ABSTRACT

Title of dissertation: Analyzing Complex Events and Human Actions in ”in-the-wild” Videos

Hyungtae Lee, Doctor of Philosophy, 2014

Dissertation directed by: Professor Larry S. Davis
Department of Electrical and Computer Engineering

We are living in a world where it is easy to acquire videos of events ranging from private picnics to public concerts, and to share them publicly via websites such as YouTube. The ability of smart-phones to create these videos and upload them to the internet has led to an explosion of video data, which in turn has led to interesting research directions involving the analysis of “in-the-wild” videos. To process these types of videos, various recognition tasks such as pose estimation, action recognition, and event recognition become important in computer vision. This thesis presents various recognition problems and proposes mid-level models to address them.

First, a discriminative deformable part model is presented for the recovery of qualitative pose, inferring coarse pose labels (e:g: left, front-right, back), a task more robust to common confounding factors that hinder the inference of exact 2D or 3D joint locations. Our approach automatically selects parts that are predictive of qualitative pose and trains their appearance and deformation costs to best discriminate between qualitative poses. Unlike previous approaches, our parts are both selected and trained to improve qualitative pose discrimination and are shared by all the qualitative pose models. This leads to both increased accuracy and higher efficiency, since fewer parts models are evaluated for each image. In comparisons with two state-of-the-art approaches on a public dataset, our model shows superior performance.

Second, the thesis proposes the use of a robust pose feature based on part based human detectors (Poselets) for the task of action recognition in relatively unconstrained videos, i.e., collected from the web. This feature, based on the original poselets activation vector, coarsely models pose and its transitions over time. Our main contributions are that we improve the original feature's compactness and discriminability by greedy set cover over subsets of joint configurations, and incorporate it into a unified video-based action recognition framework. Experiments shows that the pose feature alone is extremely informative, yielding performance that matches most state-of-the-art approaches but only using our proposed improvements to its compactness and discriminability. By combining our pose feature with motion and shape, the proposed method outperforms state-of-the-art approaches on two public datasets.

Third, clauselets, sets of concurrent actions and their temporal relationships, are proposed and explored their application to video event analysis. Clauselets are trained in two stages. Initially, clauselet detectors that find a limited set of actions in particular qualitative temporal configurations based on Allen's interval relations is trained. In the second stage, the first level detectors are applied to training videos, and discriminatively learn temporal patterns between activations that involve more actions over longer durations and lead to improved second level

clauselet models. The utility of clauselets is demonstrated by applying them to the task of “in-the-wild” video event recognition on the TRECVID MED 11 dataset. Not only do clauselets achieve state-of-the-art results on this task, but qualitative results suggest that they may also lead to semantically meaningful descriptions of videos in terms of detected actions and their temporal relationships.

Finally, the thesis addresses the task of searching for videos given text queries that are not known at training time, which typically involves zero-shot learning, where detectors for a large set of concepts, attributes, or objects parts are learned under the assumption that, once the search query is known, they can be combined to detect novel complex visual categories. These detectors are typically trained on annotated training data that is time-consuming and expensive to obtain, and a successful system requires many of them to generalize well at test time. In addition, these detectors are so general that they are not well-tuned to the specific query or target data, since neither is known at training. Our approach addresses the annotation problem by searching the web to discover visual examples of short text phrases. Top ranked search results are used to learn general, potentially noisy, visual phrase detectors. Given a search query and a target dataset, the visual phrase detectors are adapted to both the query and unlabeled target data to remove the influence of incorrect training examples or correct examples that are irrelevant to the search query. Our adaptation process exploits the spatio-temporal coocurrence of visual phrases that are found in the target data and which are relevant to the search query by iteratively refining both the visual phrase detectors and spatio-temporally grouped phrase detections (clauselets). Our approach is demonstrated on to the challenging TRECVID MED13 EK0 dataset and show that, using visual features alone, our approach outperforms state-of-the-art approaches that use visual, audio, and text (OCR) features.

ANALYZING COMPLEX EVENTS AND HUMAN ACTIONS IN "IN-THE-WILD" VIDEOS

by

Hyungtae Lee

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2014

Advisory Committee:
Professor Larry S. Davis
Professor Rama Chellappa
Professor David W. Jacobs
Professor Ramani Duraiswami
Professor Joseph F. JaJa

Dedication

I dedicate this thesis to Sanghum Lee my father and Inhye Kim my aunt who are watching me in heaven.

Acknowledgments

I owe my gratitude to all the people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Larry S. Davis for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Rama Chellappa, Prof. David W. Jacobs, Prof. Ramani Duraiswami, and Prof. Joseph F. JaJa, for their encouragement, insightful comments, and hard questions. Also I would like to thank to my master advisor Prof. HyunWook Park in KAIST, Korea.

Specially I would like to thank Vlad I. Morariu. He provides me with direction, technical support and became more of a mentor and friend. My sincere thanks also goes to Dr. Heesung Kwon, and Dr. William D. Nothwang, for offering me the further research opportunities in their groups and leading me working on diverse exciting projects.

I thank my fellow labmates in Computer Vision Laboratory, University of Maryland, Institute for Advanced Computer Studies (UMIACS): Jounghoon Beh, Jonghyun Choi, Sungmin Eom, Zhoulin Jiang, Hyunjong Cho, Yaming Wang, Joe Ng, and other members, for the stimulating discussions, for the sleepless nights we

were working together before deadlines, and for all the fun we have had in the room AVW3340.

My housemates at my place of residence have been a crucial factor in my finishing smoothly. I'd like to express my gratitude to Hyunsoo Kim, Min Choi and Jun Choi for their friendship and support.

Last but not the least, I would like to thank my family: Inwon Kim, Hyeongwook Lee, Nayoung Min, and Siho Lee for supporting me spiritually throughout my life.

It is impossible to remember all, and I apologize to those I've inadvertently left out.

Thank you all and thank God!

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

We are living in a world where it is easy to acquire videos of events ranging from private picnics to public concerts, and to share them publicly via websites such as YouTube. The ability of smart-phones to create these videos and upload them to the internet has led to an explosion of video data, which in turn has led to interesting research directions involving the analysis of "in-the-wild" videos. Video events can be expressed by several sentences, each of which consists of subject, object, scene, and action. Those sentences are connected by temporal relations (e.g. before, during). Therefore the complex event representation in the video can be modeled by these components as figure 1.1. In this thesis, we study action recognition and complex event analysis in the "in-the-wild" videos among various recognition tasks in computer vision applications.

1.1 Action Recognition

Action recognition still remains challenging due to great intra and inter variance of classes, cluttered and occluded background, etc., despite numerous recent advances. Many researchers extract local image and video features from video sequences, separate them into clusters, and generate histogram-based representations.

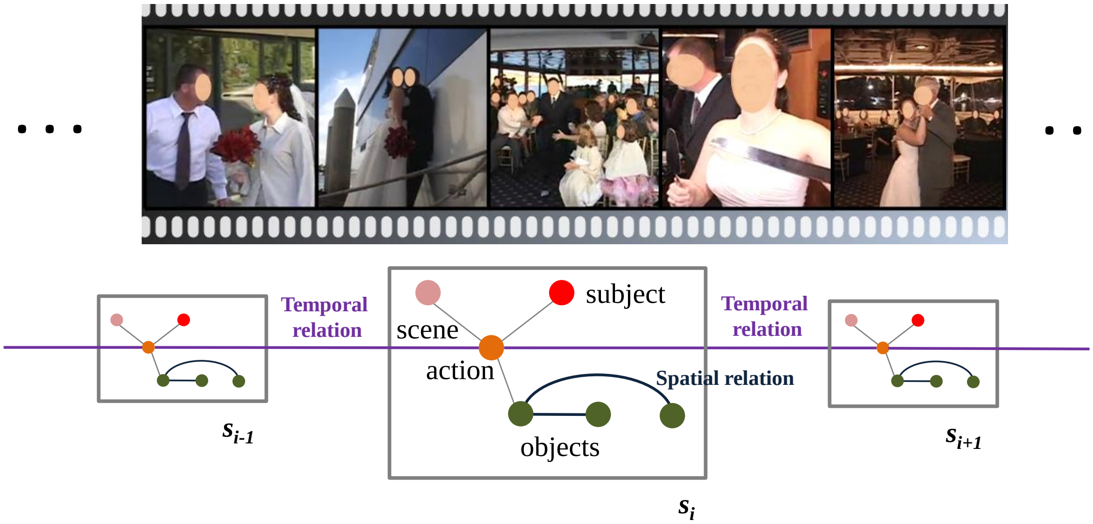


Figure 1.1: Video events can be modeled by multiple sentences consisting of subject, object, scene, and action which are connected by temporal relationships (e.g. before, during).

Interest points are often extracted by methods such as Harris3D [7], Hessian [8], etc, to capture shape and motion of local points. HOG [9], silhouettes [10], and SIFT [4] are commonly used as shape features. As a motion feature, most researchers use optical flow [10] or other custom representations of space-time volumes, e.g., Liu et al. [4] use flat gradients within 3D cuboids.

In general, actions can also be inferred from the kinematic movements of a person's limbs. However, people are highly articulated, limbs occlude each other, loose clothing conceals shape, and low resolution or motion blur lose informative features such as edges. All of these conditions confound pose estimation, making it a difficult and widely studied problem. To ameliorate these problems, researchers often introduce additional information, *e.g.*, multi-view images or depth information, if available, to reconstruct pose in 3D coordinates [11–18]. Instead of relying on additional information, our approach is based on the observation that for many action recognition tasks, it may be sufficient to infer only a qualitative description of a pose, *e.g.*, ‘bent’, ‘laying down’, ‘stretching’, ‘crouched’, ‘facing left’, ‘facing right’, even if the exact joint locations are not identified. We expect models that infer a qualitative description of a person’s pose to be more robust in the presence of the problems that confound exact joint localization. We propose a model for inferring qualitative pose labels automatically from single monocular images. Chapter 2 describes the pose model and chapter 3 employ the pose model into the action recognition framework.

1.2 Complex Event Analysis

Recent approaches to processing these types of videos use features that range from low- to mid-level, some even using features that directly correspond to words that describe portions of the videos [19]. While all of these approaches obtain competitive results on benchmark datasets, mid-level features that can also describe the semantic content of a video are desirable since they can be used to describe the video using language as well as to recognize events. We also study zero-shot learning, a problem that has received increased attention recently in the machine learning and computer vision communities. This task, where training examples of the class of interest are not needed, is appealing due to the large number of objects, actions, events, and other visual categories in the natural world and due to their long-tail nature. It is well known, for example, that only a relatively few object categories, such as people and vehicles, have large numbers of example images that can be used to train detectors, while most other object categories have too few examples to sufficiently model their appearance by current approaches. Even when enough training examples are obtained (at great cost) and annotated (at even greater cost), training detectors involves significant computational resources, making zero-shot learning even more appealing. Chapter 4 describes the mid-level video representation and chapter 5 presents how to apply the mid-level representation to the zero-shot learning task.

Chapter 2: Qualitative Pose Estimation by Discriminative Deformable Part Models

2.1 Introduction

The analysis of human actions in videos is an important task for many computer vision applications. In general, actions can be inferred from the kinematic movements of a person's limbs. However, people are highly articulated, limbs occlude each other, loose clothing conceals shape, and low resolution or motion blur lose informative features such as edges. All of these conditions confound pose estimation, making it a difficult and widely studied problem. To ameliorate these problems, researchers often introduce additional information, *e.g.*, multi-view images or depth information, if available, to reconstruct pose in 3D coordinates [11–18]. Instead of relying on additional information, our approach is based on the observation that for many action recognition tasks, it may be sufficient to infer only a qualitative description of a pose, *e.g.*, 'bent', 'laying down', 'stretching', 'crouched', 'facing left', 'facing right', even if the exact joint locations are not identified. We expect models that infer a qualitative description of a person's pose to be more robust in the presence of the problems that confound exact joint localization. We propose a model for inferring qualitative pose labels automatically from single monocular images.

We model each qualitative pose by a root filter and set of deformable parts, similar to [2]. Unlike the problem of human detection, for which part deformation is modeled to introduce invariance to slight pose changes, the problem of qualitative pose estimation benefits from models that exploit part appearance and deformation to discriminate between pose changes. While we do not require exact recovery of joint locations, it is important for part models to provide information that can be used to discriminate between qualitative poses. For this reason, when training part models, we ensure that models are tightly clustered in pose space (similar to Poselets [20]), and train multiple models covering the same physical parts in various configurations and viewing angles (see Figure 2.1), allowing the relevance of each part model to be automatically adjusted for each qualitative pose to improve discrimination. Given trained root and part filters, we optimize appearance and deformation weights for each qualitative pose and train a multi-class model that fuses the outputs of each qualitative pose model.

Our main contribution is a qualitative pose detection approach based on state-of-the-art deformable part models, with parts that are automatically initialized and trained on semantically meaningful pose clusters that are more discriminative than those initialized by random [20] or greedy [2] selection (in the latter, parts are selected to maximize the energy of the corresponding root filter subregion). A nice property of our approach is that the same parts are shared by all qualitative pose models (but are incorporated into each model using different weights), which

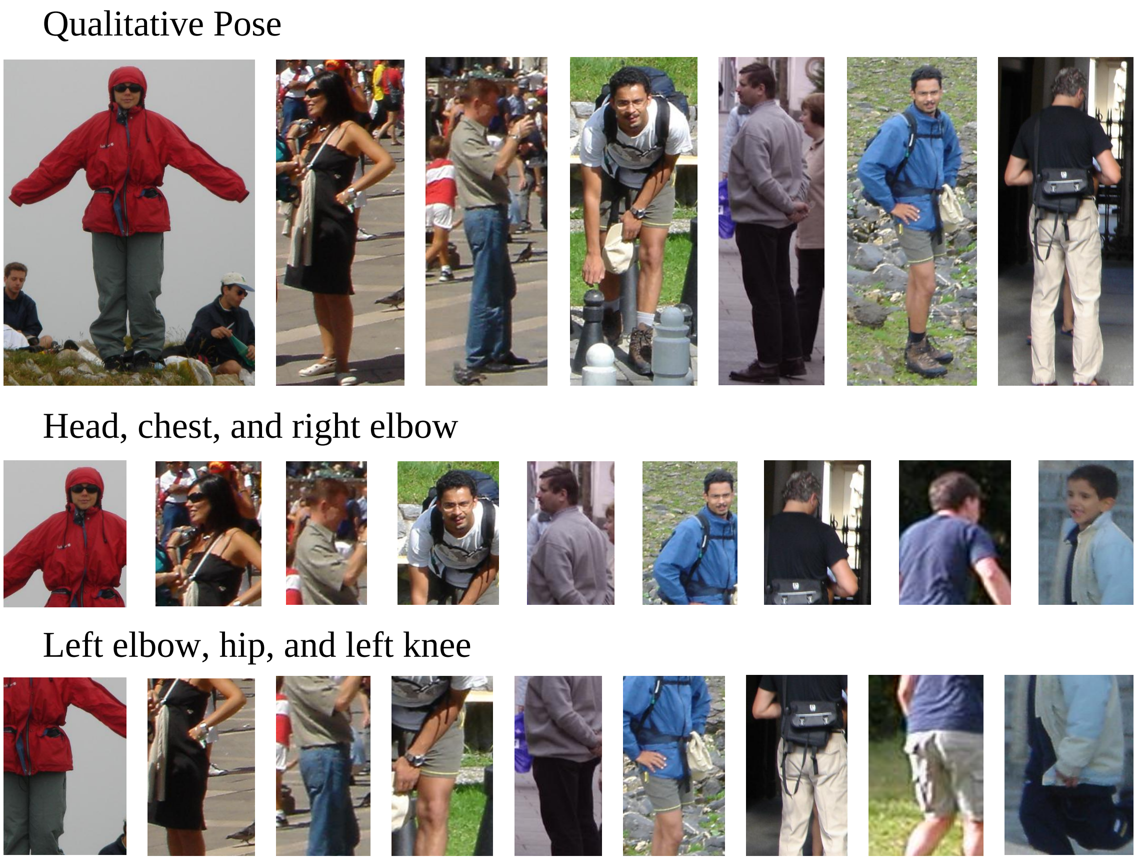


Figure 2.1: Part appearance provides information that can be used to discriminate between qualitative poses. In the first row, the figure shows various qualitative poses. The second and third rows show patches cropped to contain two sets of target joints, the combination of head-chest-right elbow (row 2) and left elbow-hip-left knee (row 3). We note that appearance of patches covering the same joints varies according to qualitative poses. Our approach takes advantage of this relationship between part patch appearance and qualitative pose.

requires that the computationally expensive sliding window search to be performed only once for each part. We demonstrate the performance of our approach on a public database and compare against two baseline approaches from [2] and [3].

In section 2.2, we discuss related work. In section 2.3, we detail our proposed model. In section 2.4, we present the experimental results that demonstrate the performance of our approach. We present our concluding remarks in section 2.5.

2.2 Related Work

The literature on pose estimation is vast and includes methods that extract 2D pose using part models [21–29] and that estimate 3D pose from single or multiple views [11–18]. We focus our discussion on 2D pictorial structure methods as they are most related to our work. Pictorial structure models were introduced by Fischler and Elschlager [21] to represent objects as a collection of parts connected by spring-like connections. Parts encode local appearance and their locations can vary subject to a specified deformation cost. While a straightforward search for the optimal locations of parts is computationally expensive, the search becomes practical under certain conditions. For example, Felzenszwalb and Huttenlocher [2] showed that if the pictorial structure is acyclic, and the relationships between pairs is expressed in a restricted form, the generalized distance transform can be used to compute the globally optimal configuration at a cost that is linear in the number of part locations. In subsequent work, Felzenszwalb et al. [2] proposed a more general deformable part model, consisting of roots, parts, and deformation costs which are all discriminitively trained using Latent SVM. Part locations are automatically initialized from an initial estimate of the root filter by a greedy cover of high energy areas of the root filter, and their deformations are optimized efficiently using the generalized distance transform.

Bourdev et al. introduce a novel concept of parts, Poselets, which are tightly clustered in both appearance and configuration space [20, 30]. As proposed, these parts do not necessarily coincide with body segments (*e.g.* upper arm, lower arm, torso), but generally capture combinations of portions of parts which are distinctive in certain views (*e.g.*, frontal face and right-shoulder combination). During training, candidate Poselets are obtained by repeatedly selecting a random patch in a training image, finding patches in other images which are near in configuration space, and training a Poselet detector using these patches. At test time, Poselet detections (or *activations*) are obtained by multi-scale sliding window search, and objects are detected either by Max Margin Hough Voting [30] or by clustering mutually consistent activations [20]. An attractive feature of Poselets is that they can easily propagate additional labels that were provided with the initial training set, *e.g.*, segmentation masks and joint locations. Consequently, Poselets have been applied to various problems, including the estimation of segmentations [31], actions [3,32], subordinate categorization [33], and attribute classification [34]. Poselets have also been incorporated into pictorial structure models for 2D pose estimation [35], with Poselets organized into a hierarchy of various sizes, covering individual parts, combinations of parts, and even the entire body.

Several exististing approaches predict discretized viewing directions (which fits our definition of qualitative pose) as part of their frameworks. Andriluka et al. [14] train eight independent view-point specific pictorial structure based detectors, whose outputs are combined using a linear SVM. Each model is trained independently and uses standard body parts (head, torso, upper/lower legs, upper/lower arms, feet). Maji et al. [3] predict discretized viewing directions as part of a static action recognition framework using the Poselet framework. To achieve this, they train 1200 Poselets, which are applied at test time.

Our approach builds on deformable part models [2] and Poselets [20], but they are optimized for the purpose of qualitative pose estimation instead of object detection. Instead of selecting Poselets by random selection [20] or greedy cover [2], our approach automatically selects clusters from sets of joints whose variations are predictive of qualitative pose. We train multiple models, one for each qualitative pose, allowing each model to select part deformation weights that best allow for discrimination between qualitative poses. Our approach is trained to maximize discrimination at all levels (parts, deformation weights, combination of output), unlike [14], and requires few part models (in our experiments we used only 64 parts, while [3] employed 1200).

2.3 Qualitative Pose Estimation

The block diagram of our approach is shown in Figure 2.2. To reduce overfitting, we divide the training dataset into two sets; one is for training root and part filters and the other is for training Q(Qualitative)-Pose models by Latent SVM [2] and calibrating them to each other. Root regions, which are defined as fixed aspect ratio bounding boxes whose vertical extents are defined by the head and the waist of a person, are first cropped from training images and are divided into sets according to their labeled qualitative pose. Root filters are learned via SVM with

HOG features constructed from the collected images in each set. Part filters cover a combination of joints, which are selected manually based on how predictive their appearance variations are of qualitative pose. In our experiments, parts are defined by three joints: head - upper torso - right elbow, head - upper torso - left elbow, right elbow - lower torso - right knee, and left elbow - lower torso - left knee. For each part, training images are divided into clusters by k-means clustering according to the similarity of joint configurations. Next, part filters are learned as for the root filter. The root and part filters are then applied to the second (held out) set of training images, and the set of activations are used to train the weights of a Latent SVM model that detects qualitative poses based on root and part filter activations. During testing, we first extract activations of trained root and part filters by sliding window search. Then, for each qualitative pose model, we select an activation for each part to maximize the joint model score, and then apply the linear model learned by multi-class SVM to predict the best matching qualitative pose.

We describe the training process in more detail in the next subsections. In section 2.3.1, we provide the model formulation. We then describe root and part filter training and model parameter optimization in section 2.3.2 and 2.3.3, respectively.

2.3.1 Model Formulation

Let $q_i,\ i = 1, 2, \cdots, Q$ denote the set of qualitative poses. A model for each qualitative pose, $M_i = \{r_i,\ P,\ A_i,\ w^i_{0:K},\ \vec{w}^i_{d;1:K}\}$ is trained, where r_i is the root filter for qualitative pose i, $P = \{p_1,\ p_2,\ \cdots,\ p_K\}$ is a set of part filters, $A_i =$

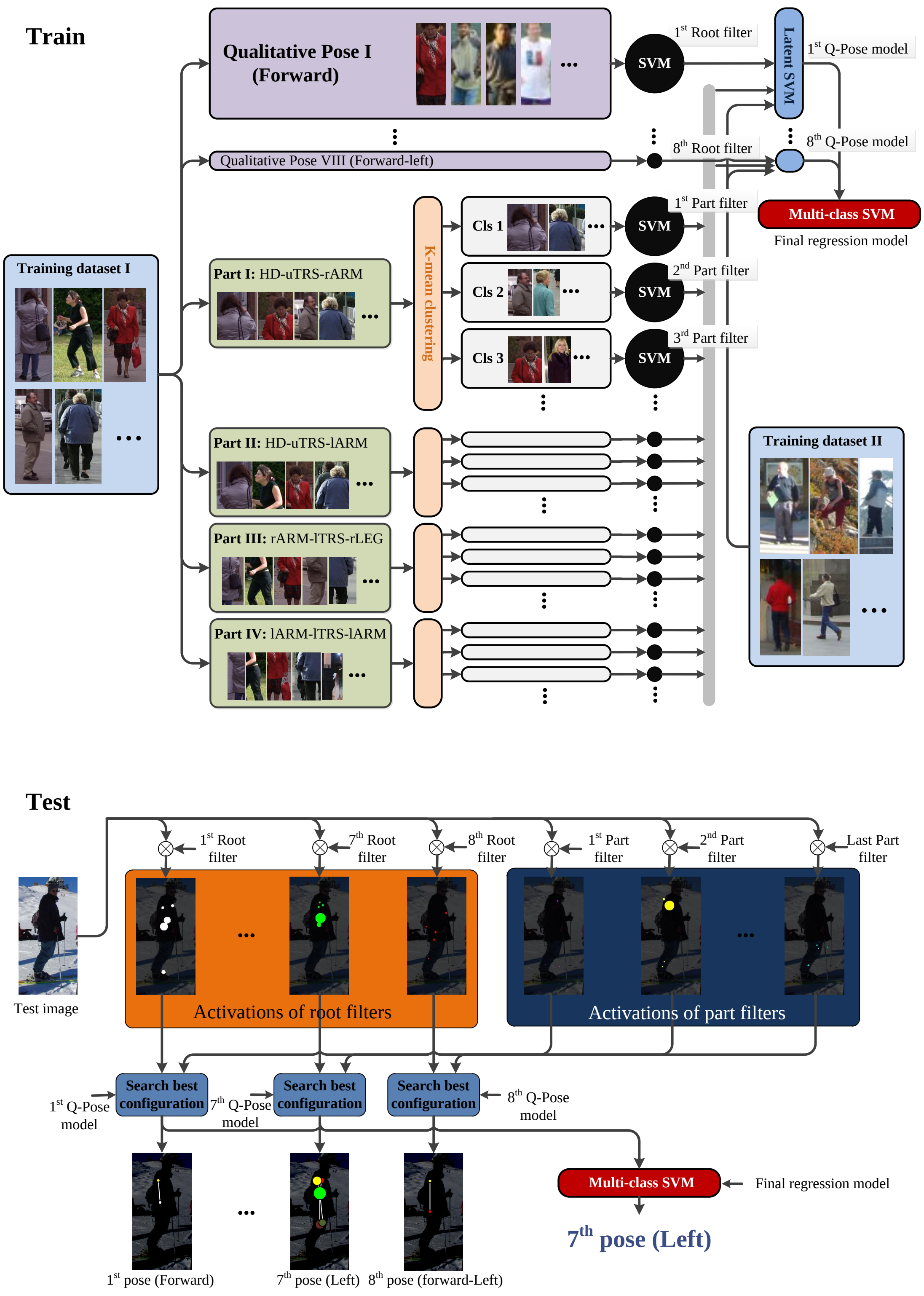


Figure 2.2: Overview of training and test procedure.

$\{a_1^i,\ a_2^i,\ \cdots,\ a_K^i\}$ is a set of anchor positions which specify the relative position of k^{th} part to the root, and $w_{0:K}^i$ and $\vec{w}_{d;1:K}^i$ are model parameters that weigh appearance and deformation costs, respectively ($\vec{w}_d$ is a vector that defines the deformation cost as in [2]). Every model uses the same set of part filters, P. Q and K denote the size of the set of qualitative poses and parts, respectively. At test time, filter activations generate a set of candidate locations for each part. A hypothesized qualitative pose is formed by selecting one of the candidate part locations for each of the root and each of the K parts, $L_i = \{l_0^i,\ l_1^i,\ \ldots,\ l_K^i\}$. We model the probability $p\left(L_i|I, M_i\right)$ that the configuration is L_i given the image I and the model M_i and decompose it as follows:

$$p(L_i|I, M_i) \propto p(I|L_i, M_i)p(L_i|M_i). \tag{2.1}$$

The distribution $p(I|L_i, M_i)$ measures the likelihood of fitting the model to a particular image given a part configuration, and $p(L_i|M_i)$ is the prior distribution that each part is placed at a particular location. The best configuration L_i can be obtained by MAP estimation as

$$L_i^* = \arg \max_{L_i \in L_a(I)} \ p(L_i|I, M_i). \tag{2.2}$$

The likelihood of configuration L_i is modeled as the product of the i^{th} root likelihood and the individual part likelihoods,

$$p(I|L_i, M_i) = p(I|r_i, l_0^i, w_0^i) \prod_{k=1}^{K} p(I|p_k, l_k^i, w_k^i). \tag{2.3}$$

where $p(I|p_k, l_k^i, w_k^i) = exp(-w_k^i m_k(I, l_k^i))$, and $m_k(I, l)$ measures the response of the k^{th} filter at position l in image I.

The prior distribution of the configurations can be expressed as a product of a root location prior and part location priors given the root location,

$$p(L_i|M_i) = p(l_0^i|M_i) \prod_{k=1}^{K} p(l_k^i|l_0^i, a_k^i, \vec{w}_{d;k}^i). \tag{2.4}$$

The prior distribution of root location, $p(l_0^i|M_i)$ is modeled as a uniform distribution, and $p(l_k^i|l_0^i, a_k^i, \vec{w}_{d;k}^i) = exp(-\vec{w}_{d;k}^{iT} f_d(l_k^i, l_0^i + a_k^i))$ is the probability that the k^{th} part is placed at l_k^i given the root location. The deformation function $f_d(l_i, l_j) = \begin{bmatrix}-dl_x & -dl_y & -dl_x^2 & -dl_y^2\end{bmatrix}^T$, where $dl = l_i - l_j$, is defined as in [2].

The score of a hypothesis is defined as the negative logarithm of equation 2.1,

$$score(I, L_i) = w_0^i m_0^i(I, l_0^i) + \sum_{k=1}^{K} w_k^i m_k(I, l_k^i) + \sum_{k=1}^{K} \vec{w}_{d;k}^{iT} f_d(l_k^i, l_0^i + a_k^i). \tag{2.5}$$

which can be more compactly represented as the dot product, $W_i^T \Phi(I, L_i)$, of model parameters W_i and a vector $\Phi(I, L_i)$ specifying a matching score and deformation cost of each part in its own location,

$$W_i = \left[w_0^i; \ldots; w_K^i; \vec{w}_{d;1}^i; \ldots; \vec{w}_{d;K}^i\ \right],$$

$$\Phi(I, L_i) = [m_0^i(I, l_0^i); \ldots; m_K(I, l_K^i); f_d(l_1^i, l_0^i + a_1^i); \ldots; f_d(l_K^i, l_0^i + a_K^i)\ ]. \tag{2.6}$$

2.3.2 Training Root and Part Filters

We define a root that represents the general position of the entire human and provides an anchor position for each part (this anchor position will vary with qualitative pose). The root is defined by the head and the waist (the width of the box is a fixed ratio of the height, which is defined as the distance between the head and the waist), and its appearance and position does not greatly change with various human poses or actions. For each qualitative pose, a root filter is trained to model the general location of parts. The part anchor positions are computed by averaging relative positions of each part to the root in all training images labeled as the specified qualitative pose. To train a root model of each qualitative pose, we collect examples cropped around the root region from images labeled as a particular qualitative pose. We crop and resize each example to a fixed height and aspect ratio. The height is set to the median value of every cropped root region and the width is calculated by dividing the height by the fixed aspect ratio. Given these examples, we extract HOG features from the collected positive examples and randomly select ten times as many negatives and train linear SVM classifiers to discriminate between positive and negative examples. As for Poselet training, we scan over background images that contain no people, collect false positive examples, and retrain linear SVM classifiers, repeating this process a few times to train the classifier efficiently with a large number of negative examples. We note that each hypothesis has one root filter.

To ensure that part filters can be used to discriminate between qualitative poses, we select parts by clustering combinations of joints that are expected to vary in predictable ways with respect to qualitative pose. Figure 2.3 shows how parts composed of certain joint triples exhibit a large spatial and appearance variation with respect to qualitative pose. We define our parts by clustering the configurations of combinations of three joints, which in our case define pairs of limbs: head - upper torso - right elbow, head - upper torso - left elbow, right elbow - lower torso - right knee, and left elbow - lower torso - left knee.

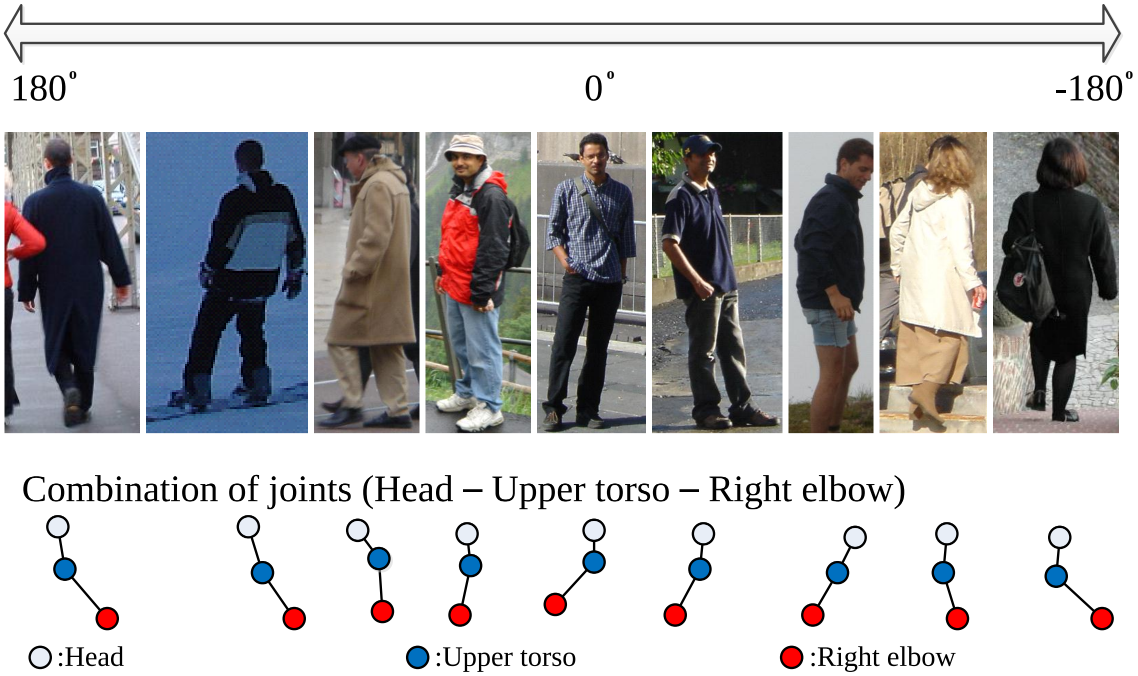


Figure 2.3: The overall appearance and arrangement of the head, upper torso, and right elbow joints varies significantly with qualitative pose, as shown by the sample images and illustration of the three corresponding nodes. For this reason, the three joints together can be considered a discriminative part.

During part filter training, images are first resized to have root regions of the same size. For each joint triplet, training examples are then selected by cropping a region containing the three selected joints. For each combination of joints, training samples are divided into n classes by k-means according to similarity of joint configuration. To cluster part training examples, the joint configuration of each part is represented as a vector of concatenated joint positions relative to the torso, and the similarity of joint configurations between two examples is computed by Euclidean distance between the configuration vectors. Each training sample is resized to the median size of the training example. As a result of this process, each joint triplet generates n parts which correspond to clusters in joint configuration space.

Note that a part, if detected, does not directly imply a certain qualitative pose, but we expect that some parts are more predictive of certain qualitative poses than others (our experimental results confirm this). Given the selected training samples for each part, we train part filters in a similar way to the root filter. The only difference is that for training the parts we use negative examples extracted from images in other clusters from the training set while for training the root we extract them from background images. By including samples from other part clusters as negative part samples, we train part filters that better discriminate between joint configurations. After root and part filter training, we obtain Q root filters and $4n$ part filters.

2.3.3 Learning Model Parameters

Each image in the training dataset is labeled with its qualitative pose. We indicate the training dataset as $\{I_n, b_n\}_{n=1}^{N}$, where I_n is an image and b_n is its label. Given an image and its label, the trained root and part filters can be applied to detect candidate locations of parts. For every qualitative pose, we learn a deformable part model over the root and part filters using the latent SVM formulation [2]. For each qualitative pose, a classifier that scores an image I is defined as

$$f_W(I) = \max_{L \in Z(I)} W^T \Phi(I, L), \tag{2.7}$$

where $Z(I)$ is the set of all possible combinations of activations (here, we only consider the locations corresponding to root and part filter activations). Model parameters can be learned by minimizing the objective function

$$L_D(W) = \tfrac{1}{2}||W||^2 + C\textstyle\sum_{i=1}^{M} \max(0, 1 - y_i f_W(I_i)), \tag{2.8}$$

$$\text{where} y_i = \begin{cases} 1 & \text{if } I_i \text{ is a positive example} \\ -1 & \text{otherwise.} \end{cases}$$

The standard hinge loss, $\max(0, 1 - y_i f_W(I_i))$ is concave when an image I_i is labeled as positive because the classifier, $f_W(I)$ is convex. Latent SVM optimization specifies the latent value L^* for every positive image and yields a linear form,

$$f_W(I) = W_t^T \Phi(I,\ L^*), \tag{2.9}$$

$$\text{where } L^* = \arg\max_{L \in Z(I)}\ W_{t-1}^T \Phi(I, L).$$

While searching for the best configuration L^*, the algorithm uses the parameter W_{t-1} learned in the previous step. In other words, the semi-convex optimization is solved by repeatedly optimizing two separate convex functions, a process called "co-ordinate descent". The first part of the optimization involves computing the overall score of each configuration of root and parts and selecting the highest scoring configuration. In our case, $Z(I)$ is a small enough set for all candidate configurations to be considered in a reasonable amount of time. In the second part of the optimization, we compute the model parameter W_t using a linear SVM.

We consider all configurations in images labeled as other qualitative poses as negative examples. To avoid considering unlikely configurations as negative examples, we collect only the best configuration for each root activation. Because negative examples are very numerous compared with positive examples, we extract a set of hard negative examples in every iteration of optimization, and ignore the remaining negatives during that iteration.

2.4 Experiments

We evaluate our framework on the public INRIA pedestrian database [9], which consists of images that contain upright pedestrians with annotated bounding boxes. Our aim in these experiments is to recognize qualitative poses by analyzing the entire body, so we did not consider datasets such as PASCAL VOC database which contain many images in which people are often only partially visible. While the INRIA pedestrian database might be considered easy for the task of human detection, it is a difficult dataset for the task of determining qualitative pose (as our experiments will show). To evaluate our approach, we assume that the person has been roughly localized, using a detector such as that of Felzenszwalb et al. [2], so we focus only on assigning a qualitative pose label to regions extracted around the annotated bounding boxes. To increase the effective size of our training set, we also flip images along the vertical axis. Since bounding boxes may exclude part of a person region due to annotation errors, we cropped images only after adding a suitably large amount of padding to the human bounding boxes. We split the database randomly into three sets using a ratio of 2:2:1; the first split is for training part and root filters, the second is for validation, and the last is for testing. We discretize the qualitative pose into 8 discrete bins of angles corresponding to the direction that a

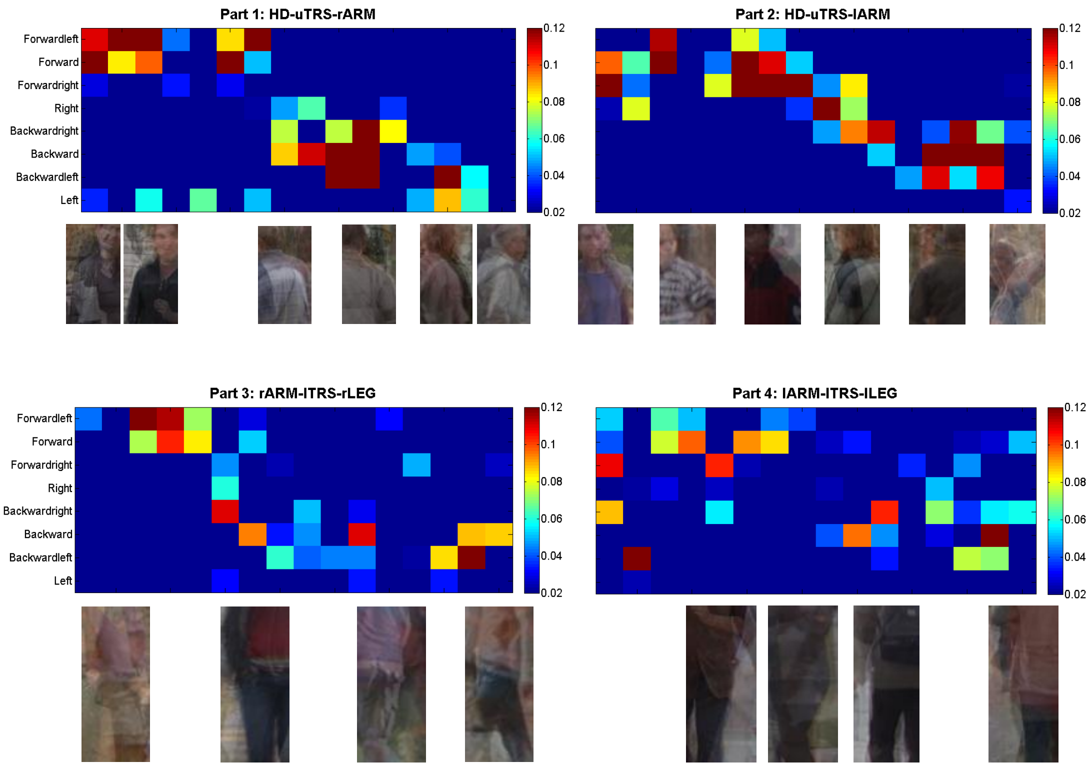


Figure 2.4: The weights obtained by the optimization in equation 2.9 for each of the 8 qualitative pose models (y-axis) and each of the 16 part models (x-axis).

person's torso is facing with respect to the camera, and so construct 8 qualitative pose models. Each bin covers 45 degrees.

To train root and part filters, we labeled the head, neck, waist, elbows, and knees, and specified the qualitative pose of each image. While training part filters, we set n, the number of clusters obtained from applying k-means of training samples to 16. We use 200 positive examples and 2000 negative examples for training each filter. We extract false positives and retrain for ten iterations.

Figure 2.4 shows the appearance weights obtained by the optimization in equation 2.9 for each of the 8 qualitative pose models. The qualitative poses are along the y-axis, ordered in circular fashion from forward-left to left. The part filters trained on the clustered joint triples are listed on the x-axis. These are also roughly ordered by the distribution of the qualitative pose labels of the training images belonging to each cluster, so that parts are also ordered circularly from forward-left to left. As expected, the strong diagonal weights in many of these images show that the parts obtained by our approach are indeed predictive of qualitative pose. Conversely, there still remains enough confusion that it is necessary to combine evidence from the multiple parts. We conclude that while upper parts (part 1 and 2) are more associated with qualitative poses, lower parts (part 3 and 4) include variations that are caused by other sources in addition to qualitative pose.

To evaluate our performance, we implement two state-of-the-art approaches for our qualitative pose estimation problem. As for our approach, pose-specific models are first trained independently on a training subset using the deformable part-based model (DPM) [2] and Poselets [3,20] using the same training, validation, and test partitions. Independent model scores are calibrated against each other on a validation set by a multiclass SVM. We applied the DPM training/testing code as provided by the original authors, with a modified input training set (the 8 qualitative poses), a single component instead of mirrored left-right models (we care about facing direction), and a subsequent multiclass calibration step. We also compare to the Poselet code of Bourdev et al. [20], but since the training code is not provided by the authors, we implement the training procedure described in [20]. We use a pose activation vector that collects detection scores of 1200 Poselets as the pose representation, as in [3]. Figure 2.5 shows the performance of each independent

Table 2.1: AUCs of each approaches. F, B, L, and R abbreviate ‘forward’, ‘backward’, ‘left’, and ‘right’, respectively. (The best result in each pose is in bold font.)

	BL	L	FL	F	FR	R	BR	B
Felzenswalb et al. [2]	0.625	0.668	0.622	0.787	0.566	0.653	0.666	0.741
Maji et al. [3]	0.591	0.719	0.658	0.779	0.715	0.615	0.615	0.742
Our approach	**0.761**	**0.854**	**0.799**	**0.896**	**0.779**	**0.809**	**0.827**	**0.897**

qualitative pose model and compares our approach with the other two alternatives on INRIA pedestrian database. Based on the ROC curves, our approach outperforms the other methods for every qualitative pose. Table 2.1, which shows the area under the ROC curves (AUC), also shows that our approach outperforms the alternatives.

Figure 2.6 shows the confusion matrix of the three approaches, obtained after the independent model outputs are combined using a multi-class SVM. The confusion matrix for our approach has a much more pronounced diagonal than the other two alternatives, which is expected, given the individual qualitative pose detection performance. As one would expect, there is a lot of confusion between neighboring poses. Commonly, ‘forward’ and ‘backward’ are well detected but subtle differences between right- or left-facing poses are often misclassified. This has also been observed by other researchers [3], who have noted that human perceptual ability also distinguishes between cardinal directions (front, back, left, right) direction better than others such as front-right, backward-left, etc. While [2] and [3] achieve different

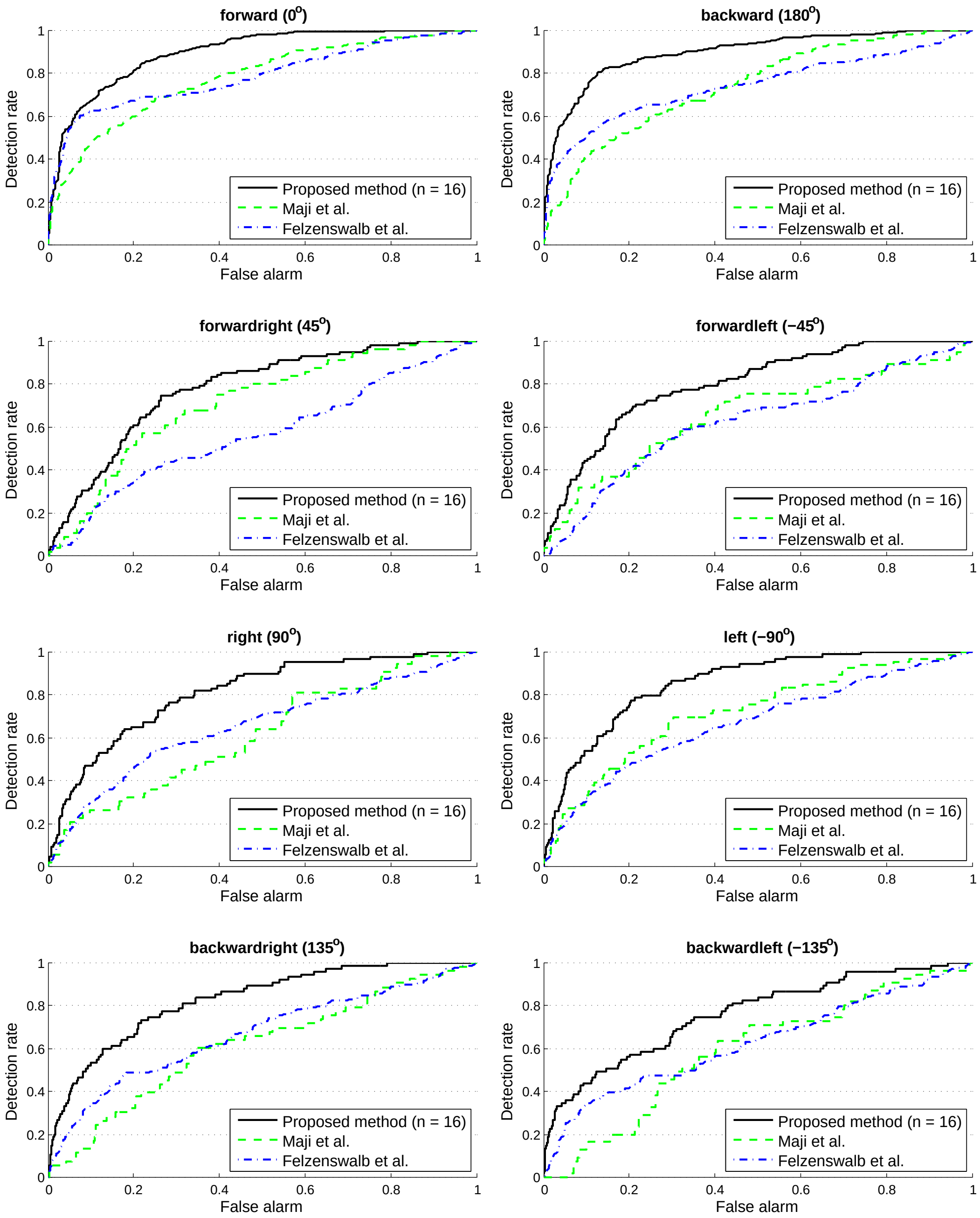


Figure 2.5: ROC curves for performance of each qualitative pose model on the INRIA person database.

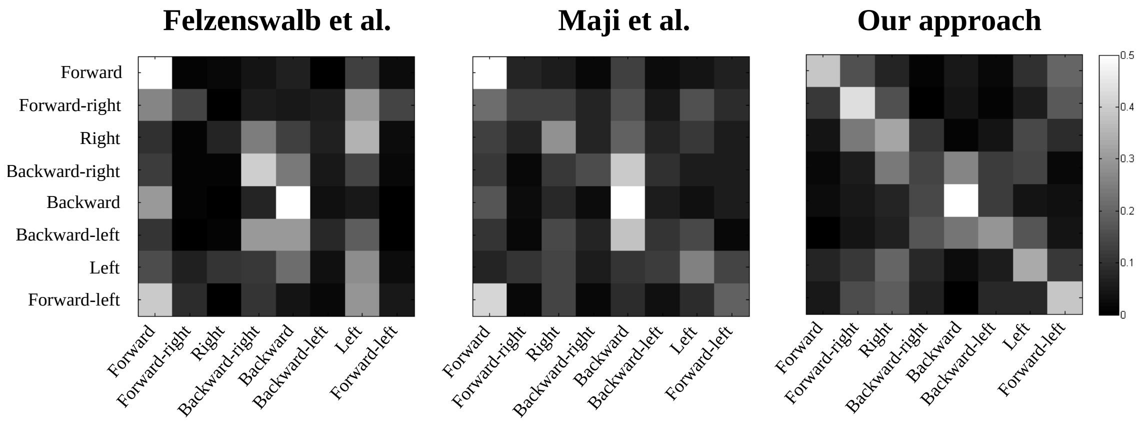


Figure 2.6: Confusion matrix of three approaches. **Left:** Felzenswalb et al. [2], **Center:** Maji et al. [3] and **Right:** our approach.

performance between qualitative poses, our approach maintains a consistent level of detection for every class. Table 2.2 shows the overall recognition rate. Errors are computed by a mean squared error from misclassified class to groundtruth, where the distance between front and front-right is 1, front and right is 2, and so on. Our approach outperforms the others using these measures, as well.

2.5 Conclusions

We presented a qualitative pose estimation approach that is based on discriminative deformable part models. Unlike previous approaches, we give special attention to the selection of part models, replacing random selection and greedy cover steps with an automatic clustering of part poses. The part appearance and deformation parameters are trained discriminatively for each qualitative pose model, and the outputs of all pose models are combined using a multi-class classifier. Our

Table 2.2: Overall recognition results of three approaches on the INRIA pedestrian database. (Bold font indicates the best result.)

	Recog. rate	Errors
Felzenswalb et al. [2]	0.2909	1.9868
Maji et al. [3]	0.2814	1.9431
Our approach	**0.3485**	**1.6810**

approach shows improved performance on the INRIA pedestrian database against two state-of-the-art approaches.

Chapter 3: Robust Pose Features for Action Recognition

3.1 Introduction

Action recognition still remains challenging due to great intra and inter variance of classes, cluttered and occluded background, etc., despite numerous recent advances. Many researchers extract local image and video features from video sequences, separate them into clusters, and generate histogram-based representations. Interest points are often extracted by methods such as Harris3D [7], Hessian [8], etc, to capture shape and motion of local points. HOG [9], silhouettes [10], and SIFT [4] are commonly used as shape features. As a motion feature, most researchers use optical flow [10] or other custom representations of space-time volumes, e.g., Liu et al. [4] use flat gradients within 3D cuboids.

While pose-based action recognition methods have also been studied [2], they have generally underperformed methods based on shape and motion features on difficult "in-the-wild" videos such as those obtained from YouTube. This is because pose estimation remains a difficult problem in uncontrolled settings and even state-of-the-art pose estimation approaches are relatively brittle.

In this work, we use a pose feature based on poselets, which captures human pose without the need for exact localization of joint locations, but instead relies on

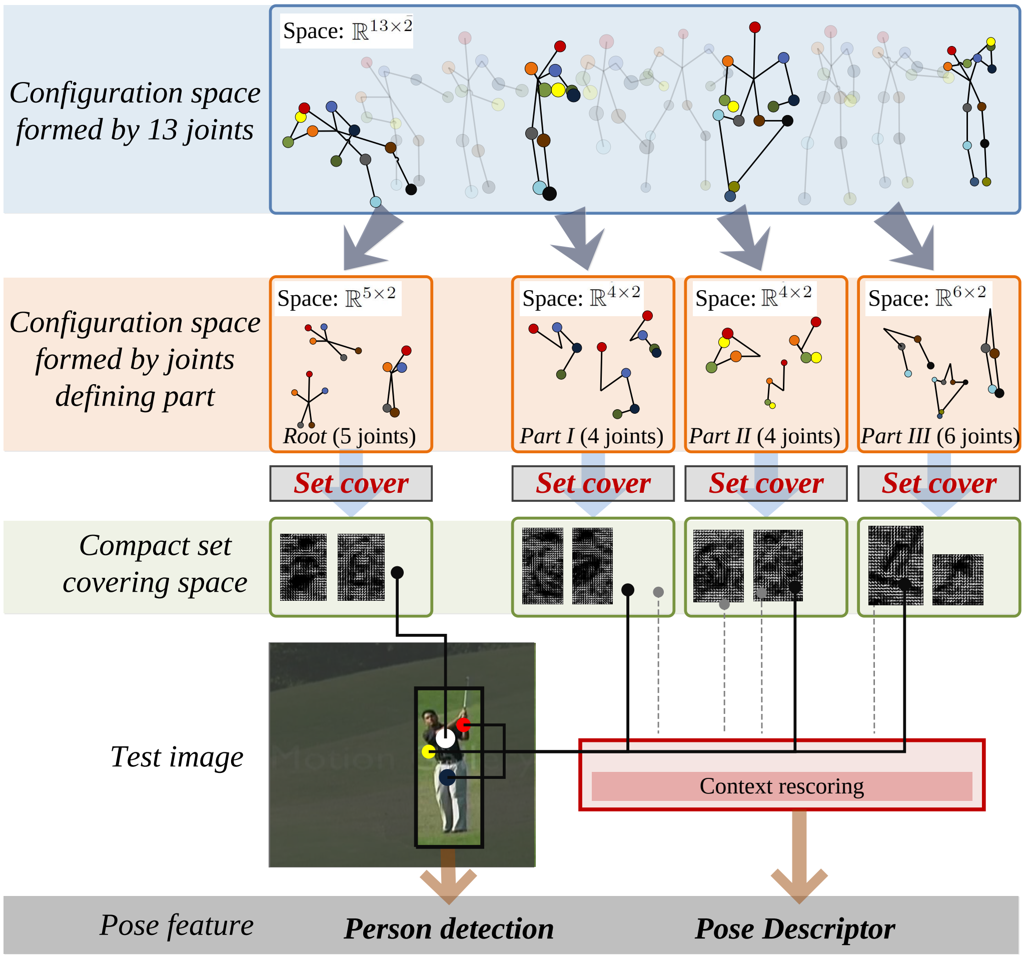

Configuration space formed by 13 joints
Space: $\mathbb{R}^{13\times2}$
Configuration space formed by joints defining part
Space: $\mathbb{R}^{5\times2}$
Root (5 joints)
Space: $\mathbb{R}^{4\times2}$
Part I (4 joints)
Space: $\mathbb{R}^{4\times2}$
Part II (4 joints)
Space: $\mathbb{R}^{6\times2}$
Part III (6 joints)
Set cover
Set cover
Set cover
Set cover
Compact set covering space
Test image
Context rescoring
Pose feature
Person detection
Pose Descriptor

Figure 3.1: Illustration of our proposed posed descriptor and its use for action recognition. The 13 joint pose configuration space is split into subsets of joints, whose smaller space of configurations we cover greedily with poselet models. This ensures that common *and* rare configurations are represented (*covered*). This improves action recognition which models transitions through pose configurations. Given an image, poselet activations are obtained, as usual, grouped by mutual consistency, and assembled into an activation vector, which is rescored to incorporate the context provided by mutually consistent activations.

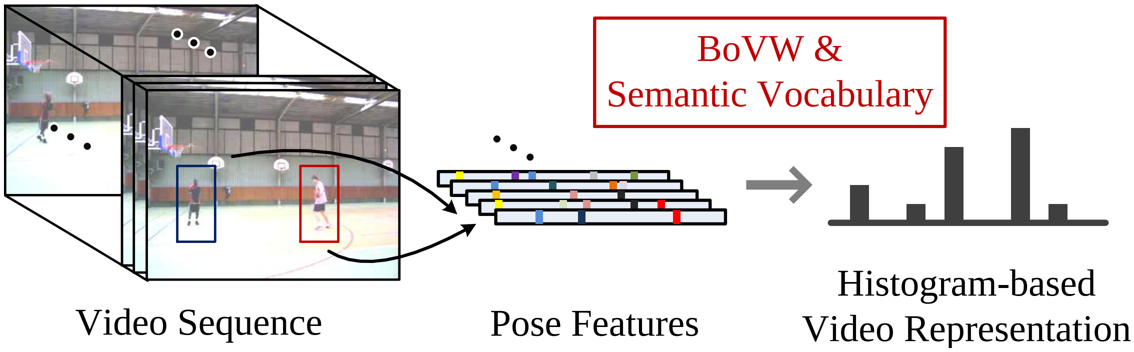


Figure 3.2: Illustration of our proposed posed descriptor and its use for action recognition. We depict the use of the proposed descriptor in a histogram based video representation.

the representation and detection of coarse qualitative poses (e.g., standing, bending) which are learned automatically from training data. Poselets [20, 30] are discriminative part models constructed to be tightly clustered in the configuration space of joints as well as in the appearance space of images, and which have been successfully used for detecting people [20, 30], describing human attributes [34], and recognizing human actions [3, 32, 36] in single images. As more poselets are used by an object detector, the detector's accuracy increases, but its efficiency decreases proportionally with the number of poselets.

While a small number of poselets might be sufficient for detection, for action recognition it becomes important to cover the space of pose variations more completely, since actions are generally modeled as transitions through pose space. However, the standard poselets training procedure requires too many poselets to adequately represent the pose space for action recognition. This leads to a loss in efficiency, increases the feature descriptor size, and ultimately leads to poor action recognition performance (as shown in our experiments). This motivates us to modify the poselet training procedure with the following goals in mind: (1) increase the coverage of the space of poses, and (2) maintain efficiency by making the set of poselets more compact. To accomplish this we partition the 13 joints into overlapping subsets (depicted in Figure 3.1), and instead of randomly selecting image rectangles to define poselets as in [30], we select seed rectangles using greedy set-cover to ensure that most joint configurations in each subset are adequately detected by a poselet. Our proposed greedy set cover algorithm ensures that each part–defined as a subset of joints–should generate poselets that cover the entire range of its configurations while avoiding redundant poselets (each poselet should detect at least one new configuration that is not detected by another poselet).

Given a test video, we obtain a pose descriptor from our compact set of poselets by constructing activation vectors from mutually consistent activations as in [30], and rescore activations using the context encoded by this vector. We construct activation vectors for each root activation and create a codebook based histogram representation using all root activations that have a high enough confidence after context rescoring. We incorporate the proposed pose features in existing action recognition [4] with traditional motion and shape features.

Figure 3.2 depicts our approach. To summarize, our contributions are the following: 1) we improve the compactness and discriminability of the original poselets by a training process that applies greedy set cover to the smaller configuration spaces of joint subsets, and 2) we are the first to our knowledge to successfully use pose as a feature for ”in-the-wild” video-based action recognition.

We evaluate our approach on two benchmarks: YouTube sports dataset [1] and YouTube action dataset [4]. Our experiments show that the proposed pose feature provides significant complementary information to the motion and shape features. In fact, the pose feature alone nearly matches state-of-the-art results, while the combination with either shape or motion alone improves over the state-of-the-art, and the combination of all three types of feature outperforms all other alternatives. In fact, on the YouTube Action dataset, our proposed approach outperforms the state-of-the-art by over 10%. Our experiments demonstrate the importance of our modified training procedure to effectively incorporate poselet features into a video-based action recognition framework.

In section 3.2, we discuss related work. In section 3.3 and 3.4, we describe details of semantic pose features and incorporating features into an action recognition framework, respectively. In section 3.5, we present the experimental results that demonstrate the performance of our approach. We present our concluding remarks in section 3.6.

3.2 Related Work

Since the literature on action recognition is vast, we describe only recent works in this section. Liu et al. [4] extract motion and shape features from videos, construct a compact yet discriminative visual vocabulary using an information-theoretic algorithm, and generate a histogram-based video representation. While this approach is effective, it does not make use of pose features. We extend this approach by incorporating our proposed pose feature to their features and followed the framework for action recognition proposed by [4]. Xie et al. [37] explore the use of deformable part models (DPM) for incorporating human detection and pose estimation into action recognition. Similar to our method, their work is also based on human poses but our part models are trained to discriminate between various poses of a person, unlike DPM's, which are trained to discriminate between patches in which a person is present or absent. Le et al. [38] learn features directly from video using independent subspace analysis that is robust to translation and selective to frequency and rotation changes. Todorovic [39] views a human activity as a space-time repetition of activity primitives and models the primitives and their repetition by a generative model-graph. Sadanand and Corso [40] propose action bank, consisting of action detectors sampled according to classes and viewpoints.

Our proposed pose feature is based on the poselets framework introduced by Bourdev and Malik [30]. Poselets are discriminative part detectors constructed from tight clusters in the configuration space of the human articulated body as well as in the appearance space of images. At test time, poselet activations are detected by multi-scale sliding windows, and persons are detected by Max Margin Hough Voting [30] or by clustering mutually consistent activations [20]. Poselets have been employed to improve results in various vision applications, including segmentations [20], subordinate categorization [33], attribute classification [34], pose [3, 41] and action recognition [3, 32, 36]. Unlike all of these extensions of poselets which are applied to static images, our method extends the use of poselets to action recognition on video sequences, producing results that improve on the current state-of-the-art.

3.3 Training Parts and Context Rescoring

3.3.1 Motivation

Poselets are successfully used in detecting humans [30] as well as recognizing actions [3] in still images but have not been used for video-based action recognition. While a small number of poselets might be sufficient for detection, for action recognition it becomes important to cover the space of pose variations more completely, so that we can observe and model transitions through the pose space. However, if the number of poselets is increased, person detection by clustering consistent activations may be impractical since the clustering complexity is quadratic in the number of poselet activations.

We modify the poselet training procedure in three ways to improve its effectiveness and efficiency. First, we manually select three sets of joints predictive of pose and introduce three parts that cover the extents of those joints in each set. We also select a set of joints corresponding to the head and torso that are stable and are suitable for use as a root for our model (similar to the root in DPM models [2], which serves as a coarse description of the person). Second, we modify the procedure for selecting a poselet seed, replacing random selection with greedy set cover to satisfy the following criteria:

1. **effectiveness:** each part should generate poselets that cover the entire range of its potential configurations,

2. **efficiency:** poselets should not be redundant.

Third, instead of clustering pairs of mutually consistent poselets to obtain detections of people, we use all root activations as potential human detections, and rescore them out by training a classifier on the feature vector containing the activation scores of the root candidate and of the parts consistent with that root candidate. This yields a clustering process whose computational requirements increase linearly (instead of quadratically) with the number of part activations, allowing for the use of a larger number of poselets in our framework.

3.3.2 Definition of Parts and Training Poselets

Definition of parts: We follow the definition of the root and the parts in [32] employing a four part star structured model to express human pose for recognizing actions. The root is defined by the head, shoulders, and hips and the three parts are defined by pairs of limbs: (head, right shoulder, right elbow, right hand), (head, left shoulder, left elbow, left hand), and (hips, knees, feet) (Fig. 3.3). Table 3.1 shows the average procrustes distance among pairs of training configurations, as well as the coverage of poselets trained on these joints. The table provides the experimental support for using the combination of the head, shoulders, and hips as a root. Only the activation vector of the root is rescored and used in the descriptor, since its coverage is high while the joints belonging to the root are relatively stable, as shown by the low procrustes distance among the root joints.

Training poselets: The appearance variations of the root and each part are cap-

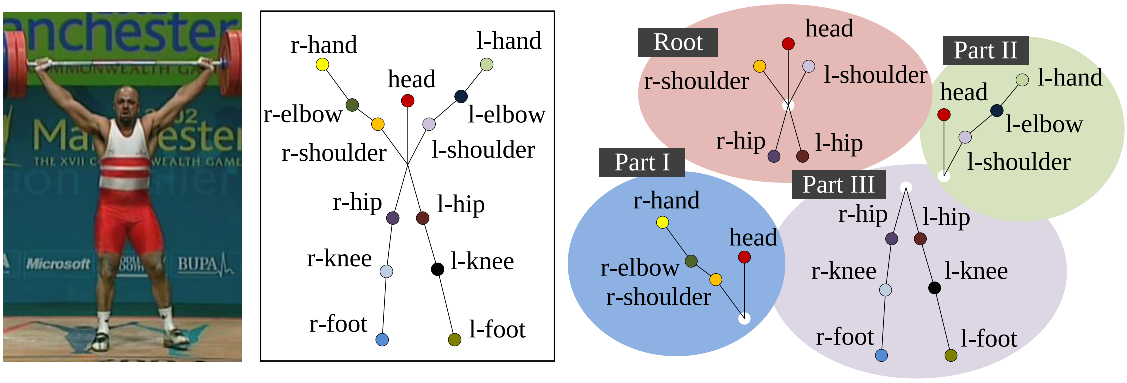


Figure 3.3: Joints annotation (left) and definition of root and parts (right).

Table 3.1: Combinations of joints which appear in more than 50 % of YouTube sports dataset [1] are selected and procrustes distance among configurations of each combination are computed. The joints that define our root (in bold) achieve the best trade-off between joint location stability and dataset coverage.

Combination of joints	Proc. dist	Coverage
l_shoulder-l_elbow-l_hip-l_knee	0.6178	0.5255
l_shoulder-l_elbow-l_hand-l_hip	1.1526	0.5658
head-l_shoulder-l_hip-l_knee	0.4509	0.5461
head-l_shoulder-l_elbow-l_hip	0.5980	0.6266
head-l_shoulder-l_elbow-l_hip	0.2490	0.6637
head-l_shoulder-l_elbow-l_hip-l_knee	0.4789	0.5238
head-l_shoulder-l_elbow-l_knee-l_hip	0.7819	0.5641
head-l_shoulder-r_shoulder-l_hip-r_hip	**0.1390**	**0.6566**

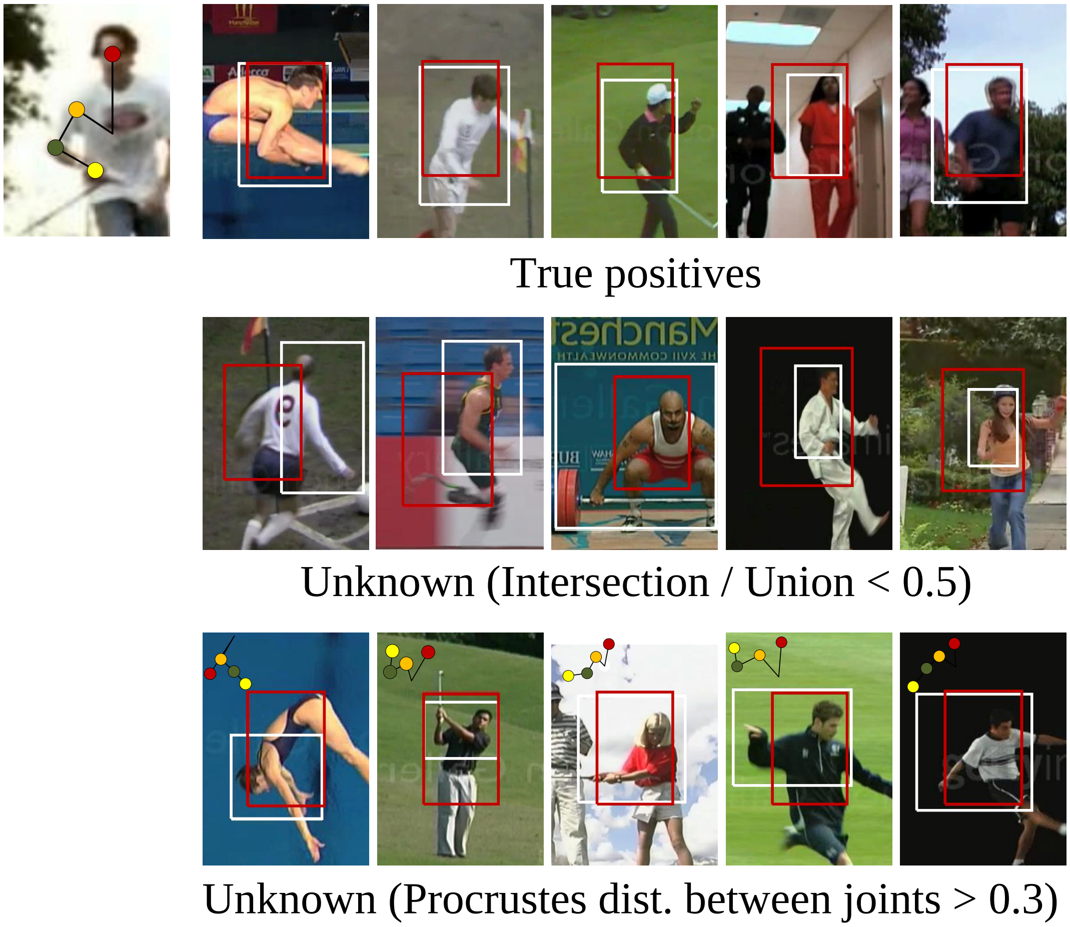
tured by multiple poselets trained by covering the configuration space of each part. Each poselet is trained by the process described in [20]. The patch (seed of a poselet) chosen in the poselet selection step (described in section 3.3.3) collects 250 patches that have similar local joint configuration and uses them as positive examples for training. The patch size is set to one of 96 x 64, 64 x 64, 64 x 96 and 128 x 64 according to the aspect ratio of the area that covers the joints comprising a part. We use the distance metric $D(P1, P2) = D_{proc}(P1, P2) + \lambda D_{vis}(P1, P2)$ proposed by [20], where D_{proc} and D_{vis} are the Procrustes distance between joint configurations of both patches and a visibility distance which is set to the intersection over union of joints present in both patches, respectively. We train a linear SVM classifier with positive examples and negative examples that are randomly selected from images which contain no person. We collect false positives with highest SVM scores as hard negatives (10 times as many as the number of positive examples) and retrain the linear SVM classifier. This process is iterated three times.

After training the poselets, we extract activations by a multi-scale sliding window scheme applied to the training images. Each activation is then labeled as a *true positive*, *false positive*, or *unknown*, using ground-truth annotations of people and their joints. For each training image, we determine matches between detections and ground-truth by comparing the detected bounding box to the ground-truth bounding box that encloses the ground-truth joints, as well as computing the Procrustes distance between the predicted joint locations (using the seed patch joint locations) and the ground-truth joint locations. Note that when computing the Procrustes distance, we exclude rotation because detecting by sliding window does not consider rotation. The latter labeling criterion, not used in [20], discards any false detection whose bounding box matches a ground-truth bounding box but whose associated joint locations are far from the ground-truth joint locations. Each activation which has an intersection over union with ground-truth more than 0.5 and whose Procrustes distance between joints is less than 0.3 is labeled as true positive. If the intersection over union with ground-truth is less than 0.1, the activation is labeled as a false positive for the purpose of the subsequent stages. Others remain unlabeled. Figure 3.4 shows some examples of activations labeled as true positives and unknown. Assuming that the joint distribution is Gaussian as in [20], the mean and variance of each joint are computed over true positive poselet activations, allowing each poselet to have an associated distribution over the position of joints.

3.3.3 Poselet Seed Selection

Our goal is to generate a set of poselets for each part that covers all appearance variations of that part over its configuration space. If we randomly choose poselet seeds and train on the nearest neighbors of those seeds as in [20, 30], we find that many of the training samples are not detected by the trained poselet (or by any other poselet), i.e., many of the training samples are not "covered" by the set of poselets. In addition to requiring that each training sample is covered by at least one poselet, we also require that the poselet covers at least one training sample that is not covered by any other poselet, otherwise the poselet would be redundant.

We introduce the poselet seed selection to generate an effective and efficient



True positives

Unknown (Intersection / Union < 0.5)

Unknown (Procrustes dist. between joints > 0.3)

Figure 3.4: Examples of activations labeled as true positives and unknown. The top-left image shows a seed window for part 1 and a configuration of its joints. In the right column, 15 examples (5 for true positives, 10 for unknown activations) are shown in a right of the seed. White and red bounding boxes depict a groundtruth and detected window, respectively. In the third column, the configuration of its joints are depicated in a top-left corner of each image.

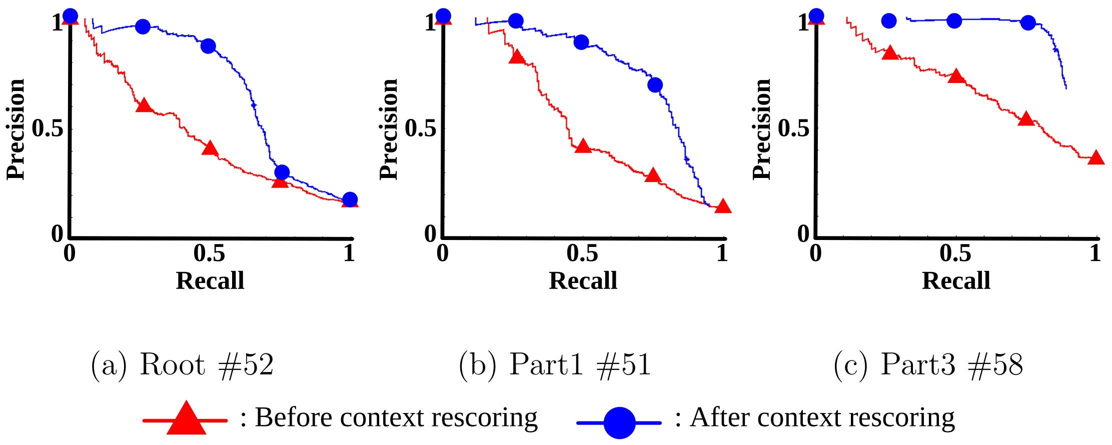
set of poselets by considering these two aspects. The poselet seed selection is an iterative process consisting of two steps: (i) *seed selection* and (ii) *set update*, and each step considers each aspect, respectively. Denote that P is a set of poselets, and C is a list of training sample IDs that are covered by P. The set T of training patches is obtained from the physical joints annotated in the training set by enclosing the annotated joints with a bounding box (plus a suitable amount of padding). First, in the seed selection step, a patch not included in C is randomly selected and its poselet is trained. If a poselet is trained, example IDs containing any of its true positive activation are added to C. Second, the set update step identifies and removes poselets that are redundant (a poselet is redundant if all the patches it covers are already covered by other poselets). Given the coverage set C, a small size P is obtained by approximately solving a set cover problem, which is to identify the smallest subset which still covers all elements. We use a greedy algorithm to approximately solve the set cover problem. First, we sort all poselets in P in an ascending order according to the size of the subset covered by the poselets. Then, starting with the poselet with the smallest coverage, we remove any poselet from P if it is redundant.

3.3.4 Context Rescoring

After training the set of poselets to detect the root and the parts, we rescore activations by exploiting context among activations of the root and the parts. This step removes activation vectors that are not consistent with the detected human pose.

We use labels of activations detected in training dataset for context rescoring. For each root activation we obtain a set of consistent part activations, where consistency between root and part activation is measured by the symmetrized KL (Kullback-Liebler) divergence of their empirical joint distributions $d_{r,p} = \frac{1}{K}\sum_k D_{SKL}(N_r^k, N_p^k)$, where $D_{SKL}(N_r^k, N_p^k) = D_{KL}(N_r^k||N_p^k) + D_{KL}(N_p^k||N_r^k)$. Here, N_r^k and N_p^k are the empirical distributions of the k^{th} joint of root and part, respectively. We treat root and part as consistent if $d_{r,p}$ is below a threshold. For each root activation, we construct an activation vector consisting of the root poselet confidence score concatenated with a vector of the confidence scores of all part poselets. The score of the root activation is placed in the first bin and all consistent activations of parts are placed in their own bins according to the poselet type; multiple consistent activations of the same type are detected, but only the maximum score is entered in the appropriate bin. The remaining bins are filled with zero.

Then, we train a linear SVM classifier with activation vectors and their labels. At test time, root activations that are classified as false positives are discarded, and part activations with no mutually consistent root are also discarded as false positives. Figure 3.5 demonstrates that this context rescoring step effectively improves the precision-recall performance of both root and part poselet detectors by discarding many false positives; in the figure, root #52, part 1 #51, and part 3 #58 were arbitrarily chosen and have typical performance.



(a) Root #52 (b) Part1 #51 (c) Part3 #58

Figure 3.5: PR curves for performance of (a) root #52, (b) part 1 #51, and (c) part 3 #58 on YouTube action dataset. Red lines are obtained before context rescoring while blue lines are after context rescoring. Typical performance is shown for three randomly selected parts.

3.4 Video Representation

We extend the framework of [4] to include our proposed pose feature in addition to motion and shape features. For all features, initial histogram-based video representations are generated via bag-of-visual words (BoVW). After the initial representation is generated for each video sequence, compact yet discriminative visual vocabularies are obtained by feature grouping. A multi-class SVM classifier is trained using as input the concatenated visual word counts for each of the three features. Details about extracting motion, shape, and pose features are given in section 3.4.1 and the method for learning semantic visual vocabulary is described in section 3.4.2

3.4.1 Motion, Shape, and Pose Features

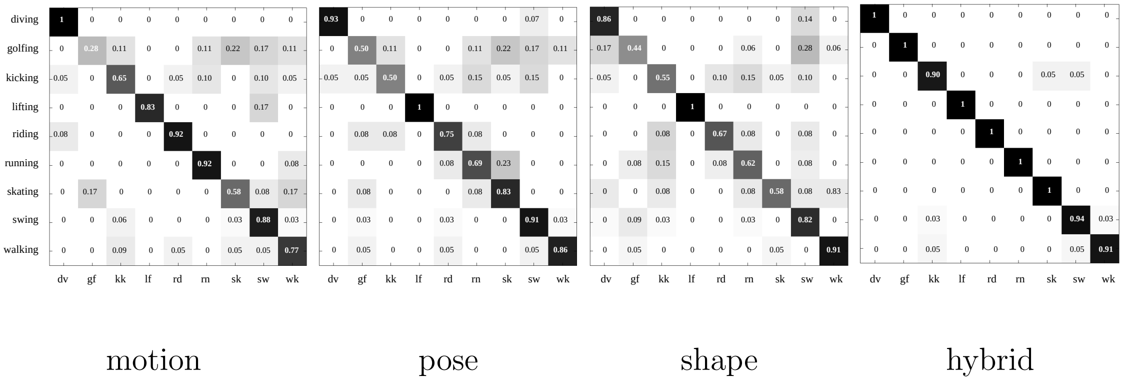
To complement our proposed pose feature, we select motion and shape features that achieve the best performances in [4, 42] on public datasets consisting of unconstrained videos.

Motion feature: We use the spatio-temporal interest point detector and descriptor proposed by Dollar et al. [43], which is described as being advantageous over other methods such as 3D Harris-Corner detector for action recognition in [4].

Shape feature: The shape feature uses the root position to compute a 3-level pyramid HOG around the root which shows the best performance among shape descriptors. [42] The region of interest side length is set to double the maximum value between the root's width and height.

Pose feature: We extract activations of root and parts by multi-scale sliding window and rescore root activations by context rescoring, using the activation vector constructed from all other mutually consistent poselet activations. Root activation vectors that are sufficiently confident after context rescoring (confidence > 0) are used as pose descriptors. The first bin in the activation vector corresponding to the root activation is excluded from the descriptor, since the root activation score is used only to confirm whether or not the root and consistent parts fit the particular qualitative pose model.

For each type of feature, we generate the histogram representation based on independent features via BoVW, which converts all features to ”codewords” using k-means based on their descriptions.



motion

	dv	gf	kk	lf	rd	rn	sk	sw	wk
diving	1	0	0	0	0	0	0	0	0
golfing	0	0.28	0.11	0	0	0.11	0.22	0.17	0.11
kicking	0.05	0	0.65	0	0.05	0.10	0	0.10	0.05
lifting	0	0	0	0.83	0	0	0	0.17	0
riding	0.08	0	0	0	0.92	0	0	0	0
running	0	0	0	0	0	0.92	0	0	0.08
skating	0	0.17	0	0	0	0	0.58	0.08	0.17
swing	0	0	0.06	0	0	0	0.03	0.88	0.03
walking	0	0	0.09	0	0.05	0	0.05	0.05	0.77

pose

	dv	gf	kk	lf	rd	rn	sk	sw	wk
diving	0.93	0	0	0	0	0	0	0.07	0
golfing	0	0.50	0.11	0	0	0.11	0.22	0.17	0.11
kicking	0.05	0.05	0.50	0	0.05	0.15	0.05	0.15	0
lifting	0	0	0	1	0	0	0	0	0
riding	0	0.08	0.08	0	0.75	0.08	0	0	0
running	0	0	0	0	0.08	0.69	0.23	0	0
skating	0	0.08	0	0	0	0.08	0.83	0	0
swing	0	0.03	0	0	0.03	0	0	0.91	0.03
walking	0	0.05	0	0	0.05	0	0	0.05	0.86

shape

	dv	gf	kk	lf	rd	rn	sk	sw	wk
diving	0.86	0	0	0	0	0	0	0.14	0
golfing	0.17	0.44	0	0	0	0.06	0	0.28	0.06
kicking	0.05	0	0.55	0	0.10	0.15	0.05	0.10	0
lifting	0	0	0	1	0	0	0	0	0
riding	0	0	0.08	0	0.67	0.08	0	0.08	0
running	0	0.08	0.15	0	0.08	0.62	0	0.08	0
skating	0	0	0.08	0	0	0.08	0.58	0.08	0.83
swing	0	0.09	0.03	0	0	0.03	0	0.82	0
walking	0	0.05	0	0	0	0	0.05	0	0.91

hybrid

	dv	gf	kk	lf	rd	rn	sk	sw	wk
diving	1	0	0	0	0	0	0	0	0
golfing	0	1	0	0	0	0	0	0	0
kicking	0	0	0.90	0	0	0	0.05	0.05	0
lifting	0	0	0	1	0	0	0	0	0
riding	0	0	0	0	1	0	0	0	0
running	0	0	0	0	0	1	0	0	0
skating	0	0	0	0	0	0	1	0	0
swing	0	0	0.03	0	0	0	0	0.94	0.03
walking	0	0	0.05	0	0	0	0	0.05	0.91

Figure 3.6: Confusion matrix for the YouTube sports [1] data set using combined feature with motion, pose, and shape feature.

3.4.2 Learning Semantic Visual Vocabulary

The initial vocabulary obtained by grouping similar features based on their appearance is far from semantically meaningful and its performance is sensitive to the size of the vocabulary, containing many redundant codewords that do not improve discrimination. We construct a compact yet discriminative visual vocabulary for each type of feature as proposed by [4]. A vocabulary is made compact by combining two bins of a BoVW if their class distributions are close to each other. Here, the distance between two distributions, p_1 and p_2 is measured by Jensen-Shannon (JS) divergence:

$$JS_\pi(p_1, p_2) = \textstyle\sum_{i=1,2} \pi_i KL(p_i, \sum_{j=1,2} \pi_j p_j),$$

$$\pi_1 + \pi_2 = 1, \tag{3.1}$$

where $KL(\cdot)$ is the KL divergence.

Let $C = c_1, c_2, \cdots, c_L$ and $X = x_1, x_2, \cdots, x_M$ represent classes and codes, respectively. Let $\hat{X} = \hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K$ be the updated clusters of X. A semantic visual vocabulary can be obtained by minimizing the loss of mutual information (MI), $Q(\hat{X}) = I(C; X) - I(C; \hat{X})$):

$$Q(\hat{X}) = \sum_{i=1}^{K} \pi(\hat{x}_i) JS(\{p(C|x_t) : x_t \in \hat{x}_i\}), \tag{3.2}$$

where $\pi(\hat{x}_i) = \sum_{x_t \in \hat{x}_t} \pi_t$, $\pi_t = p(x_t)$ is the prior. By equation 3.1, the mutual information is changed to

$$Q(\hat{X}) = \sum_{i=1}^{K} \pi(\hat{x}_i) \sum_{x_t \in \hat{x}_i} \pi_t KL(p(C|x_t), p(C|\hat{x}_i)). \tag{3.3}$$

The semantic representation $\hat{X}$ is generated by iterations of computing priors $\pi(\hat{x}_i), i = 1, 2, \cdots, K$ and updating clusters $i^*(x_t) = argmin_j KL(p(C|x_t), p(C|\hat{x}_i))$. A termination condition of the iteration is $Q(\hat{X}) < \epsilon$.

3.5 Experiments

We evaluate our framework on two benchmarks: YouTube sports dataset [1] and YouTube action dataset [4]. For both datasets, we follow the original authors' setting for evaluation. The multi-class linear SVM is used as the classifier for action recognition with vectors combining semantic representations of motion, pose, and shape feature. Each feature is normalized by L2 norm. Finally, we evaluate the boost in performance provided by our proposed poselet seed selection versus the original scheme proposed in [20]. All clustering parameters, including the size of the initial and semantic vocabulary, are obtained automatically by cross validation.

3.5.1 Experiments on YouTube Sports Dataset

The YouTube sports dataset [1] consists of a set of actions collected from various sports which are typically seen in broadcast media. For each feature, we set the initial vocabulary size to 500 and the semantic vocabulary size to 100. During training, we store for each poselet the video sequence from which its training images were selected. For clustering, we set the portion of coverage to 0.8, resulting in 123, 120, 120, and 123 poselets for the root and the three parts, respectively.

Figure 3.6 shows the confusion matrix for classification using motion, pose, shape, and hybrid (combination of all three) features. The motion feature is useful for classifying actions in which human locations change significantly, e.g., diving, horseback riding, and running. On the other hand, the pose feature outperforms others for actions consisting of distinctive poses, e.g., arm pose after golf swing or lifting and pose of legs when skating. For walking, the shape feature yields the best classification performance since walking does not involve particularly distinctive motions or poses. In table 3.2, the recognition rates using pose feature are the highest among the three types of features. Using a hybrid of motion, pose, and shape features yields an improvement in performance over Sadanand and Corso [40], the state-of-the-art.

3.5.2 Experiments on YouTube Action Dataset

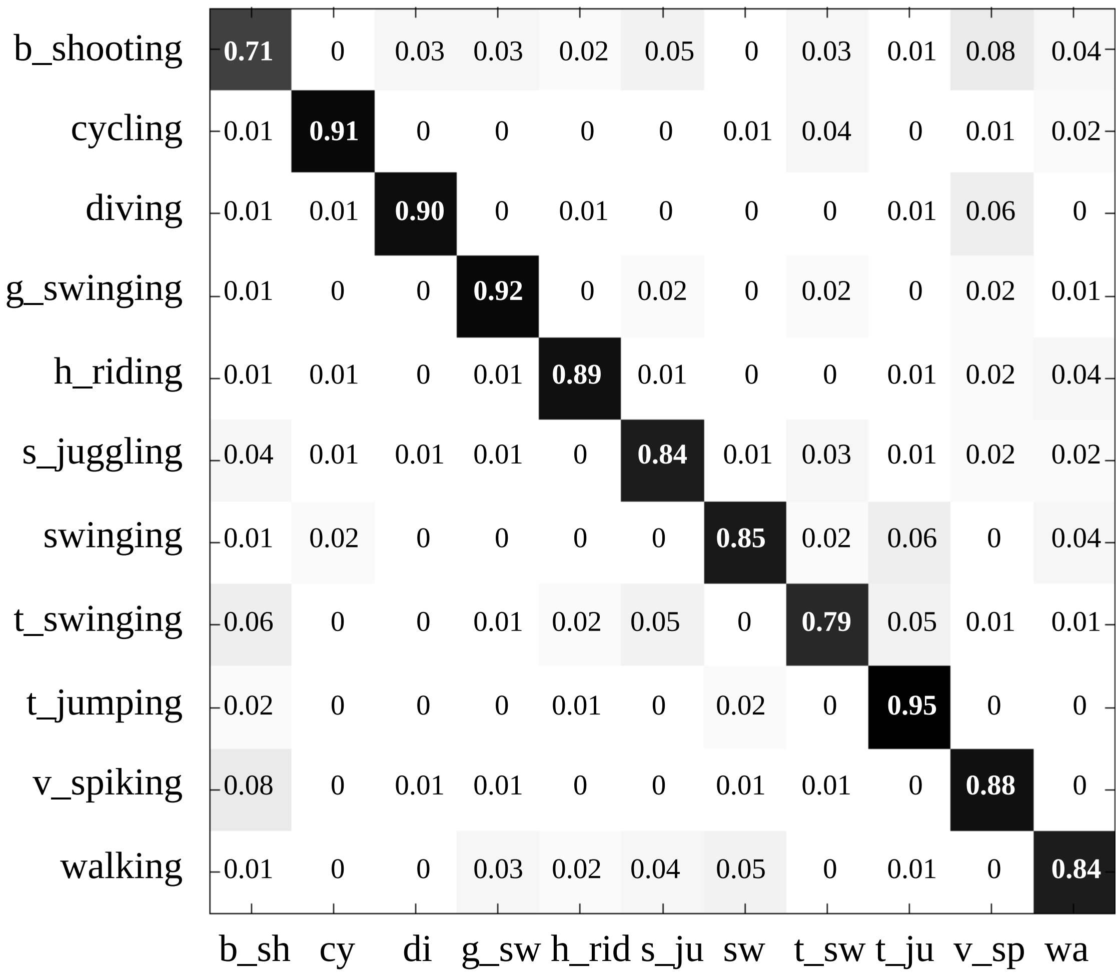
We also evaluate our framework on the challenging YouTube action dataset [4] consisting of 11 action classes. For clustering, we select 100 poselets for the root and

Table 3.2: Recognition rates on the YouTube sports data set.

Method	Accuracy (%)
Wang et al. [44]	85.6
Le et al. [38]	86.5
Kovashka and Grauman [45]	87.3
Wang et al. [35]	88.2
Wu et al. [46]	91.3
O'Hara and Draper [47]	91.3
Todorovic [39]	92.1
Sadanand and Corso [40]	95.0
Shape	71.3
Motion	75.3
Pose	76.7
Pose + Shape	84.7
Motion + Shape	86.7
Motion + Pose	90.7
Motion + Pose + Shape	**96.0**

each part. Here, we set the size of the initial vocabulary and semantic vocabulary to 1000 and 100, respectively.

Figure 3.7 shows the confusion matrix for the YouTube action dataset. Based on the confusion matrix, our framework has the worst performance on basketball



	b_sh	cy	di	g_sw	h_rid	s_ju	sw	t_sw	t_ju	v_sp	wa
b_shooting	**0.71**	0	0.03	0.03	0.02	0.05	0	0.03	0.01	0.08	0.04
cycling	0.01	**0.91**	0	0	0	0	0.01	0.04	0	0.01	0.02
diving	0.01	0.01	**0.90**	0	0.01	0	0	0	0.01	0.06	0
g_swinging	0.01	0	0	**0.92**	0	0.02	0	0.02	0	0.02	0.01
h_riding	0.01	0.01	0	0.01	**0.89**	0.01	0	0	0.01	0.02	0.04
s_juggling	0.04	0.01	0.01	0.01	0	**0.84**	0.01	0.03	0.01	0.02	0.02
swinging	0.01	0.02	0	0	0	0	**0.85**	0.02	0.06	0	0.04
t_swinging	0.06	0	0	0.01	0.02	0.05	0	**0.79**	0.05	0.01	0.01
t_jumping	0.02	0	0	0	0.01	0	0.02	0	**0.95**	0	0
v_spiking	0.08	0	0.01	0.01	0	0	0.01	0.01	0	**0.88**	0
walking	0.01	0	0	0.03	0.02	0.04	0.05	0	0.01	0	**0.84**

Figure 3.7: Confusion matrix for the YouTube action [4] data set using combined feature with motion, pose, and shape feature.

shooting and walking. Because the pose observed during shooting in basketball is similar to swinging an arm in tennis or spiking in volleyball, most of the miss-classified video sequences are classified into those classes. The reason for the low classification performance for walking is likely the same as for the previous dataset. In table 3.3, our framework outperformed other algorithms by approximately 10.4%. Interestingly, using pose feature alone provides recognition rates which matches

Table 3.3: Recognition rates on the YouTube action data set. We outperform the state-of-the-art by over 10%.

Method	Accuracy (%)
Liu et al. [4]	71.2
Zhang et al. [48]	72.9
Ikizler-Cinbis and Sclaroff [49]	75.2
Le et al. [38]	75.8
Shape	52.3
Motion	62.2
Motion + Shape	72.9
Pose	74.6
Pose + Shape	76.0
Motion + Pose	83.5
Motion + Pose + Shape	**86.2**

all the state-of-the-art. Figure 4.6 shows some examples of pose features for a qualitative evaluation.

3.5.3 Boost by Poselet Seed Selection

In this section, we compare our proposed poselet seed selection process against the random selection process of [20] in performance. The proposed selection process results in a set of poselets that cover 80% of the training examples (a training sample

Table 3.4: Top rows: the percentage of the training dataset *covered* (see text) as the number of total poselets is varied. Bottom row: the resulting action recognition rates. The right column shows the coverage and recognition rates of our proposed selection approach.

		random selection			proposed
number of poselets		400	800	1200	486
covered set (%)	root	56.1	60.6	62.3	**80.1**
	part 1	48.7	54.4	56.5	**80.0**
	part 3	53.3	59.3	61.7	**80.1**
recognition rate (%)		63.3	67.3	71.3	**76.7**

is *covered* if the poselet detector yields an activation that overlaps sufficiently with the training sample), which results in a final recognition rate of 76.7 on the youtube sports dataset [1]. The sizes of the poselets set for root, part 1, and part 3 are 123, 120, and 123, respectively. Part 1 and 2 are mirrored versions of each other, thus yielding a total of 486 poselets. Table 3.4 shows the performance over various numbers of poselets chosen by random selection versus our approach. As the number of poselet grows, the coverage of the training dataset and recognition rate improves but does not match the recognition rate obtained by our proposed poselet seed selection until training reaches 300 poselets for the root and each part (for a total of 1200 for the root and the three parts, as in [34]).

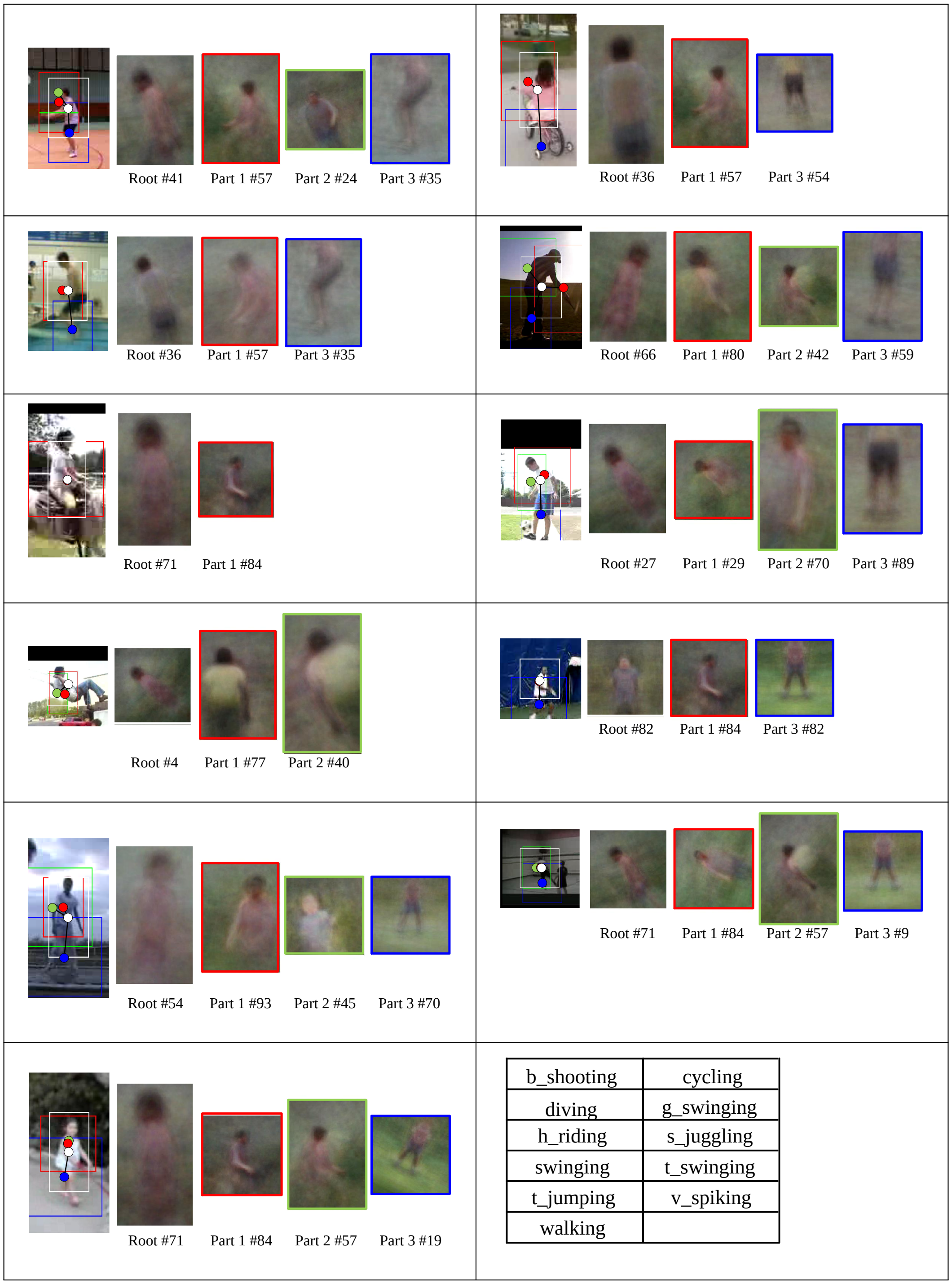


Figure 3.8: Example root and part activations for each class in the YouTube action dataset. The left-most image in each example is a region of the test image cropped by the root activation bounding box (plus padding), with consistent parts highlighted. The average image of some detected poselet is shown to the right (note: there are other activations that are not shown due to space constraints).

3.6 Conclusion

We proposed a robust pose feature based on poselets that is suitable for use in action recognition tasks involving relatively unconstrained videos. We have shown that various modifications of the poselet training process improve the representation power of the set of poselets, generating a set of features that can be seamlessly combined with existing shape and motion features. Experiments show that our proposed pose feature provides significant information alone; when in addition to motion and shape, we obtain state-of-the-art results.

Chapter 4: Clauselets: Leveraging Temporally Related Actions for Video Event Analysis

4.1 Introduction

We are living in a world where it is easy to acquire videos of events ranging from private picnics to public concerts, and to share them publicly via websites such as YouTube. The ability of smart-phones to create these videos and upload them to the internet has led to an explosion of video data, which in turn has led to interesting research directions involving the analysis of "in-the-wild" videos. Recent approaches to processing these types of videos use features that range from low- to mid-level, some even using features that directly correspond to words that describe portions of the videos [19]. While all of these approaches obtain competitive results on benchmark datasets, mid-level features that can also describe the semantic content of a video are desirable since they can be used to describe the video using language as well as to recognize events.

The detection of visual patterns that directly correspond to individual semantically meaningful actions is practical even in "in-the-wild" videos, as shown by recent works on benchmark datasets. Izadinia and Shah [50] model the joint relationship

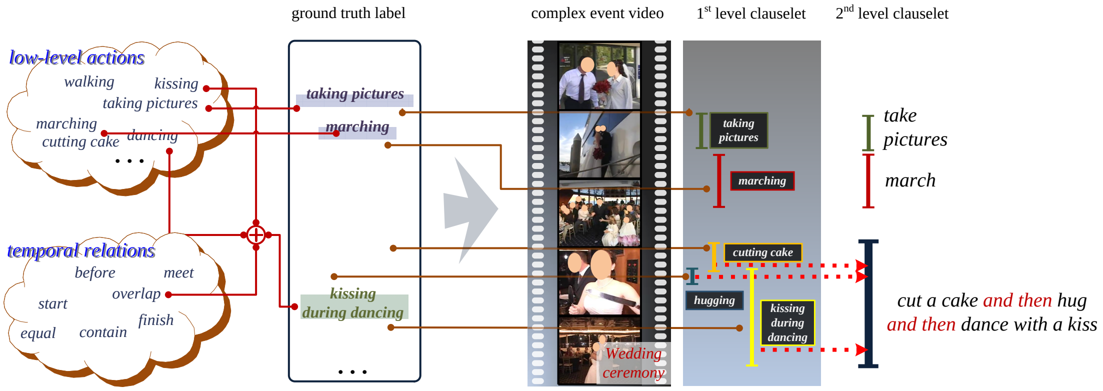


Figure 4.1: The illustration of our approach for describing the complex event video (*wedding ceremony*) with two level clauselets defined by relevant actions and temporal relationships. (*e.g. cut a cake and then hug and then dance with a kiss*) Ground truth labels contain potentially concurrent actions in particular temporal relationships. Given a video, 1^{st} level clauselets search for relevant labels with the video. 2^{nd} level clauselets group concurrent and consistent labels using coarse temporal relationships (words colored by red).

between two actions for recognizing high-level event. While pairs of actions capture more information than single actions alone, valuable information from higher order interactions remains unused. Ma et al. [51] introduce visual attributes that combine human actions with scenes, objects, and people for exploring mutual influence and mining extra information from them. Various approaches jointly model more than two local object or action detections. Bag-of-words (BOW) is a simple but still competitive video representation, which is formed by collecting local detections and generating a histogram by quantizing the feature space. Spatial-temporal pyramids collect local detections from different spatial and temporal resolutions of a video. Various graphical structures to model relations of local detections also exist. (e.g. HMMs [52], Dynamic Bayesian Networks [53], prototype trees [10], AND-OR graphs [54], latent SVM [55], Sum-Product Network [56], and Markov Logic Networks [57]). The key advantage of graphical structures is that they model the dependence of actions by local relationships while allowing for the joint optimization of a global task-dependent objective function. Our goal is to design a mid-level representation that builds on previous low- and mid-level representations, but which is able to capture higher order relationships between actions over small spatio-temporal neighborhoods without the full use of graphical structures.

We rely on temporal relationships to capture the context between actions and provide a richer description of a video than each independent action alone. We define a *clauselet* as a conjunction of actions that are reliably detected in "in-the-wild" videos and their temporal relationships. We apply this definition hierarchically at two levels of granularity, first to detect short sequences involving a limited number of action labels, and then to relate these detected sequences to each other over larger time spans and more actions. Given a set of clauselets, we scan the test video, and use the detected clauselet activations to vote for each clauselet's dominant event. We show our approach in figure 4.1. First, videos are split into clips which are annotated with one or more concurrent actions per clip. Then, 1^{st} level clauselets detect short actions patterns (*e.g. taking pictures, marches, kissing during dancing etc.*) that occur during an event ("wedding ceremony" in the example). Finally,

2^{nd} level clauselets are formed modeling the temporal relationships between 1^{st} level clauselets and other 1^{st} level clauselets that cooccur temporally to create a richer and more discriminative description of the video (*e.g. cut a cake and then hug and then dance with a kiss*).

Our contributions are that we:

1. Introduce temporal relationships between actions and groups of actions for richer video description ($1^{st}/2^{nd}$ level clauselet)

2. Propose a discriminative training process that automatically discovers action patterns and temporal relationships between them

As our experiments demonstrate, these contributions lead to improvements over state-of-the-art approaches to event classification.

In section 4.2, we discuss related works. In section 4.3 and section 4.4, we describe details of 1^{st} and 2^{nd} level clauselets and event recognition, respectively. In section 4.5, we present the experimental results that demonstrate the performance of our approach on "in-the-wild" videos from the TRECVID dataset [58]. We present our concluding remarks in section 4.6.

4.2 Related Work

We divide recent related work into three groups: low-level approaches that improve video features that capture shape and motion information, mid-level approaches that model patterns in low-level features with varying degrees of top-down supervision, and high-level approaches that apply high-level prior knowledge to low- and mid-level observations.

Low-level representations are constructed from local features including SIFT [59], Dollar et al. [43], ISA [38], STIP [60] as well as global features including GIST [61]. Low-level features alone yield competitive performance, however, they do not leverage task dependent information and higher order relationships.

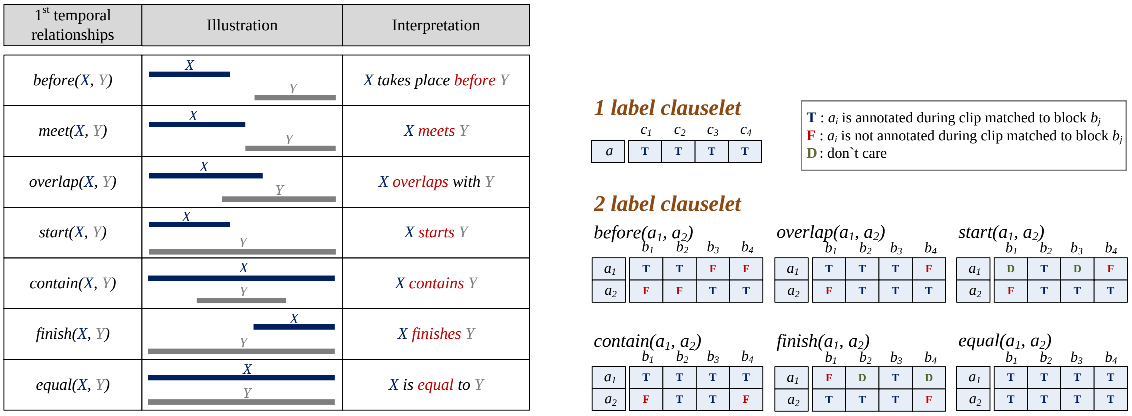
Mid-level representations add task-dependent information to extract more informative patterns from low-level features. Amer and Todorovic [56] train a sum-product network representing human activities by variable space-time arrangements of primitive actions. Jain et al. [62] introduce mid-level spatio-temporal patches that discriminate between primitive human actions, a semantic object. Song et al. [63] learn hidden spatio-temporal dynamics from observations by CRFs with latent variables and, in the test phase, group observations that have similar semantic meaning in some latent space.

High-level modeling combines or organizes low- or mid-level detections based on a knowledge base (KB). Nevatia et al. [64] define an event ontology that allows natural representation of complex spatio-temporal events common in the physical world by a composition of simpler events. Brendel et al. [65] combine the probabilistic event logic (PEL) KB with detections of primitive events for representing temporal constraints among events. Morariu and Davis [57] use the rules that agent must follow while performing activities for multi-agent event recognition. We note that in high-level recognition task, the KB is generally used to reduce false positives of low-level detections by providing spatial-temporal constraints.

Our proposed representation, the *clauselet* is a mid-level detector that bridges the gap between the low- and high-level task. Clauselets share many of the benefits of poselets [20] which are detectors trained to detect patches that are tightly clustered in both appearance and pose space, for the purpose of detecting people and their parts. However, in our case, clauselets are tightly clustered in temporal relationships and video appearance, and our goal is to construct visual event descriptions. Similar to poselets [20] we also rescore clauselet activations by mutually consistent activations, and find that this greatly improves performance.

4.3 Clauselets

Motivated by the intuition that the temporal relationships between multiple concurrent actions are important for event modeling, we propose a mid-level representation involving multiple actions and their temporal relationships. We define a *clauselet* as a conjunction of reliably detected actions and their temporal relationships. We apply this intuition hierarchically at two levels of granularity, first to detect short sequences involving a limited number of action labels (1^{st} level clauselets), and then to relate these detected sequences to each other over larger time spans and more actions (2^{nd} level clauselets).



1st temporal relationships	Illustration	Interpretation
before(X, Y)	X Y	*X takes place before Y*
meet(X, Y)	X Y	*X meets Y*
overlap(X, Y)	X Y	*X overlaps with Y*
start(X, Y)	X Y	*X starts Y*
contain(X, Y)	X Y	*X contains Y*
finish(X, Y)	X Y	*X finishes Y*
equal(X, Y)	X Y	*X is equal to Y*

(a)

1 label clauselet

	c_1	c_2	c_3	c_4
a	T	T	T	T

T : a_i is annotated during clip matched to block b_j
F : a_i is not annotated during clip matched to block b_j
D : don`t care

2 label clauselet

before(a₁, a₂)

	b_1	b_2	b_3	b_4
a_1	T	T	F	F
a_2	F	F	T	T

overlap(a₁, a₂)

	b_1	b_2	b_3	b_4
a_1	T	T	T	F
a_2	F	T	T	T

start(a₁, a₂)

	b_1	b_2	b_3	b_4
a_1	D	T	D	F
a_2	F	T	T	T

contain(a₁, a₂)

	b_1	b_2	b_3	b_4
a_1	T	T	T	T
a_2	F	T	T	F

finish(a₁, a₂)

	b_1	b_2	b_3	b_4
a_1	F	D	T	D
a_2	T	T	T	F

equal(a₁, a₂)

	b_1	b_2	b_3	b_4
a_1	T	T	T	T
a_2	T	T	T	T

(b)

Figure 4.2: 1^{st} level temporal relationships: (a) Allen's interval logic [5], (b) temporal templates used for searching positive examples by matching to ground truth annotations; here we use 1^{st} level clauselets of length k=4 blocks (each block is matched to a clip).

4.3.1 1^{st} level clauselets

4.3.1.1 Model

A 1^{st} level clauselet models sequences containing one or two actions in particular temporal relationships. We use the 7 base relations of Allen's interval logic [5] as the 1^{st} level temporal relationships: *before*, *meet*, *overlap*, *start*, *contain*, *finish*, and *equal*. Figure 4.2 (a) shows the definition of the 7 relations. In our experiments, *meet* is not used since it is too rigid to capture relations among actions annotated at a relatively large granularity (10 seconds per clip in our experiments).

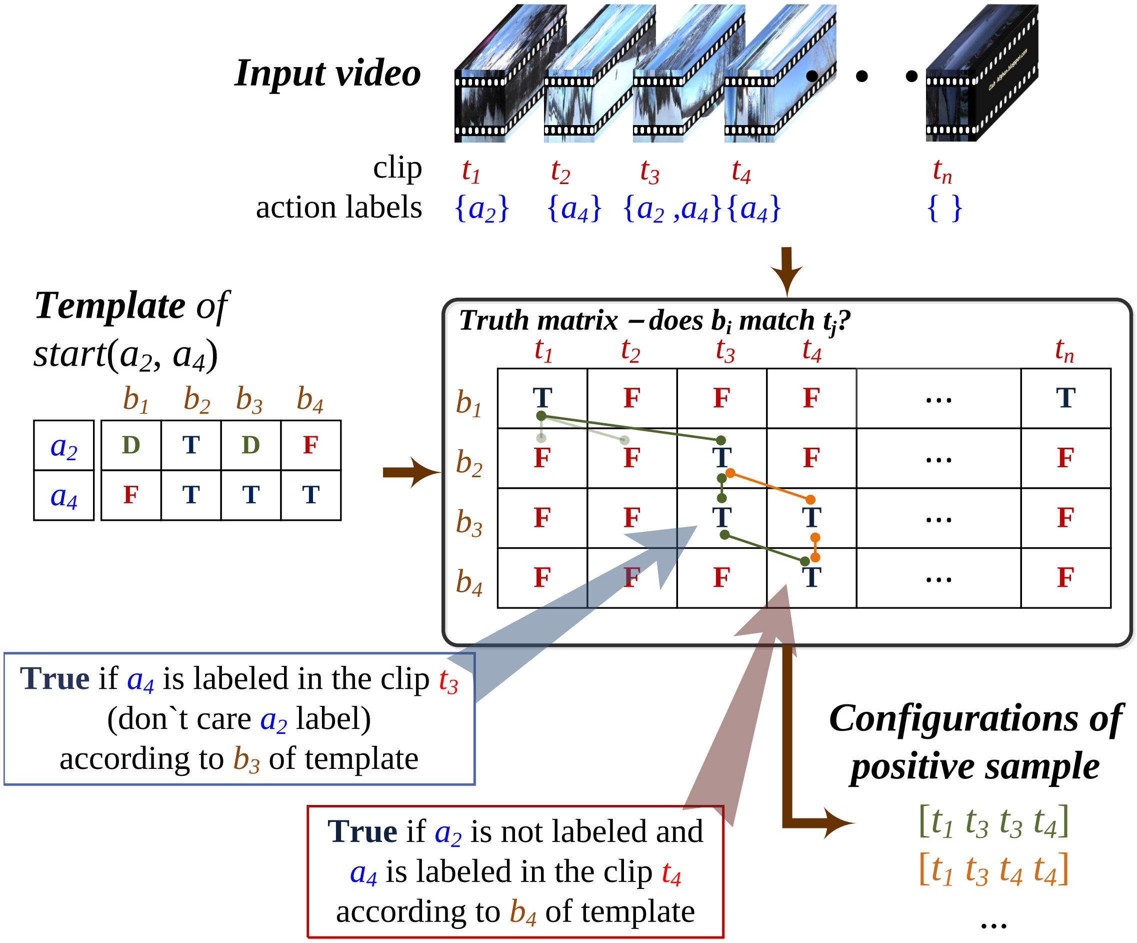
A video is split into n clips, $t_1, \cdots, t_n$, and each clip t is represented by a standard set of features concatenated into a feature vector $f(t)$ (see sec 4.5.1). A 1^{st} level clauselet c model consists of k blocks b_i for $i = 1, \ldots, k$, each of which must be matched to a video clip. Each block has an associated weight vector $w_{c,i}$ which is used to score each valid configuration $T = (T_1, \ldots, T_k)$ that matches every block b_i to a clip index $T_i \in \{1, \ldots, n\}$ as follows:

$$S_{c,T} = \sum_{i=1}^{k} w_{c,i} f(t_{T_i}). \tag{4.1}$$

A configuration T is considered valid if it satisfies a set of temporal deformation rules, i.e., $T \in \{T_{1:k} | T_1 \in \{1, \cdots, n\},\ T_{i-1} \leq T_i \leq T_{i-1} + 2,\ i = 2, \cdots, k\}$. These temporal deformations between blocks are similar to the spatial deformations of parts in a Deformable Part Model (DPM) [2], although we do not apply a deformation penalty as long as a configuration is valid. Eq. 5.3 can be evaluated using a recursive matching process, where given an initial starting clip T_1 to which the first block of the clauselet c is matched, the next block is matched to either T_1, $T_1 + 1$, or $T_1 + 2$, and so on. This process allows the k blocks of a clauselet to span 1 to $2k - 1$ clips. A configuration T of clauselet c is called an *activation* if $S_{c,T} \geq \lambda_s$, where λ_s is the activation threshold.

4.3.1.2 Training

The training process requires a set of videos whose clips are each annotated with a subset of zero, one, or more groundtruth action labels from a large vocabulary. Because 1^{st} level clauselets are intended to detect an action or pair of actions in particular temporal configurations, we define a set of temporal templates that are


Input video
clip t_1 t_2 t_3 t_4 t_n
action labels $\{a_2\}$ $\{a_4\}$ $\{a_2, a_4\}$ $\{a_4\}$ $\{\ \}$
Template of $start(a_2, a_4)$
Truth matrix – does b_i match t_j?
True if a_4 is labeled in the clip t_3 (don`t care a_2 label) according to b_3 of template
True if a_2 is not labeled and a_4 is labeled in the clip t_4 according to b_4 of template
Configurations of positive sample
$[t_1\ t_3\ t_3\ t_4]$
$[t_1\ t_3\ t_4\ t_4]$
...

	b_1	b_2	b_3	b_4
a_2	D	T	D	F
a_4	F	T	T	T

	t_1	t_2	t_3	t_4		t_n
b_1	T	F	F	F	...	T
b_2	F	F	T	F	...	F
b_3	F	F	T	T	...	F
b_4	F	F	F	T	...	F

Figure 4.3: Example of the matching process ($start(a_2, a_4)$). (Directions from truth matrix(1,1) to three successors indicates temporal deformation.) The green and orange paths denote the two possible configurations where each block matches is matched to one clip (note that two blocks might match to the same clip and that some clips might be skipped). A similar process is applied at test time, but paths are chosen to maximize SVM scores instead.

matched to groundtruth video annotations to yield a set of configurations T that all have the same temporal relationships and can be used as positive training samples.

For each template, we consider the same set of valid configurations as in the matching process described above, but instead of computing the dot product of block weights with clip features, we verify that the constraints of each template block are satisfied by the matched clip annotations. The templates are shown in figure 4.2 (b). Every template block has one of three rules: **T** means block b_j can only match a clip if the clip contains action label a_i, **F** means that the clip must *not* contain action label a_i, and **D** indicates 'don't care'. For each action and pair of action labels, we extract positive training samples by matching these templates to groundtruth annotations (see fig. 4.3). Assuming that we have A action labels and we instantiate the templates in figure 4.2 (b) for each action or pair of actions, we have $A+11A(A-1)/2$ total templates. The first term is for the 1-action template, and the second term is for the five 2-action templates that are order dependent yielding templates per action pair plus *equal*, which is order independent and yields only one template. All configurations successfully matched to one template will be used to train one clauselet. For each template, we also construct a set of negatives by randomly selecting clip groups that do not contain any of the action labels appearing in that template.

For each matched configuration we extract the features of the corresponding clips, concatenate them into a single vector, and train a linear SVM classifier to separate the positive examples from the negative sample set (which is five times the size of the positive set). The resulting SVM weights are then partitioned into the corresponding 1^{st} level clauselet block weights. We then scan over the training videos (using the learned weights this time), collect false positive activations, and retrain linear SVM classifiers, repeating this process a few times with increasingly more negative examples.

4.3.2 2^{nd} level clauselets

4.3.2.1 Model

The proposed 1^{st} level clauselets are limited in length and number of unique actions for computational reasons, since SVMs operate over high-dimensional video features, and more actions or clauselet blocks would lead to combinatorial blowup. To obtain a richer set of clauselets, which we call 2^{nd} *level clauselets*, we model the temporal relationships between the 1^{st} level clauselets, without limiting the number of action labels, and learn only configurations that are detected in the training videos instead of enumerating them as in the 1st clauselet training stage. Thus, a 2^{nd} level clauselet is defined as a group of mutually consistent 1^{st} level clauselets that coocur in particular temporal configurations.

For each 1^{st} level clauselet c_i, we obtain the set of 1^{st} level clauselets $c_{i1}, c_{i2}, \cdots$, c_{im} that are concurrent with c_i, i.e., they are nearby in time. (see Figure 4.4) For each activation, we construct a vector x consisting of the activation's score and the score of concurrent clauselet activations, grouped by clauselet type and temporal relationship type, and we use this vector to rescore the activation. Let the *head activation* be the activation that is rescored, and let a *concurrent activation* be any activation whose temporal interval overlaps the head activation temporal interval by at least one clip length. Each concurrent activation is classified into one of the

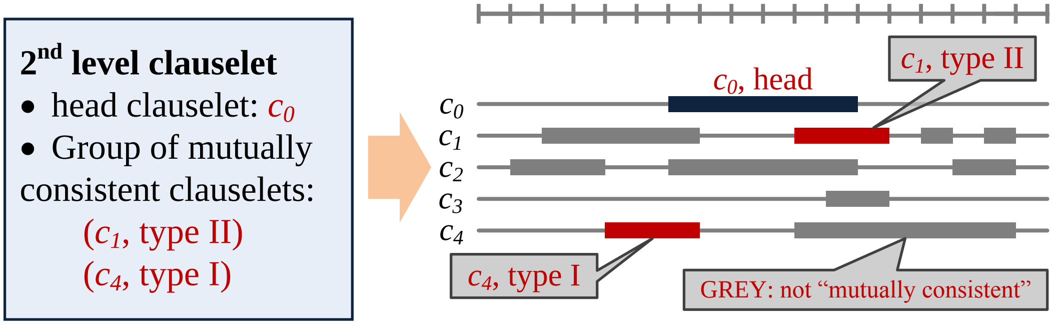


Figure 4.4: Illustration of the process selecting head clauselet and discovering concurrent activations that are mutually consistent with the head clauselet and cooccur in particular temporal configuration w.r.t. the head clauselet given the 2^{nd} level clauselet model.

2^{nd} level temporal relationships with respect to the head activation. These 2^{nd} level temporal relationships could in theory be any of the 7 base relationships in figure 4.2 (a), but we choose a coarser set of 4 relationships from figure 4.5. Our motivation for the coarser set of temporal intervals is that the temporal relationships that involve touching interval endpoints (starts, meets, equals) are less likely to occur and are more noisy, so we group them with one of our the four coarse temporal relationships (e.g., *equals* is part of the Type IV relationship, *meet* is part of Type I). Figure 4.5 shows the definition of the 4 types of 2^{nd} level temporal relationships. The vector x is constructed by placing the head activation score as the first feature, and then for each clauselet and each 2^{nd} level temporal relation, we add a feature equal to the maximum score of each activation of that clauselet (i.e., we use max-pooling if there

		Before start index $s_i < s$	After end index $e_i > e$
2^{nd} *temporal relations*	Type I	True	False
	Type II	False	True
	Type III	True	True
	Type IV	False	False

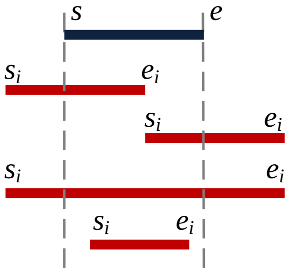


Figure 4.5: Definition and illustration of 2^{nd} temporal relationships

are multiple activations of the same clauselet and temporal relation type). The total vector length is $4n+1$, where 4 corresponds to the number of temporal relationships, n is a total number of trained clauselet models, and the 1 corresponds to the head activation. This activation vector is treated as a feature vector for rescoring the head activation.

The rescoring function is defined as

$$f_{w_s,S}(x) = w_s^T Sx, \tag{4.2}$$

where $x \in \mathbb{R}^{4n+1}$ is the input activation vector, $S \in \mathbb{R}^{m\times(4n+1)}, m \leq 4n+1$ is a subset matrix which selects m of the $4n+1$ scores in x and is formed by selecting the appropriate rows of the identity matrix I_{4n+1}. The weight vector $w_s \in \mathbb{R}^m$ is a vector that determines how the scores of selected activations are combined linearly to rescore the head activation.

4.3.2.2 Training

For each 1^{st} level clauselet, we scan over the training dataset, extract activations, and assign them as one of three labels: *positive*, *negative*, and *undecided*. If an activation overlaps 75% or more of the clips in a groundtruth positive example, it is labeled positive. If the activation clips do not contain any groundtruth action labels associated with the clauselet, it is labeled negative. Others remain undecided. The positive and negative activations are used for training 2^{nd} level clauselets.

S and w_s are optimized by minimizing the objective function below:

$$L_D(w_s, S) = \frac{1}{2} w_s^T w_s + C \sum_{i=1}^{N} \max(0, 1 - y_i f_{w_s,S}(x_i)), \tag{4.3}$$

where $y_i, i = 1, 2, \cdots, n$ is a label of the activation vector x_i defined as

$$y_i = \begin{cases} 1 & \text{if head activation of } x_i \text{ is positive} \\ -1 & \text{if head activation of } x_i \text{ is negative} \end{cases}$$

The objective function is the same to that of linear SVM model except for the score function in the hinge loss. To minimize the objective function, we use a coordinate descent approach that iteratively alternates between SVM and subset matrix optimizations as follows:

1. **Weight learning:** optimize $L_D(w_s, S)$ over w_s by learning linear SVM weights with subset of activation vector Sx

2. **Subset selection:** optimize $L_D(w_s, S)$ over S by selecting subset of features

to minimize the hinge loss of $L_D(w_s, S)$. The optimization is achieved by independently deciding whether a feature is included by checking if its inclusion/exclusion decreases the hinge loss.

The subset matrix S selects from among the concurrent activations only those that are mutually consistent (i.e. those that add to the score of the head activation), and the weight vector w_s decides how much weight each mutually consistent activation adds to the score of the head activation.

4.4 Event Recognition

We expect that clauselets will serve as useful building blocks for complex high-level reasoning (e.g., in probabilistic logical frameworks such as [5, 52]). However, to best isolate their contribution and demonstrate their utility, we employ a simple voting strategy where each clauselet activation votes for its predominant event class. Not all 1^{st} level clauselet templates lead to a trained clauselet model, because of insufficient training examples. Also, not all of the clauselet models that are trained cast a vote for an event, because they are not sufficiently predictive of a set of events. For this purpose, we find clauselets that achieve high recall and precision, defined as follows:

- *precision(e, c):* ratio of all activations of clauselet c that occur during events of class e
- *recall(e, c):* ratio of all instances of event class e containing at least one activation of clauselet c

A *precision* threshold is used to choose clauselets dominant in a certain event while a *recall* threshold is used to avoid overfitting to a few positive samples during training. Table 4.1 shows the number of clauselets used or discarded in voting according to the *precision* criterion. To avoid multiple votes by activations of the same type that are temporally close, we use non-maximum suppression, removing activations if they overlap temporally more than 50% with one or more activations with higher score. While not all 1^{st} level clauselets that are trained cast a vote for event recognition, all successfully trained 1^{st} level clauselets are used for context rescoring in 2^{nd} level clauselets.

Table 4.1: Number of 1^{st} level clauselets

	1 label (used in voting)	*1 label* (not used in voting)	*2 label* (used in voting)	*2 label* (not used in voting)
# of clets	87	6	372	17

4.5 Experiments

4.5.1 Dataset and parameter setting

We evaluate clauselet based voting event recognition on the TRECVID MED 11 dataset [58] containing 15 complex events. Each event category contains at least 111 videos whose duration varies from several seconds to longer than 10 minutes. Following [50], we split every video into 10 second clips and annotate 123 action labels in each clip. We represent each clip by the 6 features used in [50]: ISA (Independent Subspace Analysis) [38], STIP [60], Dollar et al. [43], GIST [61], SIFT [59], and MFCC (Mel-Frequency Cepstral Coefficient) [66]. For all features, histogram-based clip representations are generated via bag-of-visual words (BOVW).

We also follow the evaluation setting of [50] that randomly splits the dataset into training and test set by a ratio of 0.7. We re-split the training dataset into two sets with a ratio of 0.7 for training 1^{st} level and 2^{nd} level clauselets, respectively.

We compute *precision* and *recall* of the trained clauselets, and then empirically set their thresholds to 0.5 and 0.1, respectively, to ensure enough clauselets are trained and selected for voting. We also set the number of clauselet blocks to 4 in order to limit computational complexity and to extract sufficiently many positive examples for training (templates become more specific and rare as the number of blocks increases). We set λ_s to -0.5 to detect sufficient true positives.

4.5.2 Detection performance

We evaluate our detection performance and compare 1^{st} and 2^{nd} level clauselets while evaluating the boost obtained by adding 2-label clauselets to 1-label clauselets. Based on *precision* and *recall*, 93 action alone (out of 123) and 359 pairs of actions and their particular temporal relationships (out of 82533) are selected as 1-label and 2-label clauselets for the evaluation, respectively. The distribution of temporal relationships used in 2-label clauselets is given in table 4.2. *Before* is understandably dominant but number of other relationships seems to be large enough to be useful for describing video.

Table 4.2: Number of interval relations

Temporal relations	before	overlap	start	contain	finish	equal
#	180	25	57	34	61	32

Table 4.3 compares the detection performance of 1^{st} and 2^{nd} level clauselets (note that we are evaluating the ability of the clauselet detector to find the intended action pattern, not to perform event recognition). To confirm the utility of mutually consistent subset selection and temporal relationship binning 2^{nd} level clauselets, we evaluate 2^{nd} level clauselets in three ways: (i) rescoring by collecting all concurrent activations and without differentiating them based on temporal relationships (second row in table 4.3), (ii) applying the feature selection scheme to group concurrent and consistent activations, ignoring irrelevant activations (third row in table 4.3), and (iii) our proposed approach of applying both feature selection and coarse temporal relationships in rescoring (last row in table 4.3).

Our experiments confirm two things based on table 4.3. First, 2 label 1^{st} level clauselets are more accurate detectors than 1 label 1^{st} level clauselet (i.e., they are more effective at finding the corresponding ground truth patterns of action labels). This is consistent with the trend in computer vision where detectors of more complex pattern tend to have fewer false positives. Second, exploiting consistency among concurrent activations and selecting subset features to maximize the discriminability seems to increase the detection performance of the clauselets. We note that 2^{nd} level clauselets provide the more descriptive analysis with comparable detection

Table 4.3: Comparison of detection performance of 1^{st} and 2^{nd} level clauselets. We report the Average Precision (AP) of all clauselets, evaluated against the ground truth action patterns that the clauselets are intended to detect.

	1 label	*1&2 label*
1^{st} level clauselet	0.1497	0.1613
2^{nd} level clauselet w/o selection matrix S & tempo. relation	0.1637	0.1906
2^{nd} level clauselet w/o temporal relationships	0.1638	0.1913
2^{nd} level clauselet	**0.1703**	**0.1915**

performance, since multiple actions are related to each other temporally.

4.5.3 Performance in recognizing complex events

We evaluate the voting based event recognition performance of our model and also compare the proposed clauselets against our baseline including 1^{st} level clauselets and 2^{nd} level clauselets, excluding various components of our proposed approach such as coarse temporal relationships and feature selection, in order to evaluate the impact of each of the components of our approach. Table 4.4 shows event recognition performances of our models. Votes by relevant clauselet activations to a particular event are used to compute a mean of average precision (mAP) of the event. Table 4.4 shows that recognition performance is directly related to clauselet detection performance. We note that the rescoring scheme alone achieves state-

Table 4.4: Mean of average precision (mAP) on the event recognition task, obtained via the our proposed voting scheme.

	1 label	*1&2 label*
1^{st} level clauselet	0.3893	0.4651
2^{nd} level clauselet w/o selection matrix S & tempo. relation	0.4016	0.6596
2^{nd} level clauselet w/o selection matrix S	0.4068	0.6641
2^{nd} level clauselet	**0.4371**	**0.6730**

of-the-art performance (0.6639). By additionally including our proposed mutually consistent clauselet selection and temporal relationships we are able to obtain a richer description of the video employing various temporal relationships as well as outperform the state-of-the-art on the event recognition task.

We also compare the recognition performance of our proposed approach against that of the state-of-the-art in each event category. Table 4.5 compares the performance of our approach against two baselines: [50] and [19]. Our approach shows 1% improvement over state-of-the-art. A 1% percent improvement over the baseline is larger than the typical improvements we observed for this dataset; e.g., Ramanathan et al. [19] reported a .29% improvement over their baseline. We did not use any sophisticated optimization schemes to tailor clauselets to the complex event prediction which makes the 1% improvement more significant.

Figure 4.6 shows examples of 1^{st} and 2^{nd} level clauselet activations in some

Table 4.5: Comparison of clauselets against two baselines.

Event	[50]	[19]	clauselets
Boarding trick	0.7570	0.8402	**0.9133**
Feeding animal	**0.5650**	0.4595	0.5472
Landing fish	**0.7220**	0.6593	0.4902
Wedding ceremony	0.6750	**0.7871**	0.5696
Woodworking project	**0.6530**	0.3568	0.5241
Birthday party	0.7820	**0.9008**	0.9005
Changing tire	0.4770	0.5012	**0.6901**
Flash mob	0.9190	**0.9240**	0.8392
Vehicle unstuck	0.6910	0.6173	**0.9019**
Grooming animal	0.5100	0.5415	**0.6464**
Making sandwich	0.4190	0.5704	**0.6978**
Parade	0.7240	**0.7335**	0.5469
Parkour	**0.6640**	0.6144	0.5543
Repairing appliance	0.7820	**0.7840**	0.7329
Sewing project	0.5750	**0.6688**	0.5402
mean	0.6610	0.6639	**0.6730**

events for a qualitative evaluation. In this figure, we manually describe the video using the automatically obtained clauselet activations to show that clauselets are also useful for video event description as well as for event recognition. Note that false positives of 1^{st} level clauselet activations (e.g. *taking pictures* in an event *woodworking project*.) are removed by the 2^{nd} level clauselet.

Repairing appliance

1st level clauselet	Speaking. Speaking contains unscrewing parts. **Speaking is before holding objects.** **Speaking overlaps pointing to the object.** **Speaking start pointing to the object.** **Unscrewing parts finishes speaking.** Speaking starts unscrewing parts.
2nd level clauselet	**Speaking is before holding objects.** (*head clauselet*) **Speaking overlaps pointing to the object.** (type IV) **Speaking start pointing to the object.** (type II) **Unscrewing parts finishes speaking.** (type II)
Description	Person speaks while pointing to an object and then holds the object. After speaking, he unscrews parts from the obejct.

Flash mob

1st level clauselet	**Dancing in unison.** **Performing play.** Clapping. Lurching a pole. Reeling in is before holding objects. Playing instrument. **Yelling starts clapping.** **Dancing in unison is before clapping.**
2nd level clauselet	**Dancing in unison** (*head clauselet*) **Performing play.** (type II) **Yelling starts clapping.** (type II) **Dancing in unison is before clapping.** (type II)
Description	People dance in unison and then perform a play. Spectators yell and clap after watching their dance and performance.

Parade

1st level clauselet	**Marching.** **Squatting down.** Flipping the board. Singing in unison. **Marching contains playing instrument.** **Dancing in unison equals marching.** Fitting bolts. Marching equals playing instrument.
2nd level clauselet	**Marching** (*head clauselet*) **Marching.** (type II) **Squatting down.** (type II) **Marching contains playing instrument.** (type II) **Dancing in unison equals marching.** (type I)
Description	People dance in unison and march. Then they play instruments during marching. Someone squats down.

Woodworking project

1st level clauselet	**Cutting wood.** Taking pictures. **Speaking.** **Shaping wood.** Smoothing/sanding wood. Hammering. Speaking is before shaping wood. Speaking equals holding objects.
2nd level clauselet	**Cutting wood** (*head clauselet*) **Speaking.** (type III) **Shaping wood.** (type I) **Shaping wood.** (type II) **Shaping wood.** (type III) **Cutting wood.** (type I) **Cutting wood.** (type II) **Cutting wood.** (type III)
Description	Person cut and shape wood. Then speaking what he is doing.

Figure 4.6: Example activations of 1^{st} and 2^{nd} clauselets automatically detected for some events in TRECVID MED11 dataset. Video descriptions are manually written to emphasize the utility of clauselets for the description task (few additional words/phrases need to be added to form sentences from the detected clauselets). Bold activations in a list of 1^{st} level clauselet activations are used to rescore the *head clauselet* of each 2^{nd} level clauselet. In a list of 2^{nd} clauselet activations, a temporal relationship type of each concurrent activation toward *head clauselet* is depicted beside the activation. In the *parade* event, gray words denote the wrong description due to a false positive 2^{nd} level clauselet.

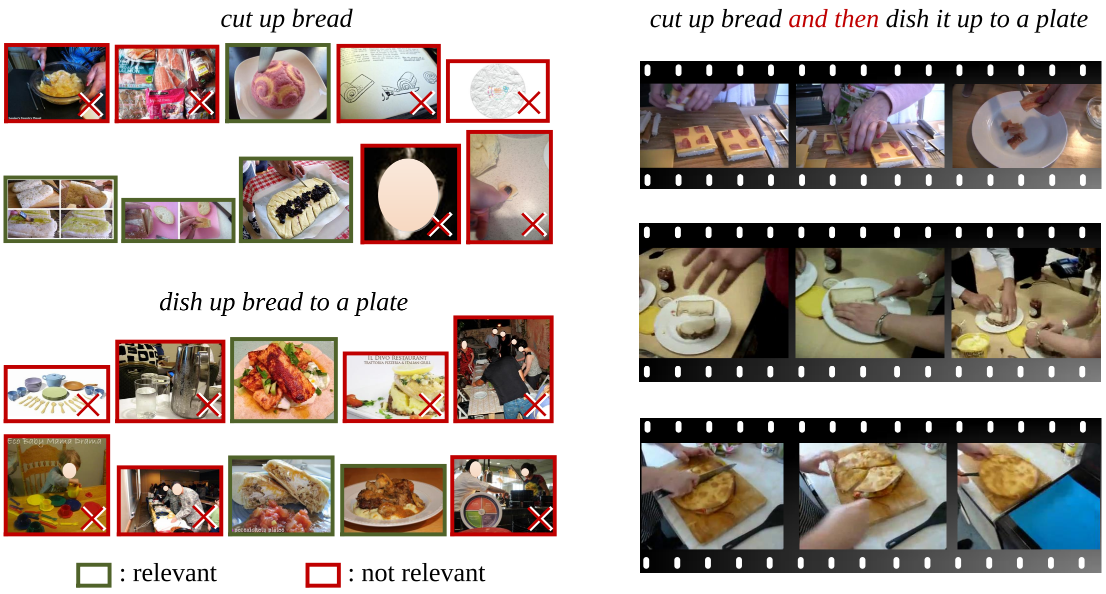
4.6 Conclusion

We proposed a new mid-level representation, a *clauselet*, that consists of a group of actions and their temporal relationships. We presented a training process that initially trains first level clauselets in a top-down fashion, and then learns more discriminative 2^{nd} level clauselets models using 1^{st} level activations that are consistent with each model and occur in particular temporal configurations. We have shown that the 2^{nd} level clauselets improve over the 1^{st} level clauselets, that they benefit from the automatic selection of which clauselets are "mutually consistent" (i.e., are assigned a non-zero weight in the model), that temporal relationships are important for both levels, and that our final model outperforms state-of-the-art recognition techniques on “in-the-wild” data when used in a simple voting scheme. Qualitative results show that clauselets are not only useful for event recognition, but the detected first and second level clauselets provide semantically meaningful descriptions of the video in terms of which actions occurred when with respect to each other.

Chapter 5: Learning Visual Clauses for Zero-shot Video Search

5.1 Introduction

The task of zero-shot learning has received increased attention recently in the machine learning and computer vision communities. The goal is to learn a classifier that can predict class labels for which data is not available at training time. This task is appealing due to the large number of objects, actions, events, and other visual categories in the natural world and due to their long-tail nature. It is well known, for example, that only a relatively few object categories, such as people and vehicles, have large numbers of example images that can be used to train detectors, while most other object categories have too few examples to sufficiently model their appearance by current approaches. Even when enough training examples are obtained (at great cost) and annotated (at even greater cost), training detectors involves significant computational resources, making zero-shot learning even more appealing. A common approach to zero-shot learning is to model visual categories by decomposing them into parts, attributes, or some other type of component that can be used to describe object classes without requiring visual examples for each class. This has been done for detecting animals by their attributes [67], actions by action components [68], and events by semantic concepts [69].



(a) Web domain

(b) Target domain

Figure 5.1: Demonstration of the mismatch between (a) the source domain (the web), in which we train our general phrase detectors and (b) the target domain where concepts (described by phrases) are in specific spatio-temporal configurations. While top ranked web search results for short phrases yield good training images, some incorrect results remain, e.g., the picture of a face for the phrase "cut up bread." Even when the image correctly matches a phrase, the image may be of a different meaning than intended for the target domain and query. Given a complex search query, our approach adapts the detectors trained on the web domain both to the target domain and to the specific search query to reduce the influence of incorrect training samples and training samples that are correct but not relevant to the search query.

Despite significant progress, general questions still remain: (1) what attributes, parts, or other components should be trained? (2) from what data source? (3) how will the training data be annotated? (4) will the data be sufficiently effective for modeling queries that are not yet known? The first question is important because we must have enough components trained to effectively model a large number of unknown classes. The second and third questions are important because it is difficult to obtain large training sets when the test domain is unknown, and it is even more difficult and costly to obtain annotations for training. And finally, the fourth question is important because machine learning approaches usually assume that training and testing data are obtained from the same distribution–in our setting; this is a problem because neither the target domain nor the set of labels are known.

Motivated by these questions, we study the task of zero-shot learning by video search using text based queries. Given a textual description of a video search query, we relate the search query to a set of pretrained visual concept detectors and rank videos from a test set according to how well they match the query. We address the first question (which attributes do we train) by assuming that we are provided a large set of generic concepts (objects, actions, scenes and attributes describing them) in the form of short text phrases, which we use to train generic visual detectors. If it is unreasonable to assume that such a large set is available a priori, our approach can also function by training phrase detectors on demand once the query is known. We address the source and annotation questions by leveraging the web–we use the known set of phrases as web search queries, and use the top-ranked image/video results for training. This not only ensures that we have access to an almost endless source of data, but we are also able to weakly associate the phrases as labels to the images or videos returned by the search, since top ranked results are relatively clean. Finally, we deal with the question of generalization by adapting the phrase detectors to the *unlabeled* target domain once the query is given, exploiting temporal and spatial patterns to adapt both to the target domain and to the target query. The goal of this step is to reduce the effect of incorrectly learned concepts (due to the weak labeling) and also of correctly learned concepts that are not relevant to the query at hand (see Figure 5.1).

The summary of our system is shown in Figure 5.2. A set of atomic phrases is trained ahead of time (or after the query is known–there is a trade-off between offline training cost and online training cost), which yields a set of general (potentially noisy) phrase detectors. In our current implementation, we train detectors for pairs of phrases to ensure that there is less ambiguity due to multiple meanings (see Figure 5.1), but these problems still remain. Once the query is provided by the user, we compute the score of each phrase pair present in the query on a dataset drawn from the target domain. Phrase detections are then partitioned into sets of probably positive and negative, which are then used to learn complex composite detectors we call “clauses” that model the spatial and temporal phrase coocurrence patterns. We then iterate between refining the phrase pair detectors to better detect clauses, and then defining new clauses that better fit the new phrase pairs. This process is based on our intuition that the intended meanings of phrases in a query are more likely to repeatedly occur in particular spatio-temporal arrangements with respect to each other compared to irrelevant or incorrect ones. Once this process is

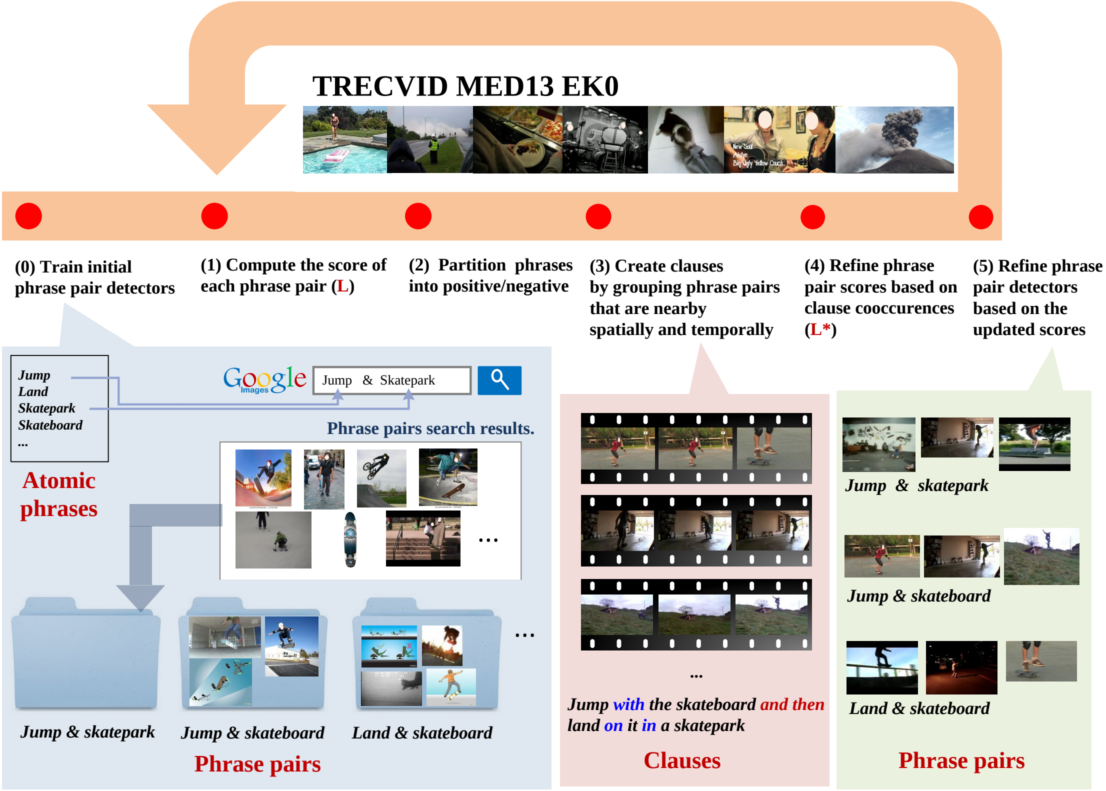

TRECVID MED13 EK0
(0) Train initial phrase pair detectors
(1) Compute the score of each phrase pair (L)
(2) Partition phrases into positive/negative
(3) Create clauses by grouping phrase pairs that are nearby spatially and temporally
(4) Refine phrase pair scores based on clause cooccurences (L*)
(5) Refine phrase pair detectors based on the updated scores
Jump
Land
Skatepark
Skateboard
...
Atomic phrases
Google Images
Jump & Skatepark
Phrase pairs search results.
...
Jump & skatepark
Jump & skateboard
Land & skateboard
Phrase pairs
Jump with the skateboard and then land on it in a skatepark
Clauses
Jump & skatepark
Jump & skateboard
Land & skateboard
Phrase pairs

Figure 5.2: Illustration of the training procedure. Given a set of atomic phrases, we first train detectors on top ranked search results for phrase pairs. Given an *unlabeled* target domain (EK0 in our experiments) and a textual search query, we adapt the initial phrase pair detectors to the query and target domain through an iterative process (steps 1 through 5). This process iterates between applying phrase detectors, grouping phrase detections into spatio-temporal groups we call "clauses", and adjusting the individual phrase detectors to better detect clauses, relying on spatio-temporal coocurrences to eliminate incorrect or irrelevant phrase meanings.

complete, we apply the clause detectors to the test set, and rank test videos based on a simple scheme where we count the number of clauses detected in each video.

Our contributions are that we:

1. automatically train a very large set of visual phrase detectors without the need for manual annotation
2. adapt the set of detectors to the target domain (if different from the source domain)
3. adapt the phrase detectors to the search query itself
4. exploit spatio-temporal coocurrences during the adaptation process

We demonstrate our approach on the TRECVID MED13 EK0 task [70], and compare to recent state-of-the-art techniques that rely on the fusion of multiple modalities (including visual, audio, text). We outperform the current baseline using just visual features alone, demonstrating the effectiveness of our approach.

5.2 Related Work

Zero-shot Learning: Lampert et al. [67] use semantic attributes for zero-shot object categorization using attribute classifiers learned from unrelated existing image datasets; it is assumed that novel objects are described in terms of these attributes. Elhoseiny et al. [71] exploit the correlation between textual descriptions of seen categories and their visual classifiers, and predict the visual classifier of an unseen category by comparing its textual description to the seen objects' descriptions.

Socher et al. [72] introduce a deep learning model differentiating on a mixture of seen and unseen classes simultaneously, with knowledge generated by unsupervised text corpora. These approaches commonly train evidence detectors in the same domain as the target and predict unseen classes based on the performance of the detectors. This requires types of annotations such as attributes labels [67] or seen category labels [71, 72]. Yu et al. [73] argue that designing informative attributes requires human effort and propose a formulation to automatically design discriminative attributes. Kankuekul et al. [74] propose self-organizing and incremental neural networks (SOINN) to learn new attributes and update existing attributes in an online incremental manner and develop a new framework to predict the unseen object by matching updated attributes relevant to the object. However, they also train attributes in a noisy domain (i.e. the potential for inconsistency with the target data exists) but unlike our approach, they do not adapt their detectors to the target domain.

Event Detection: The literature dealing with this topic is vast, so we narrow the range to the methods evaluated on the large scale, challenging TRECVID MED dataset [70]. Yang and Shah [75] propose an unsupervised approach to discover data-driven concepts from multi-modal signals (audio, scene, and motion) to describe high level semantics of videos. Ma et al. [51] leverage relevant attributes of video for event detection. Vahdat et al. [76] present a compositional model, multiple kernel learning (MKL) latent support vector machine (SVM), treating the locations of salient discriminative video segments as a latent variable. Yang et al. [77] utilize related exemplars which convey the precise semantic meaning of an event for complex event detection. Ramanathan et al. [19] employ human action and role recognition for solving the task. All these methods require a training set with annotated event labels. To eliminate the need for annotated training data, Jiang et al. [6] propose a MultiModel Pseudo Relevance Feedback (MMPRF) to select a few feedback videos, assigning assumed relevance judgements to them and ranking videos according to the statistics collected on them, repeatedly. While this method relies on multiple feature modalities (audio, video, text), it is most similar to our approach, so we compare to it as a baseline in our experiments. However, unlike their method directly applying a model to reselect training examples for updating the model, we refines the model with cooccurence information before reselecting the examples.

Exploiting Spatial and Temporal Relationships: Spatial or temporal relationships have received increased attention recently, especially for object recognition. For example, Felzenszwalb et al model spatial relationships between parts and the object as a whole [2], Bourdev et al. exploit part coocurrences (mutual context) to rescore part detections [20]. Niebles et al. [78] models activity as a complex temporal composition of simple actions. Sadeghi and Farhadi [79] and Desai and Ramanan [80] encode compositional models composing action, poses, and objects spatially related each other called phraselets (or visual phrases). Most similar to ours is NEIL (Never Ending Image Learner) a system proposed by Chen et al. [81]which uses the web to weakly label instances of visual categories, learning and exploiting their common sense relationships in the process. We additionally model temporal relationships (to model temporal events), and while the NEIL framework would fit very well with ours for modeling spatial relationships, we leave this for future work and resort to a simpler spatial model of coocurrence (within a frame), ignoring the spatial extents of objects for computational reasons. Unlike NEIL, we also address the problem of adapting to test queries and domains for zero-shot learning.

5.3 Learning Visual Clauses

Test queries will consist of textual descriptions involving observable evidence (scenes, actions, objects). Consequently, our observations will also need to be constructed from textual descriptions, so we use the terms *phrase* and *clause* to refer to representations of the video:

- *Phrases* consist of one or more short phrases from the textual description of an event, involving relevant objects, actions, and scenes. The textual description is split up into short phrases that we call *atomic phrases.*
- *Phrase pairs* are pairs of atomic phrases for which we train detectors. When detecting phrase pairs, we use the term *phrase activation* to denote a spatio-temporal window for which the phrase pair detector confidence passes a detection threshold.
- *Spatial phrase groups* are spatially coocurring phrase pairs.
- *Clauses* are groups of phrases that are spatio-temporally related each other through temporal relationships between spatial phrase groups. A *clause activation* is what we call a group of *phrase activations* that satisfy the temporal and spatial relationships of a clause.

In our current implementation, a training video is split into n clips, $t_1,\ t_2,\ \cdots,$ t_n, and each clip t is represented by a standard set of features concatenated into a feature vector $f(t)$. Phrase and clause detectors are applied to individual clips and subclips of videos, respectively. In this section, we will describe the process for modeling each detector, as well as the phrase detector refinement step based on clause cooccurence.

5.3.1 Training Initial Phrase Pair Detectors

Given a textual description for each event, the description is broken into short phrases (atomic phrases), and then every pair of atomic phrases will be used to train an associated detector[1]. Initial phrase pair detectors are trained by using web images. For each phrase pair, 50 images are downloaded via a web image-search engine (e.g. Google image, Bing, Flickr) by providing the phrase pair as the query and are used as positive examples; the images of the other phrase pairs will be used as negative examples. Initially we randomly select 500 negative examples and train the detector. Then we select hard negative examples by scanning the detector over the negative sets and collecting the top scoring 500 images and retrain the detector.

The phrase pair detector is trained by minimizing the reconstruction error. Let $\mathbf{X}$ be a matrix of n-dimensional feature vectors for N examples, i.e., $\mathbf{X} = [x_1 x_2 \cdots x_N] \in \mathbb{R}^{n \times N}$ and $\mathbf{Y}_i \in \{1, -1\}^N$ be the label vector of the i^{th} phrase pair detector. Labels of positive and negative examples are assigned as 1 and -1,

[1] We train detectors for pairs of phrases because we observed that web seach results are significantly better–though still noisy–if we search for phrase pairs instead of single phrases.

respectively. For each phrase pair detector, the model parameter $\mathbf{W}_{p,i}$ is obtained by minimizing the objective function consisting of the reconstruction error, $||\mathbf{Y}_i - \mathbf{W}_{p,i}^T\mathbf{X}||_F^2$ and a complexity term, $||\mathbf{W}_{p,i}||_F^2$ as below

$$\mathbf{W}_{p,i}^* = \arg\min_{\mathbf{W}_{p,i}} ||\mathbf{Y}_i - \mathbf{W}_{p,i}^T\mathbf{X}||_F^2 + \gamma||\mathbf{W}_{p,i}||_F^2, \tag{5.1}$$

where γ is a parameter to balance the label reconstruction error and complexity term. We solve equation 5.1 by setting its derivative with respect to the parameters to zero and obtain the optimal parameter $\mathbf{W}_{p,i}^*$ as

$$\mathbf{W}_{p,i}^* = (\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}\mathbf{X}\mathbf{Y}. \tag{5.2}$$

The reconstruction error formulation will allow us to relax labels to be continuous, enabling the adaptation process to adjust the magnitudes of the labels to be more or less positive or negative.

5.3.2 Training Clause Detectors

5.3.2.1 Model.

A clause detector models multiple phrases that are spatially or temporally related to each other. The clause detector models k clips c_i for $i = 1, 2, \cdots, k$, each of which has an associated weight vector w_i. A clause configuration T is a list of k clip indexes, one for each of the k clips. The score of a configuration is computed as follows:

$$s_T = \sum_{i=1}^{k} w_i^T f(t_{T(i)}). \tag{5.3}$$

The clause detector matches only valid configurations satisfying simple temporal deformation rules, i.e., $T \in \{T(i) | T(1) \in \{1, \cdots, n\},\ T(i-1) \leq T(i) \leq T(i-1)+2,\ i = 2,\ \cdots,\ k\}$. The configuration T becomes a clause activation if $s_T \geq \lambda_s$, where λ_s is the activation threshold.

5.3.2.2 Training.

1. Compute the score of each phrase pair detector: We scan the training dataset, compute scores, and generate the score matrix $\mathbf{L}_v \in \mathbb{R}^{F_v \times P}$, $v = 1, 2, \cdots, N_v$, where F_v, P and N_v are the number of clips of the v^{th} video, the number of phrase detectors, and the number of training videos, respectively.

2. Partition phrase pairs into positives/negatives: Based on the score matrix, we label each clip of the training video as a positive/negative example of the phrase pair by keeping the top k scoring detections for each phrase pair as positives and the rest as negatives.

3. Generate clauses from phrase pairs: Any combination of phrases related spatially and temporally to each other can be a clause candidate. Given a set of phrase pairs, we generate clauses hierarchically, first grouping phrase pairs related spatially into phrase groups and then using a sequence of propositional constraints to relate spatial phrase groups temporally (see Figure 5.3 for the sequence of propositional constraints on spatial phrase groups). For computational reasons, we place the following constraints on the groups of phrase pairs that can form a clause:

- Each clause models at most one temporal relationship between phrase groups (this means that a clause relates at most two phrase groups over a sequence of frames). Instead of considering all temporal relationships, we consider only loose versions of *before* and *during*.
- The number of spatial relationships per a phrase group varies from 2 to 4. We consider only cooccurence as spatial relationship (i.e., we ignore spatial extents for now).

After running the phrase pair detectors, we have the list of all phrase groups that coocur spatially in the training set. We then iterate over all unique phrase groups (and pairs of phrase groups), keeping only those that satisfy the above conditions. Each retained phrase group (or pair of phrase groups) when combined with one of the temporal templates in Figure 5.3, will generate a candidate clause and an associated template that encodes temporal relationships between phrase groups over a sequence of clips.

4. Collect positive and negative examples and train a clause detector: For each candidate clause, we collect positive and negative examples to train its detector weights from equation 5.3. The negative set consists of all videos that have

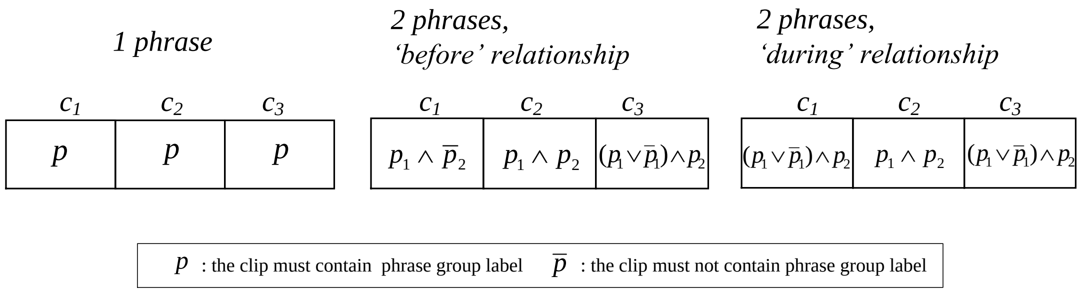

1 phrase
c_1
c_2
c_3
p
p
p
2 phrases,
'before' relationship
c_1
c_2
c_3
$p_1 \wedge \bar{p}_2$
$p_1 \wedge p_2$
$(p_1 \vee \bar{p}_1) \wedge p_2$
2 phrases,
'during' relationship
c_1
c_2
c_3
$(p_1 \vee \bar{p}_1) \wedge p_2$
$p_1 \wedge p_2$
$(p_1 \vee \bar{p}_1) \wedge p_2$
p : the clip must contain phrase group label
$\bar{p}$: the clip must not contain phrase group label

Figure 5.3: Temporal templates for clauses, Templates are used for searching positive examples by matching to labels: here we use clause detectors of length $k = 3$ clips.

no activations for phrase pairs from the query. For each clause, we identify clip sequences whose phrase activations satisfy the propositional constraints on spatial phrase groups (in Figure 5.3) and consider them as positive examples.

Once positives are obtained using the clause templates, we randomly select negative examples from the negative set as many as five times the size of the positive set and train a linear SVM classifier. We then scan over the negative videos, collect false positive activations, and retrain the linear SVM classifiers.

5.3.3 Refining Phrase Pair Scores

Clause detectors trained in the previous section can provide contextual and complementary information for updating phrase pair detectors. For example, the clause *jump with the board and then land on it* contains the contextual information that *land on the board* does not occur in the beginning of the event. Similarly, *jump* *with the board* does not end the event described by the clause. We add the contextual information provided by clause activations to the scores of phrase pair detectors $\mathbf{L}_v$ and retrain phrase pair detectors using the updated scores.

Specifically, we first detect clause activations in the training set. For an activation a of clause detector c, a configuration matrix $\mathbf{I}_c^{(a)} \in \mathbb{R}^{F_v \times k}$ indicating which clips are selected as part of the activation is constructed as follows:

$$\mathbf{I}_c^{(a)}(i,j) = \begin{cases} 1/\mathrm{S}(i) & \text{if clip index } i \text{ is selected as the } j^{th} \text{ clip of activation } a \\ & \qquad \text{of the clause detector } c \text{ (i.e., } T_c^{(a)}(j) = i) \\ 0 & \text{otherwise,} \end{cases}$$

where $S(i) = \sum_{c=1}^{C} \sum_{a=1}^{|A_{c,v}|} \sum_{j=1}^{G_c} \mathbf{1}(\mathbf{I}_c^{(a)}(i,j))$ is a normalization vector. C, $|A_{c,v}|$, and G_c are the number of the clause detectors, the number of activations of the clause detector c on the v^{th} video and, the number of phrase groups composing the clause, respectively.

Let $\mathbf{P}_c \in \mathbb{R}^{G_c \times k}$ denote the propositional constraints involving the clause c and be defined as:

$$\mathbf{P}_c(i,j) = \begin{cases} 1 & \text{if the } j^{th} \text{ clip of the clause } c \text{ contains } p_i \\ 0 & \text{if the } j^{th} \text{ clip of the clause } c \text{ contains } p_i \vee \ \bar{p}_i \\ -1 & \text{if the } j^{th} \text{ clip of the clause } c \text{ contains } \bar{p}_i. \end{cases}$$

For the clause c, a matrix $\mathbf{I}_{p,c} \in \mathbb{R}^{G_c \times P}$ indicating which phrase pairs involved in an individual phrase group of the clause is generated as below:

$$\mathbf{I}_{p,c}(i,j) = \begin{cases} 1 & \text{if } j^{th} \text{ phrase pair is involved in a group } p_i \text{ of the clause } c \\ 0 & \text{otherwise.} \end{cases}$$

The product of $\mathbf{P}_i$ and $\mathbf{I}_{p,i}$ is a matrix that encodes the contextual information from clause i projected onto phrase pair p. The intuition is that if phrase pair p appears in the clause and is not negated, its score becomes more positive; if it is negated, it becomes more negative, and remains unaffected otherwise. Let $\mathbf{L}_v^*$ denote the refined the score matrix $\mathbf{L}_v$ computed in the previous step and defined by scores of the clause activations, activated configuration, and the transition of the contextual information from the clause to the phrases as below:

$$\mathbf{L}_v^* = \mathbf{L}_v + \alpha \sum_{c=1}^{C} \sum_{a=1}^{|A_{c,v}|} s_c^{(a)} \mathbf{I}_c^{(a)} \mathbf{P}_c^T \mathbf{I}_{p,c}, \tag{5.4}$$

where $s_c^{(a)}$ is the score of activation a of clause detector c.

5.3.4 Refining Phrase Pair Detectors

To train each phrase pair detector, we use the top k clips according to their refined scores as the positive training examples. We also select negative examples from the negative training set as follows. First, we select negative examples randomly and train the detector. Then we scan the negative training set and collect hard negatives with the highest score as many as 10 times the number of positive samples and retrain the model.

We then employ the label reconstruction error optimization to obtain the phrase pair detectors. The label vector $\mathbf{Y}_i$ of the i^{th} phrase pair detector is set the refined scores for positive examples and -1 for negative examples.

5.4 Complex Event Detection

We use visual clauses for detecting complex events. For each event class, its description containing atomic phrases is given. Clauses and their detectors are defined and trained based on the relevant event's description; they are not affected by other event classes. We employ a simple voting strategy where each clause detector activation votes (equally) for its relevant event class. We count votes of all clause detectors from a class for each test video and its score is set to the number of votes.

5.5 Experiments

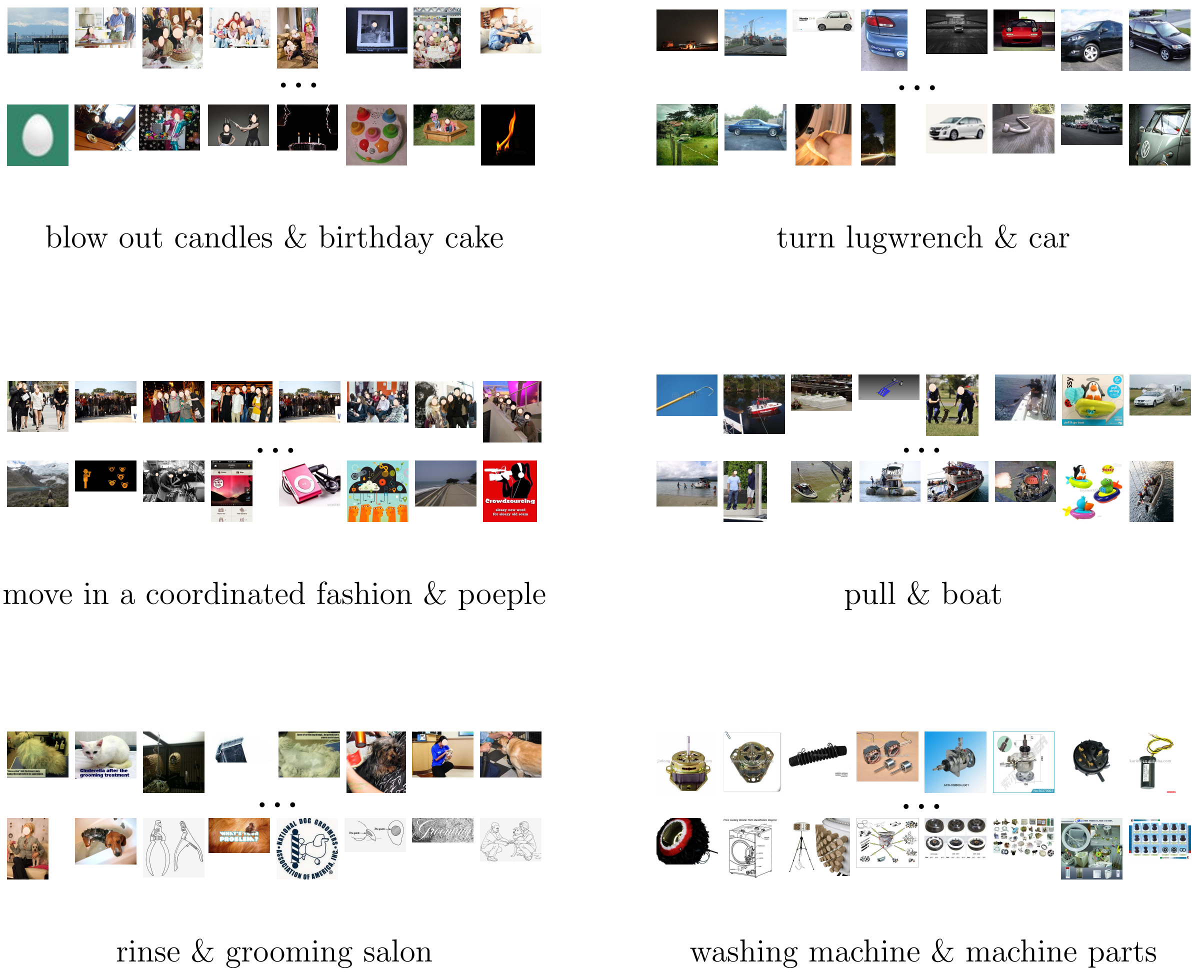
5.5.1 Dataset and Parameters

We evaluate the approach on the TRECVID MED 13 dataset [70] containing 20 complex events, half of which comes from previous challenges, MED11 [58] and MED12 [82], respectively. A MED event is a complex activity occurring at a specific place and time involving people interacting with other people and/or objects. Actions, objects/people, and scenes consisting of a MED event are loosely or tightly related temporally and spatially to the overarching activity. The MED13 event names and numbers of videos in the MED testset are listed in Table 5.1. For each event, a textual description listing the action, object, and scene that characterizes the event is provided.

Table 5.1: MED13 Evaluation Events

Events for MED13					
From MED11			From MED12		
ID	Events	#	ID	Events	#
E06	Birthday party	186	E21	Bike trick	16
E07	Changing a vehicle tire	111	E22	Cleaning an appliance	25
E08	Flash mob gathering	132	E23	Dog show	22
E09	Getting a vehicle unstuck	95	E24	Giving directions	32
E10	Grooming an animal	87	E25	Marriage proposal	33
E11	Making a sandwich	140	E26	Renovating a home	33
E12	Parade	233	E27	Rock climbing	18
E13	Parkour	104	E28	Town hall meeting	19
E14	Repairing an appliance	78	E29	Winning race without a vehicle	22
E15	Working on a sewing project	81	E30	Working on a metal crafts project	22

As a training set, various number of videos containing specified MED13 events are combined with a common set of background videos. There are three training sets referred to as EK100, EK10, and EK0 according to the number of example event videos that are provided for each query. EK0 consists of unlabeled background videos with no example event videos (the zero-shot learning task) and thus our model is trained only on the unlabeled background set. The test set is combined



blow out candles & birthday cake

turn lugwrench & car

move in a coordinated fashion & poeple

pull & boat

rinse & grooming salon

washing machine & machine parts

Figure 5.4: Improved ranking after adapting phrase pair detectors to a target domain and specific query. For each phrase pair, we show the initial web results reranked by the adapted phrase detector. The top rows and bottom rows show the highest and lowest scoring images, respectively. Note how the top scoring images much more closely match the phrase pair, and that incorrect or irrelevant meanings (bottom rows) are given low score.

with various other MED events videos (Table 5.1) and has approximately 23000 video clips.

Following the TRECVID MED protocol for EK0, which does not allow the background set to be annotated, we have three datasets: 1) the source dataset consists of web images weakly labeled through web search; 2) the target domain training set is the unlabeled EK0 background set; and 3) the MED test dataset is the test set.

Every video is represented by multiple key frames collected by selecting one frame per 10 second clip. We represent each clip by two image-based features: GIST [61] and SIFT [59]. 960 dimensional GIST feature represents an image globally and SIFT feature capture local image characteristics. For SIFT features, a histogram-based bag-of-visual words (BoVW) representation is generated using 4000 words. For training phrase pair detectors, we set the number of positive examples k to 100. We also set the threshold of clause detectors to 0 (i.e., we leave the default linear SVM decision threshold unmodified).

5.5.2 Qualitative Performance in Refining Phrase Pair Detectors

Figure 5.4 shows the web images sorted by the score given by the trained phrase pair detectors, after 5 iterations of adaptation. For each phrase pair, high scoring images are listed in the first row and low scored images are in the second row. Phrase pairs are indicated below their examples. In this section, we show the performance of the phrase pair label refinement qualitatively. Among phrase pairs in Figure 5.4, examples of *blow out candles & birthday cake*, *move in a coordinated fashion & people*, *rinse & grooming salon*, and *washing machine & machine parts*

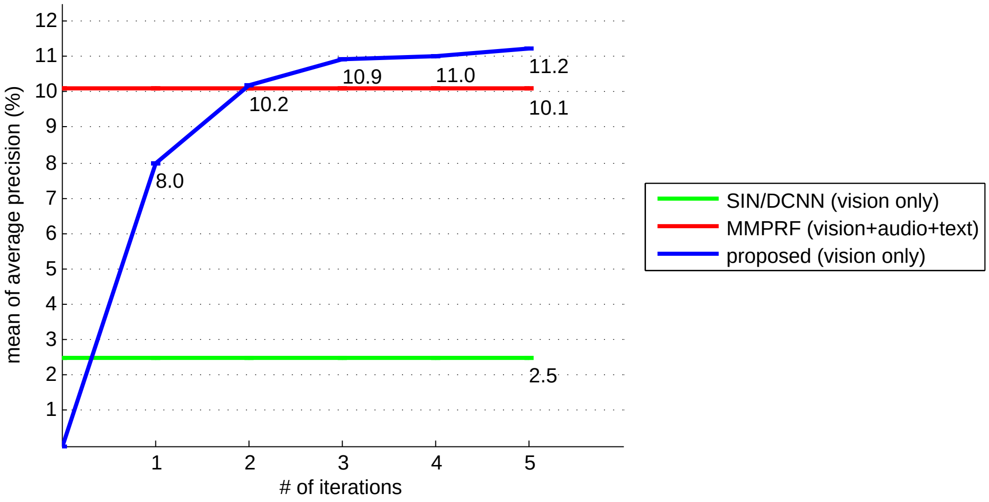


Figure 5.5: Mean of Average Precision on the TRECVID MED13 EK0 test set versus iterations of phrase and clause refinement. Improvements are large, especially in early iterations. We outperform the results reported in [6] are shown as straight lines. We use only visual features, while the best reported baseline uses a fusion of visual, audio, and text (OCR) features. For reference, we also show the performance reported by [6] for SIN/DCNN, the visual features used by MMPRF.

have an intuitive ordering: examples that are more relevant to the query and to the TRECVID MED dataset have higher rank and other noisy or irrelevant examples have lower rank. The refinement over the other two phrase pairs does not performed as expected. Only the car appears in all images in *turn lugwrench & car* and the failure is likely due to incorrect web search results. However, in *pull & boat*, images in the second row look like they contain the phrase pair correctly and seem more appropriate for the search task.

5.5.3 Quantitative Performance for Complex Event Detection

We compare our method to Jiang et al. [6] who also train their models in the TRECVID MED EK0 training setting. We use mean of average precision (mAP) as a metric to evaluate the performance in the complex event recognition, the standard for the TRECVID MED evaluation. We evaluated the clause detectors on the video search task every iteration up to the 5^{th} and show the results in Figure 5.5. The performance increases in each iteration and the proposed approach outperforms the baseline after the 2^{nd} iteration. The process converged after the 3^{rd} iteration.

Table 5.2 shows the performances of clause detectors trained after the 1^{st} and 5^{th} iteration on individual events. We can see that the performance increases for every event by at least 1.2%. Note that the variance of the performance in each class is likely related to the number of event video contained in the test set (table 5.1). For the latter 10 events, the number of video is relatively small compared to the first 10 events; their detection performance is lower as well. Table 5.3 compares the performance of the baseline [6] and our approaches. Note that the baseline approach, MMPRF, uses visual, audio and text features to achieve their results, while we use only visual features. Our approach significantly outperforms the individual feature performance reported by [6]: mAP of 5.33 for audio, 7.63 for text (OCR), and 2.50 for vision. While we use GIST and SIFT as visual features and [6] uses Semantic Indexing (SIN) and Deep Convolutional Neural Network (DCNN), the

Table 5.2: Mean Average Precision (mAP) on TRECVID MED13 EK0 pre-specified task, by event.

ID	E06	E07	E08	E09	E10	E11	E12	E13	E14	E15
itr 1	7.72	6.93	10.95	5.76	8.81	12.15	15.55	8.72	5.14	4.59
itr 5	15.59	11.00	14.54	12.62	9.56	13.53	18.53	10.28	15.39	7.01
ID	E21	E22	E23	E24	E25	E26	E27	E28	E29	E30
itr 1	6.68	5.30	7.69	7.95	8.62	8.44	7.33	5.27	8.68	7.40
itr 5	10.93	6.51	10.67	13.73	9.44	11.47	9.40	6.40	9.69	8.47

large difference in performance when restricted to visual features alone is notable. We also apply the initially trained phrase pair detectors to the task, to further evaluate their utility for this task without any other machinery. The performance gap (approx. 9.6% to that of clause detectors after 5^{th} iteration) shows how noisy the phrase pair detectors are before they are adapted to the target domain and query.

5.6 Conclusion

We demonstrated an approach to zero-shot learning of complex visual events using visual phrases learned from weak annotations automatically obtained from the web. These visual phrase detectors are noisy, and may not necessarily encode the intended meaning of a text phrase as used in a search query. In addition, it is

Table 5.3: Mean Average Precision (mAP) comparison with the baseline methods.

Method	mAP
SIN/DCNN (vision only) [6]	2.5
MMPRF [6]	10.1
Phrase Pairs	1.6
Clauses (itr 1)	8.0
Clauses (itr 5)	**11.2**

possible that the training data and test data are not from the same domain (e.g., training data could be images and test data consists of videos). For this reason, we adapt the trained visual phrases both to the search query and to the target dataset by exploiting spatio-temporal groups of visual phrases that we call visual clauses. Our experiments show that our approach successfully reduces the effect of incorrect or irrelevant training data, and outperforms state-of-the-art approaches that use audio and text (OCR) approaches in addition to visual features.

Chapter 6: Conclusion

The thesis aims to understand ”in-the-wild” videos such as YouTube. To describe the video, we study various recognition tasks such as action, pose etc. and generate event descriptions.

For action recognition, we present a qualitative pose estimation approach that is based on discriminative deformable part models and developed a robust pose feature based on this approach. Unlike previous approaches, we give special attention to the selection of part models, replacing random selection and greedy cover steps with an automatic clustering of part poses. The pose feature is suitable for use in action recognition tasks involving relatively unconstrained videos. We have shown that various modifications of the poselet training process improve the representation power of the set of poselets, generating a set of features that can be seamlessly combined with existing shape and motion features.

For complex event analysis, we proposed a new mid-level representation, a *clauselet*, that consists of a group of actions and their temporal relationships. We presented a training process that initially trains first level clauselets in a top-down fashion, and then learns more discriminative 2^{nd} level clauselets models using 1^{st} level activations that are consistent with each model and occur in particular temporal configurations. We have shown that the 2^{nd} level clauselets improve over the 1^{st} level clauselets, that they benefit from the automatic selection of which clauselets are "mutually consistent" (i.e., are assigned a non-zero weight in the model), that temporal relationships are important for both levels, and that our final model outperforms state-of-the-art recognition techniques on "in-the-wild" data when used in a simple voting scheme. We also demonstrated an approach to zero-shot learning of complex visual events using visual phrases learned from weak annotations automatically obtained from the web. These visual phrase detectors are noisy, and may not necessarily encode the intended meaning of a text phrase as used in a search query. In addition, it is possible that the training data and test data are not from the same domain (e.g., training data could be images and test data consists of videos). For this reason, we adapt the trained visual phrases both to the search query and to the target dataset by exploiting spatio-temporal groups of visual phrases that we call visual clauses.

Bibliography

[1] M. D. Rodriguez, J. Ahmed, and M. Shah. Action mach: A spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008.

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

[3] Subhransu Maji, Lubomir Bourdev, and Jitendra Malik. Action recognition from a distributed representation of pose and appearance. In *CVPR*, 2011.

[4] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *CVPR*, 2009.

[5] J. F. Allen and G. Ferguson. Actions and events in interval temporal logic. *Jour. of Logic and Computation*, 1994.

[6] L. Jiang, T. Mitamura, S. Yu, and A. G. Hauptmann. Zero-example event search using multimodal pseudo relevance feedback. In *ICMR*, 2014.

[7] I. Laptev and T. Lindeberg. Space-time interest points. In *ICCV*, 2003.

[8] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *ECCV*, 2008.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[10] Zhuolin Jiang, Zhe Lin, and Larry S. Davis. Recognizing human actions by learning and matching shape-motion prototype trees. *PAMI*, 34(3), 2012.

[11] Michael Hofmann and Dariu M. Gavrila. Multi-view 3d human pose estimation combining single-frame recovery, temporal integration and model adaptation. In *CVPR*, 2009.

[12] Xiaolin K. Wei and Jinxiang Chai. Modeling 3d human poses from uncalibrated monocular images. In *ICCV*, 2009.

[13] Peng Guan, Alexander Weiss, Alexandru O. Balan, and Michael J. Black. Estimating human shape and pose from a single image. In *ICCV*, 2009.

[14] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Monocular 3d pose estimation and tracking by detection. In *CVPR*, 2010.

[15] Ben Daubney and Xianghua Xie. Tracking 3d human pose with large root node uncertainty. In *CVPR*, 2011.

[16] Juergen Gall, Angela Yao, and Luc Val Gool. 2d action recognition serves 3d human pose estimation. In *ECCV*, 2010.

[17] Catalin Lonescu, Fuxin Li, and Cristian Sminchisescu. Latent structured models for human pose estimation. In *ICCV*, 2011.

[18] C. Chen, A. Heili, and J. Odobez. Combined estimation of location and body pose in surveillance video. In *AVSS*, 2011.

[19] Vignesh Ramanathan, Percy Liang, and Li Fei-Fei. Video event understanding using natural language descriptions. In *ICCV*, 2013.

[20] Lubomir Bourdev, Subhransu Maji, Thomas Brox, and Jitendra Malik. Detecting people using mutually consistent poselet activations. In *ECCV*, 2010.

[21] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. In *IEEE Transactions on Computer*, volume 22, 1973.

[22] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. In *International Journal of Computer Vision (IJCV)*, pages 55–79, 2005.

[23] Vivek Kumar Singh and Ram Nevatia. Action recognition in cluttered dynamic scenes using pose-specific part models. In *ICCV*, 2011.

[24] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011.

[25] Ben Sapp, Chris Jordan, and Ben Tasker. Adaptive pose priors for pictorial structures. In *CVPR*, 2010.

[26] Vivek Kumar Singh, Ram Nevatia, and Chang Huang. Efficient inference with multiple heterogeneous part detectors for human pose estimation. In *ECCV*, 2010.

[27] Benjamin Sapp, Alexander Toshev, and Ben Taskar. Cascaded models for articulated pose estimation. In *ECCV*, 2010.

[28] Marcin Eichner and Vittorio Ferrari. We are family: Joint pose estimation of multiple persons. In *ECCV*, 2010.

[29] Chunhui Gu and Xiaofeng Ren. Discriminative mixture-of-template for viewpoint classification. In *ECCV*, 2010.

[30] Lubomir Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009.

[31] Michael Maire, Stella X. Yu, and Pietro Perona. Object detection and segmentation from joint embedding of parts and pixels. In *ICCV*, 2011.

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

[33] R. Farrell, O. Oza, N. Zhang, V.I. Morariu, T. Darrell, and L.S. Davis. Subordinate categorization using volumetric primitives and pose-normalized appearance. In *ICCV*, 2011.

[34] Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Describing people: Poselet-based approach to attribute classification. In *ICCV*, 2011.

[35] Yang Wang, Duan Tran, and Zicheng Liao. Learning hierarchical poselets for human parsing. In *CVPR*, 2011.

[36] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lari Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *ICCV*, 2011.

[37] Yuelei Xie, Hong Chang, Zhe Li, Luhong Liang, Xilin Chen, and Debin Zhao. A unified framework for locating and recognizing human actions. In *CVPR*, 2011.

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

[39] Sinisa Todorovic. Human activities as stochastic kronecker graphs. In *ECCV*, 2012.

[40] Sreemanananth Sadanand and Jason J. Corso. Action bank: A high-level representation of activity in video. In *CVPR*, 2012.

[41] Hyungtae Lee, Vlad I. Morariu, and Larry S. Davis. Qualitative pose estimation by discriminative deformable part models. In *ACCV*, 2012.

[42] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Representing shape with a spatial pyramid kernel. In *CIVR*, 2007.

[43] P. Dollar, V. Raboud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatiotemporal features. In *VS-PETS*, 2005.

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

[45] A. Kovashka and K. Grauman. Learning a hierarchy of discriminative spacetime neighborhood features for human action recognition. In *CVPR*, 2010.

[46] X. Wu, D. Xu, L. Duan, and J. Luo. Action recognition using context and appearance distribution features. In *CVPR*, 2011.

[47] Stephen O'Hara and Bruce A. Draper. Scalable action recognition with a subspace forest. In *CVPR*, 2012.

[48] Yimeng Zhang, Xiaoming Liu, Ming-Ching Chang, Weina Ge, and Tsuhan Chen. Spatio-temporal phrases for activity recognition. In *ECCV*, 2012.

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

[50] Haid Izadinia and Mubarak Shah. Recognizing complex events using large margin joint low-level event model. In *ECCV*, 2012.

[51] Zhigang Ma, Yi Yang, Zhongwen Xu, Shuicheng Yan, Nicu Sebe, and Alexander G. Hauptmann. Complex event detection via multi-source video attributes. In *CVPR*, 2013.

[52] M. Amer and S. Todorvic. A chains model for localizing group activities in videos. In *ICCV*, 2011.

[53] Z. Zeng and Q. Ji. Knowledge based activity recognition with dynamic bayesian network. In *ECCV*, 2010.

[54] Kevin Tang, Bangpeng Yao, Li Fei-Fei, and Daphne Koller. Combining the right features for complex event recognition. In *ICCV*, 2013.

[55] Kevin Tang, Li Fei-Fei, and Daphne Koller. Learning latent temporal structure for complex event detection. In *CVPR*, 2012.

[56] Mohamed R. Amer and Simisa Todorovic. Sub-product networs for modeling activities with stochastic structure. In *CVPR*, 2012.

[57] Vlad I. Morariu and Larry S. Davis. Multi-agent event recognition in structured scenarios. In *CVPR*, 2011.

[58] Trecvid multimedia event detection track, 2011.

[59] D.G. Lowe. Distinctive image featrues from scale-invariant keypoints. *IJCV*, 60(2), 2004.

[60] I. Laptev. On space time interest points. *IJCV*, 2005.

[61] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42, 2001.

[62] Arpit Jain, Abhinav Gupta, Mikel Rodriquez, and Larry S. Davis. Representing videos using mid-level discriminative patches. In *CVPR*, 2013.

[63] Yale Song, Louis-Philippe Morency, and Randell Davis. Action recognition by hierarchical sequence summarization. In *CVPR*, 2013.

[64] Ram Nevatia, Tao Zhao, and Somboon Hongeng. Hierarchical language-based representation of events in video streams. In *CVPR*, 2003.

[65] William Brendel, Alan Fern, and Sinisa Todorovic. Probabilistic event logic for interval-based event recognition. In *CVPR*, 2011.

[66] L. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[67] C. H. Lampert, H. Nichisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *PAMI*, 2013.

[68] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *CVPR*, 2011.

[69] J. Liu, Q. Yu, O. Javed, S. Ali, A. Tamrakar, A. Divakaran, H. Cheng, and H. Sawhney. Video event recognition using concept attributes. In *WACV*, 2013.

[70] Trecvid multimedia event detection track 2013, 2013.

[71] M. Elhoseiny, B. Saleh, and A. Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013.

[72] R. Socher, M. Ganjoo, C. D. Manning, and A. Y. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013.

[73] F. X. Yu, L. Cao, R. S. Feris, J. R. Smith, and S. Chang. Designing category-level attributes for discriminative visual recognition. In *CVPR*, 2013.

[74] P. Kankuekul, A. Kawewong, S. Tangruamsub, and O. Hasegawa. Online incremental attribute-based zero-shot learning. In *CVPR*, 2012.

[75] Y. Yang and M. Shah. Complex events detection using data-driven concepts. In *ECCV*, 2012.

[76] A. Vahdat, K. Cannons, G. Mori, S. Oh, and I. Kim. Compositional models for video event detection: A multiple kernel learning latent variable approach. In *ICCV*, 2013.

[77] Y. Yang, Z. Ma, Z. Xu, S. Yan, and A. G. Hauptmann. How related examplars help complex event detection in web videos. In *ICCV*, 2013.

[78] J. C. Niebles, Chen C., and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010.

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

[80] Chaitanya Desai and Deva Ramanan. Detecting actions, poses, and objects with relational phraselets. In *ECCV*, 2012.

[81] X. Chen, A. Shrivastava, and A. Gupta. Neil: Extracting visual knowledge from web data. In *ICCV*, 2013.

[82] Trecvid multimedia event detection track 2012, 2012.