labels in the target domain, most previous methods for domain adaptation cannot be applied to our setting. This method samples subspaces along the geodesic between the source and target domains on the Grassman manifold. Using projections of both source and target data onto the common subspaces, they learn a discriminative classifier using partial least squares (PLS) with available labels from either domains. We ran their code using their suggested parameter settings to obtain results for their method on our task. We also show results for their method using a linear SVM as the classifier to allow for fair comparisons.

3.4.3 TRECVID MED

The 2011 TRECVID MED dataset [106] consists of a collection of Internet videos collected by the Linguistic Data Consortium from various Internet video hosting sites. There are a total of 15 complex events, and videos are labeled with either an event class or no label, where an absence of label indicates the video belongs to no event class. We select 6 object classes to learn object detectors for because they are commonly present in selected events: “Skateboard”, “Animal”, “Tire”, “Vehicle”, “Sandwich”, and “Sewing machine”. These objects appear respectively in the events “Attempting a board trick”, “Feeding an animal”, “Changing a vehicle tire”, “Getting a vehicle unstuck”, “Making a sandwich”, and “Working on a sewing project”. The video negatives were randomly sampled from the videos that were labeled with no event class.

To test our algorithm, we manually annotated approximately 200 frames with bounding boxes of positive examples for each object, resulting in 1234 annotated frames total from over 500 videos, giving us a diverse set of situations the objects can appear in. For each object, we use 20 videos from the associated event as unlabeled video training data. Results are given in Table 3.1.

3.4.4 LabelMe Video

LabelMe Video [172] is a database of real-world videos that contains a large set of annotations including object category, shape, motion, and activity information. We use the database of videos that was introduced in the original paper [172]. There are a large number of objects that are annotated in this database, and we select the most frequently occuring objects that are not scene parts, resulting in 5 objects: "Car", "Boat", "Bicycle", "Dog", and "Keyboard". The video negatives were randomly sampled from the videos that were not annotated with any of these objects.

We extract more than 200 frames with positive examples for each object class, resulting in a test set of 1137 images. For each object class, we use the remaining videos that contain the object as the unlabeled video training data, resulting in around 9 videos per object. Results are given in Table 3.2.

3.5 Discussion

From our results in Tables 3.1 and 3.2, we can observe similar patterns for most object classes. First, we note that the "VideoPosBL" baseline typically performs on par with the "InitialBL" baseline, and rarely does it post a slight gain in performance. This shows that if we discover the top K video positives and re-train our detector with all of them, we do not obtain consistent gains in performance. Our method of self-paced domain adaptation is crucial in this case, as we can see that our full method typically outperforms all other methods by significant margins. As illustrated in Figure 3.4, our method is able to add new video positives from iteration to iteration that are good examples, and remove bad examples at the same time. The method of Gopalan *et al.* [41] performs very poorly when used in conjunction with the PLS classifier, but becomes more competitive when used with an SVM. However, even then their

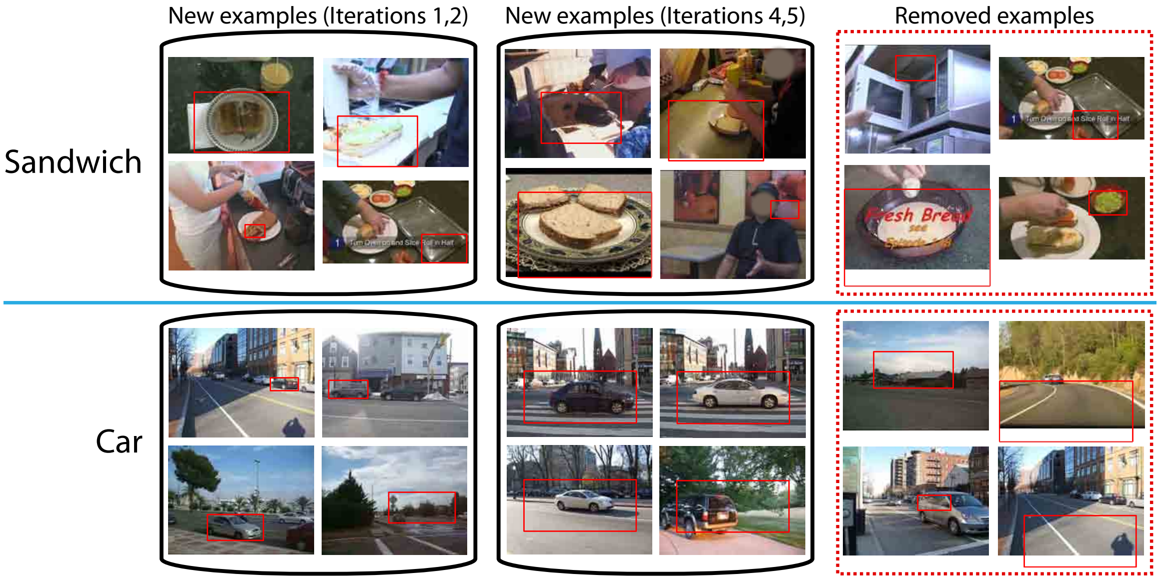


Figure 3.4: Discovered top K video positives using our method for "Sandwich" and "Car". After sets of iterations, we show samples of newly discovered video positives (left, middle columns) and bad examples that were removed (right column).

method performs much worse than our method for nearly all object classes, as it is difficult to model the underlying domain shift in feature space. This also serves to illustrate the difficulty of our problem, as poor adaptation can lead to results worse than the baselines. We show visualizations of our detections compared to baseline methods in Figure 3.5.

Observing the visualizations of the learned weights for the "Tire", "Car" and "Sandwich" classes in Figure 3.6, we see that weights trained with our method exhibit more clearly defined structure than the "InitialBL" baseline. The target features also help performance significantly. By capturing interesting patterns in the spatial context, difficult objects can become easier to detect in the target domain. For the "Sandwich" class, we can see circular weights in the spatial context surrounding the sandwich, suggesting that sandwiches typically appear on plates, and for "Car", we can clearly distinguish weights for the road beneath the car object. We observe an average AP gain of 3.93% for classes that choose models with target features versus no target features. Note that we chose to use simple spatial context as target features in our models, as they are fast to implement and easily incorporated. However,

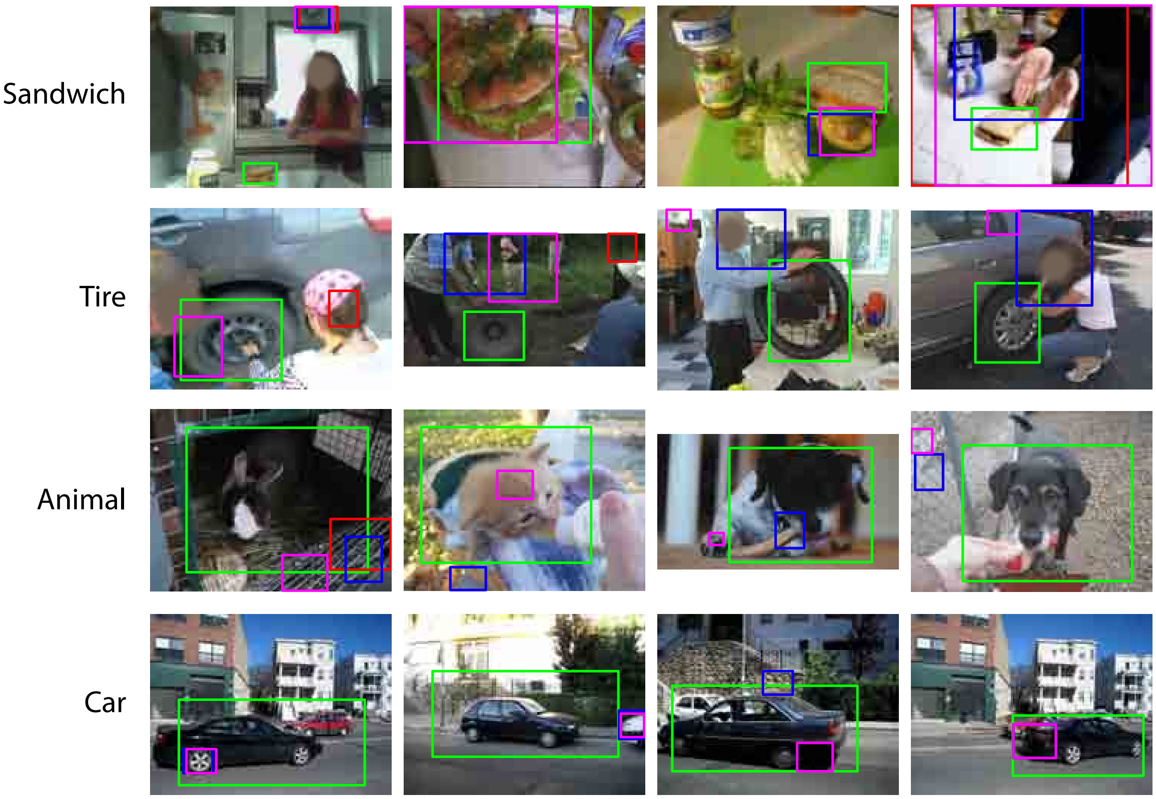


Figure 3.5: Detections for "Sandwich", "Tire", "Animal", and "Car". Green boxes detections from our method, red boxes detections from "InitialBL", blue boxes detections from "VideoPosBL", and magenta boxes detections from Gopalan *et al.*(SVM).

we hypothesize that the inclusion of more complex target features such as temporal movement could help our method achieve even better results.

We observe that for the "Vehicle" and "Keyboard" classes, the "VideoPosBL" baseline performs better than our full method. Although this is not a common occurrence, it can happen when our method of self-paced domain adaptation replaces good video positives taken in the first iteration with bad examples in future iterations. This situation arises when there are incorrect examples present in the easiest of the top K video positives, causing our detector to re-train and iteratively become worse. If we had better methods for model selection, we could also search over the number of total iterations as a model parameter, which would include the "VideoPosBL" model in our set of models to select over, as it is essentially our method run for a single

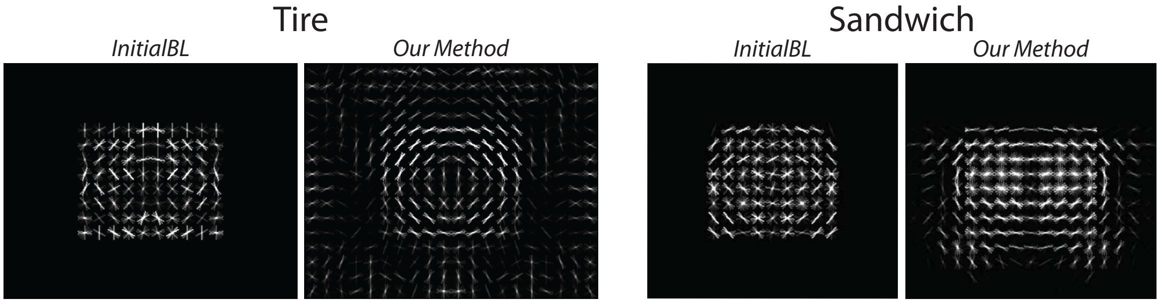


Figure 3.6: Visualizations of the positive HOG weights learned for "Tire" and "Sandwich" for the "InitialBL" baseline and our method.

iteration.

3.6 Summary

In this chapter we have introduced an approach for adapting detectors from image to video. To discover examples in the unlabeled video data, we classify tracks instead of bounding boxes, allowing us to leverage temporal continuity to avoid spurious detections, and to discover examples we would've otherwise missed. Furthermore, we introduce a novel self-paced domain adaptation algorithm that allows our detector to iteratively adapt from source to target domain, while also considering target features unique to the target domain. Our formulation is general, and can be applied to various other problems in domain adaptation. We've shown convincing results that illustrate the benefit of our approach to adapting object detectors to video.

Chapter 4

Discriminative Segment Annotation in Weakly Labeled Video

4.1 Introduction

The ease of authoring and uploading video to the Internet creates a vast resource for computer vision research, particularly because YouTube videos are frequently associated with semantic tags that identify visual concepts appearing in the video. However, since tags are not spatially or temporally localized within the video, such videos cannot be directly exploited for training traditional supervised recognition systems. This has stimulated significant recent interest in methods that learn localized concepts under weak supervision [47, 100, 113, 141]. In this chapter, we examine the problem of generating pixel-level concept annotations for weakly labeled video.

To make our problem more concrete, we provide a rough pipeline of the overall process (see Figure 4.1). Given a video weakly tagged with a concept, such as "dog", we process it using a standard unsupervised spatiotemporal segmentation method that aims to preserve object boundaries [12, 44, 88]. From the video-level tag, we know that some of the segments correspond to the "dog" concept while most probably do not. Our goal is to classify each segment within the video either as coming from

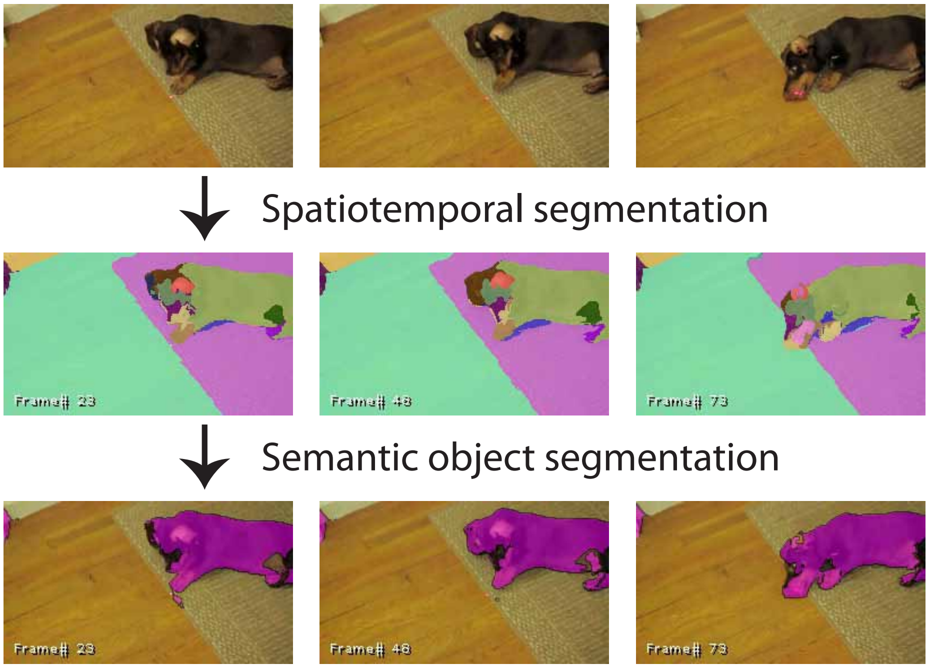


Figure 4.1: Given a weakly tagged video (top), we perform unsupervised spatiotemporal segmentation (middle) then identify segments that correspond to the label to generate a semantic segmentation (bottom).

the concept "dog", which we denote as *concept segments*, or not, which we denote as *background segments*. Given the varied nature of Internet videos, we cannot rely on assumptions about the relative frequencies or spatiotemporal distributions of segments from the two classes, neither within a frame nor across the video; nor can we assume that each video contains a single instance of the concept. For instance, neither the dog in Figure 4.1 nor most of the objects in Figure 4.10 would be separable from the complex background by unsupervised methods.

There are two settings for addressing the segment annotation problem, which we illustrate in Figure 4.2. The first scenario, which we term *transductive segment annotation* (TSA), is studied in [133]. This scenario is closely related to automatically annotating a weakly labeled dataset. Here, the test videos that we seek to annotate are compared against a large amount of negative segments (from videos not tagged with the concept) to enable a direct discriminative separation of the test video segments into two classes. The second scenario, which we term *inductive segment annotation* (ISA), is studied in [47]. In this setting, a segment classifier is trained using a large quantity of weakly labeled segments from both positively- and negatively-tagged videos. Once trained, the resulting classifier can be applied to any test video (typically not in the original set). We observe that the TSA and ISA settings parallel the distinction between transductive and inductive learning, since the test instances are available during training in the former but not in the latter. Our proposed algorithm, Concept Ranking According to Negative Exemplars (CRANE), can operate under either scenario and we show experimental results demonstrating its clear superiority over previous work under both settings.

Our contributions can be organized into three parts.

1. We present a unified interpretation under which a broad class of weakly supervised learning algorithms can be analyzed.

2. We introduce CRANE, a straightforward and effective discriminative algorithm that is robust to label noise and highly parallelizable. These properties of CRANE are extremely important, as such algorithms must handle large amounts of video data and spatiotemporal segments.

3. We introduce spatiotemporal segment-level annotations for a subset of the YouTube-Objects dataset [113], and present a detailed analysis of our method compared to other methods on this dataset for the transductive segment annotation scenario. To promote research into this problem, we make our annotations freely available. We also compare CRANE directly against [47] on the inductive segment annotation scenario and demonstrate state-of-the-art results.

4.2 Related Work

Several methods have recently been proposed for high-quality, unsupervised spatiotemporal segmentation of videos [12, 44, 88, 163, 164]. The computational efficiency of some of these approaches [44, 164] makes it feasible to segment large

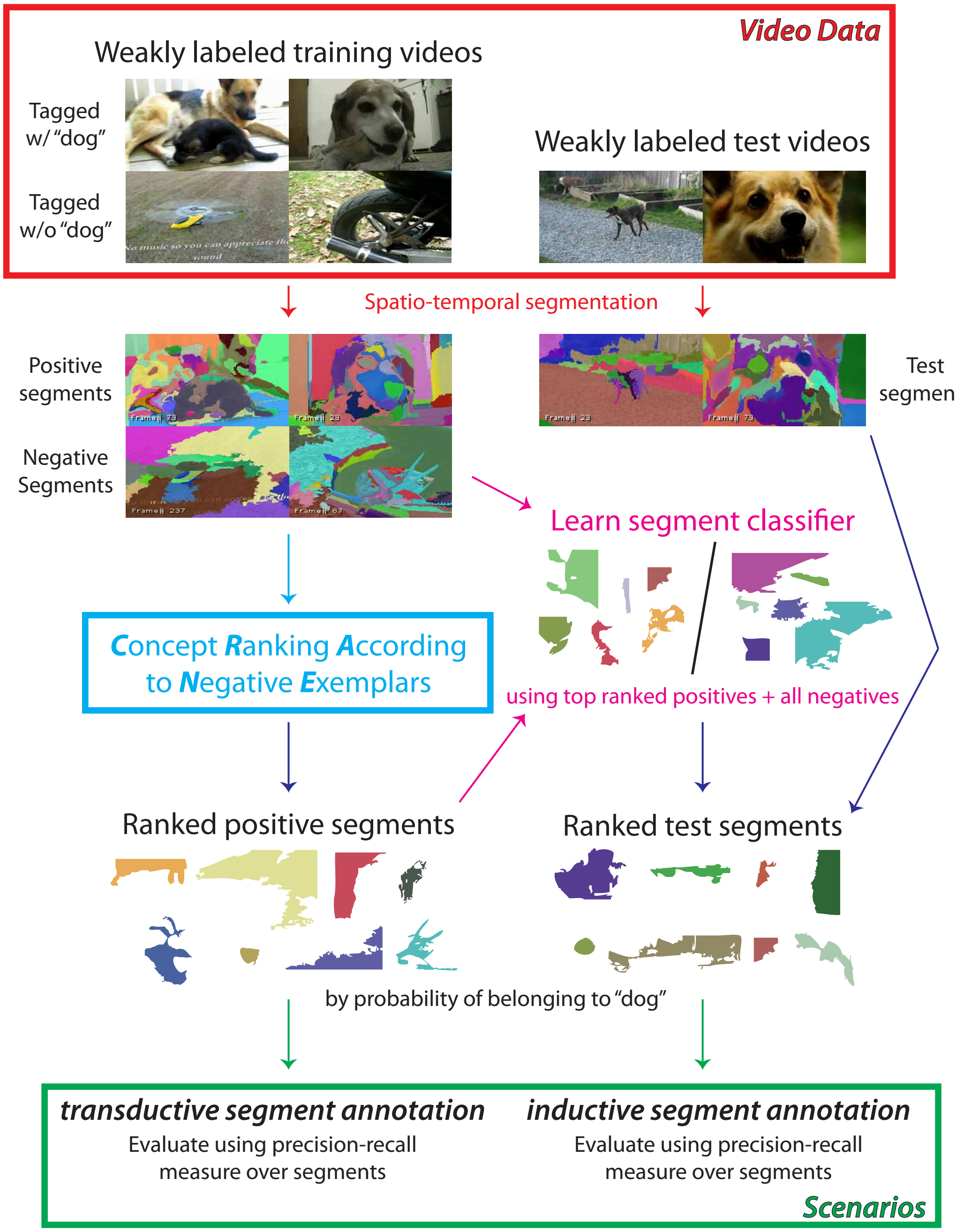

Video Data
Weakly labeled training videos
Tagged w/ "dog"
Tagged w/o "dog"
Weakly labeled test videos
Spatio-temporal segmentation
Positive segments
Negative Segments
Test segments
Learn segment classifier
Concept Ranking According to Negative Exemplars
using top ranked positives + all negatives
Ranked positive segments
Ranked test segments
by probability of belonging to "dog"
transductive segment annotation
Evaluate using precision-recall measure over segments
inductive segment annotation
Evaluate using precision-recall measure over segments
Scenarios

Figure 4.2: Overview of transductive and inductive segment annotation. In the former (TSA), CRANE is evaluated on weakly labeled training data; in the latter (ISA), we train a classifier and evaluate on a disjoint test set.

numbers of Internet videos. Several recent works have leveraged spatiotemporal segments for a variety of tasks in video understanding, including event detection [68], human motion volume generation [99], human activity recognition [11], and object segmentation [47, 86]. Drawing inspiration from these, we also employ such segments as a core representation in our work.

Lee *et al.* [86] perform object segmentation on unannotated video sequences. Our approach is closer to that of Hartmann *et al.* [47], where object segmentations are generated on weakly labeled video data. Whereas [47] largely employ variants on standard supervised methods (e.g., linear classifiers and multiple-instance learning), we propose a new way of thinking about this weakly supervised problem that leads to significantly superior results.

Discriminative segment annotation from weakly labeled data shares similarities with Multiple Instance Learning (MIL), on which there has been considerable research (e.g., [16, 156, 176, 179]). In MIL, we are given labeled bags of instances, where a positive bag contains at least one positive instance, and a negative bag contains no positive instances. MIL is more constrained than our scenario, since these guarantees may not hold due to label noise (which is typically present in video-level tags). In particular, algorithms must contend with positive videos that actually contain no concept segments as well as rare cases where some concept segments appear in negative videos.

There is increasing interest in exploring the idea of learning visual concepts from a combination of weakly supervised images and weakly supervised video [3, 27, 87, 104, 115, 153]. Most applicable to our problem is recent work that achieves state-of-the-art results on bounding box annotation in weakly labeled 2D images [133]. We show that this "negative mining" method can also be applied to segment annotation. Direct comparisons show that CRANE outperforms negative mining and is more robust to label noise.

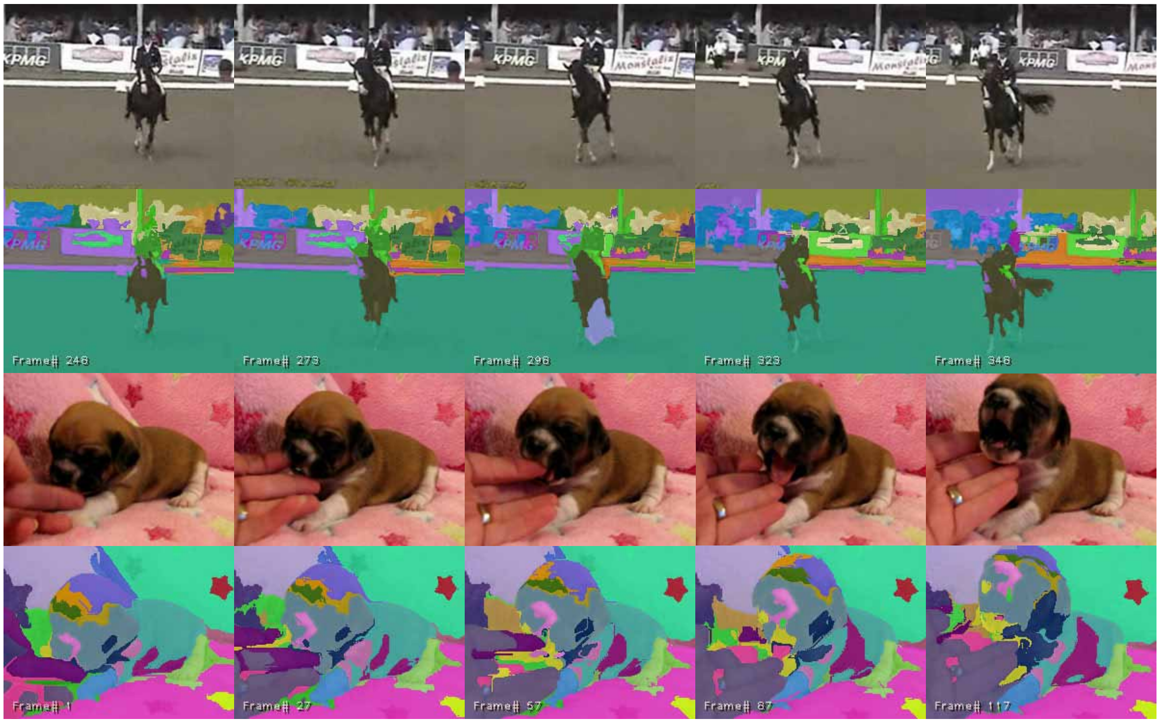


Figure 4.3: Spatiotemporal segments computed on "horse" and "dog" video sequences using [44].

4.3 Weakly Supervised Segment Annotation

As discussed earlier, we start with spatiotemporal segments for each video, such as those shown in Figure 4.3. Each segment is a spatiotemporal (3D) volume that we represent as a point in a high-dimensional feature space using a set of standard features computed over the segment.

More formally, for a particular concept c, we are given a dataset $\{\langle s_1, y_1\rangle, ..., \langle s_N, y_N\rangle\}$, where s_i is segment i, and $y_i \in \{-1, 1\}$ is the label for segment i, with the label being positive if the segment was extracted from a video with concept c as a weak label, and negative otherwise. We denote the set $\mathcal{P}$ to be the set of all instances with a positive label, and similarly $\mathcal{N}$ to be the set of all negative instances. Since our negative data was weakly labeled with concepts other than c, we can assume that the segments labeled as negative are (with rare exceptions) correctly labeled. Our task then is

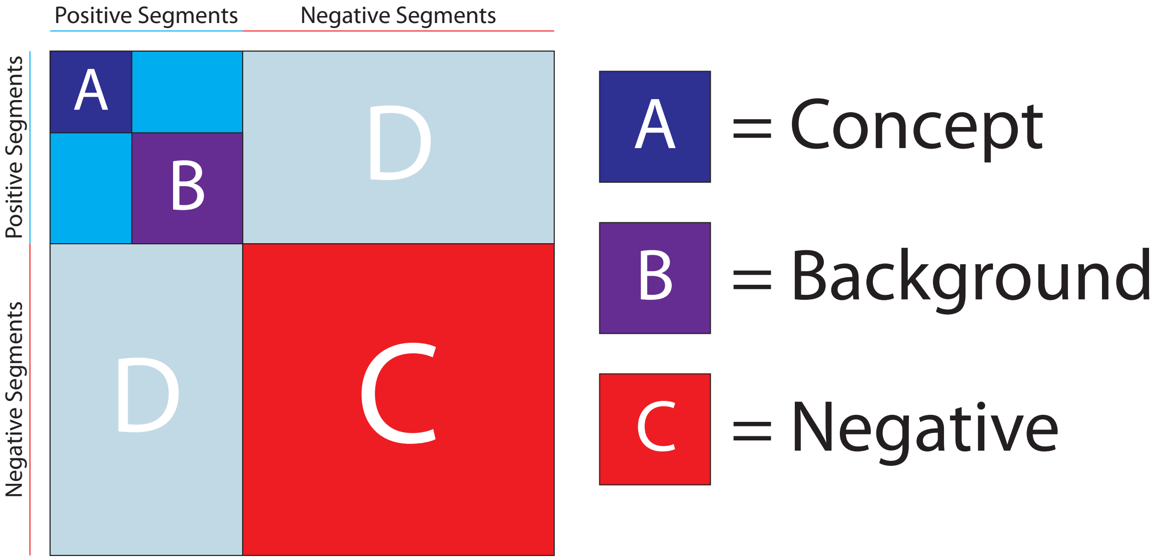


Figure 4.4: Visualization of pairwise distance matrix between segments for weakly supervised annotation.

to determine which of the positive segments $\mathcal{P}$ are concept segments, and which are background segments.

We present a generalized interpretation of transductive segment annotation, which leads to a family of methods that includes several common methods and previous works [133]. Consider the pairwise distance matrix (in the high-dimensional feature space) between all of the segments s_i from both the positive and negative videos, for a particular concept c. Across the rows and columns, we order the segments from $\mathcal{P}$ first, followed by those from $\mathcal{N}$. Within $\mathcal{P}$, we further order the concept segments $\mathcal{P}_c \subset \mathcal{P}$ first, followed by the background segments $\mathcal{P}_b = \mathcal{P} \setminus \mathcal{P}_c$. This distance matrix is illustrated in Figure 4.4. The blocks A, B and C correspond to intra-class distances among segments from $\mathcal{P}_c$, $\mathcal{P}_b$, and $\mathcal{N}$, respectively. The block circumscribing A and B corresponds to the distances among $\mathcal{P}$. Note that A and B are hidden from the algorithm, since determining the membership of $\mathcal{P}_c$ is the goal of TSA. We can now analyze a variety of weakly supervised approaches in this framework.

Rather than solely studying TSA as the problem of partitioning $\mathcal{P}$, we find it fruitful to also consider the related problem of *ranking* the elements of $\mathcal{P}$ in decreasing order of a score, $S(s_i)$ such that top-ranked elements correspond to $\mathcal{P}_c$; thresholding at a particular rank generates a partition.

Co-segmentation/Clustering. Co-segmentation [154] exploits the observation that concept segments across videos are similar, but that background segments are diverse. The purest variants of this approach are unsupervised and do not require $\mathcal{N}$ and can operate solely on the top-left 2×2 sub-matrix. The hope is that the concept segments form a dominant cluster/clique in feature space.

Kernel density estimation for $\mathcal{N}$. This principled approach to weakly supervised learning exploits the insight that the (unknown) distribution of background segments $\mathcal{P}_b$ must be similar to the (known) distribution of negative segments $\mathcal{N}$, since the latter consists almost entirely of background segments. Accordingly, we construct a non-parametric model of the probability density $P_{\mathcal{N}}(x)$ generated from the latter (block C) and employ it as a proxy for the former (block B). Then, elements from $\mathcal{P}$ that lie in high-density regions of $P_{\mathcal{N}}(.)$ can be assumed to come from $\mathcal{P}_b$, while those in low-density regions are probably the concepts $\mathcal{P}_c$ that we seek. A natural algorithm for TSA is thus to rank the elements $s_i \in \mathcal{P}$ according to $P_{\mathcal{N}}(s_i)$.

In practice, we estimate $P_{\mathcal{N}}$ using kernel density estimation, with a Gaussian kernel whose σ is determined using cross-validation so as to maximize the log likelihood of generating $\mathcal{N}$. In our interpretation, this corresponds to building a generative model according to the information in block C of the distance matrix, and scoring segments according to:

$$S_{\mathrm{KDE}}(s_i) = -P_{\mathcal{N}}(s_i) = -\frac{1}{|\mathcal{N}|}\sum_{z \in \mathcal{N}} N\Big(\mathrm{dist}(s_i, z); \sigma^2\Big), \tag{4.1}$$

where $N(\cdot; \sigma^2)$ denotes a zero-mean multivariate Gaussian with isotropic variance of σ^2.

Supervised discriminative learning with label noise. Standard fully supervised methods, such as Support Vector Machines (SVM), learn a discriminative classifier to separate positive from negative data, given instance-level labels. Such methods can be shoehorned into the weakly supervised setting of segment annotation by propagating video-level labels to segments. In other words, we learn a discriminative classifier to separate $\mathcal{P}$ from $\mathcal{N}$, or the upper 2×2 submatrix vs. block C. Unfortunately, since $\mathcal{P} = \mathcal{P}_c \cup \mathcal{P}_b$, this approach treats the background segments from positively tagged videos, $\mathcal{P}_b$ (which are typically the majority), as label noise. Nonetheless, such approaches have been reported to perform surprisingly well [47], where linear SVMs trained with label noise achieve competitive results. This may be because the limited capacity of the classifier is unable to separate $\mathcal{P}_b$ from $\mathcal{N}$ and therefore focuses on separating $\mathcal{P}_c$ from $\mathcal{N}$. In our experiments, methods that tackle weakly labeled segment annotation from a more principled perspective significantly outperform these techniques.

Negative Mining (MIN). Siva *et al.*'s negative mining method [133], which we denote as MIN, can be interpreted as a discriminative method that operates on block D of the matrix to identify $\mathcal{P}_c$. Intuitively, distinctive concept segments are identified as those among $\mathcal{P}$ whose nearest neighbor among $\mathcal{N}$ is as far as possible. Operationally, this leads to the following score for segments:

$$S_{\text{MIN}}(s_i) = \min_{t \in \mathcal{N}} \Big(\text{dist}(s_i, t) \Big). \tag{4.2}$$

Following this perspective on how various weakly supervised approaches for segment annotations relate through the distance matrix, we detail our proposed algorithm, CRANE.

4.4 Proposed Method: CRANE

Like MIN, our method, CRANE, operates on block D of the matrix, corresponding to the distances between weakly tagged positive and negative segments. Unlike MIN, CRANE iterates through the segments in $\mathcal{N}$, and each such negative instance penalizes nearby segments in $\mathcal{P}$. The intuition is that concept segments in $\mathcal{P}$ are those that are far from negatives (and therefore less penalized). While one can envision several algorithms that exploit this theme, the simplest variant of CRANE can be characterized by the following segment scoring function:

$$\begin{aligned} S_{\text{CRANE}}(s_i) = -\sum_{z\in\mathcal{N}} \mathbf{1}\Big[s_i = \operatorname*{arg\,min}_{t\in\mathcal{P}} \Big(\text{dist}(t,z)\Big)\Big] \\ \cdot f_{\text{cut}}\big(\text{dist}(s_i,z)\big), \end{aligned} \tag{4.3}$$

where $\mathbf{1}(\cdot)$ denotes the indicator function and $f_{\text{cut}}(\cdot)$ is a cutoff function over an input distance.

Figure 4.5 illustrates the intuition behind CRANE. Background segments in positive videos tend to fall near one or more segments from negative videos (in feature space). The nearest neighbor to every negative instance is assigned a penalty $f_{\text{cut}}(.)$. Consequently, such segments are ranked lower than other positives. Since concept segments are rarely the closest to negative instances, they are typically ranked higher. Figure 4.5 also shows how CRANE is more robust than MIN [133] to label noise among negative videos. Consider the points in the green box shown at the top right of the figure. Here, the unknown segment, s_i, is very close to a negative instance that may have come from an incorrectly tagged video. This single noisy instance will cause MIN to irrecoverably reject s_i. By contrast, CRANE will just assign s_i a small penalty for its proximity and in the absence of corroborating evidence from other negative instances, s_i's rank will not change significantly.

Before detailing the specifics of how we apply CRANE to transductive and inductive segment annotation tasks, we discuss some properties of the algorithm that make it particularly suitable to practical implementations. First, as mentioned above, CRANE is robust to noise, whether from incorrect labels or distorted features, confirmed in controlled experiments (see Section 4.5.1). Second, CRANE is explicitly designed to be parallelizable, enabling it to employ large numbers of negative instances. Motivated by Siva *et al.* [133]'s observation regarding the abundance of negative data,

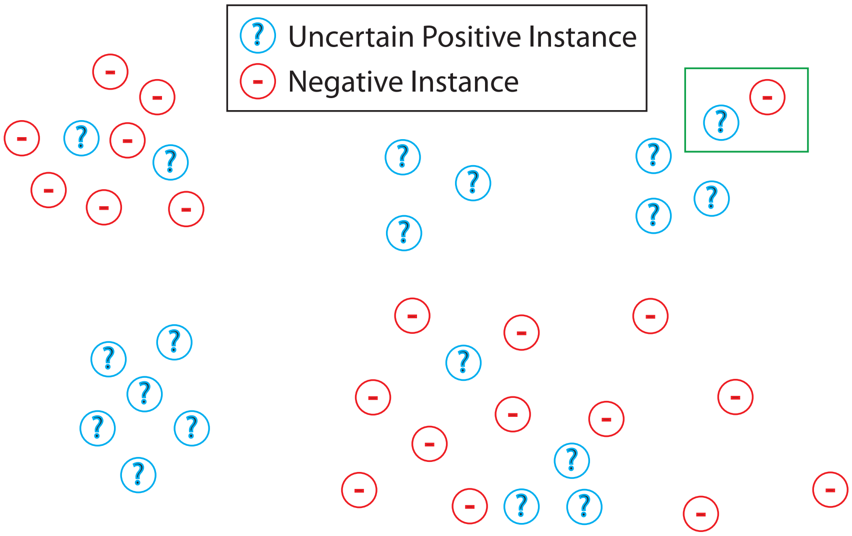


Figure 4.5: Intuition behind CRANE. Positive instances are less likely to be concept segments if they are near many negatives. The green box contrasts CRANE with MIN [133] as discussed in text.

our proposed approach enforces independence among negative instances (i.e., explicitly avoids using the data from block C of the distance matrix). This property enables CRANE's computation to be decomposed over a large number of machines simply by replicating the positive instances, partitioning the (much larger) negative instances, and trivially aggregating the resulting scores.

4.4.1 Application to transductive segment annotation

Applying CRANE to transductive segment annotation is straightforward. We generate weakly labeled positive and negative instances for each concept. Then we use CRANE to rank all of the segments in the positive set according to this score. Thresholding the list at a particular rank creates a partitioning into P_c and P_b; sweeping the threshold generates the precision/recall curves shown in Figure 4.6.

Class	Shots	Frames	Class	Shots	Frames
Aeroplane	9	1423	Cow	20	2978
Bird	6	1206	Dog	27	3803
Boat	17	2779	Horse	17	3990
Car	8	601	Motorbike	11	829
Cat	18	4794	Train	18	3270
Total Shots		151	Total Frames		25673

Table 4.1: Details for our annotations on the YouTube-Objects dataset [113]. Each shot comes from a different video, as we do not annotate multiple shots in the same video.

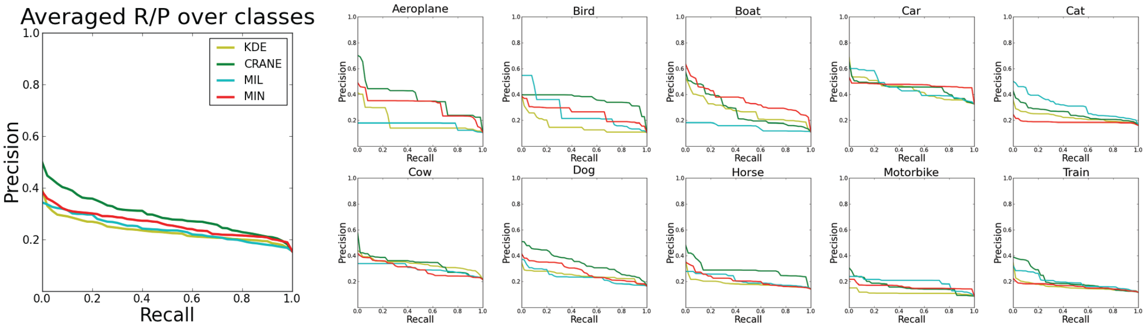


Figure 4.6: Direct comparison of several approaches for transductive segment annotation on the YouTube-Objects dataset [113].

4.4.2 Application to inductive segment annotation

In the inductive segment annotation task, for each concept, we are given a large number of weakly tagged positive and negative videos, from which we learn a set of segment-level classifiers that can be applied to arbitrary weakly tagged test videos. Inductive segment annotation can be decomposed into a two-stage problem. The first stage is identical to TSA. In the second stage, the most confident predictions for concept segments (from the first stage) are treated as segment-level labels. Using these and our large set of negative instances, we train a standard fully supervised classifier. To evaluate the performance of ISA, we apply the trained classifier to a disjoint test set and generate precision/recall curves, such as those shown in Figure 4.8.

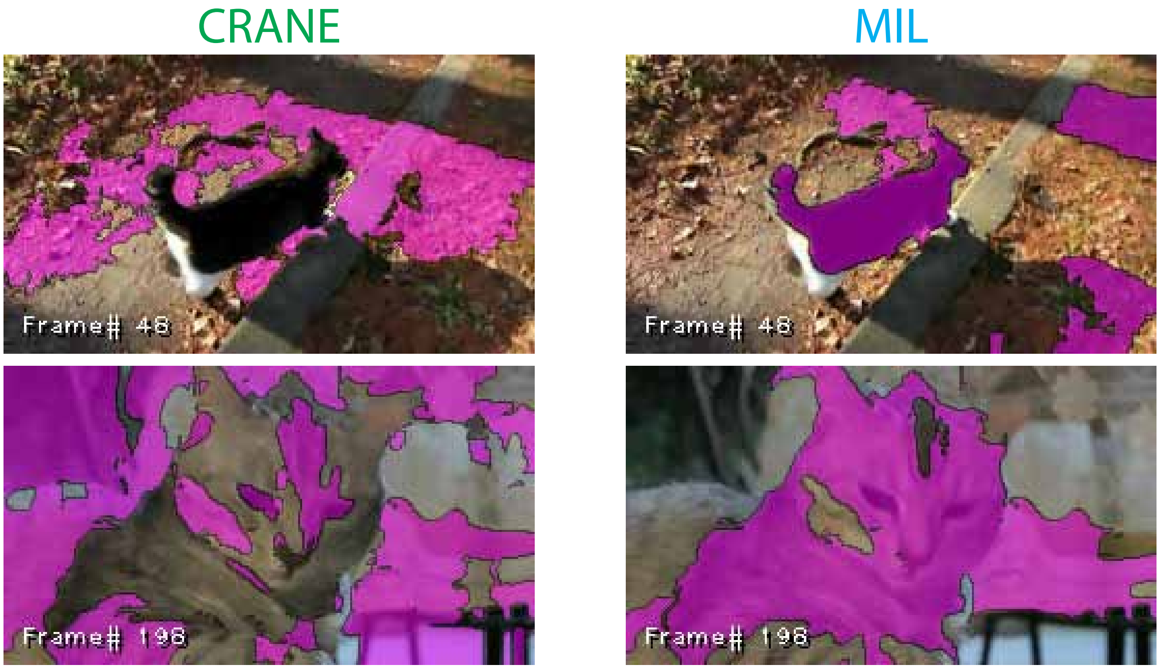


Figure 4.7: Visualizations of instances for the "cat" class where MIL is better able to distinguish between the similar looking concept and background segments.

4.5 Experiments

To evaluate the different methods, we score each segment in our test videos, rank segments in decreasing order of score and compute precision/recall curves. As discussed above, the test videos for TSA are available during training, whereas those for ISA are disjoint from the training videos.

4.5.1 Transductive segment annotation (TSA)

To evaluate transductive segment annotation, we use the YouTube-Objects (YTO) dataset [113], which consists of videos collected for 10 of the classes from the PASCAL Visual Objects Challenge [33]. We generate a groundtruthed test set by manually annotating the first shot from each video with segment-level object annotations, resulting in a total of 151 shots with a total of 25,673 frames (see Table 4.1) and 87,791 segments. We skip videos for which the object did not occur in the first shot and shots with severe undersegmentation problems. Since there is increasing interest in training image classifiers using video data [113, 143], our hope is to identify methods that can "clean" weakly supervised video to generate suitable data for training supervised classifiers for image challenges such as PASCAL VOC.

Implementation details. We represent each segment using the following set of features: RGB color histograms quantized over 20 bins, histograms of local binary patterns computed on 5×5 patches [103, 159], histograms of dense optical flow [15], heat maps computed over an 8×6 grid to represent the (x, y) shape of each segment (averaged over time), and histograms of quantized SIFT-like local descriptors extracted densely within each segment. For negative data, we sample 5000 segments from videos tagged with other classes; our experiments show that additional negative data increases computation time but does not significantly affect results for any of the methods on this dataset.

We use the L2 distance for the distance function in relevant methods, and for the cutoff function in CRANE, we simply use a constant, $f_{\text{cut}}(\cdot) = 1$. Experiments with cutoff functions such as step, ramp and Gaussian show that the constant performs just as well and requires no parameters.

Direct comparisons. We compare CRANE against several methods. MIL refers to Multiple Instance Learning, the standard approach for problems similar to our scenario. In our experiments, we use the MILBoost algorithm with ISR criterion [156], and sparse boosting with decision stumps [30] as the base classifier. MIN refers to the method of [133], which uses the minimum distance for each positive instance as the score for the instance. KDE refers to Kernel Density Estimation, which estimates the probability distribution of the negatives, and then computes the probability that each positive instance was generated from this distribution.

Discussion. Figure 4.6 shows that our method outperforms all other methods in overall precision/recall. In particular, we perform much better for the "aeroplane", "dog", "horse", and "train" classes. Interestingly, for the "cat" class, MIL performs

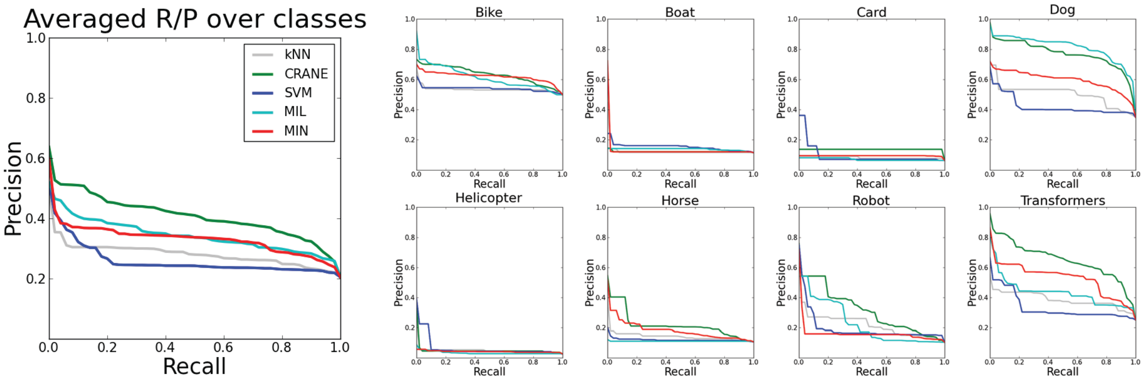


Figure 4.8: Direct comparison of several methods for inductive segment annotation using the object segmentation dataset [47].

very well whereas all other methods do poorly. By visualizing the segments (see Figure 4.7), we see that in many videos, the cat and background segments are very similar in appearance. MIL is able to focus on these minor differences while the others do not. MIN [133] performs second best on this task after CRANE. However, because it only considers the minimum distance from a positive instance to a negative instance, it is more susceptible to label noise.

The transductive segment annotation scenario is useful for directly comparing various weakly supervised learning methods in a classifier-independent manner. However, TSA is of limited practical use as it requires that each segment from every input video be compared against the negative data. By contrast, ISA assumes that once a segment-level concept model has been learned (using sufficient data to span the concept's intra-class variability), the model can be applied relatively efficiently to arbitrary input videos.

4.5.2 Inductive segment annotation (ISA)

For the task of inductive segment annotation, where we learn a segment-level classifier from weakly labeled video, we use the dataset introduced by [47], as this dataset contains a large number of weakly labeled videos and deals exactly with this task. This dataset consists of 20,000 Internet videos from 8 classes: "bike", "boat", "card",

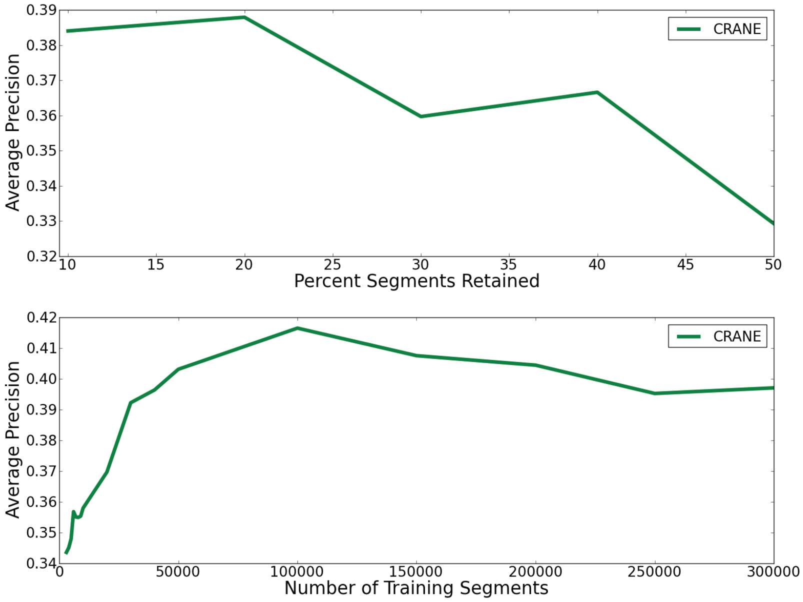


Figure 4.9: Average precision as we vary CRANE's fraction of retained segments (top) and number of training segments (bottom).

"dog", "helicopter", "horse", "robot", and "transformer". Additional videos from several other tags are used to increase the set of negative background videos. These videos are used for training, and a separate, disjoint set of test videos from these 8 concept classes is used for evaluation.

Implementation details. Due to the computational limitations of the MIL baseline, we limit the training set to 200,000 segments, equally divided among samples from $\mathcal{P}$ and $\mathcal{N}$. For segment features, we use RGB color histograms and histograms of local binary patterns. For both CRANE and MIN, we retain the top 20% of the ranked segments from $\mathcal{P}$ as positive training data for the second stage segment classifier. To simplify direct comparisons, we use k-nearest neighbor (kNN) as the second-stage classifier, with k=20 and probabilistic output for x generated as the ratio to closest negative vs. closest positive: $\min_{n\in\mathcal{N}} ||x-n|| / \min_{p\in\mathcal{P}} ||x-p||$.

Direct comparisons. In addition to several of the stronger methods from the TSA task, we add two baselines for the ISA task: (1) kNN denotes the same second-stage classifier, but using all of the data $\mathcal{P} \cup \mathcal{N}$; (2) SVM refers to a linear support vector machine implemented using LIBLINEAR [34] that was reported to do well by [47] on their task.

Discussion. Figure 4.8 shows that CRANE significantly outperforms the others in overall precision/recall and dominates in most of the per-class comparisons. In particular, we see strong gains (except on "dog") vs. MIL, which is important because [47] was unable to show significant gains over MIL on this dataset. SVM trained with label noise performs worst, except for a few low-recall regions where SVM does slightly better, but no method performs particularly well.

Figure 4.9 (top) examines how CRANE's average precision on ISA varies with the fraction of retained segments. As expected, if we retain too few segments, we do not span the intra-class variability of the target concept; conversely, retaining too many concepts risks including background segments and consequently corrupting the learned classifier. Figure 4.9 (bottom) shows the effect of additional training data (with 20% retained segments). We see that average precision improves quickly with training data and plateaus around 0.4 once we exceed 100,000 training segments.

Figure 4.10 shows example successes and failures for CRANE under both TSA and ISA settings. We stress that these results (unlike those in [47]) are the raw outputs of independent segment-level classification and employ no intra-segment post-processing to smooth labels. Observations on successes: we segment multiple non-centered objects (top-left), which is difficult for GrabCut-based methods [118]; we highlight the horse but not the visually salient ball, improving over [47]; we find the speedboat but not the moving water. CRANE can occasionally fail in clutter (top right) or when segmentations are of low quality (cruise ship + water).

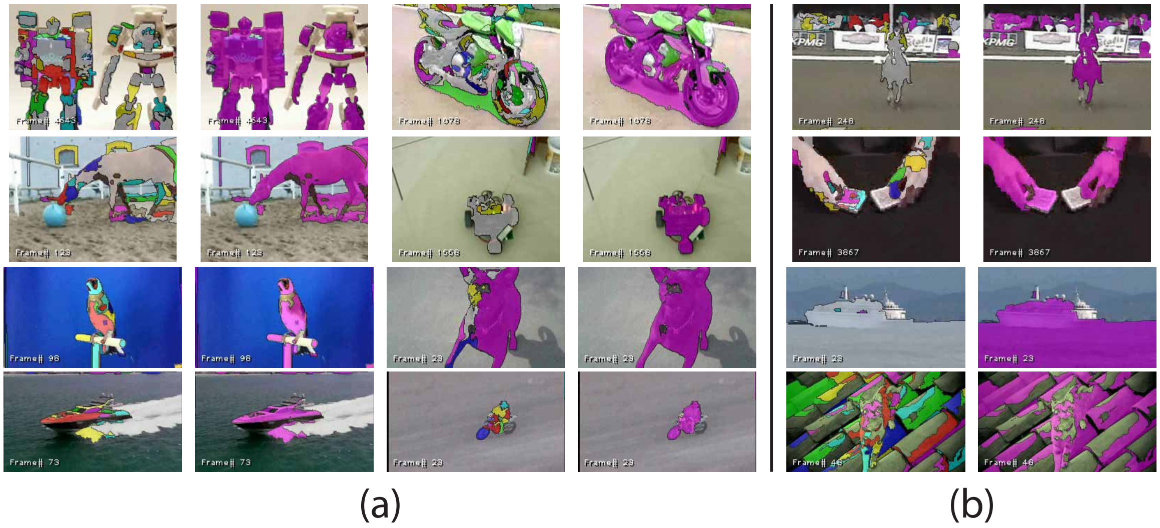


Figure 4.10: Object segmentations obtained using CRANE. The top two rows are obtained for the ISA task on the dataset introduced by [47]. The bottom two rows are obtained for the TSA task on the YouTube-Objects dataset [113]. In each pair, the left image shows the original spatiotemporal segments and the right shows the output. (a) Successes; (b) Failures.

4.6 Summary

We introduce CRANE, a surprisingly simple yet effective algorithm for annotating spatiotemporal segments from video-level labels. We also present a generalized interpretation based on the distance matrix that serves as a taxonomy for weakly supervised methods and provides a deeper understanding of this problem. We describe two related scenarios of the segment annotation problem (TSA and ISA) and present comprehensive experiments on published datasets. CRANE outperforms the recent methods [47, 133] as well as our baselines on both TSA and ISA tasks.

Chapter 5

Co-localization I: Real-World Images

5.1 Introduction

Object detection and localization has long been a cornerstone problem in computer vision. Given the variability of objects and clutter in images, this is a highly challenging problem. Most state-of-the-art methods require extensive guidance in training, using large numbers of images with human-annotated bounding boxes [48, 137]. Recent works have begun to explore weakly-supervised frameworks [28, 65, 98, 107, 134, 143], where labels are only given at the image level. Inspired by these works, we focus on the problem of unsupervised object detection through co-localization, which further relaxes the need for annotations by only requiring a set of images that each contain *some* common object we would like to localize.

We tackle co-localization in real-world settings where the objects display a large degree of variability, and worse, the labels at the image level can be noisy (see Figure 5.1). Although recent works have tried to explicitly deal with annotation noise [119, 144, 151], most previous works related to co-localization have assumed clean labels, which is not a realistic assumption in many real-world settings where we have to analyze large numbers of Internet images or discover objects with roaming robots. Our aim is therefore to overcome the challenges posed by noisy images and

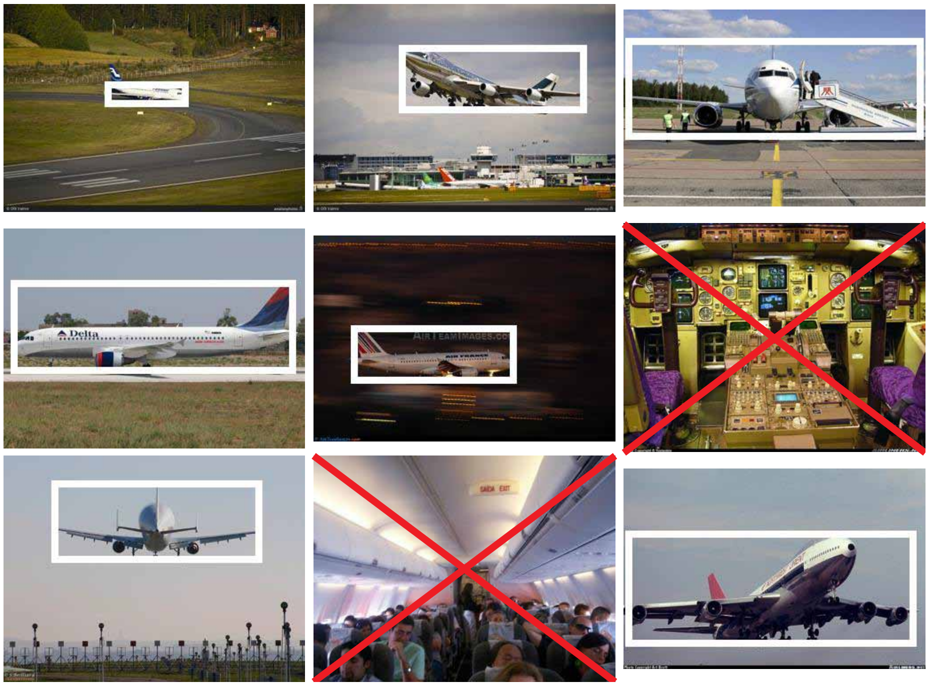


Figure 5.1: The co-localization problem in real-world images. In this instance, the goal is to localize the airplane within each image.

object variability.

We propose a formulation for co-localization that combines an image model and a box model into a joint optimization problem. Our image model addresses the problem of annotation noise by identifying incorrectly annotated images in the set, while our box model addresses the problem of object variability by localizing the common object in each image using rich correspondence information. The joint image-box formulation allows the image model to benefit from localized box information, and the box model to benefit by avoiding incorrectly annotated images.

To illustrate the effectiveness of our method, we present results on three challenging, real-world datasets that are representative of the difficulties of intra-class variation, inter-class diversity, and annotation noise present in real-world images. We outperform previous state-of-the-art approaches on standard datasets, and also show how the joint image-box model is better at detecting incorrectly annotated images. Finally, we present a large-scale study of co-localization on ImageNet [26], involving ground-truth annotations for 3,624 classes and 939,542 images. The largest previous study of co-segmentation on ImageNet consisted of ground-truth annotations for 446 classes and 4,460 images [78].

5.2 Related Work

Co-localization shares the same type of input as co-segmentation [63, 64, 71, 78, 119, 155], where we must find a common object within a set of images. However, instead of segmentations, we seek to localize objects with bounding boxes. Considering boxes allows us to greatly decrease the number of variables in our problem, as we label boxes instead of pixels. It also allows us to extract rich features from within the boxes to compare across images, which has shown to be very helpful for detection [124].

Co-localization shares the same type of output as weakly supervised localization [28, 98, 107, 134], where we draw bounding boxes around objects without any strong supervision. The key difference is that in co-localization we have a more relaxed scenario, where we do not know what the object contained in our set of images is, and are not given negative images for which we know do not contain our object. Most similar is [28], which generates candidate bounding boxes and tries to select the correct box within each image using a conditional random field. Object co-detection [6] also shares similarities, but is given additional bounding box and correspondence annotations.

Although co-localization shares similarities with both co-segmentation and weakly supervised localization, an important and new difficulty we address in this chapter is the problem of noisy annotations, which has recently been considered [119, 144, 151]. Most similar is [119], where the authors utilize dense correspondences to ignore incorrect images. We combine an image model that detects incorrectly annotated images with a box model that localizes the common object, which sets us apart from previous work. The objective functions in our models are inspired by works from outlier detection [51], image segmentation [130], and discriminative clustering [5, 63, 165]. Previous works have considered combining object detection with image classification [48, 137], but only in supervised scenarios.

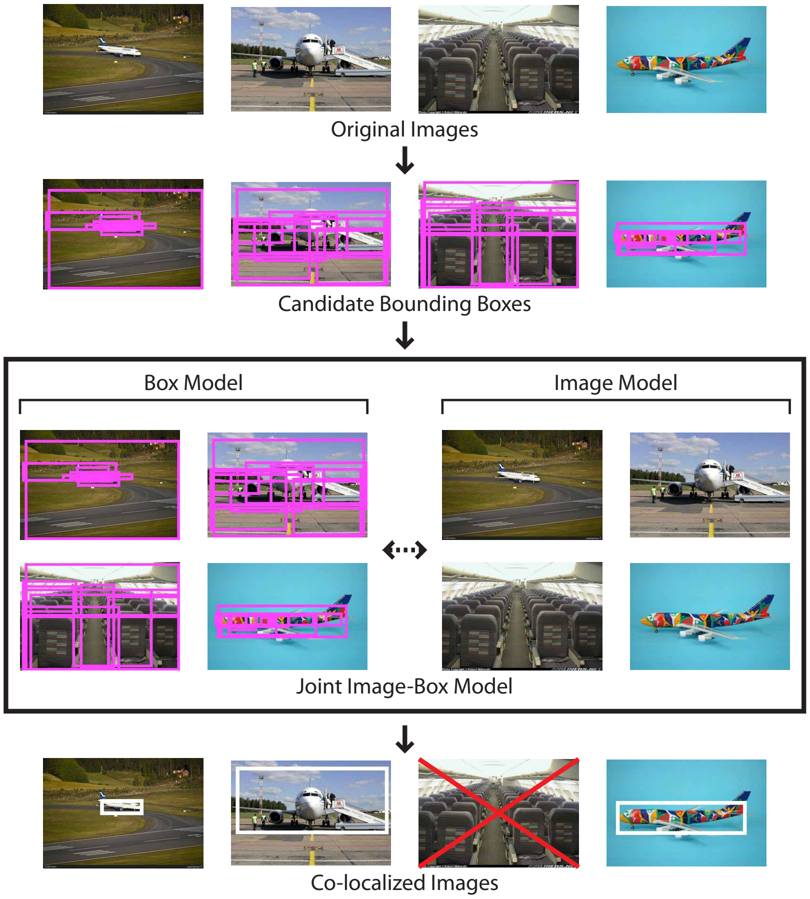


Figure 5.2: Overview of our approach to co-localization.

5.3 Our Approach

Given a set of n images $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$, our goal is to localize the common object in each image. In addition, we also consider the fact that due to noise in the process of collecting this set, some images may not contain the common object. We denote these as *noisy* images, as opposed to *clean* images, which contain the common object. Our goal is to simultaneously identify the noisy images and localize the common object in the clean images.

An overview of our approach is given in Figure 5.2. We start by generating a set of candidate boxes for each image that could potentially contain an object. Then, we formulate an image model for selecting the clean images, and a box model for selecting the box in each image that contains an instance of the common object. We denote the boxes that contain an instance of the common object as *positive* boxes, and the ones that don't as *negative* boxes.

Combining the two models into a joint formulation, we allow the image model to prevent the box model from being adversely affected by boxes in noisy images, and allow the box model to help the image model determine noisy images based on localized information in the images. Similar approaches have been considered [28], but only using a box model and only in the context of clean images.

5.3.1 Generating candidate boxes

We use the measure of objectness [2], but any method that is able to generate a set of candidate regions can be used [13, 124]. The objectness measure works by combining multiple image cues such as multi-scale saliency, color contrast, edge density, and superpixel straddling to generate a set of candidate regions as well as scores associated with each region that denote the probability a generic object is present in the region. Examples of candidate boxes generated by objectness can be seen in Figure 5.2.

Using the objectness measure, for each image $I_j \in \mathcal{I}$, we generate a set of m candidate boxes $\mathcal{B}_j = \{b_{j,1}, b_{j,2}, \ldots, b_{j,m}\}$, ordered by their objectness score.

5.3.2 Model setup

Given a set of images $\mathcal{I}$ and a set of boxes $\mathcal{B}_j$ for each image $I_j \in \mathcal{I}$, our goal is to jointly determine the noisy images and select the positive box from each clean image. To simplify notation, we define the set of all boxes as $\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2 \ldots \cup \mathcal{B}_n$ and $n_b = nm$ the total number of boxes.

Feature representation. For each box $b_k \in \mathcal{B}$, we compute a feature representation of the box as $x_k^{box} \in \mathbb{R}^d$, and stack the feature vectors to form a feature matrix $X_{box} \in \mathbb{R}^{n_b \times d}$. Similarly for each image $I_j \in \mathcal{I}$, we compute a feature representation of the image as $x_j^{im} \in \mathbb{R}^d$, and stack the feature vectors to form a feature matrix

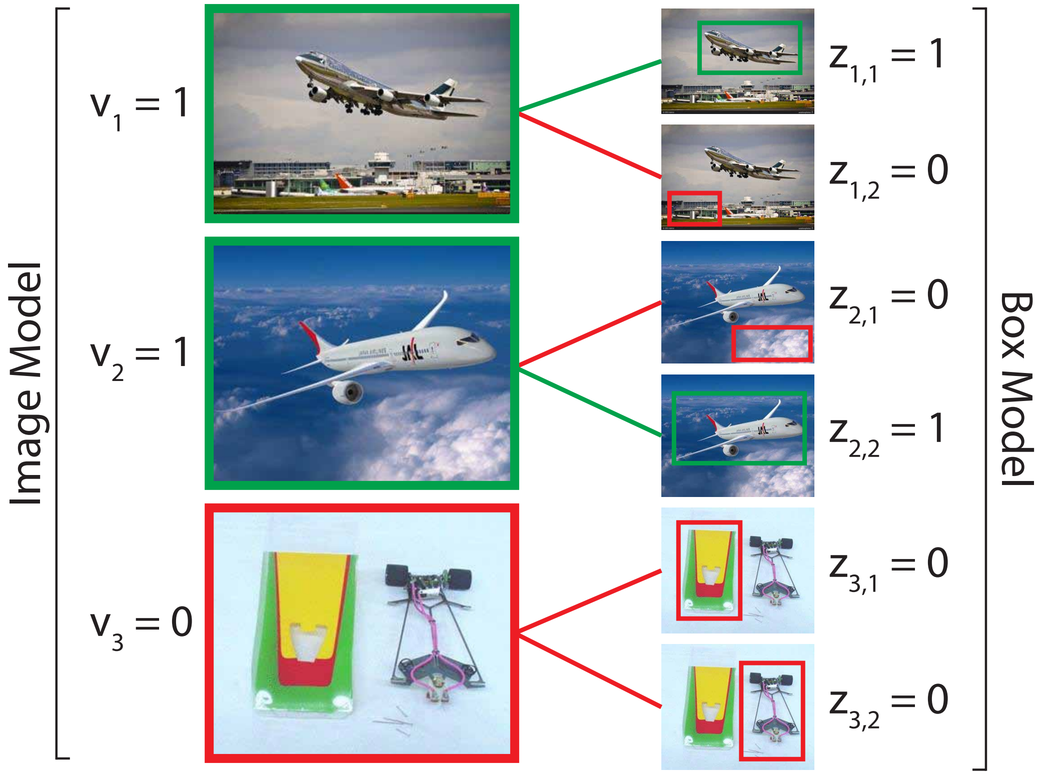


Figure 5.3: The variables v in the image model relate to the variables z in the box model through constraints that ensure noisy images (red) do not select any boxes, while clean images (green) select a single box as the postive box.

$X_{im} \in \mathbb{R}^{n \times d}$. We densely extract SIFT features [92] every 4 pixels and vector quantize each descriptor into a 1,000 word codebook. For each box, we pool the SIFT features within the box using 1×1 and 3×3 SPM pooling regions [84], and for each image, we use the same pooling regions over the entire image to generate a $d = 10,000$ dimensional feature descriptor for each box and each image.

Optimization variables. We associate with each image $I_j \in \mathcal{I}$ a binary label variable v_j, which is equal to 1 if I_j is a clean image and 0 otherwise. Similarly, we associate with each box $b_{j,k} \in \mathcal{B}_j$ a binary label variable $z_{j,k}$, which is equal to 1 if $b_{j,k}$ is a positive box and 0 otherwise. We denote by v, the n dimensional vector $v = (v_1, \ldots, v_n)^T$ and by z the n_b dimensional vector obtained by stacking the $z_{j,k}$.

Making the assumption that in each clean image there is only one positive box, and in each noisy image there are no positive boxes, we define a constraint that relates the two sets of variables:

$$\forall I_j \in \mathcal{I}, \sum_{k=1}^{m} z_{j,k} = v_j. \tag{5.1}$$

This constraint is also illustrated in Figure 5.3, where we show the relationship between image and box variables.

5.3.3 Model formulation

We begin by introducing and motivating the terms in our objective function that enable us to jointly identify noisy images and select the positive box from each clean image.

Box prior. We introduce a prior for each box that represents our belief that the box is positive. We compute an off-the-shelf saliency map for each image [17, 111], and for each box we compute the average saliency within the box, weighted by the size of the box, and stack these values into the n_b dimensional vector m_{box} to obtain a linear term that penalizes less salient boxes:

$$f_{Pbox}(z) = -z^T \log(m_{box}). \tag{5.2}$$

Although objectness also provides scores for each box, we found that the saliency measure used in objectness is dated and does not work as well.

Image prior. We introduce a prior for each image that represents our belief that the image is a clean image. For each image, we compute the χ^2 distance, defined further below, from the image feature to the average image feature in the set, and stack these values into the n dimensional vector m_{im} to obtain a linear term that penalizes outlier images:

$$f_{Pim}(v) = v^T m_{im}. \tag{5.3}$$

We experimented with several measures for outlier detection [51], but found that this simple distance worked well.

Box similarity. We encourage boxes with similar appearances to have the same label through a similarity matrix based on the box feature described above. Since this feature is a histogram, we compute a $n_b \times n_b$ similarity matrix S based on the χ^2-distance:

$$S_{ij} = \exp\left(-\gamma \sum_{k=1}^{d} \frac{(x_{ik}^{box} - x_{jk}^{box})^2}{x_{ik}^{box} + x_{jk}^{box}} \right), \tag{5.4}$$

where $\gamma = (10d)^{-\frac{1}{2}}$. We set the similarity of boxes from the same image to be 0. We then compute the normalized Laplacian matrix $L_{box} = I - D^{-\frac{1}{2}} S D^{-\frac{1}{2}}$, where D is the diagonal matrix composed of the row sums of S, resulting in a quadratic term that encourages the selection of similar boxes:

$$f_{Sbox}(z) = z^T L_{box} z. \tag{5.5}$$

This choice is motivated by the work of Shi and Malik [130], who have shown that considering the second smallest eigenvector of a normalized Laplacian matrix leads to clustering z along the graph defined by the similarity matrix, leading to Normalized Cuts when used for image segmentation. Furthermore, Belkin and Niyogi [7] have shown that minimizing Equation 5.5 under linear constraints results in an equivalent problem. The similarity term can be interpreted as a a generative term that seeks to select boxes that cluster well together.

Image similarity. We also encourage images with similar appearances to have the same label through a similarity matrix based on the image feature described above.

Replacing the box features with image features in Equation 5.4, we compute a $n \times n$ similarity matrix and subsequently the normalized Laplacian matrix L_{im} to obtain a quadratic term that encourages the selection of similar images:

$$f_{Sim}(v) = v^T L_{im} v. \tag{5.6}$$

Box discriminability. Discriminative learning techniques such as the support vector machine and ridge regression have been widely used within the computer vision community to obtain state-of-the-art performance on many supervised problems. We can take advantage of these methods even in our unsupervised scenario, where we do not know the labels of our boxes [5, 165]. Following [63], we consider the ridge regression objective function for our boxes:

$$\min_{\substack{w \in \mathbb{R}^d, \\ c \in \mathbb{R}}} \frac{1}{n_b} \sum_{j=1}^{n} \sum_{k=1}^{m} ||z_{j,k} - w x_{j,k}^{box} - c||_2^2 + \frac{\kappa}{d} ||w||_2^2, \tag{5.7}$$

where w is the d dimensional weight vector of the classifier, and c is the bias. The choice of ridge regression over other discriminative cost functions is motivated by the fact that the ridge regression problem has a closed form solution for the weights w and bias c, leading to a quadratic function in the box labels [5]:

$$f_{Dbox}(z) = z^T A_{box} z, \tag{5.8}$$

where $A_{box} = \frac{1}{n_b}(\Pi_{n_b}(I_{n_b} - X_{box}(X_{box}^T \Pi_{n_b} X_{box} + n_b \kappa I)^{-1} X_{box}^T)\Pi_{n_b})$ and $\Pi_{n_b} = I_{n_b} - \frac{1}{n_b} 1_{n_b} 1_{n_b}^T$ is the centering projection matrix. We know also that A_{box} is a positive semi-definite matrix [49]. This quadratic term allows us to utilize a discriminative objective function to penalize the selection of boxes whose features are not easily linearly separable from the other boxes.

Image discriminability. Similar to the box discriminability term, we also employ a discriminative objective to ensure that the features of the clean images should be easily linearly separable from noisy images. Replacing the box features in Equation 5.7

with image features, we can similarly substitute the solutions for w and c to obtain:

$$f_{Dim}(v) = v^T A_{im} v, \tag{5.9}$$

where A_{im} is defined in the same way as A_{box}, replacing box features with image features.

Joint formulation. Combining the terms presented above, we obtain the following optimization problem:

$$\begin{aligned}
\underset{z,v}{\text{minimize}} \quad & z^T(L_{box} + \mu A_{box})z - z^T \lambda \log(m_{box}) \\
& + \alpha(v^T(L_{im} + \mu A_{im})v + v^T \lambda m_{im}) \\
\text{subject to} \quad & v \in \{0,1\}, z \in \{0,1\} \\
& \forall I_j \in \mathcal{I}, \sum_{k=1}^{m} z_{j,k} = v_j \\
& K_0 \leq \sum_{i=1}^{n} v_i,
\end{aligned} \tag{5.10}$$

where the constraints in the formulation ensure that only a single box is selected in clean images, and none in noisy images. Using the constant K_0, we can avoid trivial solutions and incorporate an estimate of noise by allowing noisy images to not contain boxes. This prevents the boxes in the noisy images from adversely affecting the box similarity and discriminability terms.

The parameter μ controls the tradeoff between the quadratic terms, the parameter λ controls the tradeoff between the linear and quadratic terms, and the parameter α controls the tradeoff between the image and box models. Since the matrices L_{box}, A_{box}, L_{im}, and A_{im} are each positive semi-definite, the objective function is convex.

Convex relaxation. In Equation 5.10, we obtain a standard boolean constrained quadratic program. The only sources of non-convexity in this problem are the boolean

	aeroplane		bicycle		boat		bus		horse		motorbike		
Method	left	right	left	right	left	right	left	right	left	right	left	right	Average
Our Method (prior)	13.95	20.51	10.42	8.00	2.27	6.98	9.52	13.04	12.50	13.04	17.95	23.53	12.64
Our Method (prior+similarity)	39.53	35.90	**25.00**	**24.00**	0.00	2.33	23.81	34.78	37.50	43.48	48.72	58.82	31.16
Our Method (full)	**41.86**	**51.28**	**25.00**	**24.00**	**11.36**	**11.63**	**38.10**	**56.52**	**43.75**	**52.17**	**51.28**	**64.71**	**39.31**

Table 5.1: CorLoc results for various combinations of terms in our box model on PASCAL07-6x2.

constraints on v and z. We relax the boolean constraints to continuous, linear constraints, allowing v and z to take any value between 0 and 1. This becomes a convex optimization problem and can be solved efficiently using standard methods.

Given the solution to the quadratic program, we reconstruct the solution to the original boolean constrained problem by thresholding the values of v to obtain the noisy images, and simply taking the box from each clean image with the highest value of z.

5.4 Results

We perform experiments on three challenging datasets, the PASCAL VOC 2007 dataset [33], the Object Discovery dataset [119], and ImageNet [26]. Following previous works in weakly supervised localization [28], we use the CorLoc evaluation metric, defined as the percentage of images correctly localized according to the PASCAL-criterion: $\frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} > 0.5$, where B_p is the predicted box and B_{gt} is the ground-truth box. All CorLoc results are given in percentages.

5.4.1 Implementation details and runtime

We set the parameters of our method to be $\mu = 0.6$, $\lambda = 0.001$, and $\alpha = 1$, and tweaked them slightly for each dataset. We set $\kappa = 0.01$ in the ridge regression objective. Because there are no noisy images for PASCAL and ImageNet, we fix the value of $K_0 = n$ for these datasets. For the Object Discovery dataset, we set $K_0 = 0.8n$. We use 10 objectness boxes for ImageNet, and 20 objectness boxes for the other datasets.

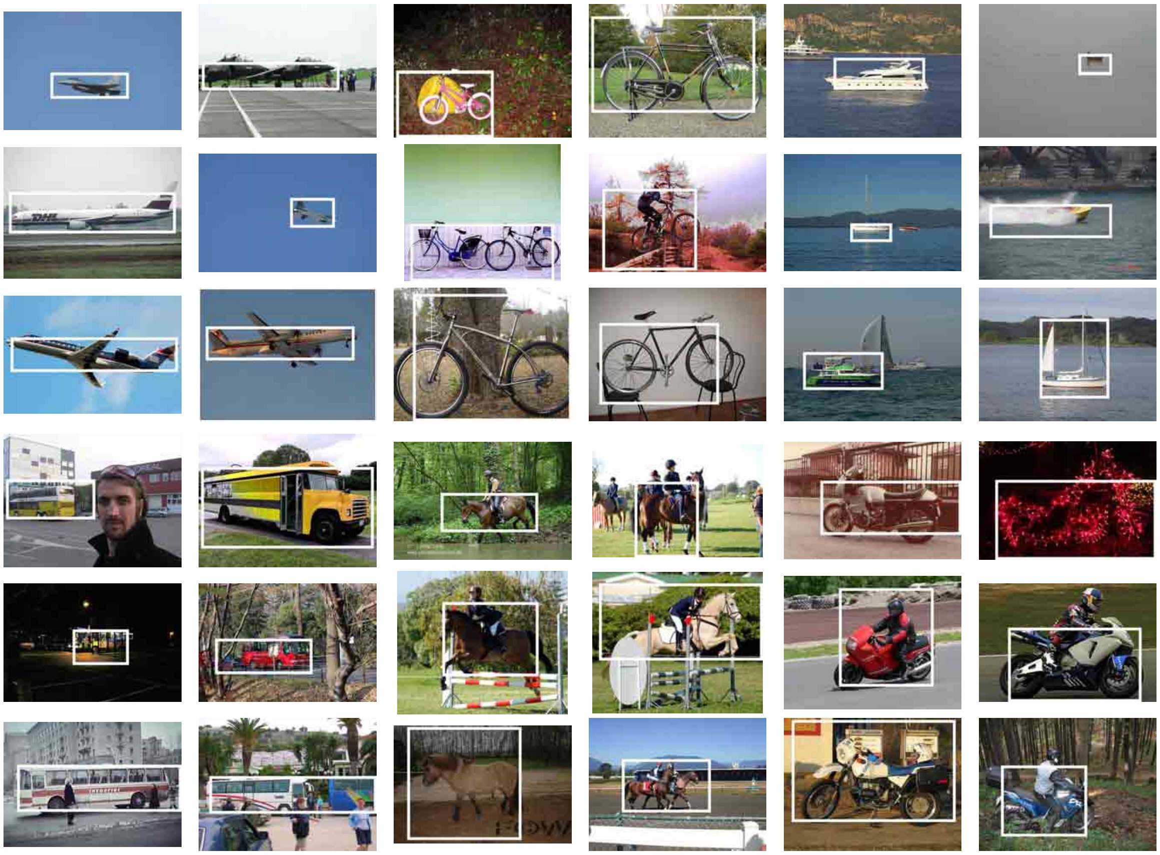


Figure 5.4: Example co-localization results on PASCAL07-6x2. Every three images in each column contains images from the same class/viewpoint combination.

After computing candidate object boxes using objectness and densely extracting SIFT features, we are able to co-localize a set of 100 images with 10 boxes per image in less than 1 minute on a single machine using code written in Python and a quadratic program solver written in C++.

5.4.2 PASCAL VOC 2007

Following the experimental setup defined in [28], we evaluate our method on the PASCAL07-6x2 subset to compare to previous methods for co-localization. This subset consists of all images from 6 classes (aeroplane, bicycle, boat, bus, horse, and motorbike) of the PASCAL VOC 2007 [33] train+val dataset from the left and right

Method	Average CorLoc
Russell *et al.* [121]	22
Chum and Zisserman [21]	33
Deselaers *et al.* [28]	37
Our Method	**39**

Table 5.2: CorLoc results compared to previous methods on PASCAL07-6x2.

Method	Airplane	Car	Horse	Average CorLoc
Kim *et al.* [71]	21.95	0	16.13	12.69
Joulin *et al.* [63]	32.93	66.29	54.84	51.35
Joulin *et al.* [64]	57.32	64.04	52.69	58.02
Rubinstein *et al.* [119]	**74.39**	87.64	63.44	75.16
Our Method	71.95	**93.26**	**64.52**	**76.58**

Table 5.3: CorLoc results on the 100 image subset of the Object Discovery dataset.

aspect each. Each of the 12 class/viewpoint combinations contains between 21 and 50 images for a total of 463 images.

In Table 5.1, we analyze each component of our box model by removing various terms in the objective. As expected, we see that results using stripped down versions of our model do not perform as well. In Table 5.2, we show how our full method outperforms previous methods for co-localization that do not utilize negative images. In addition, our method does not incorporate dataset-specific aspect ratio priors for selecting boxes. In Figure 5.4, we show example visualizations of our co-localization method for PASCAL07-6x2. In the bus images, our model is able to co-localize instances in the background, even when other objects are more salient. In the bicycle and motorbike images, we see how our model is able to co-localize instances over a variety of natural and man-made background scenes.

5.4.3 Object Discovery dataset

The Object Discovery dataset [119] was collected by automatically downloading images using the Bing API using queries for airplane, car, and horse, resulting in noisy images that may not contain the query. Introduced as a dataset for co-segmentation,

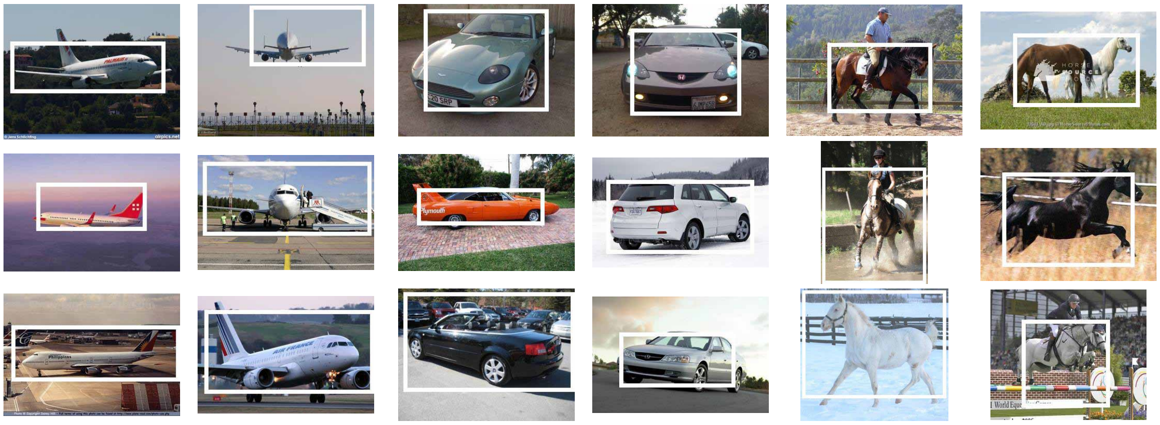


Figure 5.5: Example co-localization results on the Object Discovery dataset, with every two columns belonging to the same class.

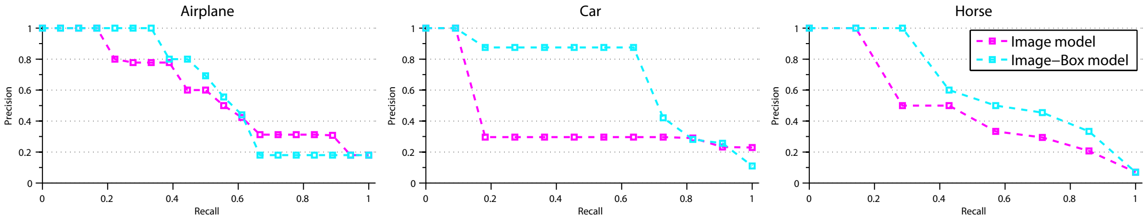


Figure 5.6: Precision-recall curves illustrating the effectiveness of our image-box model (blue) compared to the image model (pink) at identifying noisy images on the Object Discovery dataset.

we convert the ground-truth segmentations and results from previous methods to localization boxes by drawing tight bounding boxes around the segmentations. We use the 100 image subset [119] to enable comparisons to previous state-of-the-art co-segmentation methods. CorLoc results are given in Table 5.3, and example co-localization results are visualized in Figure 5.5(a). From the visualizations, we see how our model is able to handle intra-class variation, being able to co-localize instances of each object class from a wide range of viewpoints, locations, and background scenes. This is in part due to our quadratic terms, which consider the relationships between all pairs of images and boxes, whereas previous methods like [119] rely on sparse image connectivity for computational efficiency.

Method	Average CorLoc
Top objectness box [2]	37.42
Our Method	**53.20**

Table 5.4: CorLoc results on ImageNet evaluated using ground-truth annotations for 3,624 classes and 939,542 images.

We see that our method outperforms previous methods in all cases except for the airplane class. In our results, we see that since our method localizes objects based on boxes instead of segmentations [119], the airplane tail is sometimes excluded from the box, as including the tail would also include large areas of the background. This causes our method to fail in these images due to the non-convex shape of the airplane and the height of the tail.

Detecting noisy images. We also quantitatively measure the ability of our joint image-box model to identify noisy images. Because the solution to the quadratic program gives continuous values for the image variables v, we can interpret the values as a detection score for each image and plot precision-recall curves that measure our ability to correctly detect noisy images, as shown in Figure 5.6. To make comparisons fair, we compare using the best parameters for the image model alone, and the best parameters for our joint image-box model. By jointly optimizing over both image and box models, we see how the box model can correct errors made by the image model by forcing images that have good box similarity and discriminability to be clean, even if the image model believes them to be noisy.

5.4.4 ImageNet

ImageNet [26] is a large-scale ontology of images organized according to the WordNet hierarchy. Each node of the hierarchy is depicted by hundreds and thousands of images. We perform a large-scale evaluation of our co-localization method on ImageNet by co-localizing all images with ground-truth bounding box annotations, resulting in a total of 3,624 classes and 939,542 images. A similar large-scale segmentation experiment [78] only considered ground-truth annotations in 446 classes and 4,460

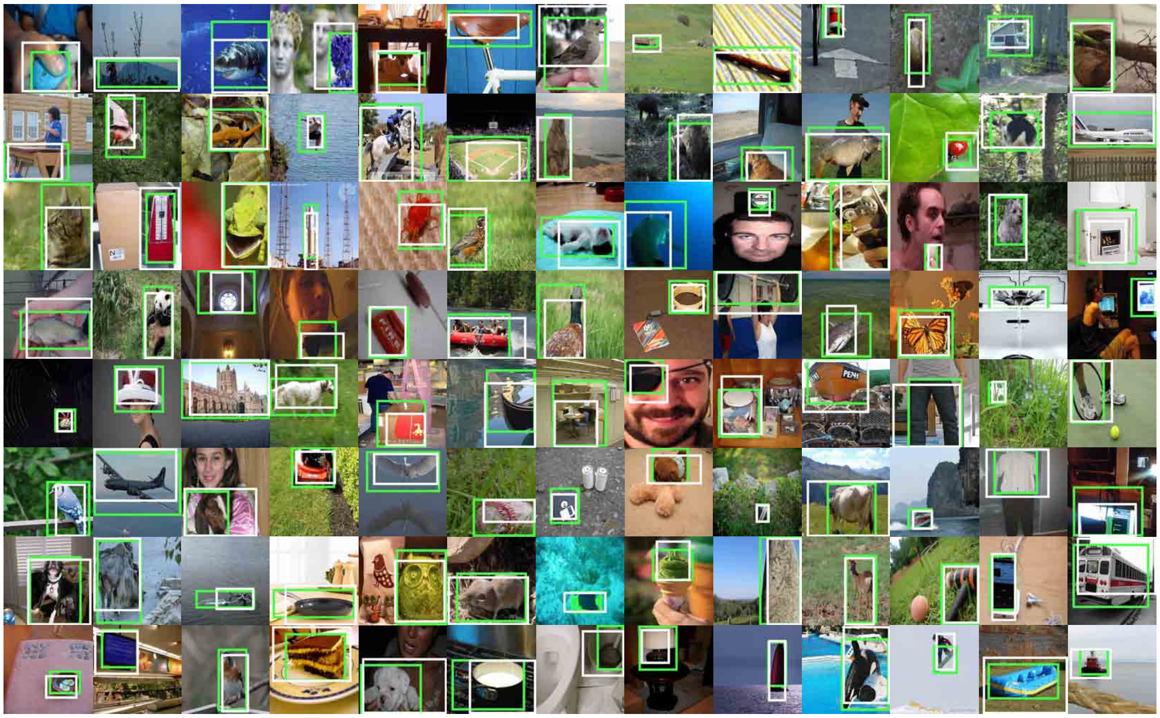


Figure 5.7: Example co-localization results on ImageNet. Each image belongs to a different class, resulting in a total of 104 classes ranging from lady bug to metronome. White boxes are localizations from our method, green boxes are ground-truth localizations.

images. At this scale, the visual variability of images is unprecedented in comparison to previous datasets, causing methods specifically tuned to certain datasets to work poorly.

Due to the scale of ImageNet and lack of code available for previous methods, we compare our method to the highest scoring objectness box [2], which gives a strong baseline for generic object detection. To ensure fair comparisons, we use the objectness score as the box prior for our model in these experiments, with CorLoc results shown in Table 5.4 and visualizations for 104 diverse classes in Figure 5.7.

Box selection. In Figure 5.8(a), we show the distribution over objectness boxes that our method selects. The boxes are ordered by decreasing objectness score, so objectness simply selects the first box in every image. By considering box similarity

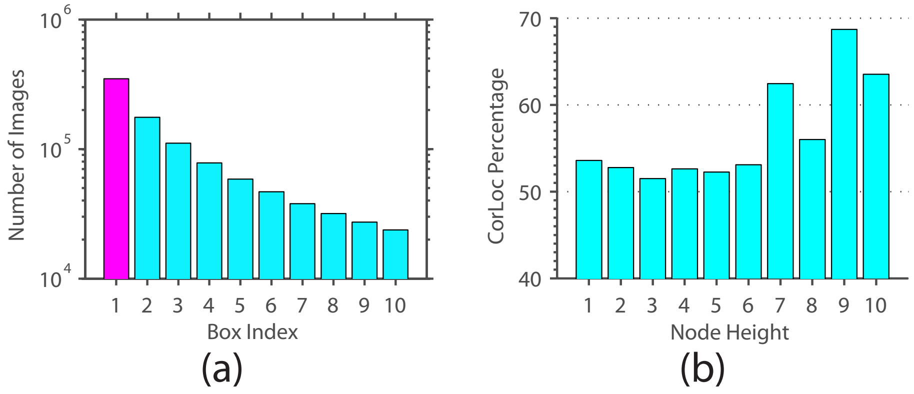


Figure 5.8: (a) Boxes selected by our method on ImageNet, ordered by descending objectness score; (b) CorLoc performance of our method separated into differing node heights of ImageNet.

and discriminability between images, our method identifies boxes that may not have very high objectness score, but are more likely to be the common object.

Effect of ImageNet node height. We also evaluate the performance of our method on different node heights in ImageNet in Figure 5.8(b). Here, a height of 1 is a leaf node, and larger values result in more generic object classes. We see that our method seems to perfom better as we go up the ImageNet hierarchy. This could be because generic objects have more images, and thus our method has more examples to leverage in the box similarity and discriminability terms.

CorLoc difference between methods. In Figure 5.9, we show the CorLoc difference between our method and objectness for all 3,624 classes. From the best CorLoc differences, we find that our method performs much better than objectness on large rooms and objects, which is probably because objectness tries to select individual objects or object parts within these large scenes, whereas our model is able to understand that the individual objects are not similar, and select the scene or object as a whole.

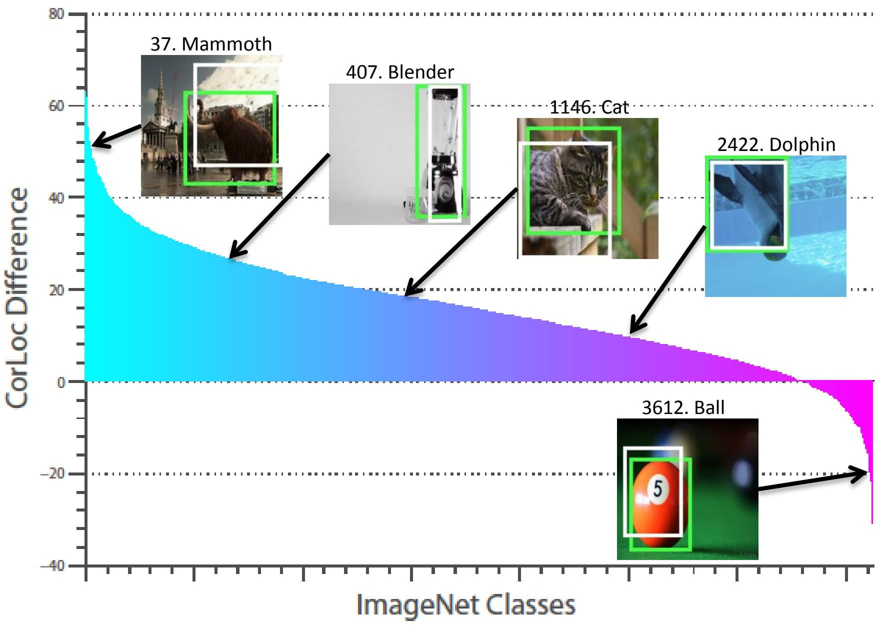


Figure 5.9: CorLoc difference between our method and objectness on all 3,624 classes from ImageNet that we evaluate on.

5.5 Summary

In this chapter, we introduce a method for co-localization in real-world images that combines terms for the prior, similarity, and discriminability of both images and boxes into a joint optimization problem. Our formulation is able to account for noisy images with incorrect annotations. We performed an extensive evaluation of our method on standard datasets, and also performed a large-scale evaluation using ground-truth annotations for 3,624 classes from ImageNet.

Chapter 6

Co-localization II: Efficient Image and Video

6.1 Introduction

With the rising popularity of Internet photo and video sharing sites like Flickr and YouTube, there is a large amount of visual data uploaded to the Internet. In addition to pixels, these images and videos are often tagged with the visual concepts they contain, leading to a natural source of weakly labeled data. Recent research has studied ways of leveraging this data, such as weakly supervised localization [28, 40, 107, 131, 133–135], co-segmentation [63, 71, 119], and co-localization [113, 142].

In this chapter, we address the problem of co-localization in images and videos. Co-localization is the problem of localizing (with bounding boxes) the common object in a set of images or videos. Recent work has studied co-localization in images with potentially noisy labels [142], and co-localization in videos [113] for learning object detectors. Building upon the success of a recent state-of-the-art method [142], we propose a formulation for co-localization in videos that can take advantage of temporal consistency with temporal terms and constraints, while still maintaining a standard quadratic programming formulation. We also show how we can combine both models to perform joint image-video co-localization, the logical way of utilizing all of the weakly supervised data we have available.

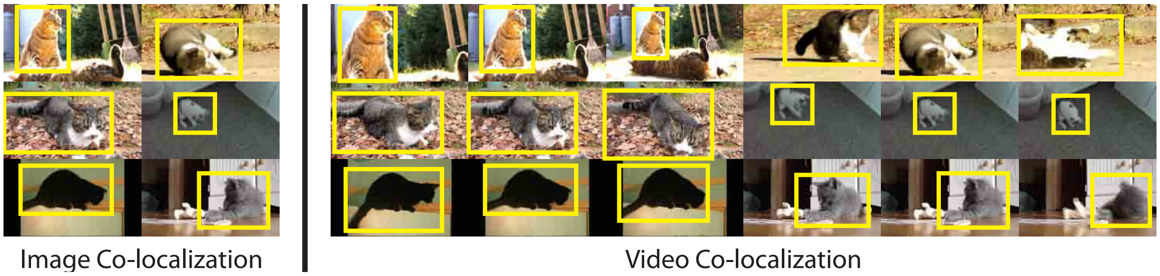


Figure 6.1: In the co-localization problem, our goal is to simultaneously localize the common object of the same class in a set of images or videos.

To efficiently perform co-localization in both images and videos, we show how our optimization problems can be reduced to a succession of simple integer problems using the Frank-Wolfe algorithm (also known as conditional gradient) [36]. For image co-localization, this results in simply taking the maximum of a set of values. For video co-localization, this results in the shortest path algorithm, which can be efficiently solved using dynamic programming.

To re-iterate, we make two key contributions in this chapter.

- **Formulation for video co-localization.** We present a novel formulation for video co-localization, extending [142] with temporal terms and constraints.
- **Frank-Wolfe algorithm for efficient optimization.** We show how the Frank-Wolfe algorithm can be used as in [14] to efficiently solve our optimization problems. We show that it leads to solving a succession of simple integer problems.

We present convincing experiments on two difficult datasets: PASCAL VOC 2007 for images [33], YouTube-Objects for videos [113]. We also show results for joint image-video co-localization by combining our models.

6.2 Related Work

The co-localization problem is similar to co-segmentation [63, 64, 71, 78, 119, 120, 155] and weakly supervised localization (WSL) [28, 40, 98, 107, 131, 133–135, 144]. In contrast to co-segmentation, we seek to localize objects with bounding boxes rather than segmentations, which allows us to greatly decrease the number of variables in our problem. Compared to weakly supervised localization (WSL), we are more flexible because we do not require any negative images for which we know do not contain our object. However, the co-localization problem and the WSL problem are extremely similar, and since both problems utilize the same experimental setup and datasets, we also perform comparisons in our results section.

Our work builds upon the formulation introduced in [142] for co-localization in images, which defines an optimization objective that draws inspiration from works in image segmentation [130] and discriminative clustering [5, 63, 165, 178]. Extending their work, we introduce a formulation for co-localization in videos that incorporates constraints and terms that capture temporal consistency, a key property in videos. We also show how the formulation in [142], as well as our video extension, are able to be efficiently solved using the Frank-Wolfe algorithm [36, 80]. Also similar is [28], which generates candidate bounding boxes and tries to select the correct box within each image. However, while they utilize a conditional random field, we adopt a quadratic programming formulation that can be relaxed and efficiently solved. Similar discrete optimization approaches have been shown to work well in various computer vision applications [10, 24, 25]. Our work is also closely related to Chari et al. [14] where they efficiently solve a quadratic program for multi-object tracking using the Frank-Wolfe algorithm.

For video co-localization, most similar is [113], which also tackles the problem of co-localization in videos by proposing candidate regions and selecting the correct one from each video. In [113], the authors try to leverage temporal information by proposing candidate tubes, which suffers from poor performance even with an optimal learning algorithm. In our formulation, we consider the temporal information directly in our model. Co-localization in video also shares similarities to co-segmentation in video, which has recently been studied in [19].

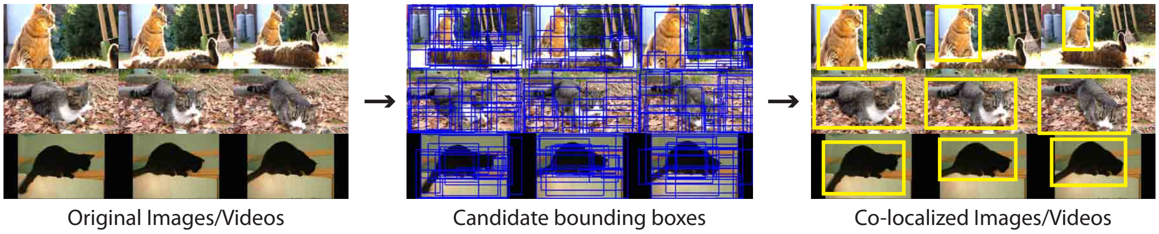


Figure 6.2: An overview of our co-localization approach for images and videos.

6.3 Our Approach

We start by briefly reviewing the co-localization model we use for images [142], and then show how it can be extended to videos. In both models, we take the approach of generating a set of candidate bounding boxes in each image/frame, and then formulating an optimization problem to jointly select the box from each image/frame that contains the common object, as shown in Figure 6.2.

6.3.1 Image Model

Given a set of n images $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$, our goal is to localize the common object in each image. Using objectness [2], we generate m candidate boxes for each image that could potentially contain an object, resulting in a set of boxes $\mathcal{B}_j$ for each image $I_j \in \mathcal{I}$. Our goal then is to jointly select the box from each image that contains the common object. To simplify notation, we define the set of all boxes as $\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2 \ldots \cup \mathcal{B}_n$ and $n_b = nm$ the total number of boxes.

Feature representation. For each box $b_k \in \mathcal{B}$, we compute a feature representation of the box as $x_k \in \mathbb{R}^d$, and stack the feature vectors to form a feature matrix $X \in \mathbb{R}^{n_b \times d}$. We densely extract SIFT features [92] at every pixel and vector quantize each descriptor into a 1,000 word codebook. For each box, we pool the SIFT features within the box using 1×1 and 3×3 SPM pooling regions [84] to generate a $d = 10,000$ dimensional feature descriptor for each box.

Model formulation. We associate with each box $b_{j,k} \in \mathcal{B}_j$ a binary label variable $z_{j,k}$, which is equal to 1 if $b_{j,k}$ contains the common object and 0 otherwise. We denote by z the n_b dimensional vector obtained by stacking the $z_{j,k}$. Making the assumption that in each image there is only one box that contains the common object, we then solve the following optimization problem to select the best box from each image:

$$\begin{aligned} &\underset{z}{\text{minimize}} && z^T(L+\mu A)z - z^T\lambda\log(m) \\ &\text{subject to} && z \in \{0,1\}, \forall I_j \in \mathcal{I} : \sum_{k=1}^{m} z_{j,k} = 1. \end{aligned} \tag{6.1}$$

The parameter μ controls the tradeoff between the quadratic terms, while the parameter λ controls the tradeoff between the linear and quadratic terms. The constraints enforce that only a single box is selected in each image. We briefly describe the terms in the objective below, but more details can be found in [142].

Box prior. The vector m is a prior for each box computed from a saliency map [111] that represents our belief that a box contains the common object given only information within the image.

Box similarity. The matrix $L = I - D^{-\frac{1}{2}}SD^{-\frac{1}{2}}$ is the normalized Laplacian matrix [130], where D is the diagonal matrix composed of the row sums of S, the $n_b \times n_b$ pairwise χ^2-similarity matrix computed from X. We set the similarity between boxes from the same image/video to 0. This matrix encourages boxes with similar appearances from different images/videos to have the same label.

Box discriminability. The matrix:

$$A = \frac{1}{n_b}(\Pi_{n_b}(I_{n_b} - X_{box}(X_{box}^T\Pi_{n_b}X_{box} + n_b\kappa I)^{-1}X_{box}^T)\Pi_{n_b}) \tag{6.2}$$

is the discriminative clustering term [5, 165], where $\Pi_{n_b} = I_{n_b} - \frac{1}{n_b}1_{n_b}1_{n_b}^T$ is the centering projection matrix. This term allows us to utilize a discriminative objective

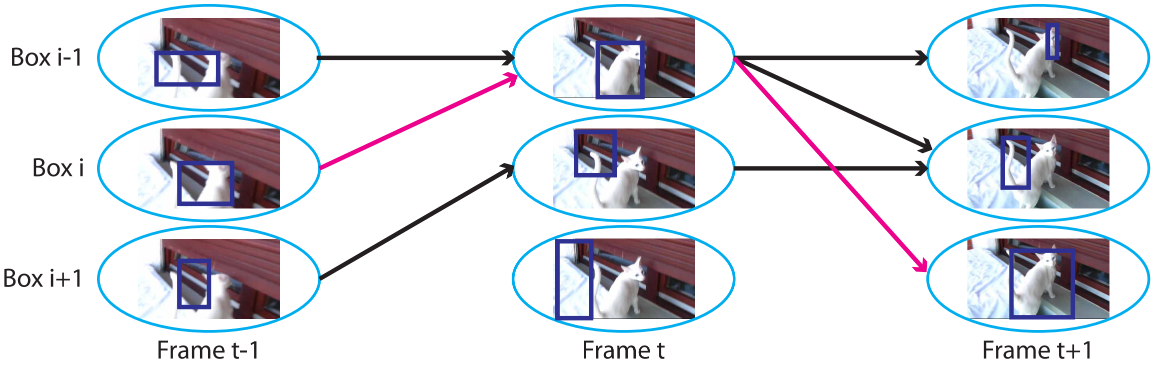


Figure 6.3: Given consecutive frames of video, we build a graph between candidate bounding boxes in adjacent frames. The magenta edges represent the optimal path through the frames.

function to penalize the selection of boxes whose features are not easily linearly separable from other boxes. Note that since the matrices L and A are each positive semi-definite, the objective function is convex.

6.3.2 Video Model

Given a set of n videos $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$, our goal is to localize the common object in each frame of each video. We approach this problem by considering each video V_i as a collection of temporally ordered frames $\mathcal{I}_i = \{I_{i1}, I_{i2}, \ldots, I_{il_i}\}$, where l_i is the length of video V_i and I_{ij} corresponds to frame j of video V_i. Similar to the image model, we generate a set of m candidate boxes $\mathcal{B}_{ij}$ for each frame of each video using objectness [2]. Our goal then is to select the box from each frame that contains the common object. Similar to the image model, we associate with each box $b_{i,j,k} \in \mathcal{B}_{i,j}$ a binary label variable $z_{i,j,k}$, and stack the variables to obtain z, the $n_b = \sum_{i=1}^{n} l_i m$ dimensional vector.

Defining $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n\}$ as the set of all frames, we can apply the same objective function and constraints from the image model to $\mathcal{I}$. The image model constraints enforce selecting a single box in each frame, and the image model objective function captures the box prior, similarity, and discriminability within and across different videos.

Incorporating temporal consistency. In video data, temporal consistency tells us that between consecutive frames, it is unlikely for objects to undergo drastic changes in qualities such as appearance, position, and size. This is a powerful prior that is often leveraged in video tasks such as tracking [4, 8, 46, 108, 112, 143, 170]. In our framework, if two boxes from consecutive frames differ greatly in their size and position, it should be unlikely that they will be selected together. Using this intuition, we can define a simple temporal similarity measure between two boxes b_i and b_j from consecutive frames as follows:

$$s_{\text{temporal}}(b_i, b_j) = \exp\Bigg(- \|b_i^{center} - b_j^{center}\|_2 - \Bigg\| \frac{|b_i^{area} - b_j^{area}|}{\max(b_i^{area}, b_j^{area})} \Bigg\|_2 \Bigg), \tag{6.3}$$

where b_i^{area} is the pixel area of box b_i and b_i^{center} are the center coordinates of box b_i, normalized by the width and height of the frame.

With this similarity metric for all pairs of boxes between adjacent frames, we obtain a weighted graph G_i for each video V_i that connects the boxes within the video based on temporal similarity, as shown in Figure 6.3. We threshold small values of similarity so that dissimilar edges have a weight of 0 and are thus disconnected. Note that as long as we can obtain a weighted graph, any similarity metric between two boxes from adjacent frames can be used. This makes our temporal framework extremely flexible, and allows us to potentially leverage state-of-the-art methods in object tracking [4, 8, 46, 108, 112, 170].

We collect all the pairwise similarities s_{temporal} between boxes in adjacent frames into a similarity matrix S_t, where $S_t(i,j) = s_{\text{temporal}}(b_i, b_j)$ if b_i and b_j are boxes in adjacent frames, and $S_t(i,j) = 0$ otherwise. With this matrix, we can compute the normalized Laplacian $U = I - D^{-\frac{1}{2}} S_t D^{-\frac{1}{2}}$, where D is the diagonal matrix composed of the row sums of S_t. This matrix encourages us to select boxes that are similar based on the temporal similarity metric.

Intuitively, the boxes we select from each video V_i should respect the corresponding graph G_i, in that the solution should follow a valid path through the graph from the first frame to the last. For each edge (a, b) in the graph G_i, we define a binary variable $y_{i,a,b}$ equal to 1 if both a and b are boxes containing the object and 0 otherwise. More precisely, we require $y \in \{0,1\}$ to follow the linear constraints for each video V_i and every box b_k (associated with binary label variable z_k) in V_i: $z_k = \sum_{l \in p(k)} y_{i,l,k} = \sum_{l \in c(k)} y_{i,k,l}$, where $p(k)$ and $c(k)$ are the parents and children of box b_k in the graph G_i, respectively.

Model formulation. Combining the temporal terms and constraints together with the original image model, we obtain the following optimization problem to select the box containing the common object from each frame of video:

$$\begin{aligned} \underset{z,y}{\text{minimize}} \quad & z^T(L + \mu A + \mu_t U)z - z^T \lambda \log(m) \\ \text{subject to} \quad & z \in \{0,1\},\ y \in \{0,1\}, \\ & \forall I_j \in \mathcal{I} : \sum_{k=1}^{m} z_{j,k} = 1, \\ & \forall V_i \in \mathcal{V},\ \forall k \in V_i,\ z_k = \sum_{l \in p(k)} y_{i,l,k} = \sum_{l \in c(k)} y_{i,k,l}, \end{aligned} \tag{6.4}$$

where z_i are the binary label variables associated with the boxes in video V_i, and μ_t weights the temporal Laplacian matrix. The additional constraint forces us to choose solutions that respect the edges defined by the underlying graphs for each video, and the additional Laplacian term in the objective function weights these edges. Note that the additional constraint is required to constrain our solutions, as the terms in the objective from the image model can still lead us to select invalid paths if we only had the temporal Laplacian matrix. This formulation allows us to incorporate temporal consistency into the image model. In the rest of this chapter, we denote by $\mathcal{P}$ the set of constraints defined in Equation 6.4.

In the next section, we present a tight convex relaxation which can be efficiently optimized using the Frank-Wolfe algorithm [36].

6.4 Optimization

A standard way of dealing with quadratic programs such as Equation 6.4 is to relax the discrete non-convex set $\mathcal{P}$ to its convex hull, conv($\mathcal{P}$). Standard algorithms such as interior point methods can be applied but leads to a complexity of $O(N^3)$ which cannot deal with hundreds of videos. We show how it is possible to design an efficient algorithm by using the specificities of our problem.

A key observation towards designing an efficient algorithm for our problem is that the constraints defining the set $\mathcal{P}$ are separable in each video and are equivalent, for each video, to the constraints used in the shortest-path algorithm. This means that if our cost function was linear, we could solve our problem efficiently using dynamic programming.

6.4.1 Frank-Wolfe Algorithm

Given a convex cost function f and a convex set $\mathcal{D}$, the Frank-Wolfe algorithm [36] finds the global minimum of f over $\mathcal{D}$ by solving a succession of linear problems [31, 55]. More precisely, at each iteration k it solves:

$$\begin{aligned} \underset{y}{\text{minimize}} \quad & y^T \nabla f(z_{k-1}) \\ \text{subject to} \quad & y \in \mathcal{D}. \end{aligned} \tag{6.5}$$

The solution y_k is then used in Frank-Wolfe updates given by:

$$z_k = z_{k-1} + \lambda(y_k - z_{k-1}), \tag{6.6}$$

where $\lambda > 0$ is found using a line search (see Algorithm 1 for details). Essentially, the Frank-Wolfe algorithm considers a linear approximation of the objective function at each iteration. Although not appropriate for all convex optimization problems, Frank-Wolfe applied to our optimization formulations results in very simple linearizations with integer solutions that are easily solved.

Frank-Wolfe algorithm on convex hull. This algorithm does not need an explicit form for $\mathcal{D}$ as long as it is possible to find the solution of a linear program over $\mathcal{D}$. This is particularly interesting when $\mathcal{D}$ is the convex hull of a set of points $\mathcal{C}$ on which it is possible to solve an integer program. Solving a linear program on $\mathcal{D}$ is then equivalent to solving an integer program over $\mathcal{C}$. This is a particularity of the Frank-Wolfe algorithm that we will exploit in our video setting.

Video model. For the video model, the Frank-Wolfe algorithm solves the following problem at each iteration:

$$\begin{aligned} \underset{y}{\text{minimize}} \quad & y^T H z_{k-1} \\ \text{subject to} \quad & y \in \text{conv}(\mathcal{P}). \end{aligned} \tag{6.7}$$

where $H = L + \mu A + \mu_t U$. The cost function and constraints are separable for each video, and optimizing Equation 6.7 results in the standard shortest path problem for each video, which can be solved efficiently using dynamic programming.

Image Model. For the image model, the linearized cost function is separable for each image, and we can efficiently find the best integer solution for this problem by computing the score for each box, $(L + \mu A)z_{k-1}$, and then simply selecting the argmin. Note that in the case of images, it is possible to use a projected gradient descent approach by projecting on the simplex. This approach is faster than ours but cannot be generalized to videos, which is the main focus of this chapter.

Since the Frank-Wolfe algorithm for images utilizes the same framework as for videos, an additional advantage is that we can easily learn a shared image/video model with a single algorithm.

6.4.2 Implementation Details

In this section, we present some details on our implementation of the Frank-Wolfe algorithm.

Algorithm 1: Frank-Wolfe algorithm with away step and rounding.

```
Data: $y_0 \in \mathcal{D}$, $\varepsilon > 0$
Result: $y^*$
Initialization: $k = 0$, $z = y_0$, $\mathcal{S}_0 = \{y_0\}$, $\alpha^0 = \{1\}$;
while $duality\_gap(z) \geq \varepsilon$ do
    $k \leftarrow k + 1$;
    $y_k \leftarrow \text{argmin}_{y \in \mathcal{D}} \langle y, \nabla f(z) \rangle$ (FW direction);
    $x_k \leftarrow \text{argmax}_{y \in \mathcal{S}_{k-1}} \langle y, \nabla f(z) \rangle$ (away direction);
    if $\langle y_k - z, \nabla f(z) \rangle \leq \langle z - x_k, \nabla f(z) \rangle$ then
        $d_k = y_k - z$;
        $\gamma_{max} = 1$;
    else
        $d_k = z - x_k$;
        $\gamma_{max} = \alpha_k(x_k)$;
    Line search: $\gamma_k = \min_{\gamma \in [0, \gamma_{max}]} f(z + \gamma d_k)$;
    $\mathcal{S}_k,\ \alpha_k \leftarrow update\_active\_set(d_k, \gamma_k)$;
    Update $z \leftarrow z + \gamma_k d_k$;
    if $f(y_k) < f(y^*)$ then
        $y^* \leftarrow y_k$ (rounding 1);
$y_r \leftarrow \text{argmax}_{y \in \mathcal{D}} \langle y, z \rangle$ (rounding 2);
if $f(y_r) < f(y^*)$ then
    $y^* \leftarrow y_r$ (combining rounding);
```

Away step. We use an accelerated version termed Frank-Wolfe with away step [162]. The details of this algorithm are given in Algorithm 1. The algorithm keeps a set of previously seen integer solutions (called active corners) $\mathcal{S}_k$ at each iteration such that the current update z is the sum of the corners in $\mathcal{S}_k$ re-weighted by α_k. The set $\mathcal{S}_k$ is used to find potentially better directions by moving "away" from an active corner (away step). This version of Frank-Wolfe has been shown to have better convergence rates [45, 79].

Line seach and duality gap. In the case of a quadratic function, both the line search and the duality gap are in closed form, which significantly improves the speed of our algorithm [80].

Parallel computation. Our constraints are separable for each image and video, allowing efficient parallel computation of the update. Note that this is a property of any first-order method, including the Frank-Wolfe algorithm. In practice, this allows us to be extremely memory efficient, as we can consider subproblems for each image or video separately.

Rounding. A typical concern with methods based on convex relaxations is obtaining a solution from the relaxed problem that satisfies the non-convex constraints from the original problem. In our case, the rounded solution must belong to the set $\mathcal{P}$. The most natural way of rounding a solution z is to find the closest element in $\mathcal{P}$ given some distance. For the ℓ_2 distance, this means solving $\min_{y \in \mathcal{P}} \|y - z\|_2^2$ which is not possible in general. However, in our case, since the ℓ_2 norm is constant on $\mathcal{P}$ (and equal to the total number of frames/images in the dataset), this projection is equivalent to:

$$\underset{y \in \mathcal{P}}{\text{maximize}} \quad \langle y, z \rangle, \tag{6.8}$$

which can be solved efficiently using the shortest-path algorithm for the video model, and simply taking the argmax in each image for the image model.

Additionally, the particular form of the Frank-Wolfe updates offers another very natural and inexpensive way of rounding our solution. We can keep track of the solution to the linear problem defined in Equation 6.7 that minimizes the cost function defined in Equation 6.4. Since this solution is in the original set $\mathcal{P}$, it automatically satisfies the constraints.

In practice, we use both rounding methods and keep the one that results in the lowest value of our cost function, as shown in Algorithm 1.

6.5 Results

We perform experiments on two challenging datasets, the PASCAL VOC 2007 dataset [33] and the YouTube-Objects dataset [113]. We also combine the two and present results for joint image-video co-localization. Following previous works in weakly supervised localization (WSL) [28, 40, 107, 131, 133–135] and co-localization [142], we use the CorLoc evaluation metric, defined as the percentage of images correctly localized according to the PASCAL-criterion: $\frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} > 0.5$, where B_p is the predicted box and B_{gt} is the ground-truth box. All CorLoc results are given in percentages.

Implementation details. We set the parameters of our method by optimizing over a small set of images/videos. For the image model, we set $\mu = 0.4$ and for the video model, we set $\mu = 0.6$ and $\mu_t = 1.8$. For both models, we found $\lambda = 0.1$ to perform best. Unless otherwise stated, we extracted 20 objectness boxes from each image. For the video model, we sampled each video every 10 frames, since there is typically little change in such a short amount of time.

6.5.1 Running Time and Rounding Experiments

In this section, we evaluate the running time of our algorithm. Our implementation is coded in MATLAB and we compare to two standard Quadratic Programming (QP) solvers, Mosek and Gurobi, which are coded in C++. All experiments are done on a single core 2.66GHz Intel CPU with 6GB of RAM.

Stopping criterion. Our stopping criterion is based on the relative duality gap defined as $d = (f - g)/g$, where f is our cost function and g is its dual. We stop when d is less than some $\varepsilon > 0$. We consider two values for ε, 10^{-2} and 10^{-3}. The choice of these values for ε is motivated by the empirical observation that our cost function remains almost constant for $d < 10^{-2}$, as show in Figure 6.4(a).

Running time analysis. In Figure 6.4(b)(c), we show how our algorithm scales in the number of videos and images compared to standard QP solvers. For fair running time comparison, we present the time for both standard QP solvers to reach a duality gap less than $\varepsilon = 10^{-2}$. When $\varepsilon = 10^{-3}$, our algorithm runs 100 times faster than standard solvers for more than 20 videos. For $\varepsilon = 10^{-2}$, this factor increases to

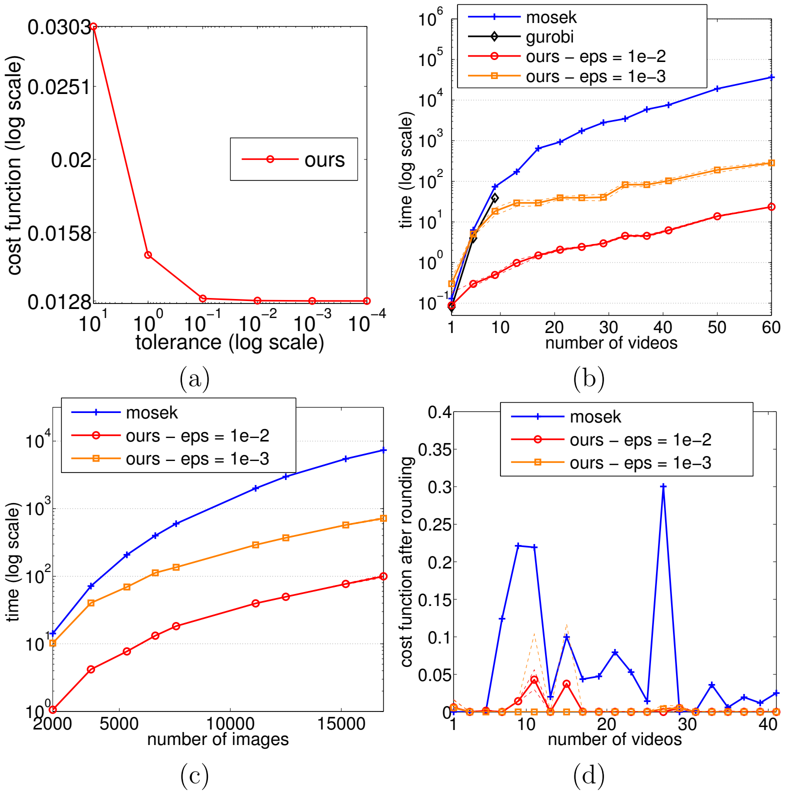


Figure 6.4: (a) Value of the cost function for 11 videos as a function of the relative duality gap (log scale). Time comparison between our algorithm and standard QP solvers (time in log scale of second) for (b) video co-localization and (c) image co-localization. (d) Comparison of the value of the cost function obtained with our algorithm and a standard QP solver.

more than 1000. Typically, for $\varepsilon = 10^{-3}$, solving our problem with 50 videos takes 3 minutes, and 80 videos takes 7 minutes. The gain in speed is mostly due to efficiently computed iterations based on a shortest path algorithm/argmin.

Rounding quality. In Figure 6.4(d), we also compare the quality of the solution obtained after rounding in terms of the original cost function. We compare the relative value of the cost function, $(f - f^*)/f^*$, where f^* is the minimum observed value of the cost function. We round the solutions by solving Equation 6.8. For fairness of comparison, we use the solution given by the QP solver for a tolerance of $\varepsilon = 10^{-10}$. Compared to the standard QP solver, our algorithm obtains a significantly better

Number of objectness boxes [2]	5	10	15	20
Our method	23.96	24.23	24.28	**24.59**
Upper bound	51.04	62.22	67.99	**71.58**

Table 6.1: Average CorLoc results and upper bound on PASCAL07.

Method	[121](w/ viewpoint)	[28](w/ viewpoint)	[135](w/o viewpoint)	Our method(w/o viewpoint)
Average CorLoc	14	23	21.2	22

Table 6.2: CorLoc results on PASCAL07-all compared to previous methods for co-localization.

rounded solution in terms of value of the cost function.

Despite numerous advantages of our solver for our specific problem, a limitation of the Frank-Wolfe algorithm with away step in the case of an exponential number of corner points (as is the case in our problem) is that it converges with no guarantee of a linear convergence rate.

6.5.2 Image Co-localization: PASCAL VOC 2007

In [142], the authors show improved co-localization performance on PASCAL07-6x2, a small subset of PASCAL VOC 2007 divided into specific viewpoints. To illustrate the benefits of the Frank-Wolfe algorithm, which allows us to efficiently consider many more images and boxes per image, we co-localize all images not labeled as difficult for all classes in the PASCAL VOC 2007 dataset [33], which we denote as PASCAL07. This makes the problem much more difficult as we now have to co-localize differing viewpoints together and a much larger set of images. To emphasize the difference in size, the “bicycle-right” class in the PASCAL07-6x2 dataset has the largest number of images at 50, whereas the “person” class in the PASCAL07 dataset has 2,008 images. In all experiments performed in [142], the authors only co-localize a maximum of 100 images at a time due to efficiency concerns. Results for our method varying the number of extracted candidate boxes are given in Table 6.1, and visualizations are shown in Figure 6.5. We also show the upper bound on the performance that can

Method	Our method (Co-localization)	[133] (WSL)	[135] (WSL)	[134] (WSL)	[131] (WSL)	[40](WSL)
Average CorLoc	24.6	30.2	30.4	32.0	36.2	38.8

Table 6.3: CorLoc results on PASCAL07 compared to previous methods for weakly supervised localization.

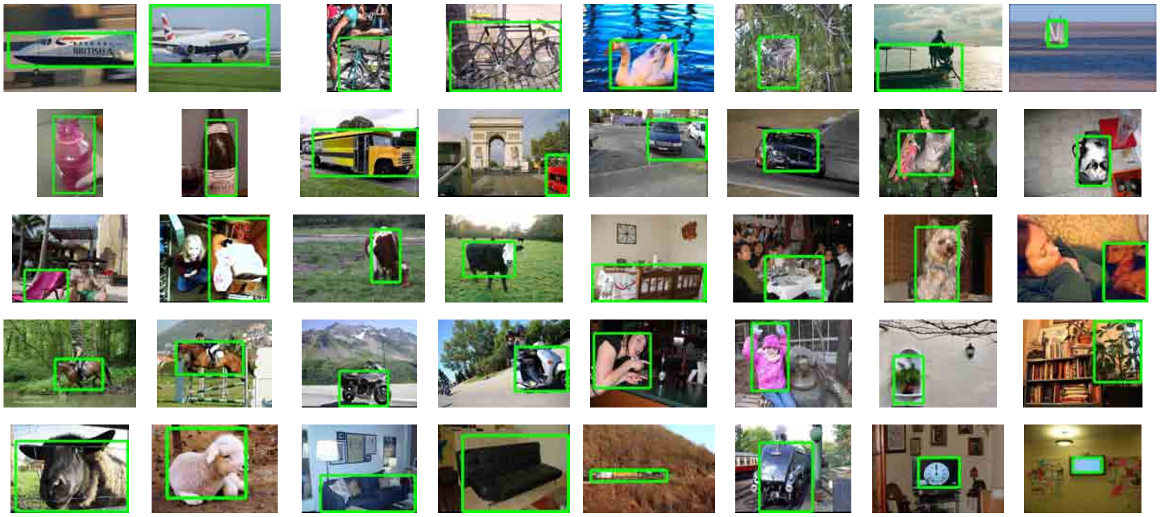


Figure 6.5: Example co-localization results on PASCAL07. From left-right, every two images belong to the same class.

be achieved with the candidate boxes, computed by selecting the box in each image with the highest CorLoc.

Number of candidate boxes. As we can see, the performance of our model increases when we increase the number of candidate boxes. We can also see that the upper bound becomes much better due to the better recall obtained with more boxes. This helps to validate the importance of efficient methods for co-localization, as they allow us to take advantage of more data in our model.

Comparisons to co-localization methods. We show results compared to previous co-localization methods in Table 6.2 for the PASCAL07-all dataset [28], which does not consider the "bird", "car", "cat", "cow", "dog", and "sheep" classes.

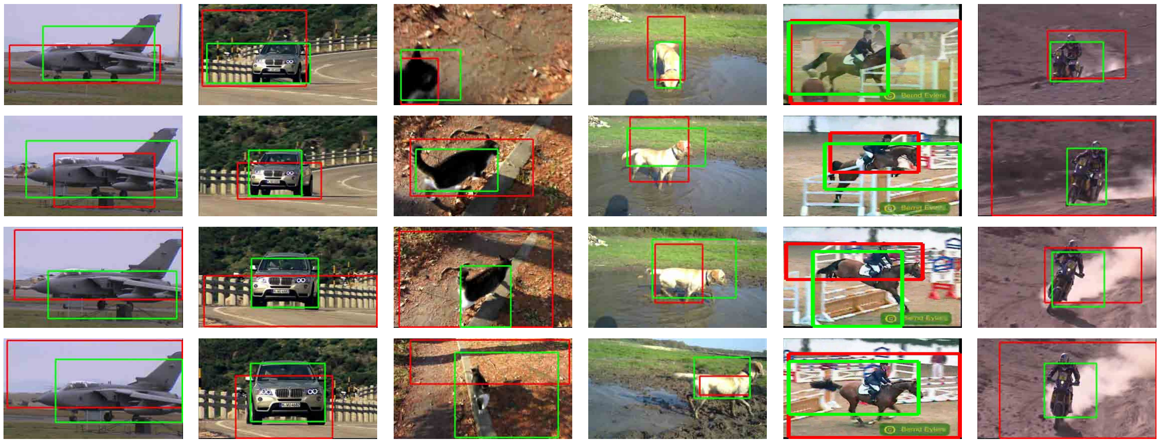


Figure 6.6: Example co-localization results on YouTube-Objects for our video model (green boxes) and our image model (red boxes). Each column corresponds to a different class, and consists of frame samples from a single video.

For [135], we used their intra-only results for co-localization. Note that all previous methods except [135] utilize additional viewpoint annotations by dividing the images for each class into separate viewpoints, and co-localizing each viewpoint separately with viewpoint-specific priors. On the other hand, our method and [135] is run on all of the viewpoints simultaneously, which is a much more difficult problem. Also, comparing results on the entire PASCAL07 dataset in Table 6.1, we obtain 24.59%, while [135] obtains 23.9%.

Comparisons to WSL methods. We also show our method compared to previous WSL methods in Table 6.3 for the PASCAL07 dataset [28]. In the WSL scenario, previous methods are also given negative data [28, 40, 107, 131, 133–135], whereas our method is not. Here, we can see that although our method is able to work in certain scenarios where there is an absence of negative data, there is still a large gap in performance that can be obtained by utilizing negative data.

6.5.3 Video Co-localization: YouTube-Objects

The YouTube-Objects dataset [113] consists of YouTube videos collected for 10 classes from PASCAL [33]: “aeroplane”, “bird”, “boat”, “car”, “cat”, “cow”, “dog”, “horse”,

Method	aeroplane	bird	boat	car	cat	cow	dog	horse	motorbike	train	Average
[113]	51.7	17.5	34.4	34.7	22.3	17.9	13.5	26.7	41.2	25.0	28.5
[109]	**65.4**	**67.3**	**38.9**	**65.2**	**46.3**	**40.2**	**65.3**	**48.4**	**39.0**	**25.0**	**50.1**
Our method (image)	18.36	19.35	28.57	32.97	32.77	25.68	38.26	30.14	15.38	21.43	26.29
Our method (image) w/ smoothing	21.26	21.51	30.95	36.26	35.29	25.68	38.26	35.62	15.38	23.21	28.34
Our method (video)	25.12	31.18	27.78	38.46	41.18	28.38	33.91	35.62	23.08	25.00	30.97

Table 6.4: CorLoc results for video co-localization on the YouTube-Objects dataset.

"motorbike", "train". For each class, bounding box annotations for the object are annotated in one frame per shot for 100-290 different shots. We perform video co-localization on all shots with annotations. Results are given in Table 6.4, where we compare to the co-localization method of [113], our image model with and without smoothing. Note that better results are obtained in [109] using unsupervised motion segmentation and appearance consistency within each video, which works particularly well for this dataset where objects of interest are moving. In contrast, our method focuses on trying to leverage appearance information across different videos in conjunction with temporal consistency.

Comparisons to [113]. From our results, we see that we outperform the previous method of [113] for most classes. For most of the classes, we obtain results which are slightly better than [113]. For the "aeroplane" and "motorbike" classes however, we perform much worse. This is likely because the candidate tube extraction algorithm used in [113] is able to effectively track simple and non-deformable objects. However, note that our method is actually agnostic to the underlying candidate region generation algorithm, and we could easily replace our objectness boxes with candidate tubes.

Comparisons to [109]. It is interesting to see that our performance is worse than [109]. In their method, they do not learn a model between the videos and instead use a novel unsupervised motion segmentation method. A possible explanation for this difference in performance is the size of the database: the YouTube-Objects dataset is relatively small and the intra-class variation is relatively high (in particular for "aeroplane"). It is thus hard to learn a common model across videos.

Method	aeroplane	bird	boat	car	cat	cow	dog	horse	motorbike	train	Average
Video only	25.12	31.18	27.78	**38.46**	41.18	28.38	33.91	35.62	**23.08**	25.00	30.97
Joint Image+Video	**27.54**	**33.33**	27.78	34.07	**42.02**	28.38	**35.65**	35.62	21.98	25.00	**31.14**

Table 6.5: CorLoc results for joint image-video co-localization on the YouTube-Objects dataset.

Comparisons to image model. Our video model outperforms the image model, which illustrates the importance of leveraging temporal consistency. From the visualizations in Figure 6.6, we see that the image model often jumps around throughout a single video. For the "dog" class however, our image model actually performs much better than our video model. This is likely due to large amounts of sporadic movement in the "dog" videos caused by both camera movement and object movement. The simple similarity metric we use for temporal consistency may not be invariant to such difficult types of motion, and thus the image model is able to perform better in this case. As noted previously, we can substitute any similarity metric into our framework, and thus potentially take advantage of methods in object tracking [4, 8, 46, 108, 112, 170] to further improve performance.

6.5.4 Joint Image-Video Co-localization

Since the classes in the YouTube-Objects dataset are a subset of the PASCAL07 classes, we can combine the images from the corresponding classes in PASCAL07 with the videos in YouTube-Objects to perform joint image-video co-localization. Results for CorLoc performance on the YouTube-Objects dataset are given in Table 6.5. We can see that our performance increases slightly for several classes, such as "aeroplane", "bird", "cat" and "dog". It is not unexpected that performance becomes worse for several classes, as there is an inherent domain adaptation problem between images and videos [113, 143]. However, our preliminary results show that with efficient algorithms for image and video co-localization, the problem of jointly considering the two domains is viable, and may present an effective way of taking advantage of all the weakly labeled data available.

6.6 Summary

In this chapter, we introduce a formulation for video co-localization that is able to naturally incorporate temporal consistency in a quadratic programming framework. In addition, we show how the image and video co-localization models that are presented can be efficiently optimized using the Frank-Wolfe algorithm. Our experiments on the PASCAL07 and YouTube-Objects datasets illustrate the benefits of our approach for image, video, and joint image-video co-localization.

Chapter 7

Learning Temporal Embeddings for Complex Video Analysis

7.1 Introduction

Video data is plentiful and a ready source of information – what can we glean from watching massive quantities of videos? At a fine granularity, consecutive video frames are visually similar due to temporal coherence. At a coarser level, consecutive video frames are visually distinct but semantically coherent.

Learning from this semantic coherence present in video at the coarser-level is the main focus of this chapter. Purely from unlabeled video data, we aim to learn embeddings for video frames that capture semantic similarity by using the temporal structure in videos. The prospect of learning a generic embedding for video frames holds promise for a variety of applications ranging from generic retrieval and similarity measurement, video recommendation, to automatic content creation such as summarization or collaging. In this chapter, we demonstrate the utility of our video frame embeddings for several tasks such as video retrieval, classification and temporal order recovery.

The idea of leveraging sequential data to learn embeddings in an unsupervised fashion is well explored in the Natural Language Processing (NLP) community. In particular, distributed word vector representations such as word2vec [95] have the

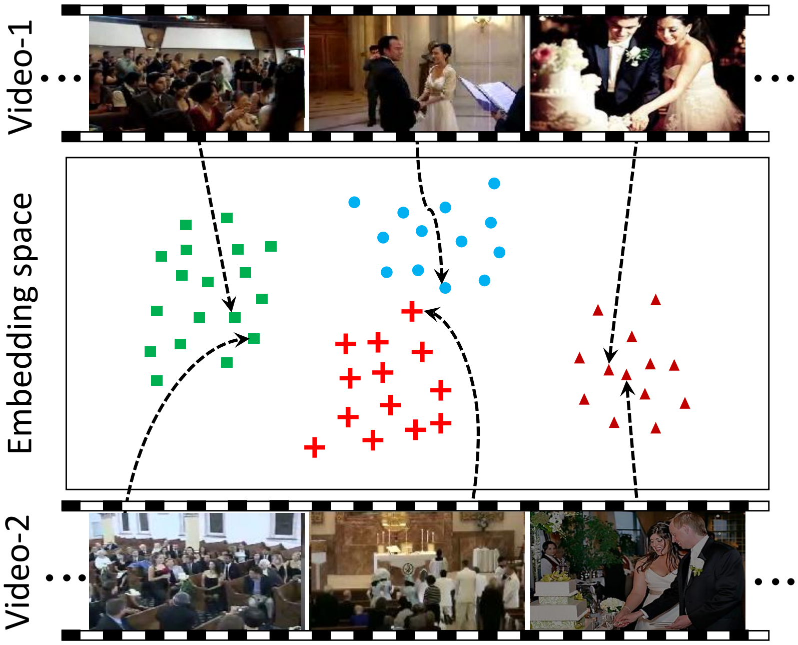


Figure 7.1: The temporal context of a video frame is crucial in determining its true semantic meaning. The middle frame from the two wedding videos correspond to visually dissimilar classes of "church ceremony" and "court ceremony". However, by observing the similarity in their temporal contexts we expect them to be semantically closer.

unique ability to encode *regularities and patterns* surrounding words, using large amounts of unlabeled data. In the embedding space, this brings together words that may be very different, but which share similar contexts in different sentences. This is a desirable property we would like to extend to video frames as well as shown in Figure 7.1. We would like to have a representation for frames which captures the semantic context around the frame beyond the visual similarity obtained from temporal coherence.

However, the task of embedding frames poses multiple challenges specific to the video domain: 1. Unlike words, the set of frames across all videos is not discrete and quantizing the frames leads to a loss in information; 2. Temporally proximal frames within the same video are often visually similar and might not provide useful contextual information; 3. The correct representation of context surrounding a frame is not obvious in videos. The main contribution of our work is to propose a new ranking loss based embedding framework, along with a contextual representation specific to videos. We also develop a well engineered data augmentation strategy to promote visual diversity among the context frames used for embedding.

We evaluate our learned embeddings on the standard tasks of video event retrieval and classification on the TRECVID MED 2011 [106] dataset, and compare to several recently published spatial and temporal video representations [54, 132]. Aside from semantic similarity, the learned embeddings capture valuable information in terms of the temporal context shared between frames. Hence, we also evaluate our embeddings on two related tasks: 1. temporal frame retrieval, and 2. temporal order recovery in videos. Our embeddings improve performance on all tasks, and serves as a powerful representation for video frames.

7.2 Related Work

Video features. Standard tasks in video such as classification and retrieval require a well engineered feature representation, with many proposed in the literature [23, 56, 61, 83, 100, 102, 105, 110, 122, 157, 158]. Deep network features learned from spatial data [58, 67, 132] and temporal flow [132] have also shown comparable results. However, recent works in complex event recognition [166, 175] have shown that spatial Convolutional Neural Network (CNN) features learned from ImageNet [26] without fine-tuning on video, accompanied by suitable pooling and encoding strategies achieves state-of-the-art performance. In contrast to these methods which either propose handcrafted features or learn feature representations with a fully supervised objective from images or videos, we try to learn an embedding in an unsupervised fashion. Moreover, our learned features can be extended to other tasks beyond classification and retrieval.

There are several works which improve complex event recognition by combining multiple feature modalities [60, 97, 140]. Another related line of work is the use of sub-events defined manually [54], or clustered from data [81] to improve recognition. Similarly, Yang et al. used low dimensional features from deep belief nets and sparse coding [169]. While these methods are targeted towards building features specifically for classification in limited settings, we propose a generic video frame representation which can capture semantic and temporal structure in videos.

Unsupervised learning in videos. Learning features with unsupervised objectives has been a challenging task in the image and video domain [57, 85, 147]. Notably, [85] develops an Independent Subspace Analysis (ISA) model for feature learning using unlabeled video. Recent work from [43] also hints at a similar approach to exploit the slowness prior in videos. Also, recent attempts extend such autoencoder techniques for next frame prediction in videos [117, 139]. These methods try to capitalize on the temporal continuity in videos to learn an LSTM [174] representation for frame prediction. In contrast to these methods which aim to provide a unified representation for a complete temporal sequence, our work provides a simple yet powerful representation for independent video frames and images.

Embedding models. The idea of embedding words to a dense lower dimension vector space has been prevalent in the NLP community. The word2vec model [95] tries to learn embeddings such that words with similar contexts in sentences are closer to each other. A related idea in computer vision is the embedding of text in the semantic visual space attempted by [37, 72] based on large image datasets labeled with captions or class names. While these methods focus on different scenarios for embedding text, the aim of our work is to generate an embedding for video frames.

7.3 Our Method

Given a large collection of unlabeled videos, our goal is to leverage their temporal structure to learn an effective embedding for video frames. We wish to learn an embedding such that the *context* frames surrounding each *target* frame can determine the representation of the *target* frame, similar to the intuition from word2vec [95]. For example, in Figure 7.1, *context* such as “crowd” and “cutting the cake” provides valuable information about the *target* “ceremony” frames that occur in between. This idea is fundamental to our embedding objective and helps in capturing semantic and temporal interactions in video.

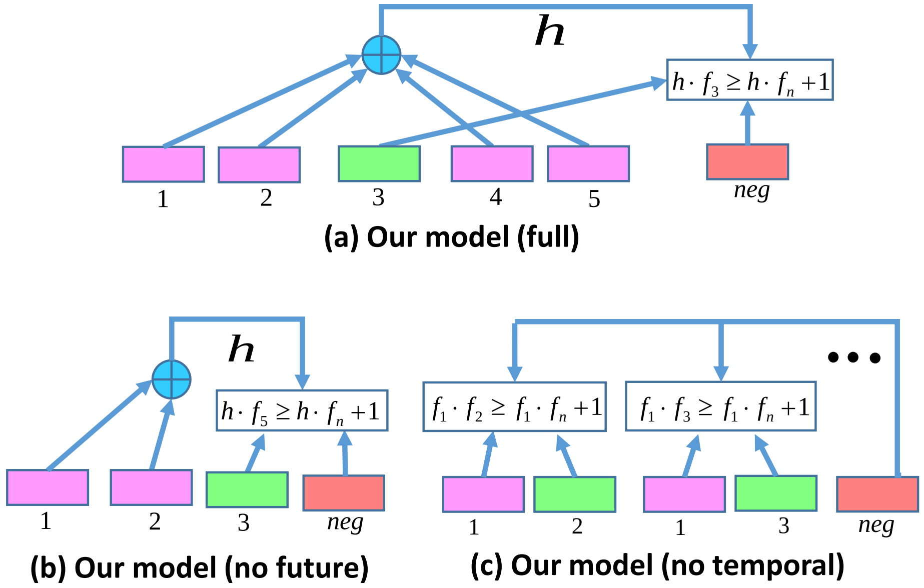
While the idea of representing frames by embeddings is lucrative, the extension from language to visual data is not straightforward. Unlike language we do not have a natural, discrete vocabulary of words. This prevents us from using a softmax objective as in the case of word2vec [95]. Further, consecutive frames in videos often share visual similarity due to temporal coherence. Hence, a naive extension of [95] does not lead to good vector representations of frames.

To overcome the problem of lack of discrete words, we use a ranking loss which explicitly compares multiple pairs of frames across all videos in the dataset. This ensures that the *context* in a video scores the *target* frame higher than others in the dataset. We also handle the problem of visually similar frames in temporally smooth videos through a carefully designed sampling mechanism. We obtain context frames by sampling the video at multiple temporal scales, and choosing hard negatives from the same video.

7.3.1 Embedding objective

We are given a collection of videos $\mathcal{V}$, where each video $v \in \mathcal{V}$ is a sequence of frames $\{s_{v1}, \ldots, s_{vn}\}$. We wish to obtain an embedding f_{vj} for each frame s_{vj}. Let $f_{vj} = f(s_{vj}; W_e)$ be the temporal embedding function which maps the frame s_{vj} to a vector. The model embedding parameters are given by W_e, and will be learned by our method. We embed the frames such that the *context* frames around the *target* frame predict the *target* frame better than other frames. The model is learned by minimizing the sum of objectives across all videos. Our embedding loss objective is shown below:

$$J(W_e) = \sum_{v \in \mathcal{V}} \sum_{\substack{s_{vj} \in v \\ s_- \neq s_{vj}}} \max\left(0, 1 - (f_{vj} - f_-) \cdot h_{vj}\right), \tag{7.1}$$


h
$h \cdot f_3 \geq h \cdot f_n + 1$
1
2
3
4
5
neg
h
$h \cdot f_5 \geq h \cdot f_n + 1$
1
2
3
neg
$f_1 \cdot f_2 \geq f_1 \cdot f_n + 1$
$f_1 \cdot f_3 \geq f_1 \cdot f_n + 1$
1
2
1
3
neg

(a) Our model (full)

(b) Our model (no future)

(c) Our model (no temporal)

Figure 7.2: Visualizations of the temporal context of frames used in: (a) our model (full), (b) our model (no future), and (c) our model (no temporal). Green boxes denote target frames, magenta boxes denote contextual frames, and red boxes denote negative frames.

where f_- is the embedding of a negative frame s_-, and the context surrounding the frame s_{vj} is represented by the vector h_{vj}. Note that unlike the word vector embedding models in word2vec [95], we do not use an additional linear layer for softmax prediction on top of the context vector.

7.3.2 Context representation

As we verify later in the experiments, the choice of context is crucial to learning good embeddings. A video frame at any time instant is semantically correlated with both past and future frames in the video. Hence, a natural choice for context representation would involve a window of frames centered around the *target* frame, similar to the skip-gram idea used in word2vec [95]. Along these lines, we propose a context representation given by the average of the frame embeddings around the *target* frame. Our context vector h_{vj} for a frame s_{vj} is:

$$h_{vj} = \frac{1}{2T}\sum_{t=1}^{T} f_{vj+t} + f_{vj-t}, \tag{7.2}$$

where T is the window size, and f_{vj} is the embedding of the frame s_{vj}. This embedding model is shown in Figure 7.2(a). For negatives, we use frames from other videos as well as frames from the same video which are outside the temporal window, as explained in Sec. 7.3.4.

Two important characteristics of this context representation is that it 1. makes use of the temporal order in which frames occur and 2. considers contextual evidence from both past and future. In order to examine their effect on the quality of the learned embedding, we also consider two weaker variants of the context representation below.

Our model (no future). This one-sided contextual representation tries to predict the embedding of a frame in a video only based on the embeddings of frames from the past as shown in Figure 7.2(b). For a frame s_{vj}, the context $h_{vj}^{nofuture}$ is given by:

$$h_{vj}^{nofuture} = \frac{1}{T}\sum_{t=1}^{T} f_{vj-t}, \tag{7.3}$$

where T is the window size.

Our model (no temporal). An even weaker variant of context representation is simple co-occurrence without temporal information. We also explore a contextual representation which completely neglects the temporal ordering of frames and treats a video as a bag of frames. The context h_{vj}^{notemp} for a target frame s_{vj} is sampled from the embeddings corresponding to all other frames in the same video:

$$h_{vj}^{notemp} \in \{f_{vk} \mid k \neq j\}. \tag{7.4}$$

This contextual representation is visualized in Figure 7.2(c).

7.3.3 Embedding function

In the previous sections, we introduced a model for representing context, and now move on to discuss the embedding function $f(s_{ij}; W_e)$. In practice, the embedding function can be a CNN built from the frame pixels, or any underlying image or video representation. However, following the recent success of ImageNet trained CNN features for complex event videos [166, 175], we choose to learn an embedding on top of the fully connected fc6 layer feature representation obtained by passing the frame through a standard CNN [74] architecture. We use a simple model with a fully connected layer followed by a rectified linear unit (ReLU) and local response normalization (LRN) layer, with dropout regularization. In this architecture, the learned model parameters W_e correspond to the weights and bias of our affine layer.

7.3.4 Data augmentation

We found that a careful strategy for sampling context frames and negatives is important to learning high quality embeddings in our models. This helps both in handling the problem of temporal smoothness and prevents the model from overfitting to less interesting video-specific properties.

Multi-resolution sampling. Complex events progress at different paces within different videos. Densely sampling frames in slowly changing videos can lead to context windows comprised of frames that are visually very similar to the target frame. On the other hand, a sparse sampling of fast videos could lead to context windows only composed of disjoint frames from unrelated parts of the video. We overcome these problems through multi-resolution sampling as shown in Figure 7.3. For every target frame, we sample context frames from multiple temporal resolutions. This ensures a good trade-off between visual variety and semantic relatedness in the context windows.

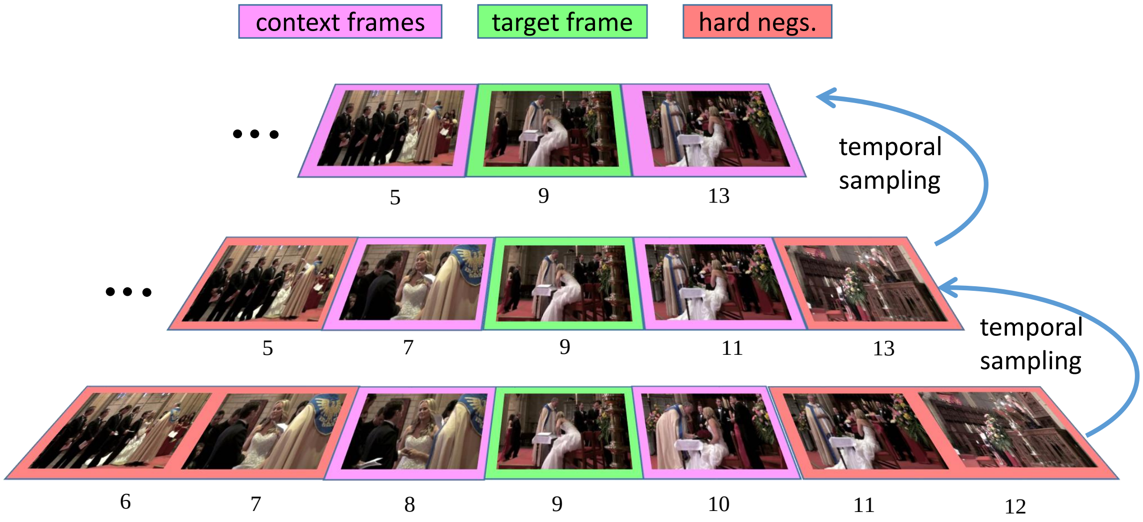


Figure 7.3: Multi-resolution sampling and hard negatives used in our full context model ($T = 1$). For a target frame (green), we sample context frames (magenta) at varying resolutions, as shown by the rows in this figure. We take hard negatives as examples in the same video that fall outside the context window (red).

Hard negatives. The context frames, as well as the target to be scored are chosen from the same video. This causes the model to cluster frames from the same video based on less interesting video-specific properties such as lighting, camera characteristics and background, without learning anything semantically meaningful. We avoid such problems by choosing hard negatives from within the same video as well. Empirically, this improves performance for all tasks. The negatives are chosen from outside the range of the context window within a video as depicted in Figure 7.3.

7.3.5 Implementation details

The context window size was set to $T = 2$, and the embedding dimension to 4096. The learning rate was set to 0.01 and gradually annealed in steps of 5000. The training is typically completed within a day on 1 GPU with Caffe [59] for a dataset of approximately 40000 videos. All videos were first down-sampled to 0.2 fps before training. The embedding code as well as the learned models and video embeddings will be made publicly available upon publication.

7.4 Experimental Setup

Our embeddings are aimed at capturing semantic and temporal interactions within complex events in a video, and thus we require a generic set of videos with a good variety of actions and sub-events within each video. Most standard datasets such as UCF-101 [138] and Sport-1M [67] are comprised of short video clips capturing a single sports action, making them unsuitable for our purpose. Fortunately, the TRECVID MED 2011 [106] dataset provides a large set of diverse videos collected directly from YouTube. More importantly, these videos are not simple single clip videos; rather they are complex events with rich interactions between various sub-events within the same video [54]. Specifically, we learn our embeddings on the complete MED11 DEV and TEST sets comprised of 40021 videos. A subset of 256 videos from the DEV and TEST set was used for validation. The DEV and TEST sets are typical random assortments of YouTube videos with minimal constraints.

We compare our embeddings against different video representations for three video tasks: video retrieval, complex event classification, and temporal order recovery. All experiments are performed on the MED11 event kit videos, which are completely disjoint from the training and validation videos used for learning our embeddings. The event kit is composed of 15 event classes with approximately $100 - 150$ videos per event, with a total of 2071 videos.

We stress that the embeddings are learned in a completely unsupervised setting and capture the temporal and semantic structure of the data. We do not tune them specifically to any event class and $\sim 0.3\%$ of the DEV and TEST sets contain videos from each category. This is not unreasonable, since any large unlabeled video dataset is expected to contain a small fraction of videos from every event.

7.5 Video Retrieval

In retrieval tasks, we are given a query, and the goal is to retrieve a set of related examples from a database. We start by evaluating our embeddings on two types of retrieval tasks: event retrieval and temporal retrieval. The retrieval tasks help to evaluate the ability of our embeddings to group together videos belonging to the same semantic event class and frames that are temporally coherent.

7.5.1 Event retrieval

In the event retrieval scenario, we are given a query video from the MED11 event kit and our goal is to retrieve videos that contain the same event from the remaining videos in the event kit. For each video in the event kit, we sort all other videos in the dataset based on their similarity to the query video using the cosine similarity metric, which we found to work best for all representations. We use Average Precision (AP) to measure the retrieval performance of each video and provide the mean Average Precision (mAP) over all videos in Table 7.1. For all methods, we uniformly sample 4 frames per video and represent the video as an average of the features extracted from them. The different baseline methods used for comparison are explained below:

- *Two-stream pre-trained*: We use the two-stream CNN from [132] pre-trained on the UCF-101 dataset. The models were used to extract spatial and temporal features from the video with a temporal stack size of 5.

- *fc6* and *fc7*: Features extracted from the ReLU layers following the corresponding fully connected layers of a standard CNN model [74] pre-trained on ImageNet.

- *Our model (no temporal)*: Our model trained with no temporal context (Figure 7.2(c)).

- *Our model (no future)*: Our model trained with no future context (Figure 7.2(b)) but with multi-resolution sampling and hard negatives.

- *Our model (no hard neg.)*: Our model trained without hard negatives from the same video.

- *Our model*: Our full model trained with multi-resolution sampling and hard negatives.

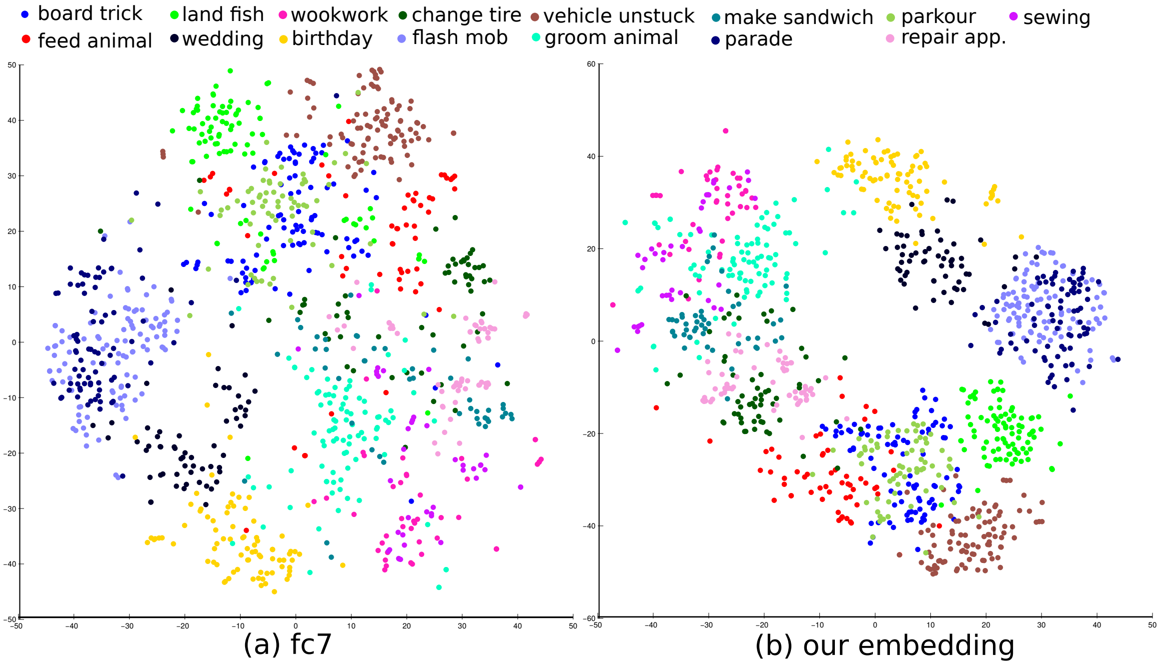
Method	mAP (%)
Two-stream pre-trained [132]	20.09
fc6	20.08
fc7	21.24
Our model (no temporal)	21.92
Our model (no future)	21.30
Our model (no hard neg.)	24.22
Our model	**25.07**

Table 7.1: Event retrieval results on the MED11 event kits.

We observe that our full model outperforms other representations for event retrieval. We note that in contrast to most other representations trained on ImageNet, our model is capable of being trained with large quantities of unlabeled video which is easy to obtain. This confirms our hypothesis that learning from unlabeled video data can improve feature representations. While the two-stream model also has the advantage of being trained specifically on a video dataset, we observe that the learned representations do not transfer favorably to the MED11 dataset in contrast to fc7 and fc6 features trained on ImageNet. A similar observation was made in [166, 175], where simple CNN features trained from ImageNet consistently provided the best results.

Our embeddings capture the temporal regularities and patterns in videos without the need for expensive labels, which allows us to more effectively represent the semantic space of events. The performance gain of our full context model over the representation without temporal order shows the need for utilizing the temporal information while learning the embeddings.

Visualizing the embedding space. To gain a better qualitative understanding of our learned embedding space, we use t-SNE [94] to visualize the embeddings in a 2D space. In Figure 7.4, we visualize the fc7 features and our embedded features by sampling a random set of videos from the event kits. The different colors in the graph correspond to each of the 15 different event classes, as listed in the figure. Visually, we can see that certain event classes such as “Grooming an animal”, “Changing a vehicle tire”, and “Making a sandwich” enjoy better clustering in our embedded framework



(a) fc7

(b) our embedding

Figure 7.4: t-SNE plot of the semantic space for (a) fc7 and (b) our embedding. The different colors correspond to different events.

as opposed to the fc7 representation.

Another way to visualize this space is in terms of the actual words used to describe the events. Each video in the MED11 event kits is associated with a short synopsis describing the video. We represent each word from this synopsis collection by averaging the embeddings of videos associated with that word. The features are then used to produce a t-SNE plot as shown in Figure 7.5. We avoid noisy clustering due to simple co-occurrence of words by only plotting words which do not frequently co-occur in the same synopsis. We observe many interesting patterns. For instance, objects such as “river”, “pond” and “ocean” which provide the same context for a “fishing” event are clustered together. Similarly crowded settings such as “bollywood”, “military”, and “carnival” are clustered together. This provides a visual clustering of the words based on shared semantic temporal context.

Event retrieval examples. We visualize the top frames retrieved for a few query frames from the event kit videos in Figure 7.6. The query frame is shown in the

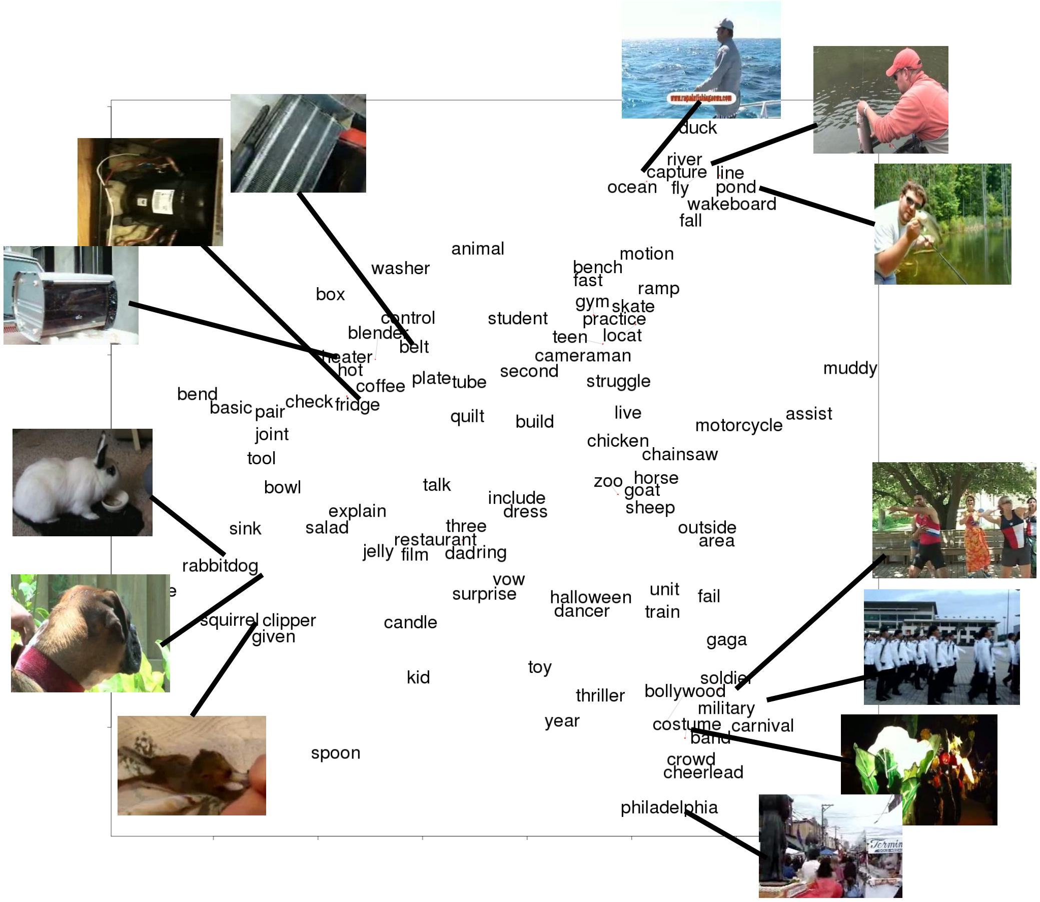


Figure 7.5: t-SNE visualization of words from synopses describing MED11 event kit videos. Each word is represented by the average of our embeddings corresponding to the videos associated with the word.

first column along with the event class corresponding to the video. The top 2 frames retrieved from other videos by our embedding and by fc7 are shown in the first and second columns for each query video, respectively.

We observe a few interesting examples where the query appears visually distinct from the results retrieved by our embedding. These can be explained by noting that the retrieved actions might co-occur in the same context as the query, which is captured by the temporal context in our model. For instance, the frame of a “bride near a car” retrieves frames of “couple kissing”. Similarly, the frame of “kneading dough” retrieves frames of “spreading butter”.

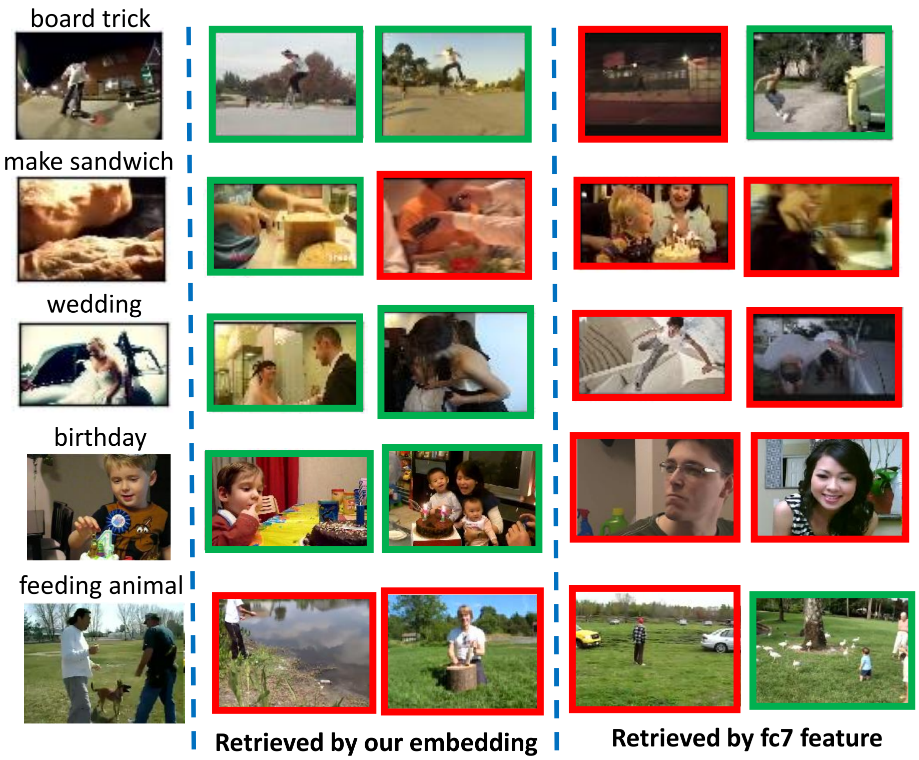


Figure 7.6: The retrieval results for fc7 (last two columns) and our embedding (middle two columns). The first column shows the query frame and event, while the top 2 frames retrieved from the remaining videos are shown in the middle two column for our embedding, and the last two columns for fc7. The incorrect frames are highlighted in red, and correct frames in green.

7.5.2 Temporal retrieval

In the temporal retrieval task, we test the ability of our embedding to capture the temporal structure in videos. We sample four frames from different time instants in a video and try to retrieve the frames in between the middle two frames. This is an interesting task which has potential for commercial applications such as ad placements in video search engines. For instance, the context at any time instant in a video can be used to retrieve the most suited video ad from a pool of video ads, to blend into the original video.

For this experiment, we use a subset of 1396 videos from the MED11 event kits which are at least 90 seconds long. From each video, we uniformly sample 4 context

Method	mAP (%)
Two-stream pre-trained [132]	20.11
fc6	19.27
fc7	22.99
Our model (no temporal)	22.50
Our model (no future)	21.71
Our model (no hard neg.)	24.12
Our model	**26.74**

Table 7.2: Temporal retrieval results on the MED11 event kits.

frames, 3 positive frames from in between the middle two context frames, and 12 negative distractors from the remaining segments of the video. In addition to the 12 negative distractors from the same video, all frames from other videos are also treated as negative distractors. For each video, given the 4 context frames we evaluate our ability to retrieve the 3 positive frames from this large pool of distractors.

We retrieve frames based on their cosine similarity to the average of the features extracted from the context frames, and use mean Average Precision (mAP) as before. We use the same baselines as the event retrieval task. The results are shown in Table 7.2.

Our embedding representation which is trained to capture temporal structure in videos is seen to outperform the other representations. This also shows their ability to capture long-term interactions between events occurring at different instants of a video.

Temporal retrieval examples. We visualize the top examples retrieved for a few temporal queries in Figure 7.7. Here, we can see just how difficult this task is, as often frames that seem to be viable options for temporal retrieval are not part of the ground truth. For instance, in the "sandwich" example, our embedding wrongly retrieves frames of human hands to keep up with the temporal flow of the video.

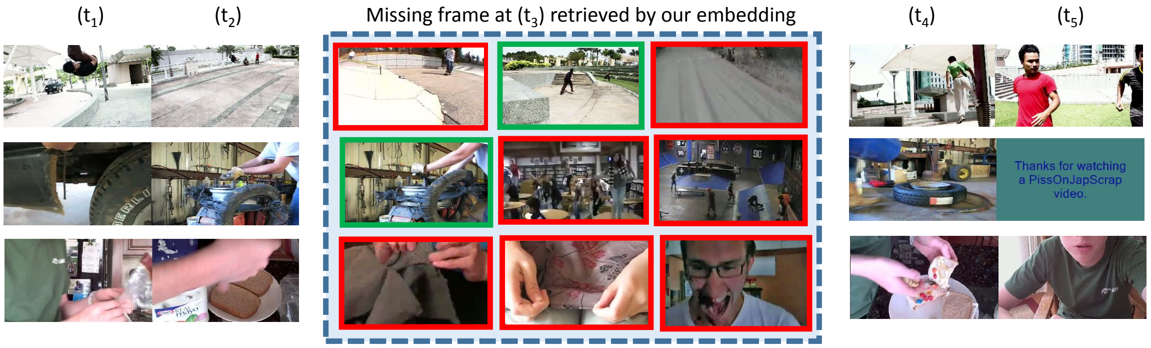

(t_1)
(t_2)
Missing frame at (t_3) retrieved by our embedding
(t_4)
(t_5)
Thanks for watching a PissOnJapScrap video.

Figure 7.7: The retrieval results for our embedding model on the temporal retrieval task. The first and last 2 columns show the 4 context frames sampled from each video, and the middle 3 columns show the top 3 frames retrieved by our embedding. The correctly retrieved frames are highlighted in green, and incorrect frames highlighted in red.

Method	mAP (%)
Two-stream fine-tuned [132]	62.99
ISA [85]	55.87
Izadinia et al. [54] linear	62.63
Izadinia et al. [54] full	66.10
Raman. et al. [116]	66.39
fc6	68.56
fc7	69.17
Our model (no temporal)	69.57
Our model (no future)	69.22
Our model (no hard neg.)	69.81
Our model	**71.17**

Table 7.3: Event classification results on the MED11 event kits.

7.6 Complex Event Classification

The complex event classification task on the MED11 event kits is one of the more challenging classification tasks. We follow the protocol of [54, 116] and use the same train/test splits. Since the goal of our work is to evaluate the effectiveness of video frame representations, we use a simple linear Support Vector Machine classifier for all methods.

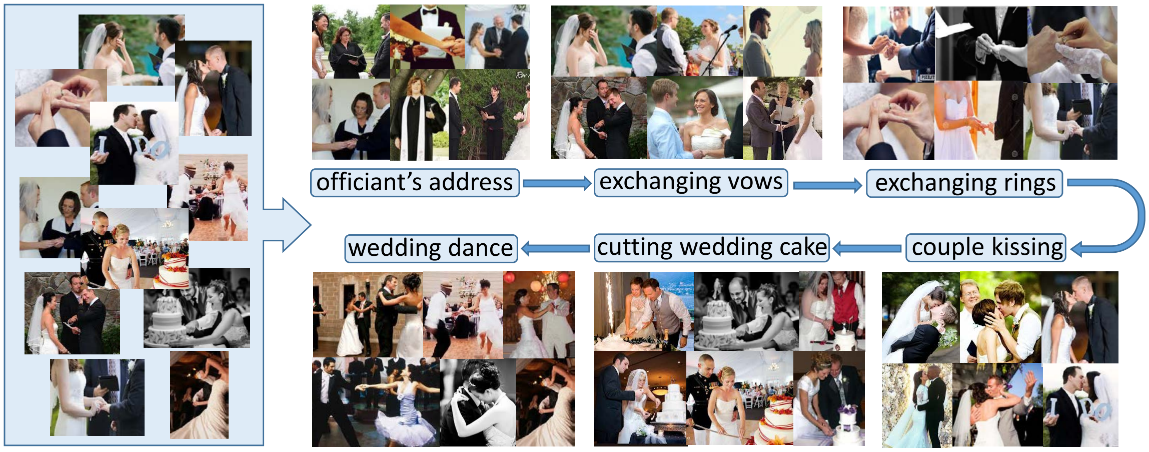
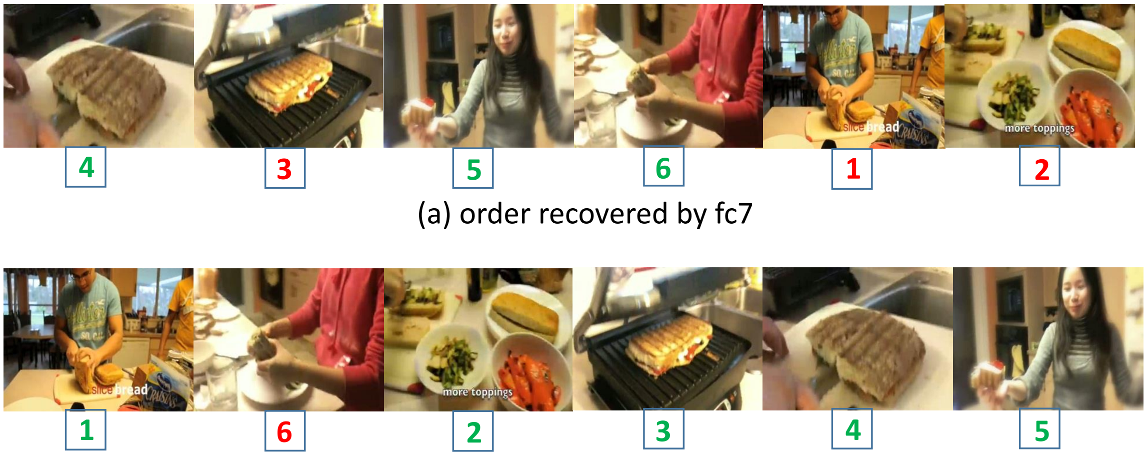


Figure 7.8: After querying the Internet for images of the "wedding" event, we cluster them into sub-events and temporally organize the clusters using our model.

Unlike retrieval settings, we are provided labeled training instances in the event classification task. Thus, we fine-tune the last two layers of the two-stream model (pre-trained on UCF-101) on the training split of the event kits, and found this to perform better than the pre-trained model.

In addition to baselines from previous tasks, we also compare with [54], [85] and [116], with results shown in Table 7.3. Note that [54, 116] use a combination of multiple image and video features including SIFT, MFCC, ISA, and HOG3D. Further, they also use additional labels such as low-level events within each video. In Table 7.3, Izadinia et al. linear refers to the results without low-level event labels.

We observe that our method outperforms ISA [85], which is also a unsupervised neural network feature representation. Additionally, the CNN features trained from ImageNet seem to perform better than most previous feature representations, which is also consistent with the retrieval results and previous work [166, 175]. Among the methods, the two-stream model holds the advantage of being fine-tuned to the MED11 event kits. However, our performance gain could be attributed to the ability of our model to use large amounts of unlabeled data to learn a better representations.



(a) order recovered by fc7

(b) order recovered by our embedding

Figure 7.9: An example of the temporal ordering retrieved by fc7 and our method for a "Making a sandwich" video. The indexes of the frames already in the correct temporal order are shown in green, and others in red.

7.7 Temporal Order Recovery

An effective representation for video frames should be able to not only capture visual similarities, but also preserve the structure between temporally coherent frames. This facilitates holistic video understanding tasks beyond classification and retrieval. With this in mind, we explore the video temporal order recovery task, which seeks to show how the temporal interaction between different parts of a complex event are inherently captured by our embedding.

In this task, we are given as input a jumbled sequence of frames belonging to a video, and our goal is to order the frames into the correct sequence. This has been previously explored in the context of photostreams [70], and has potential for use in applications such as album generation.

Solving the order recovery problem. Since our goal is to evaluate the effectiveness of various feature representations for this task, we use a simple greedy technique to recover the temporal order. We assume that we are provided the first two frames in the video and proceed to retrieve the next frame (third frame) from all other frames

Method	1.4k Videos	1k Videos
Random chance	50.00	50.00
Two-stream [132]	42.05	44.18
fc6	42.43	43.33
fc7	41.67	43.15
Our model (pairwise)	42.03	43.72
Our model (no future)	40.91	42.98
Our model (no hard neg.)	41.02	41.95
Our model	**40.41**	**41.13**

Table 7.4: Video temporal order recovery results on the MED11 event kits evaluated using the Kendell tau distance (normalized to 0-100).

in the video. This is done by averaging the first two frames and retrieving the closest frame in cosine similarity. We go on to greedily retrieve the fourth frame using the average of the second and third frames, and continue until all frames are retrieved. In order to enable easy comparison across all videos, we sample the same number of frames (12) from each video before scrambling them for the order recovery problem. An example comparing our embeddings to fc7 is show in Figure 7.9.

Evaluation. We evaluate the performance for solving the order recovery problem using the Kendall tau [69] distance between the groundtruth sequence of frames and the sequence returned by the greedy method. The Kendall tau distance is a metric that counts the number of pairwise disagreements between two ranked lists; the larger the distance the more dissimilar the lists. The performance of different features for this task is shown in Table 7.4, where the Kendall tau distance is normalized to be in the range $0 - 100$.

Similar to the temporal retrieval setting, we use the subset of 1396 videos which are at least 90 seconds long. These results are reported in the first column of the table. We observed that our performance was quite comparable to that of fc7 features for videos with visually similar frames like those from the “parade” event, as they lack interesting temporal structure. Hence, we also report results on the subset of 1000 videos which had the most visually distinct frames. These results are shown in the second column of the table. We also evaluated the human performance of this task on a random subset of 100 videos, and found the Kendell tau to be around 42. This is on par with the performance of the automatic temporal order produced by our methods, and illustrates the difficulty of this task for humans as well.

We observe that our full context model trained with a temporal objective achieves the best Kendall tau distance. This improvement is more marked in the case of the 1k Videos with more visually distinct frames. This shows the ability of our model to bring together sequences of frames that should be temporally and semantically coherent.

Ordering actions on the Internet. Image search on the Internet has improved to the point where we can find relevant images with textual queries. Here, we wanted to investigate whether or not we could also temporally order images returned from the Internet for textual queries that involve complex events. To do this, we used query expansion on the "wedding" query, and crawled Google for a large set of images. Then, based on the queries, we clustered the images into sets of semantic clusters, and for each cluster, averaged our embedding features to obtain a representation for the cluster. With this representation, we then used our method to recover the temporal ordering of these clusters of images. In Figure 7.8, we show the temporal ordering automatically recovered by our embedded features, and some example images from each cluster. Interestingly, the recovered order seems consistent with typical wedding scenarios.

7.8 Summary

In this chapter, we presented a model to embed video frames. We treated videos as sequences of frames and embedded them in a way which captures the temporal context surrounding them. Our embeddings were learned from a large collection of more than 40000 unlabeled videos, and have shown to be more effective for multiple video tasks. The learned embeddings performed better than other video frame representations for all tasks. The main thrust of our work is to push a framework for learning frame-level representations from large sets of unlabeled video, which can then be used for a wide range of generic video tasks.

Chapter 8

Conclusions and Future Directions

8.1 Conclusions

This thesis has focused on the problem of visual learning with weakly labeled video. In particular, we have addressed several standard problems, such as video classification, object localization, and video representation. While typical approaches to these problems usually involve utilizing laborious annotations, we are able to take advantage of the abundance and diversity of visual data available on the Internet by developing approaches that can effectively learn from weakly labeled video.

In Chapter 2, we addressed the problem of complex event recognition in weakly labeled video. We introduced a model for learning the latent temporal structure of complex events in weakly labeled Internet videos that are not temporally localized. Our model is simple, and lends itself to fast, exact inference, which allows us to process large numbers of videos efficiently. In addition, we trained our model in a discriminative, max-margin fashion and are able to achieve competitive accuracies on activity recognition and event detection tasks. We also showed examples of semantic structure that our model is able to automatically extract.

In Chapters 3, 4, 5, and 6, we addressed the problem of object localization in weakly labeled images and video. In Chapter 3, we introduced an approach for adapting object detectors from image to video that discovers robust examples in weakly labeled video data using feature tracks. We introduced a novel self-paced domain adaptation algorithm to iteratatively adapt to these discovered examples that is simultaneously able to consider target features unique to the video domain. In Chapter 4, we introduced CRANE, a simple yet effective algorithm for annotating spatiotemporal segments in weakly labeled videos, and presented a generalized interpretation based on the distance matrix that serves as a taxonomy for weakly supervised methods. In Chapters 5 and 6, we focused on the problem of co-localization in both images and video, and introduced a method that combines terms for the prior, similarity, and discriminability of images and boxes into a joint optimization problem. Further, we showed how the method is able to account for noisy images with incorrect anntations, and naturally incorporate temporal consistency in the form of temporal terms and constraints.

In Chapter 7, we addressed the problem of learning temporal embeddings from weakly labeled video. Here, we took advantage of the implicit weak label that videos are sequences of temporally and semantically coherent images. Using this intuition, we formulated an objective that learns temporal embeddings for frames of video by associating frames with the temporal context that they appear in. We then showed how these embeddings are able to capture semantic context, resulting in better performance for a wide variety of standard tasks in video.

8.2 Future directions

8.2.1 Weakly supervised object localization

In Chapters 3, 4, 5, and 6, we addressed the problem of weakly supervised object localization from various perspectives. These include adapting object detectors to video [143], using only negative weakly labeled data [144], and using only positive weakly labeled data [66, 142]. Each of the proposed methods/scenarios incurs different tradeoffs between factors such as runtime complexity, accuracy, and parallelization, and incorporating these approaches into a joint framework with the right balance between these tradeoffs is a possible direction for future work. In addition, simplifying the problem by supplementing the algorithms with small amounts of labeled data is also an alternative direction worth exploring that could potentially yield large improvements in accuracy.

8.2.2 Multimodal video understanding

In Chapters 2 and 7, we looked at how leveraging temporal consistency in video could be used to improve tasks that only have access to weakly labeled video. However, there are also several other multimodal dimensions to video data that could be leveraged, including modalities such as audio, and metadata such as social and location information. Using these modalities to help improve weakly supervised video classification is a possible direction for future work. Learning embeddings that can capture the interactions between these modalities using large amounts of weakly labeled video could also be a potential direction worth exploring.

Bibliography

[1] M. Albanese, R. Chellappa, N. P. Cuntoor, V. Moscato, A. Picariello, V. S. Subrahmanian, and O. Udrea. “A Constrained Probabilistic Petri Net Framework for Human Activity Detection in Video”. *IEEE Transactions on Multimedia* 10.6 (2008), pp. 982–996.

[2] B. Alexe, T. Deselaers, and V. Ferrari. “Measuring the Objectness of Image Windows”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2189–2202.

[3] K. Ali, D. Hasler, and F. Fleuret. “FlowBoost—Appearance Learning from Sparsely Annotated Video”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[4] B. Babenko, M.-H. Yang, and S. Belongie. “Robust Object Tracking with Online Multiple Instance Learning”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.8 (2011), pp. 1619–1632.

[5] F. Bach and Z. Harchaoui. “DIFFRAC: a discriminative and flexible framework for clustering”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2007.

[6] S. Y. Bao, Y. Xiang, and S. Savarese. “Object Co-detection”. *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012.

[7] M. Belkin and P. Niyogi. “Laplacian eigenmaps for dimensionality reduction and data representation”. *Neural computation* 15.6 (2003), pp. 1373–1396.

[8] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. “Multiple Object Tracking Using K-Shortest Paths Optimization”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.9 (2011), pp. 1806–1819.

[9] A. Bergamo and L. Torresani. “Exploiting weakly-labeled Web images to improve object classification: a domain adaptation approach”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS).* 2010.

[10] Y. Boykov, O. Veksler, and R. Zabih. “Fast Approximate Energy Minimization via Graph Cuts”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23.11 (2001), pp. 1222–1239.

[11] W. Brendel and S. Todorovic. “Learning Spatiotemporal Graphs of Human Activities”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2011.

[12] T. Brox and J. Malik. “Object Segmentation by Long Term Analysis of Point Trajectories”. *Proceedings of the European Conference on Computer Vision (ECCV).* 2010.

[13] J. Carreira and C. Sminchisescu. “Constrained parametric min-cuts for automatic object segmentation”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2010.

[14] V. Chari, S. Lacoste-Julien, J. Sivic, and I. Laptev. *On Pairwise Cost for Multi-Object Network Flow Tracking.* Tech. rep. arXiv, 2014.

[15] R. Chaudhry, A. Ravichandran, G. D. Hager, and R. Vidal. “Histograms of oriented optical flow and Binet-Cauchy kernels on nonlinear dynamical systems for the recognition of human actions”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2009.

[16] Y. Chen, J. Bi, and J. Wang. “MILES: Multiple-Instance Learning via Embedded Instance Selection”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.12 (2006).

[17] M.-M. Cheng, G.-X. Zhang, N. J. Mitra, X. Huang, and S.-M. Hu. "Global Contrast based Salient Region Detection". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[18] N. Cherniavsky, I. Laptev, J. Sivic, and A. Zisserman. "Semi-supervised learning of facial attributes in video". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[19] W.-C. Chiu and M. Fritz. "Multi-Class Video Co-Segmentation with a Generative Multi-Video Model". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[20] C. M. Christoudias, R. Urtasun, M. Salzmann, and T. Darrell. "Learning to recognize objects from unseen modalities". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[21] O. Chum and A. Zisserman. "An Exemplar Model for Learning Object Classes". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2007.

[22] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2005.

[23] N. Dalal, B. Triggs, and C. Schmid. "Human detection using oriented histograms of flow and appearance". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2006.

[24] A. Delong, L. Gorelick, O. Veksler, and Y. Boykov. "Minimizing Energies with Hierarchical Costs". *International Journal of Computer Vision* 100.1 (2012), pp. 38–58.

[25] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. "Fast Approximate Energy Minimization with Label Costs". *International Journal of Computer Vision* 96.1 (2012), pp. 1–27.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009.

[27] T. Deselaers, B. Alexe, and V. Ferrari. "Localizing Objects While Learning Their Appearance". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[28] T. Deselaers, B. Alexe, and V. Ferrari. "Weakly Supervised Localization and Learning with Generic Knowledge". *International Journal of Computer Vision* 100.3 (2012), pp. 275–293.

[29] L. Duan, D. Xu, I. W.-H. Tsang, and J. Luo. "Visual event recognition in videos by learning from web data". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010.

[30] J. C. Duchi and Y. Singer. "Boosting with structural sparsity". *Proceedings of the International Conference on Machine Learning (ICML)*. 2009.

[31] J. C. Dunn. "Convergence rates for conditional gradient sequences generated by implicit step length rules". *SIAM Journal on Control and Optimization* 18.5 (1980), pp. 473–487.

[32] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh. "Activity Recognition and Abnormality Detection with the Switching Hidden Semi-Markov Model". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2005.

[33] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. "The Pascal Visual Object Classes (VOC) Challenge". *International Journal of Computer Vision* (2010).

[34] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. "LIBLINEAR: A Library for Large Linear Classification". *Journal of Machine Learning Research* (2008).

[35] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. "Object Detection with Discriminatively Trained Part-Based Models". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2010).

[36] M. Frank and P. Wolfe. "An algorithm for quadratic programming". *Naval research logistics quarterly* 3.1-2 (1956), pp. 95–110.

[37] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. "Devise: A deep visual-semantic embedding model". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2013.

[38] A. Gaidon, Z. Harchaoui, and C. Schmid. "Actom Sequence Models for Efficient Action Detection". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[39] T. Gao and D. Koller. "Discriminative Learning of Relaxed Hierarchy for Large-scale Visual Recognition". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2011.

[40] R. Gokberk Cinbis, J. Verbeek, and C. Schmid. "Multi-fold MIL Training for Weakly Supervised Object Localization". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[41] R. Gopalan, R. Li, and R. Chellappa. "Domain adaptation for object recognition: An unsupervised approach". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2011.

[42] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. "Actions as Space-Time Shapes". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2007).

[43] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun. "Unsupervised Feature Learning from Temporal Data". *arXiv preprint arXiv:1504.02518* (2015).

[44] M. Grundmann, V. Kwatra, M. Han, and I. A. Essa. "Efficient hierarchical graph-based video segmentation". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010.

[45] J. Guelat and P. Marcotte. “Some comments on Wolfe’s away step”. *Mathematical Programming* 35.1 (1986), pp. 110–119.

[46] S. Hare, A. Saffari, and P. H. S. Torr. “Struck: Structured output tracking with kernels”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2011.

[47] G. Hartmann, M. Grundmann, J. Hoffman, D. Tsai, V. Kwatra, O. Madani, S. Vijayanarasimhan, I. A. Essa, J. M. Rehg, and R. Sukthankar. “Weakly Supervised Learning of Object Segmentations from Web-Scale Video”. *ECCV Workshop on Vision in Web-Scale Media.* 2012.

[48] H. Harzallah, F. Jurie, and C. Schmid. “Combining efficient object localization and image classification”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2009.

[49] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer-Verlag, 2001.

[50] M. Hoai, Z.-Z. Lan, and F. De la Torre. “Joint Segmentation and Classification of Human Actions in Video”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2011.

[51] V. J. Hodge and J. Austin. “A Survey of Outlier Detection Methodologies”. *Artificial Intelligence Review* 22.2 (2004), pp. 85–126.

[52] S. Hongeng and R. Nevatia. “Large-Scale Event Detection Using Semi-Hidden Markov Models”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2003.

[53] H. D. III. “Frustratingly Easy Domain Adaptation”. *Association of Computational Linguistics.* 2007.

[54] H. Izadinia and M. Shah. “Recognizing Complex Events using Large Margin Joint Low-Level Event Model”. *Proceedings of the European Conference on Computer Vision (ECCV).* 2012.

[55] M. Jaggi. “Revisiting Frank-Wolfe: Projection-free sparse convex optimization”. *Proceedings of the International Conference on Machine Learning (ICML)*. 2013.

[56] M. Jain, H. Jégou, and P. Bouthemy. “Better exploiting motion for better action recognition”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[57] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. “A biologically inspired system for action recognition”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2007.

[58] S. Ji, W. Xu, M. Yang, and K. Yu. “3D convolutional neural networks for human action recognition”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2013).

[59] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. “Caffe: Convolutional Architecture for Fast Feature Embedding”. *arXiv preprint arXiv:1408.5093* (2014).

[60] L. Jiang, A. G. Hauptmann, and G. Xiang. “Leveraging high-level and low-level features for multimedia event detection”. *ACM International Conference on Multimedia*. 2012.

[61] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. “Trajectory-Based Modeling of Human Actions with Motion Reference Points”. *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012.

[62] T. Joachims. “Transductive Inference for Text Classification using Support Vector Machines”. *Proceedings of the International Conference on Machine Learning (ICML)*. 1999.

[63] A. Joulin, F. Bach, and J. Ponce. “Discriminative Clustering for Image Cosegmentation”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010.

[64] A. Joulin, F. Bach, and J. Ponce. “Multi-Class Cosegmentation”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[65] A. Joulin and F. Bach. “A convex relaxation for weakly supervised classifiers”. *Proceedings of the International Conference on Machine Learning (ICML)*. 2012.

[66] A. Joulin, K. Tang, and L. Fei-Fei. “Efficient Image and Video Co-localization with Frank-Wolfe Algorithm”. *Proceedings of the European Conference on Computer Vision (ECCV)*. 2014.

[67] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. “Large-scale video classification with convolutional neural networks”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[68] Y. Ke, R. Sukthankar, and M. Hebert. “Event Detection in Crowded Videos”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2007.

[69] M. G. Kendall. “A new measure of rank correlation”. *Biometrika* (1938), pp. 81–93.

[70] G. Kim and E. P. Xing. “Jointly aligning and segmenting multiple web photo streams for the inference of collective photo storylines”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[71] G. Kim, E. P. Xing, L. Fei-Fei, and T. Kanade. “Distributed cosegmentation via submodular optimization on anisotropic diffusion”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2011.

[72] R. Kiros, R. Salakhutdinov, and R. S. Zemel. “Unifying visual-semantic embeddings with multimodal neural language models”. *arXiv preprint arXiv:1411.2539* (2014).

[73] A. Kläser, M. Marszałek, and C. Schmid. "A Spatio-Temporal Descriptor Based on 3D-Gradients". *Proceedings of the British Machine Vision Conference (BMVC)*. 2008.

[74] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep convolutional neural networks". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2012.

[75] E. Krupka and N. Tishby. "Generalization in Clustering with Unobserved Features". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2005.

[76] B. Kulis, K. Saenko, and T. Darrell. "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[77] P. Kumar, B. Packer, and D. Koller. "Self-Paced Learning for Latent Variable Models". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2010.

[78] D. Küttel, M. Guillaumin, and V. Ferrari. "Segmentation Propagation in ImageNet". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012.

[79] S. Lacoste-Julien and M. Jaggi. "An Affine Invariant Linear Convergence Analysis for Frank-Wolfe Algorithms". *arXiv preprint arXiv:1312.7864* (2013).

[80] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. "Block-Coordinate Frank-Wolfe Optimization for Structural SVMs". *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 28. 2012, pp. 1438–1444.

[81] K.-T. Lai, D. Liu, M.-S. Chen, and S.-F. Chang. "Recognizing Complex Events in Videos by Learning Key Static-Dynamic Evidences". *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2014.

[82] I. Laptev. "On Space-Time Interest Points". *International Journal of Computer Vision* (2005).

[83] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. “Learning realistic human actions from movies”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2008.

[84] S. Lazebnik, C. Schmid, and J. Ponce. “Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2006.

[85] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. “Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2011.

[86] Y. J. Lee, J. Kim, and K. Grauman. “Key-Segments for Video Object Segmentation”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2011.

[87] C. Leistner, M. Godec, S. Schulter, A. Saffari, M. Werlberger, and H. Bischof. “Improving Classifiers with Unlabeled Weakly-Related Videos”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2011.

[88] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. “Track to the Future: Spatio-temporal Video Segmentation with Long-range Motion Cues”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2011.

[89] L.-J. Li and L. Fei-Fei. “OPTIMOL: automatic Online Picture collecTion via Incremental MOdel Learning”. *International Journal of Computer Vision* (2009).

[90] J. J. Lim, R. Salakhutdinov, and A. Torralba. “Transfer Learning by Borrowing Examples for Multiclass Object Detection”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS).* 2011.

[91] J. Liu, J. Luo, and M. Shah. "Recognizing Realistic Actions from Videos "in the Wild"". 2009.

[92] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision* 60.2 (2004), pp. 91–110.

[93] B. D. Lucas and T. Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision". *International Joint Conference on Artificial Intelligence*. 1981.

[94] L. Van der Maaten and G. Hinton. "Visualizing data using t-SNE". *Journal of Machine Learning Research* 9.2579-2605 (2008), p. 85.

[95] T. Mikolov, K. Chen, G. Corrado, and J. Dean. "Efficient estimation of word representations in vector space". *arXiv preprint arXiv:1301.3781* (2013).

[96] P. Natarajan and R. Nevatia. "Coupled Hidden Semi Markov Models for Activity Recognition". *Proceedings of the IEEE Workshop on Motion and Video Computing (WMVC)*. 2007.

[97] P. Natarajan, S. Wu, S. Vitaladevuni, X. Zhuang, S. Tsakalidis, U. Park, and R. Prasad. "Multimodal feature fusion for robust event detection in web videos". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[98] M. H. Nguyen, L. Torresani, F. De la Torre, and C. Rother. "Weakly supervised discriminative localization and classification: a joint learning process". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2009.

[99] J.-C. Niebles, B. Han, A. Ferencz, and L. Fei-Fei. "Extracting Moving People from Internet Videos". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2008.

[100] J. C. Niebles, C.-W. Chen, and L. Fei-Fei. "Modeling temporal structure of decomposable motion segments for activity classification". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[101] J. C. Niebles, H. Wang, and L. Fei-Fei. “Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words”. *International Journal of Computer Vision* (2008).

[102] S. Oh, S. McCloskey, I. Kim, A. Vahdat, K. Cannons, H. Hajimirsadeghi, G. Mori, A. Perera, M. Pandey, and J. Corso. “Multimedia event detection with multimodal feature fusion and temporal concept localization”. *Machine Vision and Applications* 25 (2014).

[103] T. Ojala, M. Pietikainen, and D. Harwood. “Performance evaluation of texture measures with classification based on Kullback discrimination of distributions”. *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR).* 1994.

[104] B. Ommer, T. Mader, and J. Buhmann. “Seeing the Objects Behind the Dots: Recognition in Videos from a Moving Camera”. *International Journal of Computer Vision* 83.1 (2009).

[105] D. Oneata, J. Verbeek, and C. Schmid. “Action and event recognition with Fisher vectors on a compact feature set”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2013.

[106] P. Over, G. Awad, M. Michel, J. Fiscus, W. Kraaij, A. F. Smeaton, and G. Quenot. “TRECVID 2011 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics”. *Proceedings of TRECVID 2011.* NIST, USA. 2011.

[107] M. Pandey and S. Lazebnik. “Scene recognition and weakly supervised object localization with deformable part-based models”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2011.

[108] Y. Pang and H. Ling. “Finding the Best from the Second Bests - Inhibiting Subjective Bias in Evaluation of Visual Tracking Algorithms”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2013.

[109] A. Papazoglou and V. Ferrari. “Fast object segmentation in unconstrained video”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013.

[110] X. Peng, C. Zou, Y. Qiao, and Q. Peng. “Action recognition with stacked fisher vectors”. *Proceedings of the European Conference on Computer Vision (ECCV)*. 2014.

[111] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung. “Saliency filters: Contrast based filtering for salient region detection”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[112] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. “Color-Based Probabilistic Tracking”. *Proceedings of the European Conference on Computer Vision (ECCV)*. 2002.

[113] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. “Learning object class detectors from weakly annotated video”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[114] A. Quattoni, S. Wang, L.-P Morency, M. Collins, and T. Darrell. “Hidden-state conditional random fields”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2007.

[115] D. Ramanan, D. Forsyth, and K. Barnard. “Building models of animals from video”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.8 (2006).

[116] V. Ramanathan, P. Liang, and L. Fei-Fei. “Video Event Understanding using Natural Language Descriptions”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013.

[117] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. “Video (language) modeling: a baseline for generative models of natural videos”. *arXiv preprint arXiv:1412.6604* (2014).

[118] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut: interactive foreground extraction using iterated graph cuts". *ACM Transactions on Graphics (SIGGRAPH)* 23.3 (2004).

[119] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. "Unsupervised Joint Object Discovery and Segmentation in Internet Images". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2013).

[120] J. C. Rubio, J. Serrat, and A. M. López. "Video Co-segmentation". *Proceedings of the Asian Conference on Computer Vision (ACCV).* 2012.

[121] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. "Using Multiple Segmentations to Discover Objects and their Extent in Image Collections". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2006.

[122] S. Sadanand and J. J. Corso. "Action bank: A high-level representation of activity in video". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2012.

[123] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. "Adapting visual category models to new domains". *Proceedings of the European Conference on Computer Vision (ECCV).* 2010.

[124] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. "Segmentation as selective search for object recognition". *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* 2011.

[125] S. Sarawagi and W. W. Cohen. "Semi-Markov Conditional Random Fields for Information Extraction". *Proceedings of the Conference on Neural Information Processing Systems (NIPS).* 2004.

[126] S. Satkin and M. Hebert. "Modeling the Temporal Extent of Actions". *Proceedings of the European Conference on Computer Vision (ECCV).* 2010.

[127] C. Schüldt, I. Laptev, and B. Caputo. "Recognizing human actions: A local SVM approach". *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR).* 2004.

[128] G. Schweikert, C. Widmer, B. Schölkopf, and G. Rätsch. "An Empirical Analysis of Domain Adaptation Algorithms for Genomic Sequence Analysis". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2008.

[129] P. Sharma, C. Huang, and R. Nevatia. "Unsupervised incremental learning for improved object detection in a video". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[130] J. Shi and J. Malik. "Normalized cuts and image segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.8 (2000), pp. 888–905.

[131] Z. Shi, T. M. Hospedales, and T. Xiang. "Bayesian Joint Topic Modelling for Weakly Supervised Object Localisation". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013.

[132] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". *arXiv preprint arXiv:1409.1556* (2014).

[133] P. Siva, C. Russell, and T. Xiang. "In Defence of Negative Mining for Annotating Weakly Labelled Data". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012.

[134] P. Siva, C. Russell, T. Xiang, and L. de Agapito. "Looking Beyond the Image: Unsupervised Learning for Object Saliency and Detection". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[135] P. Siva and T. Xiang. "Weakly supervised object detector learning with model drift detection". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2011.

[136] C. Sminchisescu, A. Kanaujia, and D. Metaxas. "Conditional models for contextual human motion recognition". *Computer Vision and Image Understanding* (2006).

[137] Z. Song, Q. Chen, Z. Huang, Y. Hua, and S. Yan. “Contextualizing object detection and classification”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[138] K. Soomro, A. R. Zamir, and M. Shah. “Ucf101: A dataset of 101 human actions classes from videos in the wild”. *arXiv preprint arXiv:1212.0402* (2012).

[139] N. Srivastava, E. Mansimov, and R. Salakhutdinov. “Unsupervised Learning of Video Representations using LSTMs”. *arXiv preprint arXiv:1502.04681* (2015).

[140] A. Tamrakar, S. Ali, Q. Yu, J. Liu, O. Javed, A. Divakaran, H. Cheng, and H. Sawhney. “Evaluation of low-level features and their combinations for complex event detection in open source videos”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[141] K. Tang, L. Fei-Fei, and D. Koller. “Learning Latent Temporal Structure for Complex Event Detection”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[142] K. Tang, A. Joulin, L.-J. Li, and L. Fei-Fei. “Co-localization in Real-World Images”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014.

[143] K. Tang, V. Ramanathan, L. Fei-Fei, and D. Koller. “Shifting Weights: Adapting Object Detectors from Image to Video”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2012.

[144] K. Tang, R. Sukthankar, J. Yagnik, and L. Fei-Fei. “Discriminative Segment Annotation in Weakly Labeled Video”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[145] K. Tang, B. Yao, L. Fei-Fei, and D. Koller. “Combining the Right Features for Complex Event Recognition”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013.

[146] B. Taskar, M.-F. Wong, and D. Koller. "Learning on the Test Data: Leveraging 'Unseen' Features". *Proceedings of the International Conference on Machine Learning (ICML)*. 2003.

[147] G. Taylor, R. Fergus, Y. LeCun, and C. Bregler. "Convolutional learning of spatiotemporal features". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[148] C. Tomasi and T. Kanade. *Detection and Tracking of Point Features*. Tech. rep. CMU, 1991.

[149] T. Tommasi, F. Orabona, and B. Caputo. "Safety in numbers: Learning categories from few examples with multi model knowledge transfer". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010.

[150] P. K. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. "Machine Recognition of Human Activities: A Survey". *IEEE Transactions on Circuits and Systems for Video Technology* (2008).

[151] A. Vahdat and G. Mori. "Handling Uncertain Tags in Visual Recognition". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2013.

[152] A. Vedaldi and A. Zisserman. "Efficient Additive Kernels via Explicit Feature Maps". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010.

[153] A. Vezhnevets, V. Ferrari, and J. M. Buhmann. "Weakly supervised structured output learning for semantic segmentation". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[154] S. Vicente, V. Kolmogorov, and C. Rother. "Cosegmentation Revisited: Models and Optimization". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010.

[155] S. Vicente, C. Rother, and V. Kolmogorov. “Object cosegmentation”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[156] P. Viola, J. Platt, and C. Zhang. “Multiple Instance Boosting for Object Detection”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2005.

[157] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. “Action Recognition by Dense Trajectories”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[158] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid. “Evaluation of local spatio-temporal features for action recognition”. *Proceedings of the British Machine Vision Conference (BMVC)*. 2009.

[159] X. Wang, T. X. Han, and S. Yan. “An HOG-LBP human detector with partial occlusion handling”. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2009.

[160] X. Wang, G. Hua, and T. X. Han. “Detection by detections: Non-parametric detector adaptation for a video”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[161] Y. Wang and G. Mori. “Human Action Recognition by Semi-Latent Topic Models”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2009).

[162] P. Wolfe. “Convergence theory in nonlinear programming”. *Integer and nonlinear programming* (1970), pp. 1–36.

[163] J. Xiao and M. Shah. “Motion Layer Extraction in the Presence of Occlusion using Graph Cuts”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.10 (2005).

[164] C. Xu, C. Xiong, and J. Corso. “Streaming Hierarchical Video Segmentation”. *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012.

[165] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. "Maximum margin clustering". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2004.

[166] Z. Xu, Y. Yang, and A. G. Hauptmann. "A Discriminative CNN Video Representation for Event Detection". *arXiv preprint arXiv:1411.4006v1* (2015).

[167] J. Yang, R. Yan, and A. G. Hauptmann. "Cross-Domain Video Concept Detection Using Adaptive SVMs". *ACM International Conference on Multimedia.* 2007.

[168] M. Yang, S. Zhu, F. Lv, and K. Yu. "Correspondence driven adaptation for human profile recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2011.

[169] Y. Yang and M. Shah. "Complex events detection using data-driven concepts". *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012.

[170] A. Yilmaz, O. Javed, and M. Shah. "Object tracking: A survey". *ACM Computing Surveys* 38.4 (2006).

[171] C.-N. J. Yu and T. Joachims. "Learning Structural SVMs with Latent Variables". *Proceedings of the International Conference on Machine Learning (ICML)*. 2009.

[172] J. Yuen, B. C. Russell, C. Liu, and A. Torralba. "LabelMe video: Building a video database with human annotations". *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2009.

[173] A. L. Yuille. "The concave-convex procedure". *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2002.

[174] W. Zaremba, I. Sutskever, and O. Vinyals. "Recurrent neural network regularization". *arXiv:1409.2329* (2014).

[175] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov. "Exploiting Image-trained CNN Architectures for Unconstrained Video Classification". *arXiv preprint arXiv:1503.04144v2* (2015).

[176] Z.-J. Zha, X.-S. Hua, T. Mei, J. Wang, G.-J. Qi, and Z. Wang. “Joint Multi-Label Multi-Instance Learning for Image Classification”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2008.

[177] C. Zhang, R. Hamid, and Z. Zhang. “Taylor expansion based classifier adaptation: Application to person detection”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2008.

[178] G.-T. Zhou, T. Lan, A. Vahdat, and G. Mori. “Latent Maximum Margin Clustering”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2013.

[179] Z.-H. Zhou and M.-L. Zhang. “Multi-Instance Multi-Label Learning with Application to Scene Classification”. *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2007.