THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences et Lettres
PSL Research University

Préparée à l'École Normale Supérieure

Structured modeling and recognition of human actions in video

École doctorale n°386

SCIENCES MATHÉMATIQUES DE PARIS CENTRE

Spécialité INFORMATIQUE

COMPOSITION DU JURY :

M Patrick Pérez
Valeo AI
Rapporteur

M Cees G. M. Snoek
University of Amsterdam
Rapporteur

M Ivan Laptev
Inria
Directeur de thèse

Mme Cordelia Schmid
Inria
Directrice de thèse

M Jürgen Gall
University of Bonn
Président du jury

M Jean Ponce
Inria
Membre du jury

Soutenue par **Guilhem Chéron**
le 17 décembre 2018

ENS | PSL

Résumé

La compréhension automatique de vidéos devrait impacter notre vie de tous les jours dans de nombreux domaines comme la conduite autonome, les robots domestiques, la recherche et le filtrage de contenu, les jeux vidéo, la défense ou la sécurité. Le nombre de vidéos croît plus vite chaque année, notamment sur les plateformes telles que YouTube, Twitter ou Facebook. L'analyse automatique de ces données est indispensable pour permettre à de nouvelles applications de voir le jour.

L'analyse vidéo, en particulier en environnement non contrôlé, se heurte à plusieurs problèmes comme la variabilité intra-classe (les échantillons d'un même concept paraissent très différents) ou la confusion inter-classe (les exemples provenant de deux activités distinctes se ressemblent). Bien que ces difficultés puissent être traitées via des algorithmes d'apprentissage supervisé, les méthodes pleinement supervisées sont souvent synonymes d'un coût d'annotation élevé. Dépendant à la fois de la tâche à effectuer et du niveau de supervision requis, la quantité d'annotations nécessaire peut être prohibitive. Dans le cas de la localisation d'actions, une approche pleinement supervisée nécessite les boîtes englobantes de l'acteur à chaque image où l'action est effectuée. Le coût associé à l'obtention d'un telle annotation empêche le passage à l'échelle et limite le nombre d'échantillons d'entraînement. Trouver un consensus entre les annotateurs est également difficile et mène à des ambiguïtés dans l'étiquetage (*Où commence l'action ? Quand se termine-t-elle ? Que doit inclure la boîte englobante ?* etc.).

Cette thèse adresse les problèmes évoqués ci-dessus dans le contexte de deux tâches, la *classification* et la *localisation* d'actions humaines. La *classification* consiste à reconnaître l'activité effectuée dans une courte vidéo limitée à la durée de l'action. La *localisation* a pour but de détecter en temps et dans l'espace des activités effectuées dans de plus longues vidéos. Notre approche pour la classification d'actions tire parti de l'information contenue dans la posture humaine et l'intègre avec des descripteurs d'apparence et de mouvement afin d'améliorer les performances. Notre approche pour la localisation d'actions modélise l'évolution temporelle des actions à l'aide d'un réseau récurrent entraîné à partir de suivis de personnes. Enfin, la troisième méthode étudiée dans cette thèse a pour but de contourner le coût prohibitif des annotations de vidéos et utilise le regroupement discriminatoire pour analyser et combiner différents types de supervision.

Abstract

Automatic video understanding is expected to impact our lives through many applications such as autonomous driving, domestic robots, content search and filtering, gaming, defense or security. Video content is growing faster each year, for example on platforms such as YouTube, Twitter or Facebook. Automatic analysis of this data is required to enable future applications.

Video analysis, especially in uncontrolled environments, presents several difficulties such as intra-class variability (samples from the same concept appear very differently) or inter-class confusion (examples from two different activities look similar). While these problems can be addressed with the supervised learning algorithms, fully-supervised methods are often associated with high annotation cost. Depending on both the task and the level of required supervision, the annotation can be prohibitive. For example, in action localization, a fully-supervised approach demands person bounding boxes to be annotated at every frames where an activity is performed. The cost of getting such annotation prohibits scalability and limits the number of training samples. Another issue is finding a consensus between annotators, which leads to labeling ambiguities (*where does the action start? where does it end? what should be included in the bounding box? etc.*).

This thesis addresses above problems in the context of two tasks, namely human action *classification* and *localization*. The former aims at recognizing the type of activity performed in a short video clip trimmed to the temporal extent of the action. The latter additionally extracts the space-time locations of potentially multiple activities in much longer videos. Our approach to action classification leverages information from human pose and integrates it with appearance and motion descriptors for improved performance. Our approach to action localization models the temporal evolution of actions in the video with a recurrent network trained on the level of person tracks. Finally, the third method in this thesis aims to avoid a prohibitive cost of video annotation and adopts discriminative clustering to analyze and combine different levels of supervision.

Remerciements

Je tiens tout d'abord à remercier mes encadrants Ivan et Cordelia pour m'avoir donné cette opportunité de travailler au sein de ces deux équipes exceptionnelles : Willow et Thoth. Merci pour votre disponibilité et vos conseils avisés durant ma thèse. Vous avez su me transmettre votre passion pour la recherche, la vision et l'apprentissage. Ivan, je n'oublierai jamais nos moments passés devant CMT à uploader l'ultime version de nos papiers jusqu'à la dernière minute. Cordelia, merci de m'avoir fait profiter de ta grande expérience et de tes intuitions qui m'ont beaucoup aidé durant ces années.

Je remercie mes rapporteurs, Patrick Pérez et Cees Snoek, pour avoir accepté d'examiner mon manuscrit et Jürgen Gall pour avoir présidé mon jury. J'adresse un merci particulier à Jean en tant que membre du jury et directeur de l'équipe mais aussi pour tous ces joyeux moments passés ensemble.

Je remercie Andrew Fitzgibbon de m'avoir accueilli par deux fois chez Microsoft Research à Cambridge. Grâce à ta bonne humeur et ton enthousiasme, ce fut tous les jours un plaisir de travailler avec toi. Merci également à Laurent Massoulié sans qui ces visites à Cambridge n'auraient pas été possibles.

Ces années de thèse n'auraient pas été les mêmes sans mes formidables collègues et amis. Matthew et Julia, malgré la charge de travail, j'ai adoré nos soirées *deadline* au C314. Matthew, mon partenaire de sport et de soirées à Bastille, nous avions tant de projets, nous les réaliserons plus tard j'en suis sûr. Julia, grâce à ton soutien et à ta bonne humeur qui n'a pas fait défaut un seul instant, ce fut un bonheur quotidien et sincère de partager ce bureau avec toi. Mes amis Christophe et Maxime, j'ai passé d'excellentes pauses café à vos côtés avant que vous ne m'abandonniez. Jean-Baptiste, j'ai énormément apprécié travailler (et m'amuser) sur ce dernier projet avec toi. Merci à toi Margaux pour ta visite et nos discussions du vendredi. Merci à tous les autres membres du labo d'avoir créé un environnement aussi agréable, au travail, aux conférences ou aux *friday drinks* : Relja Arandjelovic, Sylvain Arlot, Mathieu Aubry, Dmitry Babichev, Francis Bach, Raphael Berthier, Piotr Bojanowski, Nicolas Boumal, Andrei Bursuc, Lénaic Chizat, Minsu Cho, Igor Colin, Théophile Dalens, Alexandre d'Aspremont, Alexandre Defossez, Vincent Delaitre, Aymeric Dieuleveut, Mihai Dusmanu, Thomas Eboli, Loïc Esteve, Nicolas Flammarion, Fajwel Fogel, Pierre Gaillard, Damien Garreau, Pascal Germain, Gauthier Gidel, Robert Gower, Petr Gronát, Yana Hasson, Bumsub Ham, Bastien Jacquet, Igor Kalevatykh, Vadim Kantorov, Thomas Kerdreux, Sesh Kumar, Vijay Kumar, Suha Kwak, Yann Labbé, Simon Lacoste-Julien, Rémi Lajugie, Loic Landrieu, Rémi Leblond, Zongmian Li, Antoine Miech, Guillaume Obozinski, Anton Osokin, Dmitrii Ostrovskii, Fabian Pedregosa, Loucas Pillaud Vivien,

Anastasia Podosinnikova, Antoine Recanati, Rafael Rezende, Ronan Riochet, Ignacio Rocco, Vincent Roulet, Allessandro Rudi, Damien Scieur, Guillaume Seguin, Tatiana Shpakova, Josef Sivic, Robin Strudel, Adrien Taylor, Federico Vaggi, Gül Varol, Tuan-Hung Vu, Sergey Zagoruyko and Dmitry Zhukov. Merci à mes autres collègues de l'Inria, Marion, Hélène R. et Hélène B., Emma, Anaïs, Morgane et Alain.

Je tiens à présent à remercier mes amis rennais qui à chaque retour me faisaient vivre cette ambiance bretonne si authentique. Merci notamment au groupe des plus sages, Estelle, Jérémy, David, Jude, Julien, Antoine, Maxime, Delphine, Marie B., Tiphaine et Julia, à celui des moins sages, Jamon, Noé, Maxime, Guirec, Océ, Mathieu, Ophé, Jérémy, Claire, Sami, Maria, Tam, Kiki, Guibs, Chloé, Romain, Aubin, Beyblade, Marie D., Marina, Marion, Caro, Élysa et à tous les autres. Merci à Anais et Manon pour nos soirées sur le balcon de la coloc à la fin de ma thèse. Merci à mes amis parisiens : Hugo, Adrien, Gary, Dora, Mégane, Fanny, David, Julie J., Élise, Gaétan, Nicolas, Mathilde, Sébastien, Julie H., Charlotte, Rémi, François, Daniel, Jauna, Ju, Po et les autres. Je destine une mention parisienne spéciale à Emine pour son soutien et nos soirées à refaire le monde.

Merci à l'ensemble de ma famille pour leur gentillesse. Je tiens à remercier tout particulièrement mes grands-parents, Mady, Annie et André, qui se sont fait du souci pour ma santé mentale quand les heures de travail s'accumulaient et pour leur accueil pendant une partie de la rédaction. Merci au reste de ma famille et notamment à Arnaud, Olivier, Joël, Gene, Guillaume, Margot, Patou et Julien.

Un grand merci à mes parents pour leur générosité, leur patience, et leur soutien inestimables. Merci pour votre aide, au rendez-vous depuis toujours et sans laquelle je ne serai jamais arrivé jusqu'ici. Je vous en suis profondément reconnaissant.

Je finis par toi, François, mon meilleur ami qui m'a tant appris depuis nos 3 mois d'existence. Tu as été mon modèle depuis la pataugeoire jusqu'à aujourd'hui, ne change rien. Je te dois tant. En guise de maigre retour, je te dédie cette thèse, à toi, ainsi qu'à mon grand-père, Christian, qui aurait été fier de moi.

Contents

Chapter 1

Introduction

1.1 Motivation

The need of automatic video analysis is growing together with the amount of video data. For example, in 2018, more than one billion people are using YouTube, they represent one third of the Internet users and watch one billion hours of video each day[1]. The growth of digital content is driven by new devices such as smartphones, action cameras (*e.g.*, GoPro) and tablets, allowing to upload 400 hours of videos every minute on YouTube. YouTube alone has the second internet traffic after Google[2]. Also, Twitter tells us that 82% of its users watch video content on its website[3] while Cisco predicts that video will represents 82% of the Internet traffic in 2021[4].

Tools for analyzing this enormous amount of data will have a large impact on our everyday life and will create new opportunities in a wide range of applications. Among them, analyzing the content of online platforms is essential to filter undesired videos

[1] Youtube statistics, https://www.youtube.com/intl/en-GB/yt/about/press/

[2] Top 500 websites on Alexa, https://www.alexa.com/topsites

[3] Twitter, https://blog.twitter.com/marketing/en_us/a/2015/new-research%2Dtwitter-users-love-to-watch-discover-and-engage-with-video.html

[4] Cisco, https://www.cisco.com/c/en/us/solutions/service-provider/visual%2Dnetworking-index-vni/index.html

(parental controls, terrorist propaganda, fake news, *etc.*) and to enable movie narration (*e.g.*, generating audio description for blind people). Moreover, video indexing is necessary to build search engines and organize personal video collections. But video analysis is also a crucial topic in many other areas such as robotics. Indeed, for enabling robots helping us at home – looking after kids, cooking, washing – it will be necessary to design smart assistants capable of understanding their environments. Autonomous vehicles (cars, trucks, drones, submarines, *etc.*) will be useful in many fields and the real-time and reliable video analysis will be required to avoid car accidents. Video analysis also becomes increasingly important in human-machine interaction, notably in gaming (*e.g.*, Kinect) and is used in many devices (*e.g.*, Hololens, Google glass). Moreover, industries have an increasing need for controlling quality of goods and detecting possible anomalies in production chains. Finally, event detection will be crucial in security and video surveillance to localize suspicious behaviors.

Computers start to extend human capacities and open up new perspectives for the future. Indeed, it is not possible for humans to fully exploit the huge amount of video data without automatic methods. For example, machines have recently attained competence approaching that of top-performing humans in face recognition and it has been shown that maximum accuracy was obtained by a human-machine collaboration[5]. Important progress in many other domains can be achieved by exploiting computer abilities. The different components of video analysis, such as action recognition, person detection, tracking, pose estimation, retrieval, restoration, *etc.*, will be improved. The hope is that, all together, these components will allow new possibilities that people could not achieve, enabling for example computers to be better and safer driver than humans. To make this possible, it is necessary and exciting to develop new algorithms for video analysis.

[5]Face recognition at NIST, https://www.nist.gov/news-events/news/2018/05/nist-study-shows-face-recognition-experts-perform-better-ai-partner

Finally, while Laptev [2013] shows that 35% of pixels from the content we are used to share, watch and produce belong to people, this thesis focuses on analyzing *human-based* video content by designing algorithms to accomplish this task automatically.

1.2 Goal

Figure 1.1: Typical actions encountered in classification datasets. Each column respectively corresponds to HMDB-51 [Kuehne et al., 2011], UCF-101 [Soomro et al., 2012], Kinetics [Kay et al., 2017] and MPII Cooking Activities [Rohrbach et al., 2012] datasets. Class names from left to right, top to bottom: *clap*, *pull-up*, *pizza tossing*, *punch*, *brush painting*, *wrapping present*, *wash hands* and *squeeze*.

Action classification and *action localization* are required for video analysis as people behaviors generally constitute an essential component for understanding a scene. It is therefore not surprising that they belong to the widely studied topics in the computer vision community. As it will be presented in Chapter 2, the literature devoted to these tasks is substantial. In this thesis, our high-level goal is to classify and localize human actions in real-world videos.

In Chapter 4, we target action *classification*. This task consists in assigning videos to a closed set of class labels. The input videos are short clips temporally trimmed to the

Figure 1.2: Spatio-temporal action localization consists in three sub-tasks: recognizing the action (*drinking*), finding the temporal extent (one horizontal bar for each of the 3 instances) and spatially localizing the actor (person bounding boxes). Examples from the DALY [Weinzaepfel et al., 2016] dataset.

action extent. In Figure 1.1, we illustrate typical action instances encountered in recent real-world video datasets, for example, they can be sport-oriented or come from daily activities or gestures.

Classifying actions such as *drinking*, *diving*, or *playing harmonica* is not as challenging as localizing them in space and time. This harder task is addressed by spatio-temporal action *localization*, or *detection*. It analyzes longer videos – potentially containing several instances belonging to multiple classes – and aims at finding the temporal extent of actions, as well as the corresponding spatial location of their actors. We provide an example of detecting three instances of the "drinking" action in a video depicted in Figure 1.2. The action localization task is addressed in Chapter 5 and Chapter 6.

The three main goals of this thesis can be summarized as follow:

- studying how to use human pose for action classification (Chapter 4);
- performing fully-supervised spatio-temporal action localization (Chapter 5);
- making use of weak supervision for spatio-temporal action localization (Chapter 6).

Action recognition models aim at extracting meanings from digital content (pixel values) by detecting repetitive patterns. However, video samples are too complex and prone to

high variability preventing manual adjustment of model parameters. To address this issue, *machine learning* provides algorithmic solutions capable of extracting generalization from data. To detect these patterns and build numerical representation of a concept, we have to give several samples as input of the learning method. In general, the more data the algorithm processes the better it generalizes. The methods we introduce in this thesis are all based on such learning approach. When designing models, some structure can be imposed by integrating additional information. For example, this can be achieved *e.g.*, using either human pose as considered in Chapter 4, or person tracks as in Chapter 5.

In order to train a model, supervised methods require examples with corresponding annotations. Collecting video datasets typically requires a large amount of time-consuming and tedious manual annotation of videos. Figure 1.3 illustrates a typical annotation scenario for collecting action recognition datasets on *Amazon Mechanical Turk* (AMT) where the annotator, or *worker*, has to answer whether the video contains an instance of a specific action. Action labels can be entered by the annotator or automatically proposed *e.g.* from the video title. Table 1.1 provides statistics on some recent action classification datasets. In Chapter 4, we aim at exploiting this type of annotation for performing action classification.

Spatio-temporal action localization datasets contain richer supervision. Indeed, in addition to the action class, annotators need to annotate each action with its temporal interval together with its spatial extent represented by a rectangle, or *bounding box*, around the actor. In Chapter 5, we design models for action localization based on such annotation.

Finally, when composing a dense labeling used in fully supervised settings, finding the exact temporal boundaries as well as drawing bounding boxes in all frames can be extremely tedious. Fortunately, weakly-supervised learning can deal with weaker levels of supervision during training. By relying on such techniques, the last goal of this thesis – presented in Chapter 6 – is to compare different levels of supervision in the context

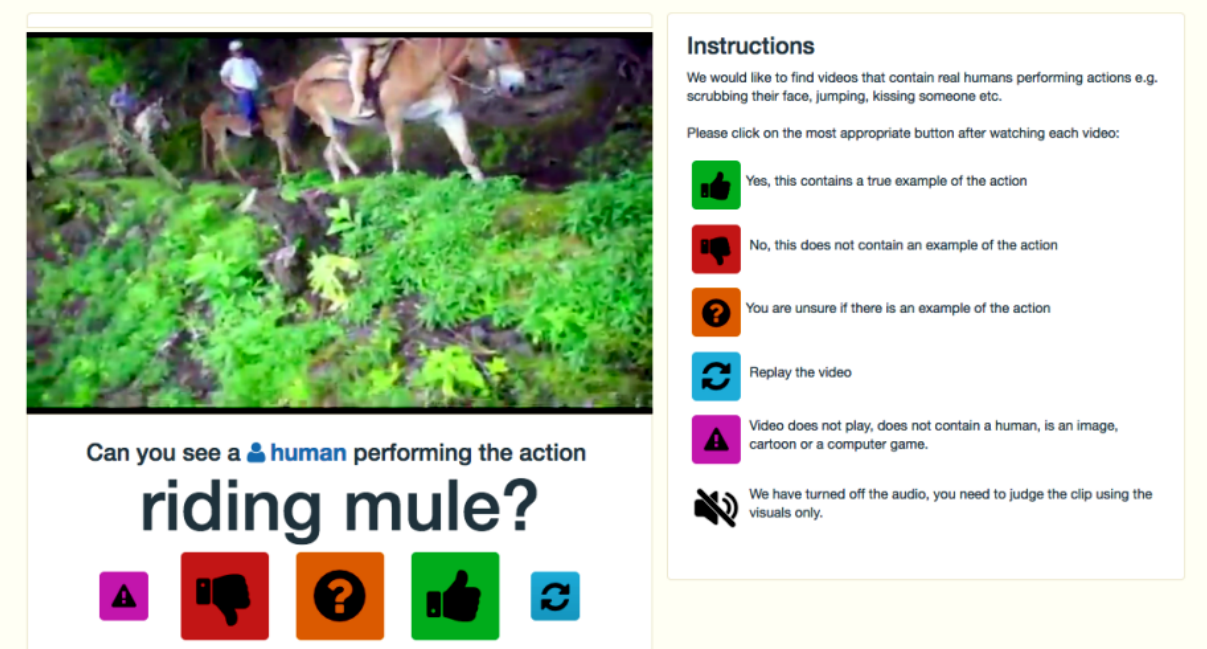


Figure 1.3: One way to collect annotation is to use an Amazon Mechanical Turk (AMT) labeling interface. This example comes from [Kay et al., 2017] and illustrates annotation of the Kinetics video dataset for action classification. A video clip is presented to an annotator who has to answer whether a given action, here *riding a mule*, is performed, or not. This proposed class name either come from the YouTube video title or by running an existing classifier.

of action detection. Being able to exploit weak supervision and understanding what information is truly necessary are essential for reducing the annotation effort as it comes with several issues. Indeed, dense labeling is problematic since its high cost limits the number of training examples. We address this issue in Chapter 6.

1.3 Challenges

The difficulties underlying automatic video analysis are related to several challenges. One of the main issue come from the fact that two scenes depicting the same concept can appear very differently. This *intra-class variability* is driven by many factors especially in uncontrolled video environments. It includes recording conditions like: view point,

Dataset	Year	Classes	Clips per class	Total clips
HMDB-51 [Kuehne et al., 2011]	2011	51	min. 102	6,766
UCF-101 [Soomro et al., 2012]	2012	101	min. 101	13,320
ActivityNet-200 [Heilbron et al., 2015]	2015	200	avg. 141	28,108
Kinetics [Kay et al., 2017]	2017	400	min. 400	306,245

Table 1.1: Statistics for recent human action classification datasets.

Figure 1.4: There is more than one way to "walk a dog". Intra-class variability comes from different execution styles, scalings, dog appearances, involved objects, person/dog spatial relations, *etc.* Images come from the Unrel dataset [Peyre et al., 2017].

occlusion, motion blur, lighting, camera motion or acquisition noise. But also subject or object attributes (scale, shape, color, texture, dress, *etc.*) and scene (background clutter, indoor/outdoor, crowded/not crowded, *etc.*). Additionally, performing an action with different styles or duration of execution also largely affects the appearance of the scene. For examples, Figure 1.4 shows that there is more than one way to "walk a dog". Video sources (TV show, amateur, news, movie, report) also influence how we perceive a scene. Finally, arbitrary decisions made in problem specification (*e.g.* level of labeling granularity) modify sample appearance inside a given class. Indeed, in a scenario where the goal is to recover instances of a high-level action like "riding", "riding a horse" candidates look very different from "riding a bike" ones.

Inter-class confusion also affects video understanding due to class similarities. A difficult example from fine-grained action recognition is to differentiate cooking activities "slicing" and "dicing". In action localization even less obvious similarities may lead to confusing classes such as "brushing teeth" with "drinking" as hands appear to be close to the mouth in both cases.

Interpretation of dynamic scenes may be ambiguous when observing only a single frame. Indeed, several frames would be necessary to answer the questions in Figure 1.5 as it is hard to explain from a single image what is going on with motion information being removed. These observations have motivated the modeling of dynamic cues and have especially made *optical flow* usage more and more popular in action recognition.

Even though significant progress has been achieved in video analysis recently, its associated tasks are far from being solved and many problems remain. For example, the best action localization performance reported on the challenging DALY dataset only reaches 31.0%@0.3 mAP [Chéron et al., 2018b]. To improve results, taking into account the challenges presented so far is necessary. Hence, we aim at designing models that can handle different actions taking visual variability into account and address the following challenges.

Figure 1.5: Looking only at a single frame is often ambiguous for interpreting dynamic scenes. Is the man sitting up or down? Is the photo going to be shot or was it already taken? Is the lady getting in or out of the car? Those pictures respectively come from HMDB-51 [Kuehne et al., 2011], DALY [Weinzaepfel et al., 2016] and Hollywood [Laptev et al., 2008] datasets. Answers: the man is sitting down, the photo was already taken and the lady is getting out of the car.

In order to improve action recognition performance by correcting the lack of structure in the data, we use human pose to "organize" features coming from different body parts (Chapter 4). In spatio-temporal localization, the temporal extent detected with sliding window or dynamic programming approaches is often very approximative. In the same

context, we propose (Chapter 5) a more accurate solution and training strategy while designing and comparing different model architectures.

One last challenge comes from fully supervised methods which necessitate a large amount of annotations. Such dense labeling is usually extremely tedious to obtain and reducing the required annotation effort is a part of our goal in this thesis. The annotation becomes particularly problematic in the case of spatio-temporal action localization where one needs to provide annotations in each frame. Besides that the full manual video annotation does not scale to many samples and classes, finding a consensus among annotator is also complicated – *e.g.* , does the *brushing teeth* action starts when the toothbrush is in actor's hand or mouth? when does it end? – and leads to arbitrary decisions in the definition of actions. Additionally, the high cost of dense labeling drastically limits the number of training samples preventing from potentially large performance gain. In Chapter 6, we introduce a flexible model capable of leveraging information from different levels of supervision, and of potentially mixing them. It also gives insights on which annotation is necessary to train methods for action localization.

1.4 Contributions and outline

This thesis first addresses the action classification task [Chéron et al., 2015b] (Chapter 4) and the localization problem [Chéron et al., 2018b] (Chapter 5), both in a fully-supervised setting, *i.e.*, when models are trained based on a set of videos where dense annotation is provided. Finally, we study the possibility of using less supervision (*weakly-supervised* setting) and the ability of mixing different levels of labeling for the task of action localization [Chéron et al., 2018a] (Chapter 6). The fully supervised models introduced in this thesis outperform the state of the art. The weakly supervised approach achieves comparable performance while using less manual annotation. We summarize our contributions

in more details below.

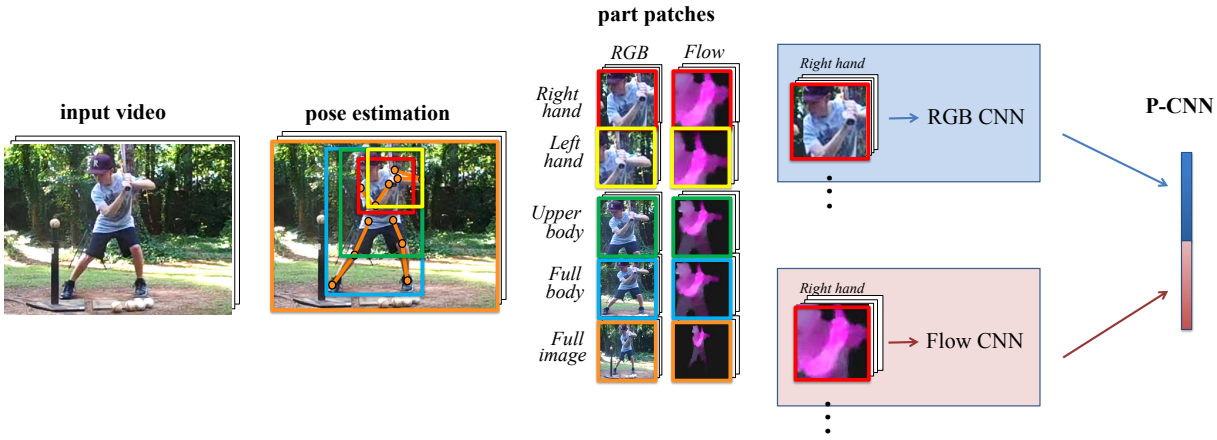


Figure 1.6: Our P-CNN features for action classification. From left to right: Input video. Human pose and corresponding human body parts for one frame of the video. Patches of appearance (RGB) and optical flow for human body parts. One RGB and one flow CNN descriptor is extracted per frame and per part (an example is shown for the human body part *right hand*) and aggregated over time. Video descriptors are concatenated over parts and streams (appearance and flow).

P-CNN: Pose-based CNN Features for Action Recognition

Here we address fully-supervised action classification in trimmed video clips. While previous methods have typically represented actions by statistics of local video features, here we argue for the importance of a representation derived from human pose. To this end, we propose a new pose-based convolutional neural network descriptor (P-CNN) for action recognition. The descriptor aggregates motion and appearance information along tracks of human body parts (Figure 1.6). We investigate different schemes of temporal aggregation and experiment with P-CNN features obtained both for automatically estimated and manually annotated human poses. We evaluate our method on the JHMDB and MPII Cooking datasets. For both datasets, our method shows consistent improvement over the state of the art at the time of publication. This work has been published in ICCV'15 [Chéron et al., 2015b] and is presented in Chapter 4.

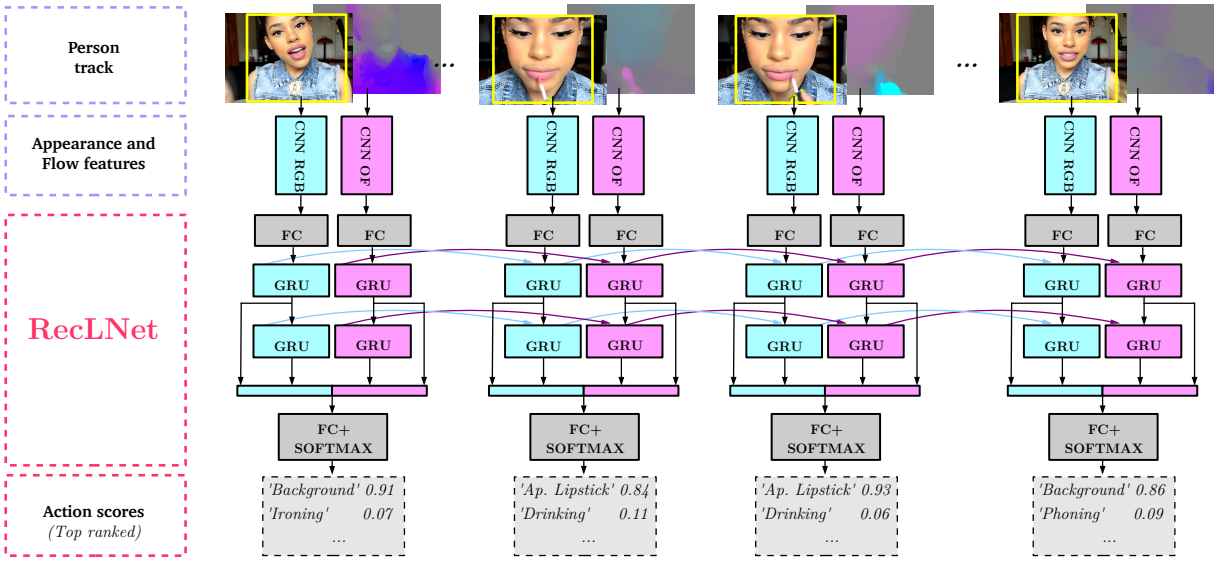


Figure 1.7: Our RecLNet approach for spatio-temporal action localization. The input is a person track where for each spatially localized actor bounding-box, we extract appearance (RGB) and optical flow (OF) CNN features. Each stream is processed by our recurrent network. Outputs are class probabilities for each frame.

Modeling Spatio-Temporal Human Track Structure for Action Localization

We address spatio-temporal localization of human actions in the video. Compared to the work in Chapter 4, the input videos are no longer trimmed around the target actions but represent longer dynamic scenes where the goal is to localize both the temporal interval and the spatial location of the action. At train time, we are given the temporal extent of action instances and the corresponding person bounding boxes in this interval. To localize actions in time, we propose a recurrent localization network (RecLNet) designed to model the temporal structure of actions on the level of person tracks. Our model is trained to simultaneously recognize and localize action classes in time and is based on two layer gated recurrent units (GRU) applied separately to two streams, *i.e.* appearance and optical flow streams (Figure 1.7). When used together with state-of-the-art person detection and tracking, our model is shown to improve substantially spatio-temporal action

localization in videos by better temporal localization. We evaluate our method on recent datasets for spatio-temporal action localization, UCF101-24 and DALY, and demonstrate a significant improvement of the state of the art. This work is under submission [Chéron et al., 2018b] and is presented in Chapter 5.

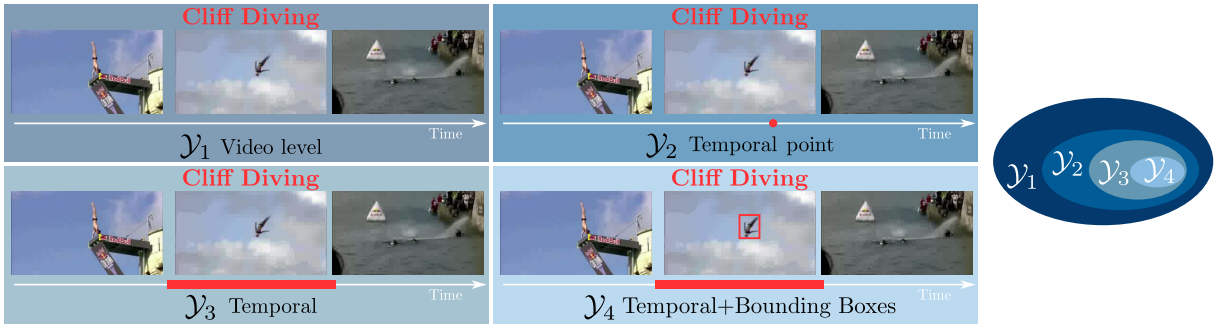


Figure 1.8: Varying levels of supervision handled by our flexible method. Our method estimates a matrix $Y \in \mathcal{Y}_s$ assigning human tracklets to action labels. Different types of supervision define particular constraints $\mathcal{Y}_s$. The increasing level of supervision imposes stricter constraints, e.g. $\mathcal{Y}_1 \supset \mathcal{Y}_2 \supset \mathcal{Y}_3 \supset \mathcal{Y}_4$ as illustrated for the Cliff Diving example above.

A flexible model for training action localization with varying levels of supervision

Spatio-temporal action detection in videos is typically addressed in a fully-supervised setup with manual annotation of training videos required at every frame. Since such annotation is extremely tedious and prohibits scalability, there is a clear need to minimize the amount of manual supervision. We propose a unifying approach that can handle and combine varying types of less-demanding *weak supervision*. Our model is based on discriminative clustering and integrates different types of supervision as constraints on the optimization. We investigate applications of such a model to training setups with alternative supervisory signals ranging from video-level class labels over temporal points or sparse action bounding boxes to the full per-frame annotation of action bounding boxes

(Figure 1.8). Experiments on the challenging UCF101-24 and DALY datasets demonstrate competitive performance of our method at a fraction of supervision used by previous methods. The flexibility of our model enables joint learning from data with different levels of annotation. Experimental results demonstrate a significant gain by adding a few fully supervised examples to otherwise weakly labeled videos. This work has been published in NIPS'18 [Chéron et al., 2018a] and is presented in Chapter 6.

Chapter 2

Related work

In 1878, Eadweard Muybridge's work, which might be the first "video" comprehension attempt, has illustrated that a video can be seen as a sequence of frames. Among all his studies on motion, a famous one aims to answer the question whether all four feet of a horse are off the ground at the same time while galloping. The movement of a galloping horse is too fast for a human eye to answer this question. To resolve the riddle, Muybridge took a sequence of pictures from 12 different cameras (depicted in Figure 2.1) and was able to resolve the question with a positive answer. Beyond the original goal, this experiment has highlighted the importance of video analysis and its close relation to the study of static images. When presenting modern methods for action classification and localization below, we try to make links with the closely related techniques for still image analysis. Indeed, we will see that video comprehension models have been largely shaped by the latter. In the rest of this chapter we first review methods for action classification in Section 2.1 and then discuss literature on localization in Section 2.2.

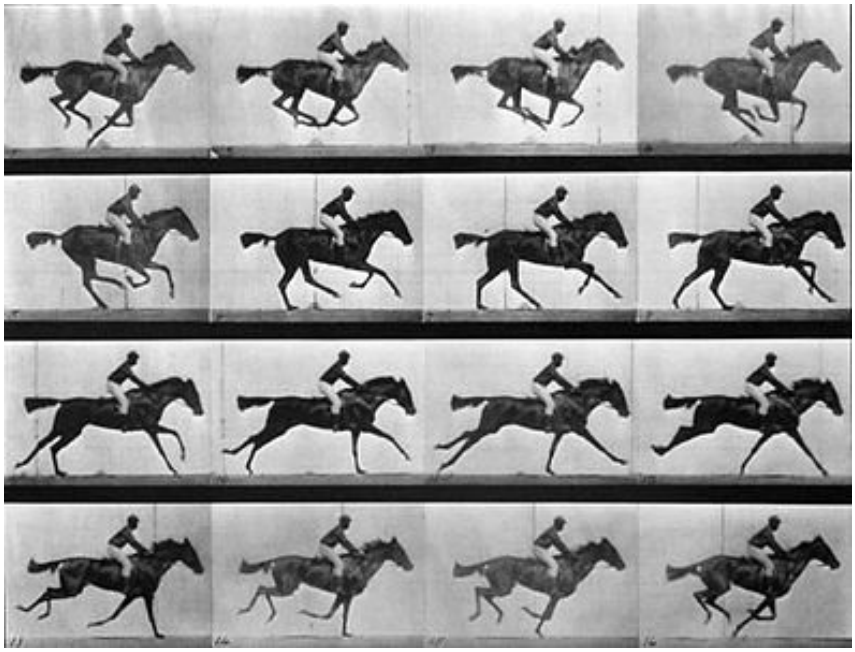


Figure 2.1: Eadweard Muybridge's study of horse movement might be the first work on "video" analysis.

2.1 Action classification

Among all computer vision areas related to video analysis, action classification is certainly one of the most studied topic. As introduced in Section 1.2, action classification aims to assign videos with labels from a closed set of action classes. While any type of video could be considered, current methods and datasets typically focus on short video clips trimmed to the temporal boundaries of human actions. In this thesis we only consider recognition of actions performed by *people* and not by animals as in [Xu et al., 2015]. In the following, we introduce classical methods developed for this task and discuss their relation to the methods of static image classification. The literature review covers methods using local descriptors (Section 2.1.1), convolutional neural networks (Section 2.1.2) and person-specific cues (Section 2.1.3).

2.1.1 Local descriptors

Action recognition in the last decade has been dominated by methods involving local features [Laptev et al., 2008; Schuldt et al., 2004; Wang and Schmid, 2013]. Such approaches usually take advantage of local space-time features originally built for still images. Indeed, among the wide range of methods developed by the computer vision community for still image comprehension, designing local feature representation is certainly one of the most important aspect that influenced works on video recognition.

In the still image domain Schmid and Mohr [1997] proposed to represent an image by local descriptors extracted at specific regions (aka. interest points). Such locations can be obtained, for example, with a corner detector [Harris and Stephens, 1988] which takes advantage of the auto-correlation function in order to catch positions where the image signal significantly changes in two directions. One key property of such points is the *repeatability*, *e.g.*, in the task of matching a pair of images, most of the interest points should be detected in both images (see Figure 2.2). This idea has been reused in video analysis when Laptev [2005] has adapted the corner detector of Harris and Stephens [1988] to detect spatio-temporal interest points (see Figure 2.3).

To locally describe regions of interest, geometrically-invariant representations have been studied in [Mikolajczyk and Schmid, 2005]. Based on the histogram of image gradient, the scale-invariant feature transform (SIFT) [Lowe, 1999] is certainly one of the most influential local descriptors for still images. This work has inspired other frequently used representations in computer vision, *e.g.*, the histograms of oriented gradients (HOG) [Dalal and Triggs, 2005] and the speeded-up robust features (SURF) [Bay et al., 2006]. For action recognition, Laptev and Lindeberg [2004] have used histograms of gradients and flow to describe video regions. This work has been extended to compensate for extra camera motions in [Laptev et al., 2007] and has inspired methods introduced

Figure 2.2: Interest point detection (red dots) in a pair of images with different lighting condition and view point.

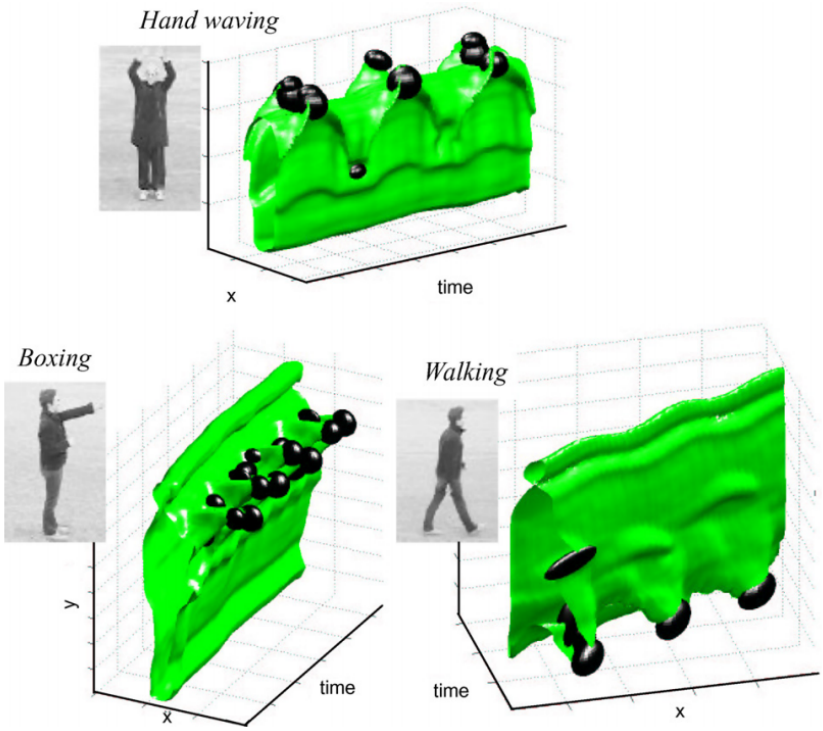


Figure 2.3: Examples of local motion event detection [Laptev, 2005]. The illustrations from [Laptev et al., 2007] show one image from the image sequence and a level surface of image brightness over space–time. The events are illustrated as dark ellipsoids.

in [Laptev and Pérez, 2007; Laptev et al., 2008]. Another approach, notably used in scene classification or face recognition, consists in performing a dense extraction of such representations by replacing *interest points* by a regular grid [Fei-Fei and Perona, 2005; Lazebnik et al., 2006; Simonyan et al., 2013]. In video, Klaser et al. [2008] have extended the widely used HOG image descriptor to a 3D version in order to take the temporal dimension into account.

Along with the spatial descriptors developed for still images, motion provides important additional information in videos. It can be estimated by *optical flow* algorithms providing the vertical and horizontal point displacements for consecutive frames. A basic assumption for the computation optical flow is the constancy of the brightness along the displacement [Horn and Schunck, 1981; Lucas et al., 1981]. Over the years, many approaches have been proposed to extract optical flow [Brox et al., 2004; Weinzaepfel et al., 2013; Zach et al., 2007]. Transforming optical flow output into video descriptors has achieved initial progress for action classification. Notably, in the manner of HOG features using histograms of quantized gradients, Laptev et al. [2008] have introduced the histogram of optical flow (HOF). Also, motion boundary histograms (MBH) [Dalal et al., 2006; Wang et al., 2013], based on differential optical flow, have been developed to cancel constant motion such as camera translation, and to emphasize motion boundaries.

Local descriptor aggregation into image statistics was decisive for good performance. Adapted from information retrieval in natural language processing, the bag-of-feature model was widely used to represent the distribution of local image descriptors [Cula and Dana, 2001; Csurka et al., 2004; Sivic and Zisserman, 2003]. It consists in first defining a set (or vocabulary) of *visual words* by clustering image features. Then, local descriptors are assigned to their closets visual word. Vocabulary occurrences are quantized into an histogram serving as final image representation. Commonly used representations such as vector of aggregated local descriptors (VLAD) [Jegou et al., 2012] and Fisher

vector [Perronnin et al., 2010b] can be seen as an extension of the bag-of-feature model involving a soft assignment of image descriptors. Finally, a classifier is learned on top of these representations, typically using support vector machines (SVM) [Cortes and Vapnik, 1995]. While such methods are used both in image and video analysis (it first appeared in [Schuldt et al., 2004] for action recognition), other feature quantization techniques have been specifically designed for videos. For example, Laptev et al. [2008] extend the work of Lazebnik et al. [2006] and impose some time order. To this purpose, they define a spatio-temporal pyramid by computing aggregations for several video volumes. Similarly, Gaidon et al. [2011] model an action as a sequence of *actoms* and concatenate their bag-of-feature representations keeping the temporal structure of action instances. Finally, Peng et al. [2014] and Taralova et al. [2014] pool the features over supervoxels.

To summarize and illustrate the importance of local descriptors in action classification, we now present an interesting example still used nowadays in video analysis. This method is called *dense trajectory* (DT) [Wang et al., 2011] and has shown excellent performance in several action recognition benchmarks [Oneata et al., 2013; Wang et al., 2013]. At several scales, the method first densely samples points which are tracked to the next frame using optical flow. Similar to points of interest extraction presented previously, dynamic information is kept while static trajectories are pruned based on the autocorrelation matrix. This leads to a good compromise between saliency and density. For each remaining trajectory, several descriptors are extracted in the aligned spatio-temporal volume including HOG, HOF and MBH. This is summarized in Figure 2.4. While several adaptations of this model have been made, Wang and Schmid [2013] have notably introduced the improved dense trajectory (IDT). In addition to the DT approach, it computes a homography supported by optical flow and SURF point matches pruning outliers with random sample consensus (RANSAC) [Fischler and Bolles, 1981]. Then, it removes trajectories consistent with the camera motion and warps optical flow according to the estimated homography before

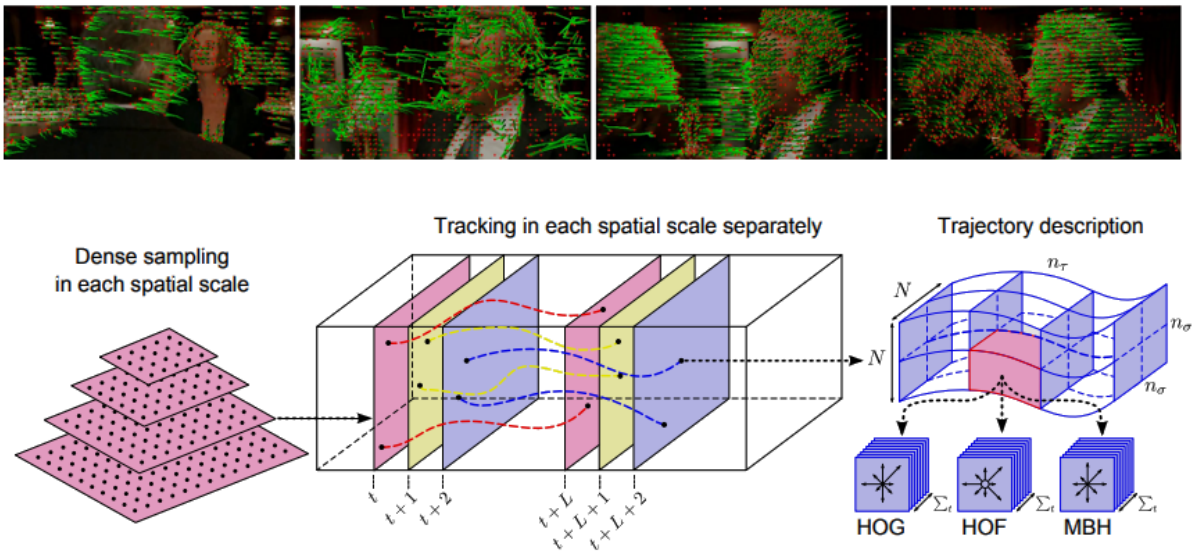


Figure 2.4: The dense trajectory approach [Wang et al., 2011, 2013].
Top: Examples of the resulting *trajectories* on a "kissing" action. Red dots indicate the point positions in the current frame.
Bottom: The trajectories are obtained by dense sampling and tracking at several scales. Feature extraction (HOG, HOF, MBH) is performed along trajectories.

flow descriptor computation. In the manner of still image feature quantization, IDT combined with Fisher vector aggregation [Perronnin et al., 2010b] was dominating the action recognition field [Oneata et al., 2013]. In our work of Chapter 4, we notably use IDT with Fisher vector encoding as a strong baseline and experimentally demonstrate its complementarity to our method. To draw the parallel between videos and images, we note that the dense trajectory pipeline (Figure 2.4) is closely related to a typical still image analysis approach summarized in Figure 2.5. Indeed, they both contain all the important steps presented in this section. Namely, they start by defining locations (interest points or dense grid) before extracting local descriptors, then aggregate feature representations and finally perform classification. In the manner of IDT which outperformed other methods in action classification, such procedure based on local descriptors has also dominated the still image field during several years for many tasks like face recognition [Simonyan et al., 2013], image classification [Perronnin et al., 2010b] or retrieval [Perronnin et al., 2010a].

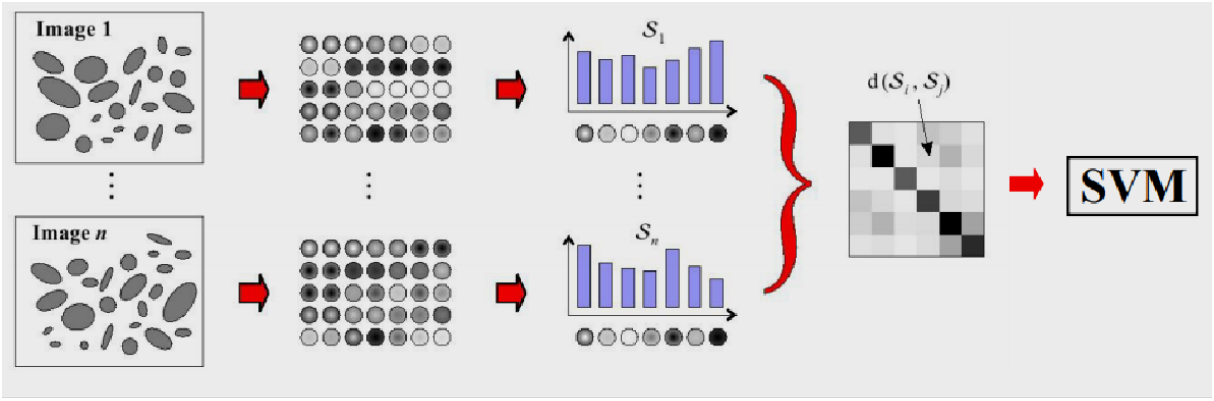


Figure 2.5: A common computer vision pipeline for still image analysis [Nowak et al., 2006; Marszałek et al., 2007].

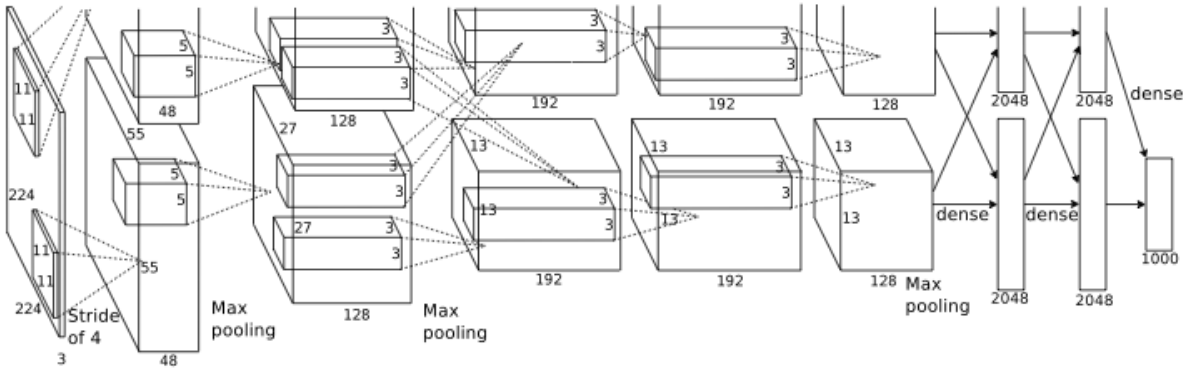


Figure 2.6: The *AlexNet* architecture [Krizhevsky et al., 2012] has won the large scale visual recognition challenge (ILSVRC) [Russakovsky et al., 2015] in 2012.

2.1.2 Deep learning approach

More recently neural networks and especially CNN-based approaches have been investigated to learn video representations for action recognition. Again, this phenomena was made possible thanks to an important step forward in still image analysis.

The winning of the ImageNet large scale visual recognition challenge (ILSVRC) [Russakovsky et al., 2015] by the CNN-based method [Krizhevsky et al., 2012] (depicted in Figure 2.6) in 2012 has marked a breakthrough in computer vision. The ILSVRC is an

image classification challenge built on the ImageNet dataset [Deng et al., 2009] containing one million images and one thousand classes. This success was possible thanks to a fast implementation of image convolution based on GPU, a large amount of labeled training data (ImageNet) and the heavy usage of regularization (weight decay, data augmentation and dropout). Also, faster optimization convergence was obtained by replacing network non-linearities by the rectified linear unit (ReLU) allowing to propagate gradient easily. A wide range of indepth studies were already performed on convolutional neural networks (CNN) [LeCun et al., 1998a,b] and on their associated training techniques [Bottou, 1998] such as *backpropagation* [Rumelhart et al., 1985]. The revelation of Krizhevsky et al. has then allowed the community to apply this considerable work to a wide range of problems (including video analysis) and *deep learning* quickly entered most areas of computer vision. For example, impressing advances in object and person detection have been reached by the region-based convolutional neural networks (R-CNN) [Girshick et al., 2014] but also more specifically in human pose estimation [Toshev and Szegedy, 2014; Chen and Yuille, 2014; Newell et al., 2016; Wei et al., 2016]. In the meantime, Taigman et al. [2014] approached the human performance in face recognition. CNN approach was adopted for video classification in [Simonyan and Zisserman, 2014; Yue-Hei et al., 2015]. Initial CNN-based models were combined with IDT to reach state-of-the-art performance [Tran et al., 2015; de Souza et al., 2016; Feichtenhofer et al., 2016a,b; Varol et al., 2017]. In their *two-streams* approach, Simonyan and Zisserman [2014] propose a combination of appearance and motion information learned by two separate CNN networks. In the manner of HOG and HOF descriptors, the *spatial* stream takes a single frame as input and models its static appearance while the *temporal* stream extracts motion features from a stack of optical flow fields. The CNN architecture was then adapted to the video dimensionality and alternative methods based on spatio-temporal convolutions have been studied in Ji et al. [2010] and Taylor et al. [2010] and more recently using the C3D [Tran et al., 2015]

and I3D [Carreira and Zisserman, 2017] architectures.

Video datasets with action labels are significantly smaller in terms of samples compared to still-image datasets. This complicates the training of CNN-based models for action recognition in video. The transfer of pre-trained network parameters to problems with limited training data (*transfer learning*) has shown success [Girshick et al., 2014; Oquab et al., 2014; Simonyan and Zisserman, 2014] and is commonly used in video analysis. This approach, known as *fine-funing*, adapts pre-trained parameters of the network to a new task with limited training data. The fine-tuning is often performed for the top layers of the network only, as the bottom convolutional kernels tend to be generic. Carreira and Zisserman [2017] notably take advantage of transfer learning by training 3D architectures on a large video dataset [Kay et al., 2017] leading to one of the currently best performing model for action recognition.

Several works use recurrent neural networks (RNN) for aggregating video information along time [Baccouche et al., 2011; Donahue et al., 2015; Ng et al., 2015; Pigou et al., 2017; Singh et al., 2016; Srivastava et al., 2016]. Although RNN-based models have shown promise for the task of gesture recognition [Pigou et al., 2017], such models so far have not shown significant improvements for action classification [Yue-Hei et al., 2015].

2.1.3 Action recognition with person-centric cues

Person-centric cues, such as the position of human body joints, also called "human pose", have been used in several models for action recognition. We review some of these works related to our P-CNN descriptor presented in Chapter 4.

As discussed earlier, most of the recent methods for action recognition deploy global aggregation of local video descriptors. Such representations provide invariance to numerous variations in the video but may fail to capture important spatio-temporal structure. On the other hand, human-centric models tend to focus on the subject of interest, the

person, while removing irrelevant background noise.

For fine-grained action recognition, previous methods have represented person-object interactions by joint tracking of hands and objects [Ni et al., 2014] or, by linking object proposals [Zhou et al., 2015], followed by feature pooling in selected regions. Alternative methods represent actions using positions and temporal evolution of body joints. While reliable human pose estimation is still a challenging task, the recent study of Jhuang et al. [2013] reports significant gains provided by dynamic human pose features in cases when reliable pose estimation is available. In Chapter 4, we extend the work of Jhuang et al. [2013] and design a new CNN-based representation for human actions combining positions, appearance and motion of human body parts. We then extend previous global CNN methods such as Simonyan and Zisserman [2014] and Yue-Hei et al. [2015] and address action recognition using CNN descriptors at the local level of body parts.

Our work in Chapter 4 also builds on methods for human pose estimation in images [Pishchulin et al., 2013; Sapp and Taskar, 2013; Toshev and Szegedy, 2014; Yang and Ramanan, 2011] and video sequences [Cherian et al., 2014; Sapp et al., 2011]. In particular, we build on the method of Cherian et al. [2014] and extract temporally-consistent tracks of body joints from video sequences. While our pose estimator is imperfect, we use it to derive CNN-based pose features providing significant improvements for action recognition for two challenging datasets.

2.1.4 Datasets and evaluation metrics

Recent datasets for human action recognition include UCF-101 [Soomro et al., 2012], HMDB-51 [Kuehne et al., 2011], ActivityNet-200 [Heilbron et al., 2015] and Sports-1M [Karpathy et al., 2014]. The large Kinetics video dataset [Kay et al., 2017] that contains 400 action classes with at least 400 video clips per action has notably allowed to train the previously mentioned I3D architecture [Carreira and Zisserman, 2017] in order

to get state-of-the-art performance. In our experiments of Chapter 4 we use the JHMDB [Jhuang et al., 2013] and MPII Cooking Activities [Rohrbach et al., 2012] datasets as well as their respective subsets, sub-JHMDB and sub-MPII Cooking. We present them and their associated metric in the following.

JHMDB [Jhuang et al., 2013] is a subset of HMDB-51 [Kuehne et al., 2011], see Figure 4.2 (left). It contains 21 human actions, such as *brush hair*, *climb*, *golf*, *run* or *sit*. Video clips are restricted to the duration of the action. There are between 36 and 55 clips per action for a total of 928 clips. Each clip contains between 15 and 40 frames of size 320×240. Human pose is annotated in each of the 31838 frames. There are 3 train/test splits for the JHMDB dataset and evaluation averages the results over these three splits. The metric used is accuracy: each clip is assigned to an action label corresponding to the maximum value among the scores returned by the action classifiers. The hard assignment prediction is considered as correct or incorrect and the per-video accuracy is reported.

In our experiments we additionally use the same subset of JHMDB as defined by Jhuang et al. [2013], referred to as **sub-JHMDB**. This subset includes 316 clips distributed over 12 actions in which the human body is fully visible. Again there are 3 train/test splits and the evaluation metric is accuracy.

MPII Cooking Activities [Rohrbach et al., 2012] contains 64 fine-grained actions and an additional background class, see Figure 4.2 (right). Actions take place in a kitchen with static background. There are 5609 action clips of frame size 1624×1224. Some actions are very similar, such as *cut dice*, *cut slices*, and *cut stripes* or *wash hands* and *wash objects*. Thus, these activities are qualified as "fine-grained". There are 7 train/test splits and the evaluation is reported in terms of mean Average Precision (mAP) using the code provided with the dataset. For an action class k, true positive (TP) and false positive (FP) assignment is computed from all test video scores and the corresponding

precision-recall curve is drawn. The average precision (AP) of class k can be seen as the area under this curve. The mAP metric is the average of the AP for all classes.

We have also defined a subset of MPII cooking, referred to as **sub-MPII cooking**, with classes *wash hands* and *wash objects*. We have selected these two classes as they are visually very similar and differ mainly in manipulated objects. To analyze the classification performance for these two classes in detail, we have annotated human pose in all frames of sub-MPII cooking. There are 55 and 139 clips for *wash hands* and *wash objects* actions respectively, for a total of $29,997$ frames.

2.2 Action localization

Action localization aims to find spatial and temporal extents as well as classes of actions in the video. It answers the following questions: *what* are the performed actions? *when* do they happen? and *where* do they take place? As discussed in Section 1.2, this task is more challenging than action classification which only answers the *what* question. Techniques based on local descriptors and deep learning introduced in Section 2.1.1 and Section 2.1.2 are also used for action localization. Here, we review existing work dedicated to temporal (Section 2.2.1), spatio-temporal (Section 2.2.2) and weakly-supervised (Section 2.2.4) localization. Note that in this thesis we aim to recover both temporal and spatial extents of the action (Chapter 5 and Chapter 6). This is a more difficult problem compared to the temporal localization discussed in Section 2.2.1.

2.2.1 Temporal localization

Temporal action localization aims both to identify temporal extents and to classify actions in longer video clips than the ones encountered in action recognition benchmarks. Approaches for joint action classification and temporal segmentation have explored dynamic

programming [Hoai et al., 2011] and temporal grammars [Pirsiavash and Ramanan, 2014]. More recently, several RNN-based methods have shown gains for this task [Bagautdinov et al., 2017; Ma et al., 2016; Singh et al., 2016; Yeung et al., 2016; Yuan et al., 2016]. For example, Ma et al. [2016], Singh et al. [2016] and Yuan et al. [2016] explore variants of LSTM to improve per-frame action classification whereas Yeung et al. [2016] learn to directly predict action boundaries. An alternative approach based on 3D CNN and temporal action proposals has shown competitive results in [Shou et al., 2016]. Similar proposals are first expended in time by Zhao et al. [2017] then segmented into different stages. This allows them to evaluate their "completenes" based on structured pyramid pooling. A much more challenging task, that requires to localize actions in space and time, is presented next.

2.2.2 Spatio-temporal localization

Spatio-temporal action detection consists in finding the location of action instances in a video volume, both in space and time. Some of the earlier works exploit volumetric features while scanning the video clip with 3D window detectors [Ke et al., 2005]. Laptev and Pérez [2007] extend this idea by detecting key-frames to obtain action hypotheses with predefined temporal extents further classified with a boosted action classifier (see Figure 2.7). Jain et al. [2014], Oneata et al. [2014] and van Gemert et al. [2015] have extended the idea of object proposals in still images [Arbeláez et al., 2014; Uijlings et al., 2013] by adapting it to video action proposals.

Currently, the dominant strategy [Gkioxari and Malik, 2015; Peng and Schmid, 2017; Saha et al., 2016; Singh et al., 2017; Weinzaepfel et al., 2015] is to obtain per-frame action or person detections then to link resulting bounding boxes into continuous spatio-temporal tracks. The temporal localization is finally achieved by using temporal sliding windows [Weinzaepfel et al., 2015, 2016] or dynamic programming [Peng and Schmid, 2017;

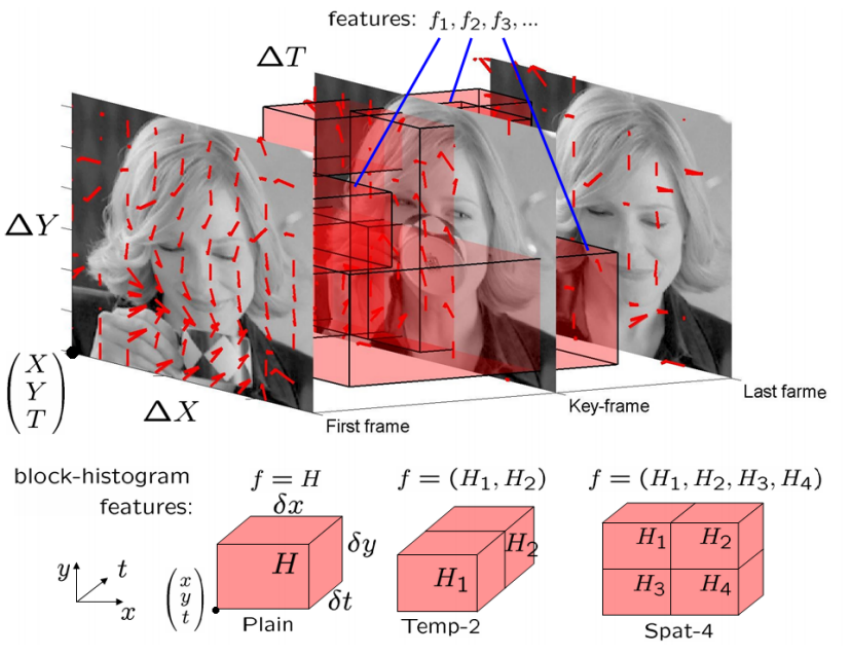


Figure 2.7: (Top): action volume in space-time represented by three frames of a drinking action. Arrows on the frames correspond to the computed optic flow vectors. Transparent blocks in red demonstrate some of the volumetric features of a boosted space-time classifier. (Bottom): Three types of volumetric features with different arrangement of histogram blocks. Figure from [Laptev and Pérez, 2007].

Saha et al., 2016, 2017; Singh et al., 2017]. The most recent methods [Gu et al., 2018; Hou et al., 2017; Kalogeiton et al., 2017a; Saha et al., 2017; Zolfaghari et al., 2017] operate on multiple frames and leverage temporal information to determine actions. Indeed, instead of relying on per-frame detections, Saha et al. [2017] regress pairs of successive frames while Hou et al. [2017] generate clip proposals from 3D feature maps. The approach of Zolfaghari et al. [2017] uses a CNN to sequentially fuse human pose features in addition to the standard appearance and optical flow modalities. Kalogeiton et al. [2017a] stack features from several frames relying on the SSD detector [Liu et al., 2016b] adapted to spatio-temporal anchors. This generates short candidates called *tubelets* and final human tracks are obtained by linking. The recent work of Gu et al. [2018] applies I3D features [Carreira and Zisserman, 2017] to spatio-temporal volumes.

As decribed in Section 2.2.1, several RNN-based approches have shown performance gain for temporal action localization. In Chapter 5, we follow these observations by introducing our recurrent RecLNet model trained on the level of person tracks which instead address the more challenging task of localizing actions in space and time. Our method both classifies and localizes actions within tracks supported by a thresholding and filtering method. Our analysis shows that RecLNet provides significant gains due to accurate temporal localization and its complementarity to the recent method [Kalogeiton et al., 2017a] with more accurate spatial localization but approximate temporal localization.

The methods discussed above require the spatial supervision (person bounding boxes) at the training time. Such supervision can be found in several datasets designed for action recognition that contain human annotations [Jhuang et al., 2013; Rodriguez et al., 2008; Soomro et al., 2012]. However, some of them [Jhuang et al., 2013; Rodriguez et al., 2008] have been created by collecting video clips temporally trimmed to the action extent and are then not suited for our task. We then present in Section 2.2.3 the datasets allowing action localization.

2.2.3 Datasets and evaluation metrics

Recent datasets for the temporal localization task include THUMOS [Idrees et al., 2016], Activity-Net [Heilbron et al., 2015] and MPII Cooking [Rohrbach et al., 2016]. The spatio-temporal action localization, addressed in our work, can be evaluated in UCF101-24 [Soomro et al., 2012] and DALY [Weinzaepfel et al., 2016].

UCF101-24. The original version of the UCF-101 dataset [Soomro et al., 2012] is designed for action classification and contains 13321 videos for 101 action classes. The THUMOS challenge [Gorban et al., 2015] has defined a subset of 24 action classes dispatched in 3207 videos for the task of spatio-temporal action localization. We refer to

this subset as "UCF101-24". The dataset contains relatively short sport oriented videos and the actions usually last a large part of the duration (at least 80% of the video length for half of the classes). For example, "biking" or "rope climbing" are among long actions whereas some last few frames such as "basketball dunk" or "tennis swing". The annotation is exhaustive, *i.e.*, for each action instance, the full person track (sequence of bounding boxes) within the temporal interval of the action is manually annotated. In our training and testing, we use recently corrected ground truth tracks [Saha et al., 2016][1]. Each UCF-101 video contains action instances of a single class. There is only one split containing 2293 train videos and 914 test videos.

DALY [Weinzaepfel et al., 2016] is a recent large-scale dataset for action localization with 10 different daily activities: "applying make up on lips", "brushing teeth", "cleaning floor", "cleaning windows", "drinking", "folding textile", "ironing", "phoning", "playing harmonica" and "taking photos or videos". It contains 510 videos for a total of 31 hours arranged in a single split containing at least 31 train and 20 test videos per class. The average length of the videos is 3min 45s. They are composed of several shots and – in contrast to the UCF101-24 dataset – all actions are short with respect to the full video length. This makes the task of temporal action localization more challenging. Also, DALY samples may contain multiple action classes in the same video. For each action instance, its temporal extent is provided while bounding boxes are spatially annotated for a sparse set of frames (called "keyframes" in Weinzaepfel et al. [2016]) within action intervals.

Performance evaluation. At test time, action localization methods produce *candidate tracks* that are compared to the available annotations referred to as *ground-truth tracks*. In order to evaluate the detection performance, we use the standard spatio-temporal intersection over union (IoU) criterion between candidate tracks and ground-truth tracks.

[1] https://github.com/gurkirt/corrected-UCF101-Annots

It is defined for the UCF101-24 benchmark as the product of the temporal IoU between the time segments of the tracks and the average spatial IoU taken on the frames where both tracks are present. The candidate detection is correct if its intersection with the ground-truth track is above a threshold and if both tracks belong to the same action class. Duplicate detections are considered as false positives and are penalized by the standard precision-recall measure. The overall performance is evaluated in terms of mean average precision (mAP). Note that since only sparse spatial annotation is available for some keyframes in the DALY dataset, its benchmark computes spatial IoU at keyframe locations only (temporal IoU computation remains the same).

2.2.4 Weakly-supervised action localization

The methods we have presented so far are fully supervised and require a large quantity of annotations. As described in Section 1.3, this information is usually extremely tedious to obtain, leads to annotation ambiguities and limits the number of training examples. Weakly-supervised learning for action understanding is very promising since it can reduce the annotation effort significantly. This is the goal of our work presented in Chapter 6. In this section, we first review the related work addressing weakly-supervised learning for temporal action localization then for spatio-temporal action localization.

Temporal action localization

A number of approaches address temporal action detection with different types of weak supervision. We present several of these techniques in the following.

Prior work has explored the use of readily-available sources of information such as movie scripts [Duchenne et al., 2009; Bojanowski et al., 2013] to discover actions in clips using text analysis. Duchenne et al. [2009] first extract coarse localization of temporal action boundaries from such scripts. The authors make use of discriminative clustering [Bach

and Harchaoui, 2007; Xu et al., 2004] in order to refine the localization allowing to select training clips with more precise action boundaries. Such samples are used to train an SVM classifier to recognize "open door" and "sit down" instances. Temporal detection is finally performed using the classifier in a sliding window manner. Recent approaches have also explored more complex forms of weak supervision such as the ordering of actions [Bojanowski et al., 2014; Huang et al., 2016; Richard et al., 2017]. While Bojanowski et al. [2014] assign actions to temporal segments based on discriminative clustering, Bojanowski et al. [2015] extend this work and, instead, align natural text descriptions to video. Similar to movie scripts, narrated instructional videos are sources of information of particular interest. Indeed, video narration explaining the course of actions is naturally correlated with what is happening in the video. Another advantage is that text data can be extracted by automatic speech recognition. In this line of work, Alayrac et al. [2016] temporally localize an ordered sequence of steps necessary to achieve a complex task such as "changing a tire" (see Figure 2.8). Authors first align textual narrations to extract the main steps of a task. Then, the action steps are localized in time by incorporating temporal constraints, imposed by speech information, within a model based on discriminative clustering.

The UntrimmedNet [Wang et al., 2017] and Hide-and-Seek [Singh and Lee, 2017] methods are the current state of the art in weakly-supervised temporal action detection. Wang et al. [2017] introduce a feed forward neural network composed of two branches – one for classification and another one for selecting relevant frames – that can be trained end-to-end from clip level supervision only. Alternatively, Singh and Lee [2017] claim that models trained with weakly-supervised learning tend to only focus on the most discriminative portions of images or videos and present it as an issue. Then, authors alleviate this problem by hiding random patches of the input images at training time, forcing the network to focus on less discriminative parts. In this way, more precise object boxes or temporal action boundaries are obtained.

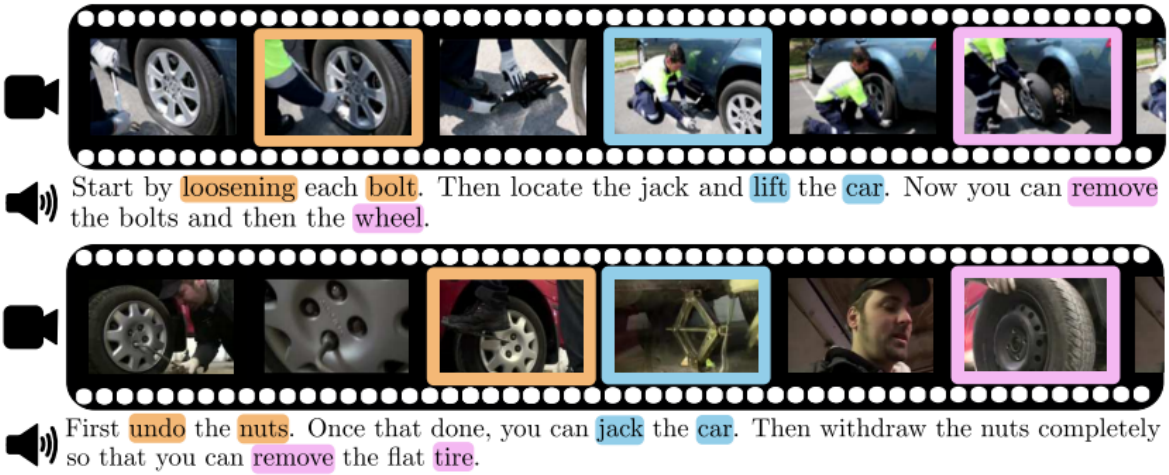


Figure 2.8: Three main steps (highlighted by the same colors in both videos) are automatically discovered for the task "changing a tire". Each step is associated with its corresponding narration and temporal detection in each narrated instructional video. Figure from [Alayrac et al., 2016].

Spatio-temporal action localization

In this thesis (Chapter 6), we seek not to only localize the action temporally but also to look for spatial localization from weak supervision. In [Soomro and Shah, 2017; Yang and Yuan, 2017], the authors propose unsupervised methods to localize actions by using graph clustering algorithms to first discover action classes and then localize actions within each cluster. Other works assume that clip-level annotations are available [Chen and Corso, 2015; Siva and Xiang, 2011]. These methods often rely on action proposals, followed by a selection procedure to find the most relevant segment. Contrary to this line of work, we do not use action proposals. We rely instead on recent advances in object detection [Girshick et al., 2018] and use human tracks. More recently Mettes et al. [2017] make use of pseudo annotation, to integrate biases (e.g. presence of objects, the fact that the action is often in the center of the video...) to further improve performance. Most of this work uses multiple instance learning (MIL) in order to select discriminative instances from the set of all potential candidates. In our approach presented in Chapter 6, we use

discriminative clustering, which does not require EM optimization but instead relies on a convex relaxation with convergence guarantees.

Most related to our method, are the works from Mettes et al. [2016] and Weinzaepfel et al. [2016] that also study the trade-off between annotation cost and final performance. Mettes et al. [2016] show that one can simply use spatial points instead of bounding boxes and still obtain reasonable performance. Weinzaepfel et al. [2016] demonstrate that only a few frames with bounding box annotation are necessary for good performance. In Chapter 6, we introduce a method that can leverage varying levels of supervision and enables to study action localization as a function of different levels of supervision. The spatio-temporal action localization datasets used in this work have been presented in Section 2.2.3.

Chapter 3

Background

In the following we present the discriminative clustering method DIFFRAC together with the Frank-Wolfe optimization algorithm. Both techniques will be used in Chapter 6 in the context of weakly-supervised learning.

Discriminative clustering. Let us assume that we have M data points $X \in \mathbb{R}^{M \times d}$ in a d-dimensional space and K clusters. Discriminative clustering [Bach and Harchaoui, 2007; Xu et al., 2004] is a learning method which separates the data so that the clusters can be easily recovered by a discriminative classifier. The general problem is framed as recovering an assignment matrix of data points to clusters (classes) $Y \in \{0,1\}^{M \times K}$ and a classifier $W \in \mathbb{R}^{d \times K}$ that minimizes a clustering cost inspired by the classification loss:

$$\min_{Y \in \mathcal{Y}_s, W} l(Y, XW) + \Omega(W) \tag{3.1}$$

where $\mathcal{Y}_s$ is the set of possible assignments, l is the loss function and Ω is the regularization function. In Chapter 6, $\mathcal{Y}_s$ is defined based on the available supervision while we employ the discriminative clustering method called DIFFRAC proposed by Bach and Harchaoui [2007]. The latter is appealing due to its simple analytic form, its flexibility and its recent

successes for weakly supervised methods in computer vision [Alayrac et al., 2016, 2017; Bojanowski et al., 2013; Joulin et al., 2010].

DIFFRAC. As stated above, the intuition behind discriminative clustering is to separate the data so that the clustering is easily recoverable by a classifier over the input features. In the particular case of the DIFFRAC setting [Bach and Harchaoui, 2007], the loss and regularization functions defined in (3.1) correspond to the square loss and the Tikhonov regularization respectively while the classifier is linear. It leads to the following discriminative clustering cost:

$$h(Y) = \min_{W \in \mathbb{R}^{d \times K}} \frac{1}{2M} \|XW - Y\|_F^2 + \frac{\lambda}{2} \|W\|_F^2, \tag{3.2}$$

where $W \in \mathbb{R}^{d \times K}$ corresponds to the classifier weights for each of the K classes, $\lambda \in \mathbb{R}^+$ is a regularization parameter and $\|.\|_F$ is the standard Frobenius matrix norm. We then want to solve the corresponding optimization problem over the assignment matrix Y:

$$\min_{Y \in \mathcal{Y}_s} h(Y). \tag{3.3}$$

In (3.2), following [Bach and Harchaoui, 2007], we can solve the minimization in W in closed form to obtain after plugging the solution:

$$h(Y) = \frac{1}{2M} \mathrm{Tr}(YY^T B), \tag{3.4}$$

where $\mathrm{Tr}(\cdot)$ is the matrix trace and B is a strictly positive definite matrix (hence h is strongly convex) defined as:

$$B := I_M - XP, \tag{3.5}$$

with:

$$P := (X^T X + M\lambda I_M)^{-1} X^T, \tag{3.6}$$

where I_d stands for the d-dimensional identity matrix. In the following, we will describe how we optimize the problem in (3.3) over Y belonging to the set of possible assignments $\mathcal{Y}_s$.

Relaxation. Directly minimizing the function defined in (3.3) over $\mathcal{Y}_s$ is NP hard due to the integer constraints. To address this challenge, we follow recent approaches such as [Bojanowski et al., 2014] and propose a convex relaxation of the problem. The latter hence becomes the convex quadratic program under linear constraints:

$$\min_{Y \in \bar{\mathcal{Y}}_s} h(Y), \tag{3.7}$$

where $\bar{\mathcal{Y}}_s$ is the convex hull of $\mathcal{Y}_s$. It then allows us to find a continuous solution of the relaxed problem using an efficient method for convex optimization, *e.g.* the Frank-Wolfe algorithm.

Frank-Wolfe. The Frank-Wolfe algorithm is a method to solve the following problem:

$$\min_{x \in \mathcal{M}} f(x), \tag{3.8}$$

where f is smooth and convex and $\mathcal{M}$ is a convex and compact (closed and bounded in finite dimension) domain. The problem defined in (3.7) satisfies these conditions. Now let us consider the gradient step at iteration $k+1$ with step size δ_k associated to (3.8):

$$s_{k+1} = x_k - \delta_k \nabla f(x_k). \tag{3.9}$$

The standard *projected gradient algorithm* deployed to solve such problem projects s_{k+1} back to the constraint domain $\mathcal{M}$, leading to the following update:

$$x_{k+1} = \Pi_{\mathcal{M}}[s_{k+1}], \tag{3.10}$$

with the projection:

$$\Pi_{\mathcal{M}}[x] = \arg\min_{z \in \mathcal{M}} \|z - x\|. \tag{3.11}$$

Such method can be used to solve (3.7) over the $\bar{\mathcal{Y}}_s$ domain. However, the projection in (3.11) can be very costly for some applications such as temporal ordering [Bojanowski et al., 2014]. That is why we prefer the Frank-Wolfe algorithm [Frank and Wolfe, 1956] which only requires to know how to solve linear programs over the constraint set and does not necessitate any projection step. The iterative procedure starts by computing an affine approximation of the objective function thanks to the gradient at the k-th iterate $\nabla f(x_k)$. Then, the following linear program is minimized:

$$p_k^* = \arg\min_{p \in \mathcal{M}} \langle \nabla f(x_k), p \rangle_F, \tag{3.12}$$

Note that solving this problem is potentially cheaper than the projection in (3.11). The next iterate is obtained by combining p_k^* with the current iterate:

$$x_{k+1} = (1 - \gamma_k) x_k + \gamma_k p_k^*, \tag{3.13}$$

with γ_k obtained by line search or using $\gamma_k = \frac{2}{k+2}$. The Frank-Wolfe algorithm is summarized in Algorithm 1. In the following, we discuss the procedure for block coordinate Frank-Wolfe and notably how to use it for solving the problem (3.7).

Algorithm 1 The Frank-Wolfe algorithm

Initialize:
  $x_0 \in \mathcal{M}$
for $k = 0$ to $N_{iter} - 1$ **do**
  $p_k^* \leftarrow \arg\min_{p \in \mathcal{M}} \langle \nabla f(x_k), p \rangle_F$ ▷ Linear oracle
  $\gamma_k \leftarrow \frac{2}{k+2}$ or line-search ▷ Step-size
  $x_{k+1} \leftarrow (1 - \gamma_k) x_k + \gamma_k p_k^*$ ▷ Iterate update
end for

Block coordinate Frank-Wolfe. The block coordinate Frank-Wolfe [Lacoste-Julien et al., 2013] (BCFW) can be used when the set of constraints decomposes over multiple subsets $\mathcal{M} = \mathcal{M}^1 \times \cdots \times \mathcal{M}^n$. The idea behind BCFW is to randomly sample a subset of constraints (a.k.a. block) at each iteration then to perform a Frank-Wolfe step on this subset. It allows one to deal with a large number of data points while obtaining optimization speed-up. In particular, Lacoste-Julien et al. [2013] have shown that the BCFW algorithm has the same convergence rate as the batch version – which requires a full pass on the dataset – but iterations are n time cheaper.

In the particular case of DIFFRAC and when $\bar{\mathcal{Y}}_s = \bar{\mathcal{Y}}_s^1 \times \cdots \times \bar{\mathcal{Y}}_s^N$, problem (3.7) can be solved as follows. The block i corresponds to the subset of data samples $X^{(i)}$ and constraints $\bar{\mathcal{Y}}_s^i$ and its gradient related to (3.4) is

$$\nabla h_{(i)}(Y) = \frac{1}{M}(Y^{(i)} - X^{(i)} P Y). \tag{3.14}$$

Note that from (3.5) and (3.6), the B and P matrices only depend on X and λ and can then be computed only once. The recent adaptation of the block coordinate Frank-Wolfe for the discriminative clustering objective [Miech et al., 2017] is summarized in Algorithm 2.

In Chapter 6, all short human tracks (*tracklets*) contained in a single video belong to the same block (constraints decompose over videos). It then allows us to scale to large

Algorithm 2 BCFW for Discriminative Clustering

Initialize:
  Sample Y_0 randomly in $\bar{\mathcal{Y}}_s$
  $P := (X^T X + M\lambda I_M)^{-1} X^T$
  $W_0 = PY_0$
  $g_i = +\infty, \forall i$
for $t = 0$ to $N_{iter} - 1$ **do**
  $i \leftarrow$ sample from distribution proportional to g
  $\nabla_{(i)} h(Y_t) \leftarrow \frac{1}{M}(Y_t^{(i)} - X^{(i)} W_t)$ ▷ Block gradient
  $Y_{min} \leftarrow \arg\min_{y \in \bar{\mathcal{Y}}_s^{(i)}} \langle \nabla_{(i)} h(Y_t), y \rangle_F$ ▷ Linear oracle
  $D \leftarrow Y_{min} - Y_t^{(i)}$
  $g_i \leftarrow -\langle \nabla_{(i)} h(Y_t), D \rangle_F$ ▷ Block gap
  $\gamma \leftarrow \min(1, \frac{g_i}{\frac{1}{M}\langle D, D - X^{(i)} P^{(i)} D \rangle_F})$ ▷ Line-search
  $W_{t+1} \leftarrow W_t + \gamma P^{(i)} D$ ▷ W update
  $Y_{t+1}^{(i)} \leftarrow Y_t^{(i)} + \gamma D$ ▷ Block update
end for

datasets with several hundreds of videos.

Chapter 4

Pose-based CNN features for action classification.

4.1 Introduction

Recognition of human actions is an important step toward fully automatic understanding of dynamic scenes. Despite significant progress in recent years, action recognition remains a difficult challenge. Common problems stem from the strong variations of people and scenes in motion and appearance. Other factors include subtle differences of fine-grained actions, for example when manipulating small objects or assessing the quality of sports actions.

The majority of recent methods recognize actions based on statistical representations of local motion descriptors [Laptev et al., 2008; Schuldt et al., 2004; Wang and Schmid, 2013]. These approaches are successful in recognizing coarse action (standing up, hand-shaking, dancing) in challenging scenes with camera motions, occlusions, multiple people, etc. Global approaches, however, are lacking structure and may not be optimal to recognize subtle variations, e.g. to distinguish correct and incorrect golf swings or to recognize

fine-grained cooking actions illustrated in Figure 4.5.

Fine-grained recognition in static images highlights the importance of spatial structure and spatial alignment as a pre-processing step. Examples include alignment of faces for face recognition [Berg and Belhumeur, 2013] as well as alignment of body parts for recognizing species of birds [Duan et al., 2012]. In analogy to this prior work, we believe action recognition will benefit from the spatial and temporal detection and alignment of human poses in videos. In fine-grained action recognition, this will, for example, allow one to better differentiate *wash hands* from *wash object* actions.

In this chapter we design a new action descriptor based on human poses. Provided with tracks of body joints over time, our descriptor combines motion and appearance features for body parts. Given the recent success of Convolutional Neural Networks (CNN) [Krizhevsky et al., 2012; LeCun et al., 1998a], we explore CNN features obtained separately for each body part in each frame. We use appearance and motion-based CNN features computed for each track of body parts, and investigate different schemes of temporal aggregation. The extraction of proposed *Pose-based Convolutional Neural Network* (P-CNN) features is illustrated in Figure 4.1.

Pose estimation in natural images is still a difficult task [Yang and Ramanan, 2011; Chen and Yuille, 2014; Tompson et al., 2014; Toshev and Szegedy, 2014; Newell et al., 2016; Wei et al., 2016]. In this chapter we investigate P-CNN features both for automatically estimated as well as manually annotated human poses. We report experimental results for two challenging datasets: JHMDB [Jhuang et al., 2013], a subset of HMDB [Kuehne et al., 2011] for which manual annotation of human pose have been provided by Jhuang et al. [2013], as well as MPII Cooking Activities [Rohrbach et al., 2012], composed of a set of fine-grained cooking actions. Evaluated on both datasets, our method consistently outperforms the human pose-based descriptor HLPF [Jhuang et al., 2013]. Combination of our method with Dense trajectory features [Wang and Schmid, 2013] improves the state

of the art for both datasets at the time of publication. Our implementation of P-CNN features is available from [Chéron et al., 2015a].

Related work. Action recognition in the last decade has been dominated by local features [Laptev et al., 2008; Schuldt et al., 2004; Wang and Schmid, 2013] and in particular IDT-FV [Wang and Schmid, 2013] (improved version of dense trajectories with Fisher vector encoding) that we use as a strong baseline and experimentally demonstrate its complementarity to our method.

Application of CNNs to action recognition in video, however, has shown only limited improvements so far [Simonyan and Zisserman, 2014; Yue-Hei et al., 2015]. We extend previous global CNN methods and address action recognition using CNN descriptors at the local level of human body parts.

Alternative methods represent actions using positions and temporal evolution of body joints. While reliable human pose estimation is still a challenging task, the recent study of Jhuang et al. [2013] reports significant gains provided by dynamic human pose features in cases when reliable pose estimation is available. We extend the work of Jhuang et al. [2013] and design a new CNN-based representation for human actions combining positions, appearance and motion of human body parts.

Our work also builds on methods for human pose estimation in images [Yang and Ramanan, 2011; Pishchulin et al., 2013; Sapp and Taskar, 2013; Chen and Yuille, 2014; Tompson et al., 2014; Toshev and Szegedy, 2014; Newell et al., 2016; Wei et al., 2016] and video sequences [Cherian et al., 2014; Sapp et al., 2011]. In particular, we build on the method [Cherian et al., 2014] and extract temporally-consistent tracks of body joints from video sequences. While our pose estimator is imperfect, we use it to derive CNN-based pose features providing significant improvements for action recognition for two challenging datasets.

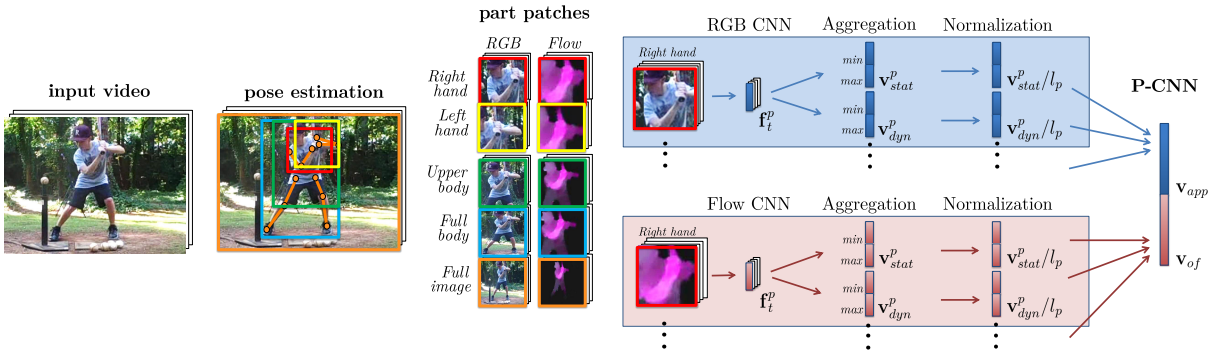


Figure 4.1: P-CNN features. From left to right: Input video. Human pose and corresponding human body parts for one frame of the video. Patches of appearance (RGB) and optical flow for human body parts. One RGB and one flow CNN descriptor $\mathbf{f}_t^p$ is extracted per frame t and per part p (an example is shown for the human body part *right hand*). Static frame descriptors $\mathbf{f}_t^p$ are aggregated over time using *min* and *max* to obtain the video descriptor $\mathbf{v}_{stat}^p$. Similarly, temporal differences of $\mathbf{f}_t^p$ are aggregated to $\mathbf{v}_{dyn}^p$. Video descriptors are normalized and concatenated over parts p and aggregation schemes into appearance features $\mathbf{v}_{app}$ and flow features $\mathbf{v}_{of}$. The final P-CNN feature is the concatenation of $\mathbf{v}_{app}$ and $\mathbf{v}_{of}$.

Outline. The chapter is organized as follows. Section 4.2 introduces our P-CNN features. We summarize baseline methods (state of the art at the time of publication) used and compared to in our experiments in Section 4.3. Section 4.4 evaluates our method and compares it to baseline methods. Section 4.5 concludes the chapter.

4.2 P-CNN: Pose-based CNN features

We believe that human pose is essential for action recognition. Here, we use positions of body joints to define informative image regions. We further borrow inspiration from the work of Simonyan and Zisserman [2014] and represent body regions with motion-based and appearance-based CNN descriptors. Such descriptors are extracted at each frame and then aggregated over time to form a video descriptor, see Figure 4.1 for an overview. The details are explained below.

To construct P-CNN features, we first compute optical flow [Brox et al., 2004] for each consecutive pair of frames. The method of Brox et al. [2004] has relatively high speed, good accuracy and has been recently used in other flow-based CNN approaches [Gkioxari and Malik, 2015; Simonyan and Zisserman, 2014]. Following [Gkioxari and Malik, 2015], the values of the motion field v_x, v_y are transformed to the interval $[0, 255]$ by $\tilde{v}_{x|y} = av_{x|y} + b$ with $a = 16$ and $b = 128$. The values below 0 and above 255 are truncated. We save the transformed flow maps as images with three channels corresponding to motion $\tilde{v}_x$, $\tilde{v}_y$ and the flow magnitude.

Given a video frame and the corresponding positions of body joints, we crop RGB image patches and flow patches for *right hand*, *left hand*, *upper body*, *full body* and *full image* as illustrated in Figure 4.1. Each patch is resized to 224×224 pixels to match the CNN input layer. To represent appearance and motion patches, we use two distinct CNNs with an architecture similar to [Krizhevsky et al., 2012]. Both networks contain 5 convolutional and 3 fully-connected layers. The output of the second fully-connected layer with $D = 4096$ values is used as a frame descriptor ($\mathbf{f}_t^p$). For RGB patches we use the publicly available "VGG-f" network from Chatfield et al. [2014] that has been pre-trained on the ImageNet ILSVRC-2012 challenge dataset [Deng et al., 2009]. For flow patches, we use the motion network provided by Gkioxari and Malik [2015] that has been pre-trained for action recognition task on the UCF101 dataset [Soomro et al., 2012].

Given descriptors $\mathbf{f}_t^p$ for each part p and each frame t of the video, we then proceed with the aggregation of $\mathbf{f}_t^p$ over all frames to obtain a fixed-length video descriptor. We consider *min* and *max* aggregation by computing minimum and maximum values for each descriptor dimension $i \in [1, D]$ over T video frames

$$\begin{aligned} m_i &= \min_{1 \le t \le T} \mathbf{f}_t^p(i), \\ M_i &= \max_{1 \le t \le T} \mathbf{f}_t^p(i). \end{aligned} \tag{4.1}$$

The *static video descriptor* for part p is defined by the concatenation of time-aggregated frame descriptors as

$$\mathbf{v}^p_{stat} = [m_1, ..., m_D, M_1, ..., M_D]^\top . \tag{4.2}$$

To capture temporal evolution of per-frame descriptors, we also consider temporal differences of the form $\Delta\mathbf{f}^p_t = \mathbf{f}^p_{t+\Delta t} - \mathbf{f}^p_t$ for $\Delta t = 4$ frames. Similar to (4.1) we compute minimum Δm_i and maximum ΔM_i aggregations of $\Delta\mathbf{f}^p_t$ and concatenate them into the *dynamic video descriptor*

$$\mathbf{v}^p_{dyn} = [\Delta m_1, ..., \Delta m_D, \Delta M_1, ..., \Delta M_D]^\top . \tag{4.3}$$

Finally, video descriptors for motion and appearance for all parts and different aggregation schemes are normalized and concatenated into the P-CNN feature vector. The normalization is performed by dividing video descriptors by the average L_2-norm of the $\mathbf{f}^p_t$ from the training set.

In Section 4.4 we evaluate the effect of different aggregation schemes as well as the contributions of motion and appearance features for action recognition. In particular, we compare "*Max*" vs. "*Max/Min*" aggregation where "*Max*" corresponds to the use of M_i values only while "*Max/Min*" stands for the concatenation of M_i and m_i defined in (4.2) and (4.3). *Mean* and *Max* aggregation are widely used methods in CNN video representations. We choose *Max-aggr*, as it outperforms *Mean-aggr* (see Section 4.4). We also apply *Min* aggregation, which can be interpreted as a "non-detection feature". Additionally, we want to follow the temporal evolution of CNN features in the video by looking at their dynamics (*Dyn*). Dynamic features are again aggregated using *Min* and *Max* to preserve their sign keeping the largest negative and positive differences. The concatenation of static and dynamic descriptors will be denoted by "*Static+Dyn*".

The final dimension of our P-CNN is $(5 \times 4 \times D) \times 2 = 40D$, i.e., 5 body parts, 4

different aggregation schemes, D-dimensional CNN descriptor for appearance and motion. Note that such a dimensionality is comparable to the size of Fisher vector [Chatfield et al., 2011] used to encode dense trajectory features [Wang and Schmid, 2013]. P-CNN training is performed using a linear SVM.

4.3 Baseline methods

In this section we present the baseline methods used and compared to in our experiments. We first present the approach for human pose estimation in videos [Cherian et al., 2014] used in our experiments. We then present high-level pose features (HLPF) [Jhuang et al., 2013] and improved dense trajectories [Wang and Schmid, 2013] baselines.

4.3.1 Pose estimation

To compute P-CNN features as well as HLPF features, we need to detect and track human poses in videos. We have implemented a video pose estimator based on [Cherian et al., 2014]. We first extract poses for individual frames using the approach of Yang and Ramanan [2011]. Their approach is based on a deformable part model to locate positions of body joints (head, elbow, wrist...). We re-train their model on the FLIC dataset [Sapp and Taskar, 2013].

Following [Cherian et al., 2014], we extract a large set of pose configurations in each frame and link them over time using dynamic programming (DP). The poses selected with DP are constrained to have a high score of the pose estimator [Yang and Ramanan, 2011]. At the same time, the motion of joints in a pose sequence is constrained to be consistent with the optical flow extracted at joint positions. In contrast to Cherian et al. [2014] we do not perform *limb recombination*. See Figure 4.2 for examples of automatically extracted human poses.

Figure 4.2: Illustration of human pose estimation used in our experiments [Cherian et al., 2014]. Successful examples and failure cases on JHMDB (left two images) and on MPII Cooking Activities (right two images). Only left and right arms are displayed for clarity.

4.3.2 High-level pose features (HLPF)

High-level pose features (HLPF) encode spatial and temporal relations of body joint positions and were introduced by Jhuang et al. [2013]. Given a sequence of human poses P, positions of body joints are first normalized with respect to the person size. Then, the relative offsets to the head are computed for each pose in P. We have observed that the head is more reliable than the torso used in [Jhuang et al., 2013]. Static features are, then, the distances between all pairs of joints, orientations of the vectors connecting pairs of joints and inner angles spanned by vectors connecting all triplets of joints.

Dynamic features are obtained from trajectories of body joints. HLPF combines temporal differences of some of the static features, i.e., differences in distances between pairs of joints, differences in orientations of lines connecting joint pairs and differences in inner angles. Furthermore, translations of joint positions (dx and dy) and their orientations ($\arctan(\frac{dy}{dx})$) are added.

All features are quantized using a separate codebook for each feature dimension (descriptor type), constructed using K-means with 20 clusters. A video sequence is then represented by a histogram of quantized features and the training is performed using an SVM with a χ^2-kernel. More details can be found in [Jhuang et al., 2013]. To compute HLPF features we use the publicly available code with minor modifications, i.e., we con-

sider the head instead of the torso center for relative positions. We have also found that converting angles, originally in degrees, to radians and L2 normalizing the HLPF features improves the performance.

4.3.3 Dense trajectory features

Dense Trajectories (DT) [Wang et al., 2011] are local video descriptors that have recently shown excellent performance in several action recognition benchmarks [Oneata et al., 2013; Wang et al., 2013]. The method first densely samples points which are tracked using optical flow [Farnebäck, 2003]. For each trajectory, 4 descriptors are computed in the aligned spatio-temporal volume: HOG [Dalal and Triggs, 2005], HOF [Laptev et al., 2008] and MBH [Dalal et al., 2006]. A recent approach [Wang and Schmid, 2013] removes trajectories consistent with the camera motion (estimated computing a homography using optical flow and SURF [Bay et al., 2008] point matches and RANSAC [Fischler and Bolles, 1981]). Flow descriptors are then computed from optical flow warped according to the estimated homography. We use the publicly available implementation [Wang and Schmid, 2013] to compute improved version of DT (IDT).

Fisher Vectors (FV) [Perronnin et al., 2010b] encoding has been shown to outperform the bag-of-word approach [Chatfield et al., 2011] resulting in state-of-the-art performance at the time of publication for action recognition in combination with DT features [Oneata et al., 2013]. FV relies on a Gaussian mixture model (GMM) with K Gaussian components, computing first and second order statistics with respect to the GMM. FV encoding is performed separately for the 4 different IDT descriptors (their dimensionality is reduced by the factor of 2 using PCA). Following Perronnin et al. [2010b], the performance is improved by passing FV through signed square-rooting and L_2 normalization. As in [Oneata et al., 2013] we use a spatial pyramid representation and a number of $K = 256$ Gaussian components. FV encoding is performed using the Yael library [Douze and Jégou, 2014]

and classification is performed with a linear SVM.

4.4 Experimental results

	JHMDB-GT			MPII Cooking-Pose [Cherian et al., 2014]		
Parts	App	OF	App + OF	App	OF	App + OF
Hands	46.3	54.9	57.9	39.9	46.9	51.9
Upper body	52.8	60.9	67.1	32.3	47.6	50.1
Full body	52.2	61.6	66.1	-	-	-
Full image	43.3	55.7	61.0	28.8	56.2	56.5
All	**60.4**	**69.1**	**73.4**	**43.6**	**57.4**	**60.8**

Table 4.1: Performance of appearance-based (App) and flow-based (OF) P-CNN features. Results are obtained with max-aggregation for JHMDB-GT (% accuracy) and MPII Cooking Activities-Pose [Cherian et al., 2014] (% mAP).

This section describes our experimental results and examines the effect of different design choices. First, we evaluate the complementarity of different human parts in Section 4.4.1. We then compare different variants for aggregating CNN features in Section 4.4.2. Next, we analyze the robustness of our features to errors in the estimated pose and their ability to classify fine-grained actions in Section 4.4.3. Finally, we compare our features to baseline methods and show that they are complementary to the popular dense trajectory features in Section 4.4.4.

4.4.1 Performance of human part features

Table 4.1 compares the performance of human part CNN features for both appearance and flow on JHMDB-GT (the JHMDB dataset with ground-truth pose) and MPII Cooking-Pose [Cherian et al., 2014] (the MPII Cooking dataset with pose estimated by [Cherian et al., 2014]). Note, that for MPII Cooking we detect upper-body poses only since full bodies are not visible in most of the frames in this dataset.

Conclusions for both datasets are similar. We can observe that all human parts (hands, upper body, full body) as well as the full image have similar performance and that their combination improves the performance significantly. Removing one part at a time from this combination results in the drop of performance (results not shown here). We therefore use all pose parts together with the full image descriptor in the following evaluation. We can also observe that flow descriptors consistently outperform appearance descriptors by a significant margin for all parts as well as for the overall combination *All*. Furthermore, we can observe that the combination of appearance and flow further improves the performance for all parts including their combination *All*. This is the pose representation used in the rest of the evaluation.

In this section, we have applied the max-aggregation (see Section 4.2) for aggregating features extracted for individual frames into a video descriptor. Different aggregation schemes will be compared in the next section.

4.4.2 Aggregating P-CNN features

CNN features $\mathbf{f}_t$ are first extracted for each frame and the following temporal aggregation pools feature values for each feature dimension over time (see Figure 4.1). Results of max-aggregation for JHMDB-GT are reported in Table 4.1 and compared with other aggregation schemes in Table 4.2. Table 4.2 shows the impact of adding min-aggregation (*Max/Min-aggr*) and the first-order difference between CNN features (*All-Dyn*). Combining per-frame CNN features and their first-order differences using max- and min-aggregation further improves results. Overall, we obtain the best results with *All-(Static+Dyn)(Max/Min-aggr)* for App + OF, i.e., 74.6% accuracy on JHMDB-GT. This represents an improvement over *Max-aggr* by 1.2%. On MPII Cooking-Pose [Cherian et al., 2014] this version of P-CNN achieves 62.3% mAP (as reported in Table 4.4) leading to an 1.5% improvement over max-aggregation reported in Table 4.1.

Aggregation scheme	**App**	**OF**	**App+OF**
All(Max-aggr)	60.4	69.1	73.4
All(Max/Min-aggr)	60.6	68.9	73.1
All(Static+Dyn)(Max-aggr)	62.4	**70.6**	74.1
All(Static+Dyn)(Max/Min-aggr)	**62.5**	70.2	**74.6**
All(Mean-aggr)	57.5	69.0	69.4

Table 4.2: Comparison of different aggregation schemes: *Max-*, *Mean-*, and *Max/Min-*aggregations as well as adding first-order differences (*Dyn*). Results are given for appearance (*App*), optical flow (*OF*) and App + OF on JHMDB-GT (% accuracy).

We have also experimented with second-order differences and other statistics, such as mean-aggregation (last row in Table 4.2), but this did not improve results. Furthermore, we have tried temporal aggregation of classification scores obtained for individual frames. This led to a decrease of performance, e.g. for *All (App)* on JHMDB-GT score-max-aggregation results in 56.1% accuracy, compared to 60.4% for features-max-aggregation (top row, left column in Table 4.2). This indicates that early aggregation works significantly better in our setting.

In summary, the best performance is obtained for *Max-aggr* on single-frame features, if only one aggregation scheme is used. Addition of *Min-aggr* and first order differences *Dyn* provides further improvement. In the remaining evaluation we report results for this version of P-CNN, i.e., *All* parts *App+OF* with *(Static+Dyn)(Max/Min-aggr)*.

4.4.3 Robustness of pose-based features

This section examines the robustness of P-CNN features in the presence of pose estimation errors and compares results with the pose features HLPF [Jhuang et al., 2013]. We report results using the code of Jhuang et al. [2013] with minor modifications described in Section 4.3.2. Our HLPF results are comparable to [Jhuang et al., 2013] in general and are slightly better on JHMDB-GT (77.8% vs. 76.0%). Table 4.3 evaluates the impact of

automatic pose estimation versus ground-truth pose (GT) for sub-JHMDB and JHMDB. We can observe that results for GT pose are comparable on both datasets and for both types of pose features. However, P-CNN is significantly more robust to errors in pose estimation. For automatically estimated poses P-CNN drops only by 5.7% on sub-JHMDB and by 13.5% on JHMDB, whereas HLPF drops by 13.5% and 52.5% respectively. For both descriptors the drop is less significant on sub-JHMDB, as this subset only contains full human poses for which pose is easier to estimate. Overall the performance of P-CNN features for automatically extracted poses is excellent and outperforms HLPF by a very large margin (+35.8%) on JHMDB.

	sub-JHMDB		
	GT	Pose [Yang and Ramanan, 2011]	Diff
P-CNN	72.5	66.8	5.7
HLPF	78.2	51.1	27.1

	JHMDB		
	GT	Pose [Cherian et al., 2014]	Diff
P-CNN	74.6	61.1	13.5
HLPF	77.8	25.3	52.5

Table 4.3: Impact of automatic pose estimation versus ground-truth pose (GT) for P-CNN features and HLPF [Jhuang et al., 2013]. Results are presented for sub-JHMDB and JHMDB (% accuracy).

We now compare and evaluate the robustness of P-CNN and HLPF features on the MPII cooking dataset. To evaluate the impact of ground-truth pose (GT), we have manually annotated two classes "washing hand" and "washing objects", referred to by sub-MPII Cooking. Table 4.4 compares P-CNN and HLPF for sub-MPII and MPII Cooking. In all cases P-CNN outperforms HLPF significantly. Interestingly, even for ground-truth poses P-CNN performs significantly better than HLPF. This could be explained by the better encoding of image appearance by P-CNN features, especially for object-centered

	sub-MPII Cooking		
	GT	Pose [Cherian et al., 2014]	Diff
P-CNN	83.6	67.5	16.1
HLPF	76.2	57.4	18.8

	MPII Cooking
	Pose [Cherian et al., 2014]
P-CNN	62.3
HLPF	32.6

Table 4.4: Impact of automatic pose estimation versus ground-truth pose (GT) for P-CNN features and HLPF [Jhuang et al., 2013]. Results are presented for sub-MPII Cooking and MPII Cooking (% mAP).

actions such as "washing hands" and "washing objects". We can also observe that the drop due to errors in pose estimation is similar for P-CNN and HLPF. This might be explained by the fact that CNN hand features are quite sensitive to the pose estimation. However, P-CNN still significantly outperforms HLPF for automatic pose. In particular, there is a significant gain of +29.7% for the full MPII Cooking dataset.

4.4.4 Comparison to baseline methods

In this section we compare to dense trajectory features [Wang and Schmid, 2013] encoded by Fisher vectors [Oneata et al., 2013] (IDT-FV), briefly described in Section 4.3.3. We use the online available code, which we validated on Hollywood2 (65.3% versus 64.3% [Wang and Schmid, 2013]). Furthermore, we show that our pose features P-CNN and IDT-FV are complementary and compare to other baseline approaches on JHMDB and MPII Cooking.

Table 4.5 shows that for ground-truth poses our P-CNN features outperform IDT-FV descriptors significantly (8.7%). If the pose is extracted automatically both methods are on par. Furthermore, in all cases the combination of P-CNN and IDT-FV obtained by late fusion of the individual classification scores significantly increases the performance

Method	JHMDB		MPII Cook.
	GT	Pose [Cherian et al., 2014]	Pose [Cherian et al., 2014]
P-CNN	74.6	61.1	62.3
IDT-FV	65.9	65.9	67.6
P-CNN + IDT-FV	**79.5**	**72.2**	**71.4**

Table 4.5: Comparison to IDT-FV on JHMDB (% accuracy) and MPII Cooking Activities (% mAP) for ground-truth (GT) and pose [Cherian et al., 2014]. The combination of P-CNN + IDT-FV performs best.

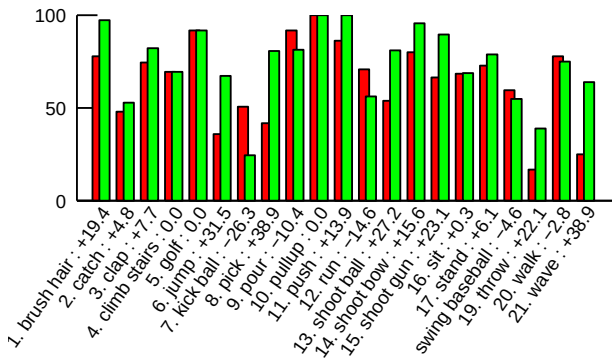


Figure 4.3: Per class accuracy on JHMDB-GT for P-CNN (green) and IDT-FV (red) methods. Values correspond to the difference in accuracy between P-CNN and IDT-FV (positive values indicate better performance of P-CNN).

over using individual features only. Figure 4.3 illustrates per-class results for P-CNN and IDT-FV on JHMDB-GT.

Table 4.6 compares our results to other methods on MPII Cooking. Our approach outperforms the state of the art on this dataset and is on par with the recently published work of Zhou et al. [2015] (at the time of publication). We have compared our method with HLPF [Jhuang et al., 2013] on JHMDB in the previous section. P-CNN performs on par with HLPF for GT poses and significantly outperforms HLPF for automatically estimated poses. Combination of P-CNN with IDT-FV improves the performance to 79.5% and 72.2% for GT and automatically estimated poses respectively (see Table 4.5). This outperforms the result reported in [Jhuang et al., 2013].

Method	MPII Cook.
Holistic + Pose [Rohrbach et al., 2012]	57.9
Semantic Features [Zhou et al., 2014]	70.5
Interaction Part Mining [Zhou et al., 2015]	72.4
P-CNN + IDT-FV (our)	71.4

Table 4.6: State of the art on the MPII Cooking at the time of publication (% mAP).

Qualitative results comparing P-CNN and IDT-FV are presented in Figure 4.4 for JHMDB-GT. See Figure 4.3 for the quantitative comparison. To highlight improvements achieved by the proposed P-CNN descriptor, we show results for classes with a large improvement of P-CNN over IDT-FV, such as *shoot gun*, *wave*, *throw* and *jump* as well as for a class with a significant drop, namely *kick ball*. Figure 4.4 shows two examples for each selected action class with the maximum difference in ranks obtained by P-CNN (green) and IDT-FV (red). For example, the most significant improvement (Figure 4.4, top left) increases the sample ranking from 211 to 23, when replacing IDT-FV by P-CNN. In particular, the *shoot gun* and *wave* classes involve small localized motion, making classification difficult for IDT-FV while P-CNN benefits from the local human body part information. Similarly, the two samples from the action class *throw* also seem to have restricted and localized motion while the action *jump* is very short in time. In the case of *kick ball* the significant decrease can be explained by the important dynamics of this action, which is better captured by IDT-FV features. Note that P-CNN only captures motion information between two consecutive frames.

Figure 4.5 presents qualitative results for MPII Cooking-Pose [Cherian et al., 2014] showing samples with the maximum difference in ranks over all classes.

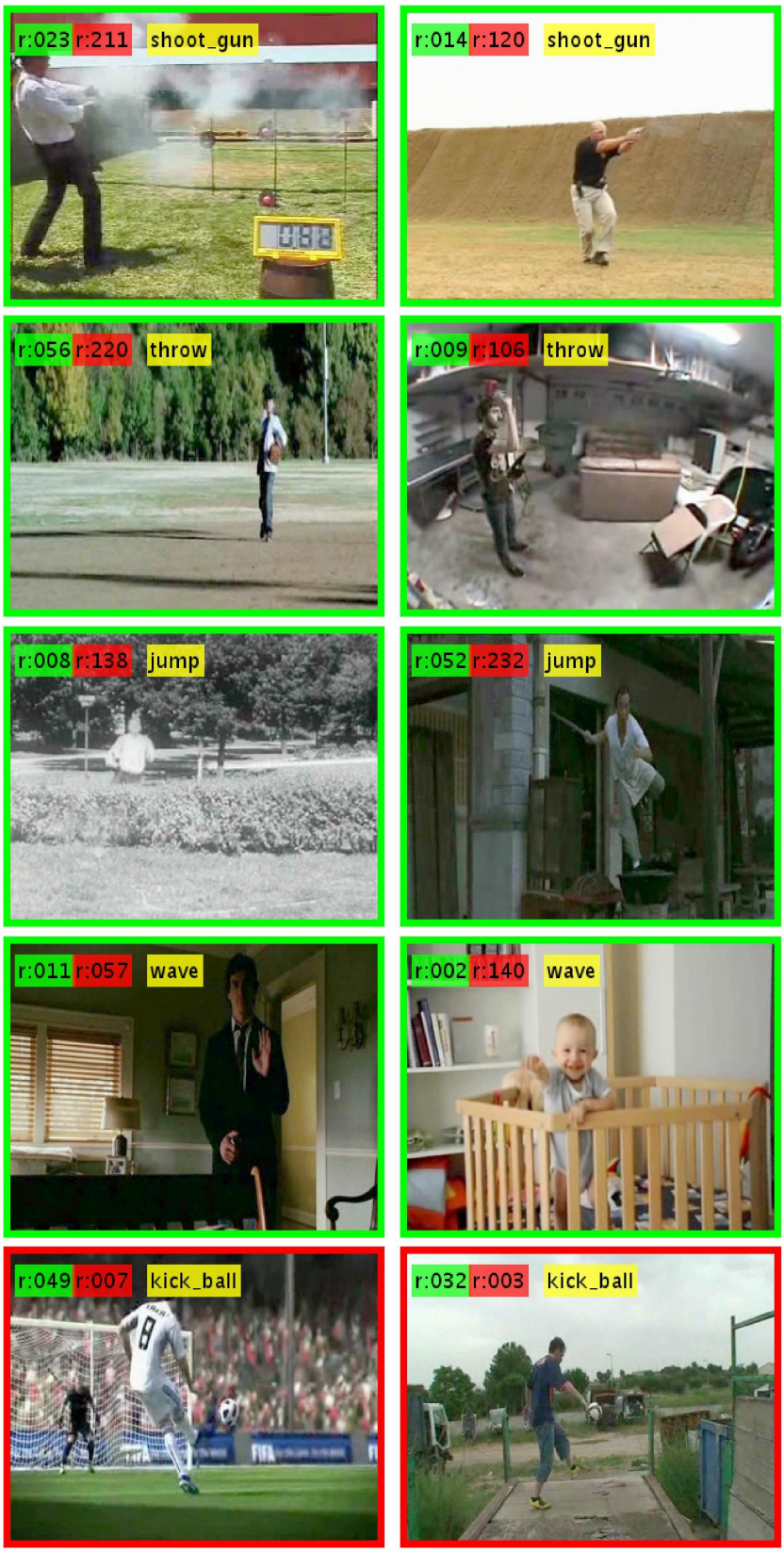


Figure 4.4: Results on JHMDB-GT (split 1). Each line corresponds to an action class. Video frames on the top lines (green) illustrate two test samples per action with the largest improvement in ranking when using P-CNN (rank in green) and IDT-FV (rank in red). Examples on the last line (red) illustrate samples with the largest decreases in the ranking. Actions with large differences in performance are selected according to Figure 4.3. Each video sample is represented by its middle frame.

Figure 4.5: Results on MPII Cooking-Pose [Cherian et al., 2014] (split 1). Examples on the top lines (green) show the 8 best ranking improvements (over all classes) obtained by using P-CNN (rank in green) instead of IDT-FV (rank in red). Examples on the last line (red) illustrate video samples with the largest decrease in the ranking. Each video sample is represented by its middle frame.

4.5 Conclusion

This chapter introduces pose-based convolutional neural network features (P-CNN). Appearance and flow information is extracted at characteristic positions obtained from human pose and aggregated over frames of a video. Our P-CNN description is shown to be significantly more robust to errors in human pose estimation compared to existing pose-based features such as HLPF [Jhuang et al., 2013]. In particular, P-CNN significantly outperforms HLPF on the task of fine-grained action recognition in the MPII Cooking Activities dataset. Furthermore, P-CNN features are complementary to the dense trajectory features and significantly improve the state of the art at the time of publication for action recognition when combined with IDT-FV.

Our study confirms conclusions of Jhuang et al. [2013], namely, that correct estimation of human poses leads to significant improvements in action recognition. This implies that pose is crucial to capture discriminative information of human actions. Pose-based action recognition methods have a promising future due to the recent progress in pose estimation, notably using CNNs [Chen and Yuille, 2014]. An interesting direction for future work is to adapt CNNs for each P-CNN part (hands, upper body, etc.) by fine-tuning networks for corresponding image areas. Another promising direction is to model temporal evolution of frames using RNNs [Donahue et al., 2015].

Chapter 5

Spatio-temporal action localization

5.1 Introduction

In the previous chapter we focused on action classification aiming to answer *what* is happening in a video when the input videos are restricted to action intervals. Here, we address the more challenging task of *spatio-temporal action localization*. In addition to answering the *what*, this task addresses spatial (*where* does the activity take place?) and temporal (*when* does it arise?) localization. Successful action localization is required for many applications, including autonomous car driving, preventing crime, searching our video collections and will eventually enable robots to serve us at home. Most of the current methods and benchmarks for action recognition, however, only address action classification [de Souza et al., 2016; Simonyan and Zisserman, 2014], *i.e.* assuming temporally segmented action intervals as input.

Identifying the beginning and the end of an action naturally suggests the need of temporal models for video sequences. Sequence models have previously been explored for sound, speech and text understanding. In particular, recurrent neural network models (RNNs) have recently shown success for speech recognition [Dahl et al., 2012] and text generation [Sutskever et al., 2011] as well as for image and video captioning [Donahue

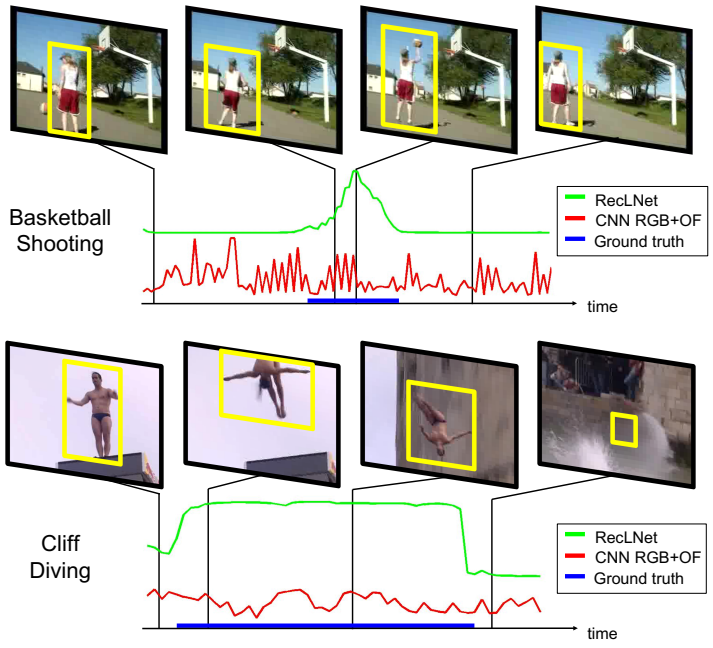


Figure 5.1: Spatio-temporal action localization using a CNN baseline (red) and our RecLNet (green) both applied on the level of person tracks. Our approach provides accurate temporal boundaries when the action happens.

et al., 2015; Karpathy and Fei-Fei, 2015; Vinyals et al., 2015]. RNNs have also been explored for action classification in video [Donahue et al., 2015; Ng et al., 2015], but have shown limited improvements for this task so far.

Action classification may not require sophisticated temporal models if classes can be distinguished solely by the presence of action-specific features. On the other hand, if the structure of the video is required as the output, explicit spatio-temporal models of the video can be beneficial. Recent work [Liu et al., 2016a; Ma et al., 2016; Singh et al., 2016; Yeung et al., 2016; Yuan et al., 2016] has indeed shown improvements in temporal action localization achieved with recurrent models of video sequences. Here we develop and investigate recurrent models for spatio-temporal action localization.

Our goal is to localize the acting person in the video frame and to identify temporal

boundaries of corresponding actions. To this end, we propose a recurrent localization network (RecLNet) with gated recurrent units (GRU) [Cho et al., 2014] for modeling actions on the level of person tracks. Our method starts by person detection and tracking similar to [Gkioxari and Malik, 2015; Peng and Schmid, 2017; Saha et al., 2016; Weinzaepfel et al., 2015]. Differently to previous work, we train our RecLNet to score actions and detect temporal boundaries within each person track (see Figure 5.1). This scoring is achieved by two-stream recurrent units exploiting appearance and motion Fast-RCNN [Girshick, 2015] features pooled from person boxes, while final detections are obtained using our simple and effective temporal localization method composed of filtering and thresholding. We provide a thorough experimental evaluation and analyze the impact of recurrence on spatio-temporal action localization by making the following contributions:

- we show spatio-temporal action localization improvement supported by our RecLNet trained on a track-level and compare different standard and recurrent architectures;
- we empirically diagnose *temporal* localization as being a weakness of existing methods which our method is able to correct;
- our method is complementary to most recent works [Kalogeiton et al., 2017a] mainly focusing on increasing spatial boxes precision and we identify the spatial aspect as being our principal room for improvement;
- results are reported on the two largest datasets for our task, namely UCF101-24 detection [Soomro et al., 2012] and DALY [Weinzaepfel et al., 2016], for both of these datasets our method results in significant improvements over the state of the art.

Related work. Our work is mostly related to methods for human action classification, temporal action localization and spatio-temporal action detection in video. Currently

best performing action classification methods combine IDT features with CNN-based representations of motion and appearance [de Souza et al., 2016; Feichtenhofer et al., 2016a,b; Varol et al., 2017] or pure 3D CNN architecture [Carreira and Zisserman, 2017] but RNNs, having shown promise for the task of gesture recognition in [Pigou et al., 2017], did not show significant improvements for the more general task of action classification. In this work, we compute appearance and motion CNN features extracted on the level of person tracks and use them as input to our RecLNet for action localization.

Temporal action localization aims both to classify and identify temporal extents of actions in longer video clips and several RNN-based methods have shown gains for this task in [Bagautdinov et al., 2017; Ma et al., 2016; Singh et al., 2016; Yeung et al., 2016; Yuan et al., 2016]. For example, Ma et al. [2016], Singh et al. [2016] and Yuan et al. [2016] explore variants of LSTM to improve per-frame action classification whereas the method in [Yeung et al., 2016] learns to directly predict action boundaries. Here we use GRU as recurrent units for temporal modeling of actions in our RecLNet. Unlike previous work on temporal action localization, however, we address the task of localizing actions in space and time.

The list of datasets for this task is limited: UCF101-24 – a subset of UCF-101 [Soomro et al., 2012] used in the THUMOS challenge [Gorban et al., 2015], the DALY dataset [Weinzaepfel et al., 2016]. Other datasets that we are aware of have either a very limited number of examples or consist of temporally trimmed videos. The common strategy in [Gkioxari and Malik, 2015; Peng and Schmid, 2017; Saha et al., 2016; Singh et al., 2017; Weinzaepfel et al., 2015, 2016] is to localize actions in each frame with per-frame action or person detectors and to link resulting bounding boxes into continuous tracks. The temporal localization is then achieved by dynamic programming except in [Weinzaepfel et al., 2015, 2016] who use a temporal sliding windows. Most recent method [Kalogeiton et al., 2017a], instead of relying on per-frame detections, adapts SSD detector [Liu et al., 2016b]

to spatio-temporal anchors and generates human tracks with tubelets linking. In this chapter, we propose a recurrent localization network (RecLNet) that both classifies and localizes actions within tracks supported by a thresholding and filtering method. Our analysis shows that RecLNet provides significant gains due to accurate temporal localization and its complementarity to the recent method of Kalogeiton et al. [2017a] with more accurate spatial localization but approximate temporal localization. Our resulting approach outperforms the state of the art in spatio-temporal action detection [Kalogeiton et al., 2017a; Peng and Schmid, 2017; Saha et al., 2016; Singh et al., 2017; Weinzaepfel et al., 2016] on two challenging benchmarks for this task.

Outline. The chapter is organized as follows. Section 5.2 introduces our RecLNet model, its architecture and our threshold temporal localization technique. Section 5.4 presents our experimental and qualitative results. Section 5.5 draws conclusions.

5.2 Action localization

This section presents our method for spatio-temporal action localization. The overview of the method is illustrated in Figure 5.2. A spatially localized person track (Figure 5.2, row 1) is passed to appearance and optical flow feature extractors (Figure 5.2, row 2). These descriptors feed our localization network composed of 3 layers for each stream and 1 fusion layer. In each stream, the first layer (Figure 5.2, row 3) processes either appearance or flow features before sending them to a 2-layer GRU. The GRU outputs from both layers and both streams are concatenated (Figure 5.2, row 5) and converted by a fully-connected layer (Figure 5.2, row 6) to action probabilities (Figure 5.2, last row).

In the following, we first briefly present the inputs of our method, namely the human tracks and their associated features. Then, we introduce our spatio-temporal action localization method based on the recurrent localization network (RecLNet). Finally, we discuss

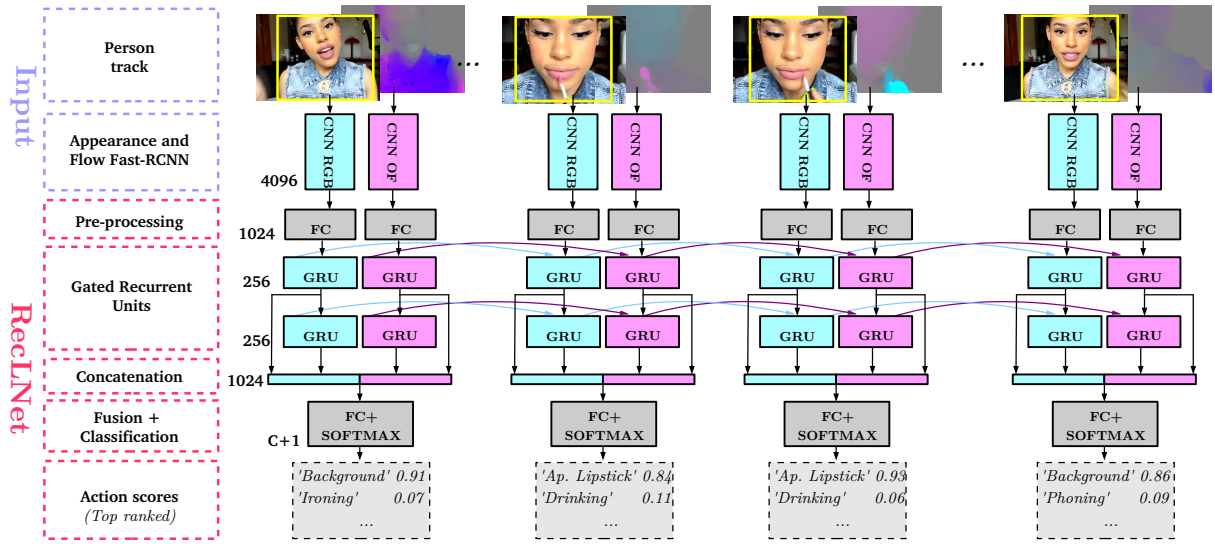


Figure 5.2: Our RecLNet approach for spatio-temporal action localization (output dimensions are shown for each layer). The input is a person track where for each spatially localized actor bounding-box, we extract appearance (RGB) and optical flow (OF) CNN features. Each stream is pre-processed by a fully-connected layer and fed into a two-layer GRU. The outputs from both GRU levels and from both streams are concatenated then classified with a fully-connected layer combined with softmax scoring. Outputs are class probabilities for each frame.

how to post-process the action detection scores in order to output final spatio-temporal human action tubes.

5.2.1 Person tracks

We obtain human tracks as in [Saha et al., 2016; Weinzaepfel et al., 2016] by first running an action or person detector in each video frame, and then linking detections into human tracks spanning the entire video clip. Our method aims at segmenting the track in time to obtain temporal action boundaries. For this purpose, we associate to each time frame its corresponding bounding-box in the human track. This box is used as a pooling region to extract per-frame descriptors. Such features are obtained by Fast-RCNN [Girshick, 2015] appearance and flow ROI-pooling. The details on state-of-the-art human tracks and their associated features used in this work will be discussed in Section 5.3.

5.2.2 Temporal action localization

This section presents our model for action localization and its training procedure on the level of person tracks. In our work, we choose to adopt a model with memory links, like in a recurrent neural network (RNN), which we call recurrent localization network (RecLNet), to temporally localize actions. RNNs have been demonstrated to successfully model sequential data especially for language tasks such as speech recognition [Dahl et al., 2012], machine translation [Bahdanau et al., 2014] or image captioning [Karpathy and Fei-Fei, 2015]. Given this success, we believe that recurrent networks are well-suited for modeling temporal sequences of appearance and motion in person tracks. Our final RecLNet model is composed of gated recurrent units (GRU) described below.

The LSTM and GRU architectures. Features from a human track of length T can be seen as an input sequence $\mathbf{x} = (x_1, ..., x_T)$ where for each x_i we aim to provide action activation h_i, forming the output $\mathbf{h} = (h_1, ..., h_T)$. To generate such an output, we investigate two types of recurrent networks for our RecLNet, namely LSTM and GRU as defined below.

In the long short-term memory (LSTM) architecture [Hochreiter and Schmidhuber, 1997], one memory cell and three 'gates' give LSTM the ability of discovering long-range temporal relationships by reducing the vanishing gradient problem compared to vanilla RNN. This is a useful property in our task since we need to handle particularly long human tracks. The LSTM cell takes as input features x_t at time step t together with the

output h_{t-1} at the previous time step $t-1$ and operates as follows:

$$\begin{aligned} f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{5.1}$$

where σ is the sigmoid function and f_t, i_t, o_t, c_t are the 'forget gate', 'input gate', 'output gate' and 'memory cell', respectively. Matrices $W_.$, $U_.$ and vectors $b_.$ denote the parameters of the cell.

The GRU [Cho et al., 2014] cell differs from LSTM by the absence of the output gate and operates as follows:

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z) \\ r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) \\ h_t &= z_t \odot h_{t-1} \\ &\quad + (1 - z_t) \odot \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \end{aligned} \tag{5.2}$$

where z_t and r_t are the 'update gate' and 'reset gate', respectively. The GRU cell is simpler, has fewer parameters and has shown some improvement on video tasks as in [Tokmakov et al., 2017] while in other settings mitigated results have been reported when compared to LSTM [Chung et al., 2014] claiming that the choice of using LSTM or GRU recurrent unit may depend heavily on the dataset and corresponding task. An empirical comparison of LSTM and GRU cells is given in Section 5.4.

We define the **Recurrent Localization Network** (RecLNet) as a multi-class recurrent network trained to classify actions against background. As shown in Figure 5.2, at each time step, appearance (RGB) and optical flow (OF) Fast-RCNN networks take bounding

boxes of human tracks as object proposals and extract features. Each stream (RGB and OF) is processed independently by feeding Fast-RCNN outputs (FC7 layers) to our RecLNet which produces action scores at each time step. Each stream of our RecLNet consists of a fully-connected layer that pre-processes the input features (appearance or flow) and a 2-layer GRU. The output of the two recurrent layers is concatenated to a $2 \times M$-dimensional stream output (where M is the memory size). We stack several recurrent layers as it is common practice in many applications [Graves et al., 2013; Ng et al., 2015] while we set their number to two as we want to limit the complexity of our architecture. Finally the appearance and flow branch outputs are concatenated and a last fully-connected layer with softmax converts the recurrent output to an action probability for all actions (and background). The network is trained using the standard negative log-likelihood loss w.r.t. all C action classes and background boxes:

$$L(W) = -\sum_{i=1}^{N}\sum_{c=1}^{C+1} \mathbb{1}\{y_i = c\} \log P(y_i = c|x_i, W), \tag{5.3}$$

with probabilities defined by softmax:

$$P(y_i = c|x_i, W) = \frac{\mathrm{e}^{f_c(x_i,W)}}{\sum_{c'=1}^{C+1} \mathrm{e}^{f_{c'}(x_i,W)}}. \tag{5.4}$$

Here $f_c(x_i, W)$ is the output of the last fully-connected layer corresponding to class c, x_i and y_i denote features and labels at time step i, symbol W denotes network parameters, N is the number of boxes in the training set, C is the number of action classes, $\mathbb{1}\{.\}$ is the indicator function.

To train RecLNet, we set the ground-truth targets y_i in the following way. We assign label c to a frame bounding-box detection from a person track if it overlaps more than 0.3 spatial IoU with one ground truth annotation from action label c, otherwise this input is considered as background. Having low IoU threshold allows us to get more positives. Note that bounding-box detections outside of the temporal ground-truth intervals are

considered as background.

Appearance and optical flow fusion. Single-stream networks are first independently trained then we consider three fusion methods to combine their appearance (RGB) and optical flow (OF) ouputs. The *average* simply averages both softmax RGB and OF network outputs. Both next fusion methods are trained on top of the two stream networks (their weights remaining fixed). The *gating layer* learns a per-class linear combination of the RGB and OF network outputs then applies softmax. The third method, *fusion layer*, trains a fully-connected classification layer (followed by softmax) on top of the concatenated memory units outputs from each stream network (concatenation layer in Figure 5.2).

5.2.3 Post-processing of human action tracks

RecLNet output represents action scores at each time step of a human track. While being spatially localized, these scored tracks have to be segmented in time in order to produce the final spatio-temporal detections. For the track ranking, it is also necessary to score each of the final detections.

In this section, we then describe our method to get the final spatio-temporal detections and their associated score. Temporal localization is performed within each track.

Threshold. Let a be the action to segment and τ a human track spanning from frame 1 to T of scores $\mathbf{s}_\tau = (s_1^a, ..., s_T^a)$ with s_t^a the score associated to the person box at time t. The goal of temporal segmentation is to extract from τ one or several sub-tracks ν (of time interval $[t_1^\nu, t_2^\nu] \subseteq [1, T]$) as final spatio-temporal detections.

For this purpose, we first get smoother scores $\hat{s}_i^a$ by applying a median window filtering on the s_i^a. Then, we temporally segment the track by selecting consecutive boxes with scores above a certain threshold θ, while others are rejected. More formally, considering that consecutive boxes $(b_{t_0}, ..., b_{t-1})$ from time t_0 to $t-1$ have already been added to the

sub-track ν, the next box b_t is added to ν if $\hat{s}_t^a \geq \theta$. Otherwise, ν ends and is returned as final detection. This method allows us to break the initial track into several sub-track candidates of arbitrary lengths and is then able to capture several repetitive action instances happening on the same human track (like *drinking*, *applying make up*, *playing harmonica*, see Figure 5.4 and Figure 5.5). Here, model outputs must be smooth in order to get accurate action temporal boundaries. The recurrent units (like GRU, LSTM) are then well-suited for this localization method while appearance and flow CNNs output is generally noisier. Our temporal localization technique is referred to as *threshold*.

Temporally segmented track scoring. In order to rank detections and perform the average-precision (AP) evaluation, we need to set a score r_ν for each final spatio-temporal detection ν. Following, e.g., [Saha et al., 2016], we define r_ν as the average of the top 40 action scores s_i^a contained in ν. The same scores are also used for non-maximum-suppression (NMS) of spatio-temporal detection candidates based on their scores and overlap. For NMS, we use spatio-temporal Intersection-over-Union[1] as overlap criterion.

5.2.4 Implementation details

RecLNet parameters. The FC7 output of each Fast-RCNN is 4096-dimensional and the first fully-connected layers convert each stream to a 1024-dimensional vector. The memory size M is equal to 256 and the last fully-connected layer input is 1024-dimensional ($2 \times 2 \times M$, i.e. the memory stack from both GRU layers of both streams).

Training. Both appearance and flow branches of RecLNet are separately trained using the Adam optimizer [Kingma and Ba, 2014] with a weight decay set to 5.10^{-4} to avoid overfitting. Note that to train the single-stream networks, we halve the input dimension of the last fully-connect layer of RecLNet (concatenation layer in Figure 5.2). Training batches contain 100 different tracks of temporal length 20. Backpropagation through time

[1] The spatio-temporal IoU between two tracks is defined as a product of temporal IoU between the time segments of the tracks and average spatial IoU on the frames where both tracks are present.

(BPTT) is then performed every 20 time steps.

Detection. In all experiments, the NMS overlap threshold is set 0.2, the action localization *threshold* θ is set to 0.1 and the median window size is 25.

Optical flow. To obtain the optical flow data, we compute horizontal and vertical flow for each consecutive pair of frames using the approach of Brox et al. [2004]. Following [Gkioxari and Malik, 2015; Weinzaepfel et al., 2015], flow maps are saved as 3-channels images corresponding to optical flow in x and y direction and its magnitude with all the values restricted to the interval $[0, 255]$.

5.3 Experimental setup

In Section 2.2.3, we have described the UCF101-24 [Soomro et al., 2012] and DALY [Weinzaepfel et al., 2016] datasets used for evaluation of our method. For both datasets, we provide experimental details on the use of ground truth annotation and data preprocessing.

5.3.1 UCF101-24

Short vs. long classes. While some of the UCF101-24 action classes are short (*basketball dunk* or *tennis swing*) other continuous actions (*biking* or *rope climbing*) typically last for the full duration of the video. To better evaluate the temporal localization on UCF101-24, we define a subset with short action classes that on average last less than a half of the video length. This subset contains six actions (*Basketball*, *Basketball Dunk*, *Cricket Bowling*, *Salsa Spin*, *Tennis Swing* and *Volleyball Spiking*) and we call them "short classes". In Section 5.4, we evaluate localization for all 24 action classes and for short classes separately.

Action-specific human tracks. To enable direct comparison of our method with Saha et al. [2016], for UCF101-24 experiments we use the same person tracks as in [Saha et al.,

2016]. These tracks are obtained by linking per-frame action detections with dynamic programming (DP). Action detections are obtained with the Fast-RCNN method [Girshick, 2015] trained with appearance and flow input for the task of spatial action localization. Note that the action bounding-boxes come from the appearance network while their detection scores are boosted by combining them with the motion detection scores. As our model performs its own temporal localization, we do not run the temporal segmentation of Saha et al. [2016] (second DP pass) and keep 5 action proposals per action covering the whole video. Per-frame input features for our RecLNet are obtained from the same Fast-RCNN detector used for track detection.

5.3.2 DALY

Human tracks and associated features. To enable direct comparison with Weinzaepfel et al. [2016] we use tracks provided by Weinzaepfel et al. [2016]. An accurate Faster-RCNN [Ren et al., 2015] person detector is trained on the large MPII human pose dataset [Andriluka et al., 2014]. The tracks are obtained by linking human detections with a tracking-by-detection approach. The latter consists in combining the score from the human detector with an instance level detector (a linear SVM learned on the features from the last fully-connected layer of the Faster R-CNN). The current box is tracked in the next frame by selecting the highest scored box according to the combined score around the current box location. Note that both detection and tracking are performed on appearance images only. Similarly to UCF101-24, the appearance and flow features are obtained with a Fast-RCNN detector trained to detect actions on the annotated DALY frames. The action scores from this detector will be used as a baseline in Section 5.4 and referred to as *CNN RGB+OF*. Following [Mathias et al., 2014], we compensate the annotation bias by adapting human box sizes to action annotation. To achieve this, we train a linear regression of bounding boxes. For each DALY annotated keyframe from the

training set we associate its most overlapping bounding-box from the human tracks (with at least 0.5 IoU) to train the linear model.

DALY tracks labeling. To compensate for the sparse action annotation in DALY, we extend ground truth to all frames of the actions with automatic tracking. We use the Siamese Fully-Convolutional Network for online non-class-specific tracking method[2] [Bertinetto et al., 2016] initialized on all ground-truth keyframes. For a given action instance, we aggregate all tracks at each frame by a median bounding box. To assess the quality of generated ground-truth tracks, we computed the track proposals recall (92.2% at 0.3) within the action time interval. Also, the IoU overlap between ground-truth tracks and annotated keyframes is greater than 0.8 (resp. 0.5) for 91.4% (resp. 99.0%) of keyframes.

5.4 Experiments

In this section, we first evaluate the impact of the *threshold* temporal localization, recurrent architecture and the fusion method (Section 5.4.1). We then evaluate the potential gain due to *action classification* (Section 5.4.2) followed by an extensive analysis on *temporal localization* (Section 5.4.3). Next, we show an improvement if using I3D and temporal person tracks (Section 5.4.4). We compare our RecLNet to the state of the art on UCF101-24 and DALY datasets (Section 5.4.5). Section 5.4.6 concludes this experimental part by presenting qualitative results.

5.4.1 Impact of localization method, recurrent architecture and fusion

Localization method. In [Saha et al., 2016], temporal localization is performed using the Viterbi algorithm on top of action tracks spanning the whole video. To temporally trim these tracks, one binary label (action vs. background) is associated to each bounding-box

[2] https://github.com/bertinetto/siamese-fc

Method	Localization	UCF101-24 @0.3	UCF101-24 @0.5
CNN RGB+OF	Viterbi [Saha et al., 2016]	55.5	35.9
CNN RGB+OF	threshold	60.4	42.7

Table 5.1: Localization method analysis on UCF101-24. We compare the temporal localization of Saha et al. [2016] which uses *Viterbi* algorithm to our *threshold* method. We apply the two methods on original detection scores from Saha et al. [2016] (*CNN RGB+OF*). Evaluation is spatio-temporal action localization at IoU 0.3 and 0.5 (mAP).

by maximizing an energy function where unary potentials represent box action scores while pairwise potentials control the final action length (or track smoothness). This smoothness score is weighted per class and then might over control the track length towards the action durations seen on the training set while, in our method, median filtering encodes smoothness more explicitly and suffers less from the dataset biases described in 5.3.1.

In Table 5.1, we refer to the track action scores from Saha et al. [2016] as *CNN RGB+OF*, since they come from two Fast-RCNN where appearance and optical flow outputs have been fused. We recall that to enable a direct comparison with Saha et al. [2016] we are using the same human tracks as Saha et al. [2016] but score them differently (see Section 5.3 for details). Here, we compare the Viterbi algorithm originally used in Saha et al. [2016] for temporal localization and our *threshold* method on original scores (*CNN RGB+OF*). Interestingly we observe that our *threshold* technique improves over their original results by +4.9% (resp. +6.8%) at IoU 0.3 (resp. 0.5) and conclude that thresholding combined with median filtering works better than this typically used localization method while being simpler. The next paragraph shows the impact of differently scoring, among others, tracks from Saha et al. [2016]. In the following, only our *threshold* temporal localization is used.

Recurrent architecture. Table 5.2 compares models based on LSTM and GRU units (see Section 5.2 for details). As a baseline we use a model with no temporal connections

Method	UCF101-24 (@0.3/@0.5)			DALY (@0.3/@0.5)		
	OF	RGB	RGB+OF	OF	RGB	RGB+OF
CNN Saha et al.	58.6/41.9	55.2/39.9	60.4/42.5	-	-	-
CNN	-	-	-	9.8/3.7	11.4/6.3	13.6/7.5
FC	59.9/42.4	58.4/42.0	64.1/45.2	11.5/5.3	12.9/6.7	16.5/8.5
LSTM	65.2/43.7	58.1/42.8	66.5/46.1	13.0/4.2	11.4/6.2	16.2/7.7
GRU	**67.2/44.6**	**59.7/43.2**	**67.4/46.3**	**14.4/6.1**	**14.2/6.7**	**17.6/9.0**

Table 5.2: Performance on UCF101-24 and DALY for different recurrent architectures (*LSTM* and *GRU*) and baselines with no temporal connections (*CNN* and *FC*). Models are evaluated with flow (OF) and/or appearance (RGB) features as input (RGB+OF averages both stream outputs) for spatio-temporal localization at IoU 0.3 and 0.5 (mAP).

but similar architecture: the recurrent units are replaced by a stack of 2 fully-connected layers with non-linearity and has the same number of parameters as the GRU unit. This additional fully-connected classifier is referred to as *FC*. For UCF101-24 evaluation, we again report original track scores (*CNN [Saha et al., 2016]* as in Table 5.1) while for DALY, *CNN* represents the Fast-RCNN scores similarly retrained as Weinzaepfel et al. [2016] on their human tracks (see Section 5.3.2 for details). Input modalities are either, optical flow features (*OF*), appearance features (*RGB*) or both (*RGB+OF*) where action scores from the two stream outputs are averaged in this case.

In the following, we analyse spatio-temporal action localization evaluated on UCF101-24 and DALY datasets at spatio-temporal IoU of 0.3. We first observe that training an additional classifier (*FC*) is better than directly taking Fast-RCNN outputs (*CNN*) and improves performance by +3.2% and +2.9% on UCF101-24 and DALY respectively when both appearance and optical flow streams are used (*RGB+OF*). Also, non-recurrent baselines (*CNN* and *FC*) perform worse than the recurrent variants (*LSTM* and *GRU*). When using both *RGB+OF* as input, the gain due to recurrence is +7.0% and +4.0%

on UCF101-24 and DALY respectively when comparing *GRU* to *CNN* and +3.3% and +1.1% when comparing *GRU* to *FC*. Interestingly, recurrent unit improvement is larger on optical flow +6.5% on UCF101-24 and +2.9% on DALY when comparing *GRU* to *FC*. This shows that temporal memory links are beneficial for better spatio-temporal action localization performance. We also note, as this is often the case when working with videos (e.g. Tokmakov et al. [2017]), that *GRU* outperforms *LSTM* while being a simpler model. Similar observations are made at IoU 0.5 but with limited gain on DALY as achieving good results at this precision is really challenging on this dataset. State-of-the-art methods report lower IoU values which are evaluated in Section 5.4.5.

This experiment has shown *GRU* memory unit achieves the best action localization accuracy. In the following, *GRU* will then be used in our recurrent model for determining the action temporal extent in all experiments.

Fusion method. Table 5.3 explores the different fusion strategies described in Section 5.2 to combine appearance (RGB) and flow (OF) features in our localization model (with *GRU* layer which was validated in the previous experiment). We first note that all fusion methods improve action localization results on both UCF101-24 and DALY. However, the simple averaging method has difficulties to capture feature complementarity between appearance and flow features especially on UCF101-24 where it gets +1.3% improvement at IoU 0.5 but only +0.2% at IoU 0.3 compared to the best performing OF features while the *gating layer* is on par with the average fusion. Finally, the *fusion layer* captures better feature complementarity and improves action localization mAP on UCF101-24 and DALY respectively by +1.8% and +4.3% (+1.9% and +3.2% at IoU 0.5) compared to the single flow features and by +9.3% and +4.5% (+3.3% and +2.6% at IoU 0.5) compared to appearance features.

In the following, our final recurrent model uses the *fusion layer* and is referred to as *RecLNet* in all experiments.

Finally, when using our *threshold* method, re-scoring the tracks with our RecLNet instead of taking Fast-RCNN outputs (*CNN*) as in [Saha et al., 2016; Weinzaepfel et al., 2016] improves spatio-temporal action localization results by +8.6% and +5.1% on UCF101-24 and DALY respectively (from Table 5.2). The improvement is even larger (+13.5%) when comparing RecLNet accuracy (69.0%) to the original result using scores and temporal localization from Saha et al. [2016] (55.5% on UCF101-24 from Table 5.1).

Given that we are using the same tracks, this first gives an insight that our method improvement compared to [Saha et al., 2016; Weinzaepfel et al., 2016] might be due to *temporal* localization. This question is further studied in the next sections.

Features	Fusion	UCF101-24 (@0.3 / @0.5)	DALY (@0.3 / @0.5)
OF	-	67.2 / 44.6	14.4 / 6.1
RGB	-	59.7 / 43.2	14.2 / 6.7
RGB+OF	*average*	67.4 / 46.3	17.6 / 9.0
RGB+OF	*gat. layer*	67.3 / 46.1	17.6 / 8.2
RGB+OF	*fusion layer*	**69.0 / 46.5**	**18.7 / 9.3**

Table 5.3: Performance on UCF101-24 and DALY for different features and fusion strategies *average*, *gating layer* and *fusion layer* (see Section 5.2 for details). Models with *GRU* memory units take optical flow (OF) and/or appearance (RGB) features as input and are evaluated for spatio-temporal localization at IoU 0.3 and 0.5 (mAP).

5.4.2 Analyzing the performance gain: *action classification*

Spatio-temporal action localization is composed of *spatial localization*, *action classification* and *temporal localization*. In this section we focus on action classification and analyze its performance given person tracks and pre-defined temporal action boundaries.

Evaluating action classification. Here, we do not require temporal localization by restricting the track to the ground-truth time interval in order to evaluate the performance gain due to action classification. As described in Section 5.2, the track scoring is obtained

with the average over the top 40 action scores s_i^a in the trimmed interval. Table 5.4 shows our method (*RecLNet*) increases action classification by approximately +2% compared to scores from Saha et al. [2016] (*CNN RGB+OF*) when evaluated on UCF101-24 classes. This improvement is moderate compared to the +13.5% spatio-temporal localization boost of Section 5.4.1.

IoU	0.2	0.3	0.4
CNN RGB+OF [Saha et al., 2016]	84.3	81.6	75.6
RecLNet	86.9	83.7	77.6

Table 5.4: Performance on UCF101-24 for clips trimmed to the ground-truth temporal interval (mAP at IoU 0.2, 0.3 and 0.4).

Table 5.5 shows the same experiments on DALY. We observe that our *RecLNet* method performs slightly worse (from −0.1% to −0.9%) compared to the *CNN RGB+OF* baseline (see Section 5.3.2 for details). This might be due to DALY track labeling which, contrary to UCF101-24, is obtained by automatic tracking of ground-truth sparse annotations (see Section 5.3.2) which introduces some noise and can slightly affect action classification. We, therefore, conclude that the spatio-temporal action localization gain of our method over the baseline, demonstrated in Section 5.4.1, cannot come from action classification.

IoU	0.2	0.3	0.4
CNN RGB+OF	65.5	64.8	63.6
RecLNet	65.4	64.5	62.7

Table 5.5: Performance on DALY for clips trimmed to the ground-truth temporal interval (mAP at IoU 0.2, 0.3 and 0.4).

Overall, this experiment confirms results in the literature [Ng et al., 2015] that recurrent units (RNN, LSTM, GRU) do not really improve the performance for action classification. The spatial localization (the person boxes positions) being fixed, the gain

of our method for spatio-temporal action localization has to come from better temporal localization. This observation is analyzed in detail in Section 5.4.3.

5.4.3 Analyzing the performance gain: *temporal localization*

In this section, we show that the main gain of RecLNet comes from better temporal localization and demonstrate its complementarity with the state of the art [Kalogeiton et al., 2017a]. We first motivate this study by explaining the UCF101-24 bias, then a per-class analysis compares our RecLNet to [Kalogeiton et al., 2017a] and to the method we build on [Saha et al., 2016]. Finally, we investigate the potential room for improvement.

UCF101-24 bias. As described in Section 5.3.1, UCF101-24 contains only six short action classes lasting less than half of the video duration, while 17 actions (74% of the dataset classes) span at least 70% of the video length in which 11 of them (48% of the dataset classes) span even more than 90% of the video length. UCF101-24 results are then biased toward long actions which do not require temporal localization. Indeed, tracks spanning the whole video already achieves good temporal localization for these classes (note that such detections would totally fail on the DALY dataset). To avoid this bias and to focus on challenging cases of temporal localization, we next compare our method to [Kalogeiton et al., 2017a; Saha et al., 2016] on the subset of short classes along with whole dataset.

Per-class performance on UCF101-24. Table 5.6 compares the per-class results of our approach with [Saha et al., 2016] since we are using their human tracks and with the best performing state-of-the-art method [Kalogeiton et al., 2017a] (see Section 5.4.5 for comparison with the state of the art). We report mAP for short classes only (*mAP-short*) and also for all classes (*mAP*). We outperform [Saha et al., 2016] by +31.7% on average on short classes and by +67.3% on "Basketball". As we have already seen previously, the overall improvement is +13.5%. We can observe that for some "long" actions the

Actions	Saha et al. [2016]	RecLNet	Kalogeiton et al. [2017a]
Basketball	9.8	**77.1**	18.0
BasketballDunk	11.9	**45.2**	44.1
CricketBowling	7.3	**50.4**	23.5
SalsaSpin	18.8	**25.4**	20.7
TennisSwing	27.2	38.4	**39.1**
VolleyballSpiking	1.0	**34.9**	1.0
mAP-short	12.7	**44.4**	24.4
Biking	56.1	65.0	**72.0**
CliffDiving	67.1	77.8	**82.0**
Diving	82.9	**93.4**	84.8
Fencing	80.5	80.6	**81.3**
FloorGymnastics	**99.7**	99.6	99.2
GolfSwing	59.8	66.2	**76.7**
HorseRiding	88.1	93.0	**94.6**
IceDancing	**64.1**	55.4	60.8
LongJump	40.1	65.6	**82.5**
PoleVault	49.6	56.9	**90.4**
RopeClimbing	78.4	**99.3**	92.4
SkateBoarding	85.1	91.5	**92.0**
Skiing	70.7	**84.8**	84.0
Skijet	**92.7**	89.3	80.4
SoccerJuggling	84.9	**92.3**	91.4
Surfing	55.4	51.6	**65.9**
TrampolineJumping	36.9	47.5	**62.3**
WalkingWithDog	67.1	**80.5**	76.8
mAP	55.5	**69.0**	67.3

Table 5.6: Per-class AP for IoU 0.3 on UCF101-24 comparing our RecLNet to state of the art [Saha et al., 2016; Kalogeiton et al., 2017a]. We report mean AP for short classes, *mAP-short*, and all classes, *mAP*. The 6 short classes are reported on the top lines (before *mAP-short* column).

performance drops slightly. Similarly, we outperform the state of the art [Kalogeiton et al., 2017a] by only +1.7% overall while we reach a large improvement of +20% on short classes.

	All classes			Short classes		
IoU	0.3	0.5	0.75	0.3	0.5	0.75
Kalogeiton et al. [2017a]	67.3	**51.4**	**22.7**	24.4	2.6	**0.0**
RecLNet	**69.0**	46.5	10.3	**44.4**	**6.6**	**0.0**

Table 5.7: Performance on UCF101-24 when differentiating short classes from others. Spatio-temporal action localization (mAP) is evaluated at IoU 0.3, 0.5 and 0.75).

Table 5.7 now compares our model to the best performing method [Kalogeiton et al., 2017a] at higher IoU. We observe that even if short classes get extremely difficult to detect at highest IoU (0% mAP at IoU 0.75), at IoU 0.5, our model still outperforms the state of the art by +4.0% on short classes while loosing −4.9% overall (Section 5.4.5 studies this latter result when comparing to the state of the art).

These experiences on per-class performance show that our model is able to improve [Saha et al., 2016] by producing more accurate temporal action localization. Also, while Kalogeiton et al. [2017a] takes advantage from strong features and spatially accurate person boxes to get excellent accuracy on long classes (e.g. compared to Saha et al. [2016]), its performance on short classes suffers from approximate temporal localization. This demonstrates the potential complementarity between our RecLNet and the current best performing method [Kalogeiton et al., 2017a]. The room for improvement of RecLNet will be studied in the next section.

Correct temporal loc.	Correct spatial loc.	Correct class	class mAP @0.75 short	class mAP @0.75 all
-	-	-	0.0	10.3
✓	-	-	11.1	16.7
-	✓	-	9.7	**53.2**
-	-	✓	0.0	17.1
✓	-	✓	16.6	25.2
-	✓	✓	18.1	60.8
✓	✓	-	54.0	73.3
✓	✓	✓	56.6	77.1

Table 5.8: RecLNet performance on UCF101-24 short and all classes when considering different components to be correct. Spatio-temporal action localization (mAP) is evaluated at IoU 0.75). **Spatial** localization is the largest room for improvement of our method.

Toward action localization improvement. This paragraph analyzes how to improve the spatio-temporal action localization performance of our model especially when evaluating at very high IoU (0.75). To distinguish the potential improvements, we can consider *action classification*, *spatial localization* and/or *temporal localization* as being correct. We proceed as follow. When we suppose perfect *action classification*, we set final spatio-temporal detection score r_ν (see Section 5.2.3) to 0 for all false positives (note this is equivalent to the recall). Let o_s (resp. o_t) be the spatial IoU overlap of a final detection with its ground-truth. Thereby, when considering perfect *spatial* (resp. *temporal*) localization, if o_s (resp. o_t) is greater than 0.3, it is then set to 1 in the spatio-temporal IoU computation. We choose 0.3 as it seems fair enough 1) not to get a track completely shifted in time ($o_t \simeq 0$) for which the spatial IoU o_s would have been computed on only one frame or a few and 2) to ensure that the track at least approximately spatially "follows" ($o_s \neq 0$) the person. Table 5.8 presents the combinations of these assumptions. First, assuming correct *temporal localization* or *action classification* only improves our model performance by respectively +6.4% and +6.8%. However, the *spatial* assumption is by far

the best room for improvement of our model (+42.9%) and achieves 53.2% accuracy. It also shows that one reason why our method gets 0% accuracy on short classes is because of inaccurate human track bounding-boxes as the spatial assumption improves it to 9.7%. Of course, combining several assumptions further improve the performance (note that last line does not reach 100% since the track recall is not 100% and that the above "0.3 criterion" eliminates some track candidates). However, we observe that combining correct *temporal localization* and *action classification* (25.2%) is still far from the single *spatial localization* assumption (53.2%). This experiment confirms that improving spatial human detection is the direction to take further enhancements and demonstrates that temporal localization is already a strong component of RecLNet.

This section has shown that the UCF101-24 bias is not in favour of our model while the short classes are by far the hardest to localize and then are source of potentially large improvement for current action localization methods. Also, we observed that our model already benefits from accurate temporal localization while its considerable room for improvement is the spatial localization component (the temporal localization being the one with the less potential for RecLNet). Consequently, the state of the art [Kalogeiton et al., 2017a], which mostly relies on spatially more accurate human tracks and better features that the ones we build on [Saha et al., 2016] (as previously compared in Table 5.8), and our model, which greatly improves temporal localization, are definitely complementary.

5.4.4 Improved tracks and descriptors

By using the same tracks and features as input, the previous sections have shown our RecLNet is able to correct the temporal localization weakness of state-of-the-art methods. Here, we substitute these tracks and features (introduced in Section 5.2.1 and 5.3) by improved models. In the following, we first explain the track modification then the features substitution and finally analyze their impact.

Person action tracks with temporal integration. In order to improve person tracking, we integrate temporal information in the detector. Supported by our analysis in Section 5.4.3, showing that better spatial localization would greatly improve our results, and by Kalogeiton et al. [2017a] who show large detection improvement by stacking features coming from several neighboring frames, we design a new approach for tracks. In the same spirit as Kalogeiton et al. [2017a] who stack frame features in the SSD detector [Liu et al., 2016b], we adapt Faster-RCNN [Ren et al., 2015] to perform accurate detection by temporally integrating stacks of K images. We introduce the following modifications to the Faster-RCNN pipeline. First, the inference pass, producing the feature map on which the ROI-pooling applies, is performed independently on K consecutive frames. The K output feature maps are stacked along the channel dimension and will serve at computing scores and regressions. Second, the RPN computes at each anchor location a global objectness score for the stack and K regressions. Third, anchors labels are computed based on the mean overlap between the K ground-truth boxes and the K regressed proposals. Fourth, the 3D proposals are further regressed to action proposals (tubelets) by computing one action score and K regressions per class. Tubelets are linked into action tracks using the code of Kalogeiton et al. [2017a]. This is an online method that iteratively aggregates tubelets sorted by action score to a set of current links based on spatio-temporal overlap. The detector is used on the ResNet-101 architecture [He et al., 2016] and we choose $K = 5$ since it is a good trade-off between tubelet quality and efficiency. Since only sparse keyframes are spatially annotated on DALY, automatic tracking is performed to propagate the ground truth (see Section 5.3.2). Here, stacks used at train time on DALY are then composed of one annotated keyframe and its 2 frames tracked forward and backward. These action tracks are referred to as *stack* in the following.

I3D features. Recent results have shown large action recognition improvement using the I3D [Carreira and Zisserman, 2017] architecture. We replace the per-frame features by

descriptors extracted with the I3D RGB and flow networks both trained on the Kinetics dataset [Kay et al., 2017]. We extract features after the 7-th inception block, before the max-pooling, as it is a good balance between deepness, for strong classification results, and accurate resolution, for precise track pooling. After this block, the temporal receptive field is roughly one hundred frames. As input, we use a spatial resolution of 320×240 pixels resulting in feature maps of size 20×15 with 832 channels. These feature maps are extracted at intervals of 4 frames. The box temporally aligned with the middle of the interval is used as pooling region to obtain a 832-dimensional I3D descriptor representing the track at this time step.

			UCF101-24			DALY		
method	feat.	tracks	0.3	0.4	0.5	0.1	0.2	0.3
RecLNet	f. based	f. based	69.0	57.5	46.5	28.4	24.6	18.7
RecLNet	I3D	f. based	72.3	63.2	50.8	37.9	33.9	26.8
RecLNet	I3D	*stack*	**77.4**	**68.5**	**57.5**	**41.1**	**37.6**	**31.0**
FC	I3D	f. based	71.8	62.3	49.8	35.5	32.1	25.0
FC	I3D	*stack*	75.4	67.4	56.3	38.7	34.9	28.6

Table 5.9: Spatio-temporal action localization (mAP) performance on UCF101-24 (at IoU 0.3, 0.4 and 0.5) and DALY (at IoU 0.1, 0.2 and 0.3) when substituting the frame-based features by I3D and the frame-based tracks by *stack*. Results are reported for RecLNet and the fully connected classifier (FC).

Experimental evaluation. Here, frame-based features and frame-based tracks we refer to are the ones described in Section 5.3. Table 5.9 first shows that when substituting the baseline features by the I3D descriptors our RecLNet model obtains a performance boost of around 4% on UCF101-24 (+4.3% at IoU 0.5) and 8-9% on DALY (+8.1% at IoU 0.3). This result is inline with [Gu et al., 2018]. When we further replace the frame-based tracks by ours obtained with stack-Faster-RCNN (*stack*), RecLNet results get a second improvement of 5-7% on UCF101-24 and 3-4% on DALY. This gain of performance val-

IoU	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.75
Weinzaepfel et al. [2015]	54.3	51.7	46.8	37.8	-	-	-	-
Hou et al. [2017]	54.7	51.3	47.1	39.2	-	-	-	-
Zolfaghari et al. [2017]	65.2	59.5	47.6	38.0	-	-	-	-
Weinzaepfel et al. [2016]	71.1	-	58.9	-	-	-	-	-
Peng and Schmid [2017]	78.8	77.3	72.9	65.7	-	-	-	-
Saha et al. [2016]	79.1	76.6	66.8	55.5	46.4	35.9	26.8	-
Mettes et al. [2016]	-	-	34.8	-	-	-	-	-
Singh et al. [2017]	-	-	73.5	-	-	46.3	-	15.0
Saha et al. [2017]	-	71.3	63.1	51.6	-	33.1	-	-
Kalogeiton et al. [2017a]	-	-	76.5	65.2	-	49.2	-	19.7
Gu et al. [2018]	-	-	-	-	-	**59.9**	-	-
RecLNet	79.0	78.8	75.3	69.0	57.5	46.5	36.1	10.3
RecLNet++	86.6	86.1	83.4	77.4	68.5	57.5	46.0	23.9
RecLNet++ (FT)	**87.1**	**86.6**	**84.1**	**77.7**	**69.5**	59.6	**47.2**	**24.2**

Table 5.10: Comparison to the state of the art on UCF101-24 (mAP for IoU values ranging from 0.05 to 0.75).

idates what was shown by Kalogeiton et al. [2017a], i.e, integrating temporal information in the detector increases the track spatial precision leading to action localization improvement (as we observed in Section 5.4.3). For comparison we also report the fully connected classifier (FC) results as in Section 5.4.1. Given the large temporal receptive field of I3D features mentioned earlier, we notice RecLNet still improves the performance, e.g, it gets +2.0% and +2.4% mAP at IoU 0.3 when using the *stack* tracks on UCF101-24 and DALY respectively. In the following, the RecLNet model with the I3D features and the improved tracks (*stack*) is referred to as RecLNet++.

5.4.5 Comparison to the state of the art

UCF101-24. Table 5.10 compares RecLNet and RecLNet++ models to [Gu et al., 2018; Hou et al., 2017; Kalogeiton et al., 2017a; Mettes et al., 2016; Peng and Schmid, 2017; Saha et al., 2016, 2017; Singh et al., 2017; Weinzaepfel et al., 2015, 2016; Zolfaghari et al.,

2017] on UCF101-24 spatio-temporal action localization. RecLNet is our recurrent network with GRU memory units and *fusion layer* to combine optical flow and appearance features performing localization with our *threshold* approach and RecLNet++ uses I3D features and our improved *stack* tracks. RecLNet already significantly outperforms all other methods at most IoU thresholds. Due to our significantly improved temporal localization, our approach outperforms [Saha et al., 2016] on which we build by +9.3% at high IoU (0.6). We also outperform [Peng and Schmid, 2017], despite the fact that they use more sophisticated features combining a number of different human parts as well as multi-scale training and testing. Both these components are complementary to our approach. Our significant boost in performance can be explained by the fact that most current action localization methods rely on spatial features, for example per-frame CNN descriptors for optical flow and appearance, to detect action both spatially and temporally. However, such features are not designed for temporal localization. By relying on an accurate temporal model such as our recurrent RecLNet, we can clearly improve temporal detection and methods we build on that do not focus on this aspect. Recent method [Kalogeiton et al., 2017a], relying on more accurate spatial localization obtained by spatio-temporal cuboid regression, outperforms our model at high IoU. Indeed, they benefit from more precise human track bounding-boxes than the one we build on. Such an approach is used by RecLNet++, the enhanced version of RecLNet. This is shown to improve the state of the art by a large margin outperforming [Kalogeiton et al., 2017a] by +12.2%, +8.3% and +4.2% at IoU 0.3, 0.5 and 0.75 respectively. We also compare to very recent work [Gu et al., 2018] which only reports results at IoU 0.5. The method fine-tunes its localization network based on the I3D models end-to-end leading to a training procedure on a very large number of parameters. In our case, we originally wanted to keep a network with limited number of parameters capable to build on existing tracks and features without fine-tuning the input data. To obtain comparable results while keeping the same training

scores	localization	0.1	0.2	0.3
Weinzaepfel et al. [2016]	*sliding window*	-	14.5	-
CNN RGB+OF	*threshold*	24.3	18.5	13.6
RecLNet	*threshold*	28.4	24.6	18.7
RecLNet++	*threshold*	**41.1**	**37.6**	**31.0**

Table 5.11: State of the art on DALY (mAP for IoU 0.1, 0.2 and 0.3). Localization indicates the method used for temporal localization.

Action classes	CNN RGB+OF	RecLNet
App. MakeUp Lips	3.4	**14.7**
Brushing Teeth	9.5	**18.0**
Cleaning Floor	12.6	**23.9**
Cleaning Windows	9.4	**12.3**
Drinking	**20.6**	12.3
Folding Textile	7.3	**12.8**
Ironing	25.4	**29.0**
Phoning	7.3	**19.3**
Playing Harmonica	**33.1**	31.9
Taking Photos/Videos	7.6	**12.5**
mAP	13.6	**18.7**

Table 5.12: Per-class performance on DALY (mAP at IoU 0.3).

strategy, we adapt the input data. We fine-tuned the I3D models on UCF101-24 leading to input features more adapted to this dataset. Our network referred to as RecLNet++ (FT) uses these fine-tuned descriptors and achieves similar performance than Gu et al. [2018] (59.6% vs. 59.9%).

DALY. The recent DALY dataset [Weinzaepfel et al., 2016] is very challenging for spatio-temporal localization because it contains only short actions compared to the video length making temporal segmentation crucial and difficult. Table 5.11 compares our results to the state of the art. We can see that our RecLNet approach significantly outperforms [Wein-

zaepfel et al., 2016] by more than 10%. We also report results with our CNN scores *CNN RGB+OF* and temporal thresholding of the action scores without using RecLNet. Interestingly, the baseline approach outperforms the state of the art, but performs significantly worse than our (*RecLNet*) method. Also, Table 5.12 shows RecLNet improves over the baseline for all classes except *Drinking* and *Playing Harmonica*. Again, since we build on tracks from Weinzaepfel et al. [2016] and extract similar features (see Section 5.2), the improvement should be attributed to the more accurate temporal localization achieved by our RecLNet model. RecLNet++ further improves the results and outperforms [Weinzaepfel et al., 2016] by +23.1%.

5.4.6 Qualitative results

Good temporal localization needs precise evolution of action score with clear temporal boundaries. Having smooth scores also help our *threshold* localization method presented in Section 5.2.3. Here, we provide qualitative results to illustrate that our RecLNet possesses these properties and is well-suited for such localization.

Figure 5.3 shows scores from Saha et al. [2016] (red curves) versus our RecLNet response (green curves) for the 5 action track proposals used on UCF101-24. Thus, there are in total 10 curves per graph, one curve per track for the two compared methods. A bold curve (resp. dashed curve) shows that its corresponding track overlaps with more (resp. less) than 0.5 spatial IoU with a ground-truth action annotation. The ground-truth action interval is represented by the horizontal blue bar below the x-axis. The x-axis represents time. For each graph, we show the frame corresponding to the maximum RecLNet response (localized by the vertical yellow line) with its associated human detection (yellow box).

We show scores for 3 short actions *basketball*, *tennis swing* and *volleyball spiking*, and 2 long actions, *cliff diving* and *diving*. We can observe that the scores of the RecLNet

correspond to the temporal boundaries more precisely than the CNN scores of Saha et al. [2016]. Scores from Saha et al. [2016] are often flat and have high values throughout the entire video length (Figure 5.3, row 3). In general, they are not precise temporally (e.g., Figure 5.3, row 4). Also, a lot of dashed lines are highly scored (Figure 5.3, rows 3 and 5) which adds false positives. Finally, these graphs show that it is easy to set a threshold on RecLNet responses in order to obtain good temporal localization, while scores from Saha et al. [2016] (and scores from CNNs in general) are often imprecise temporally.

Similarly, Figures 5.4, 5.5 show qualitative results for temporal action localization on the DALY dataset, comparing again our RecLNet (green curves) and the *CNN RGB+OF* baseline (red curves). Each curve corresponds to one person track in the video. Each row corresponds to one action class. The last (third) column in Figures 5.4, 5.5 corresponds to failure cases or examples where our RecLNet model does not improve temporal localization. Similarly to results for the UCF101-24 dataset, the two first columns show that scores of RecLNet are better aligned with the ground truth action boundaries compared to scores of the *CNN RGB+OF* baseline. At the same time, dashed lines indicate many high-scoring false positive detections for the *CNN RGB+OF* method (e.g. Figure 5.4, row 1, plot 2 and Figure 5.5, row 2, plot 2). The last column in Figures 5.4, 5.5 indicates that DALY is a challenging dataset with many difficult examples where, most of the time, both methods fail to detect the correct temporal action boundaries.

5.5 Conclusion

This chapter shows that training a recurrent model (GRU) on the level of person tracks for modeling the temporal structure of actions improves localization in time significantly. Building on current state-of-the-art methods that obtain very good results for spatial localization, RecLNet improves significantly the detection of action time boundaries and hence the overall performance of spatio-temporal localization. As demonstrated in our

analysis, improving spatial precision of human tracks, inspired by most recent methods, and using better features further improve our results (RecLNet++). Our method outperforms the state of the art on the two challenging datasets, namely UCF101-24 and DALY.

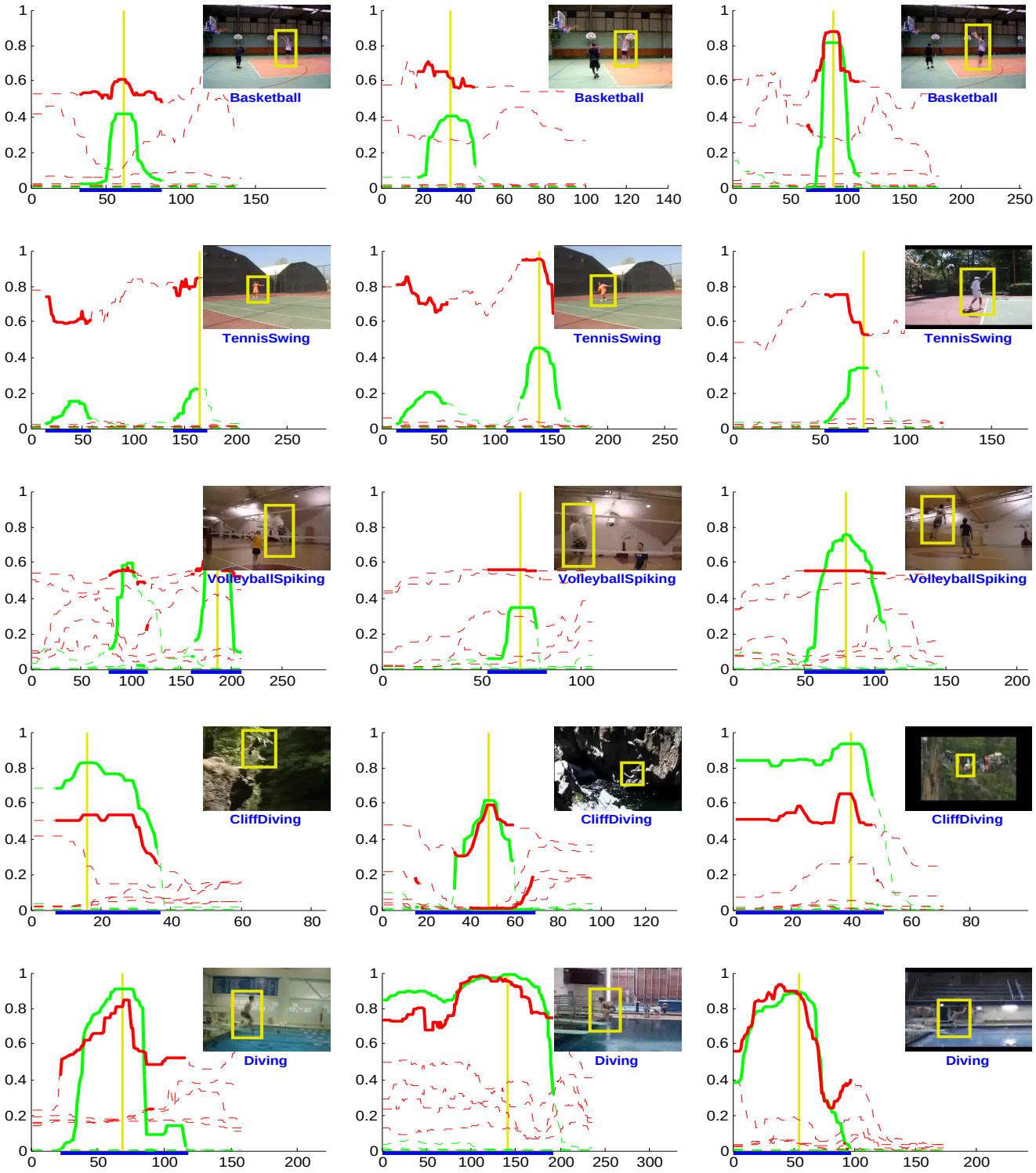


Figure 5.3: Qualitative results for temporal localization on UCF101-24. Each curve represents a human track, green curves correspond to our RecLNet scores and red ones to scores from Saha et al. [2016]. A bold curve (resp. dashed curve) shows that the track overlaps with more (resp. less) than 0.5 spatial IoU with a ground-truth action. The horizontal blue bar represents the ground-truth time segment. The video frame corresponds to the maximum RecLNet response (localized at the vertical yellow line) with its associated human detection (yellow box). The x-axis represents frame numbers.

Figure 5.4: Qualitative results for temporal localization on DALY. Each curve represents a human track, green curves correspond to our RecLNet scores and red ones to scores from CNN features *CNN RGB+OF*. A bold curve (resp. dashed curve) shows that the track overlaps with more (resp. less) than 0.5 spatial IoU with a ground-truth action. The horizontal blue bar represents the ground-truth time segment. The video frame corresponds to the maximum RecLNet response (localized at the vertical yellow line) with its associated human detection (yellow box). The last column corresponds to failure cases or examples where our RecLNet model does not improve temporal localization. Each row corresponds to one action class, namely *Applying Make Up On Lips*, *Brushing Teeth*, *Cleaning Floor*, *Cleaning Windows* and *Drinking*. The x-axis represents frame numbers.

Figure 5.5: Qualitative results for temporal localization on DALY. Each curve represents a human track, green curves correspond to our RecLNet scores and red ones to scores from CNN features *CNN RGB+OF*. A bold curve (resp. dashed curve) shows that the track overlaps with more (resp. less) than 0.5 spatial IoU with a ground-truth action. The horizontal blue bar represents the ground-truth time segment. The video frame corresponds to the maximum RecLNet response (localized at the vertical yellow line) with its associated human detection (yellow box). The last column (red boxes) corresponds to failure cases or examples where our RecLNet model does not improve temporal localization. Each row corresponds to one action class, namely *Folding Textile*, *Ironing*, *Phoning*, *Playing Harmonica* and *Taking Photos Or Videos*. The x-axis represents frame numbers.

Chapter 6

Toward less supervision: a flexible model for training action localization with varying levels of supervision

6.1 Introduction

The previous chapter addressed action localization in a fully-supervised setting. Such an approach needs a dense labeling which, as we will see next, leads to several drawbacks. To avoid a fully-supervised setting, this chapter introduces a weakly-supervised method capable to leverage information from several, potentially *weak*, levels of annotation. Given the cost of full annotation, a weakly-supervised approach will enable training from much larger datasets.

Action localization aims to find spatial and temporal extents as well as classes of actions in the video, answering questions such as *what* are the performed actions? *when* do they happen? and *where* do they take place? This is a challenging task with many potential applications in surveillance, autonomous driving, video description and search. To address this challenge, a number of successful methods have been proposed Gkioxari and Malik [2015]; Gu et al. [2018]; Kalogeiton et al. [2017a]; Peng and Schmid [2017]; Saha et al. [2016]; Singh et al. [2017]; Weinzaepfel et al. [2015]. Such methods, however, typically rely on exhaustive supervision where each frame of a training action has to be

manually annotated with an action bounding box.

Manual annotation of video frames is extremely tedious. Moreover, achieving consensus when annotating action intervals is often problematic due to ambiguities in the start and end times of an action. This prevents fully-supervised methods from scaling to many action classes and training from many examples. To avoid exhaustive annotation, recent works have developed several weakly-supervised methods. For example, Weinzaepfel et al. [2016] learns action localization from a sparse set of frames with annotated bounding boxes. Mettes et al. [2016] reduces bounding box annotation to a single spatial point specified for each frame of an action. Such methods, however, are designed for particular types of weak supervision and can be directly used neither to compare nor to combine various types of annotation.

In this work we design a unifying framework for handling various levels of supervision. Our model is based on discriminative clustering and integrates different types of supervision in a form of optimization constraints as illustrated in Figure 6.1. We investigate applications of such a model to training setups with alternative supervisory signals ranging from video-level class labels over temporal points or sparse action bounding boxes to the full per-frame annotation of action bounding boxes. Experiments on the challenging UCF101-24 and DALY datasets demonstrate competitive performance of our method at a fraction of supervision used by previous methods. We also demonstrate a significant gain by adding a few fully supervised examples to weakly-labeled videos. In summary, the main contributions of our work are (i) a flexible model with ability to adopt and combine various types of supervision for action localization and (ii) an experimental study demonstrating the strengths and weaknesses of a wide range of supervisory signals and their combinations.

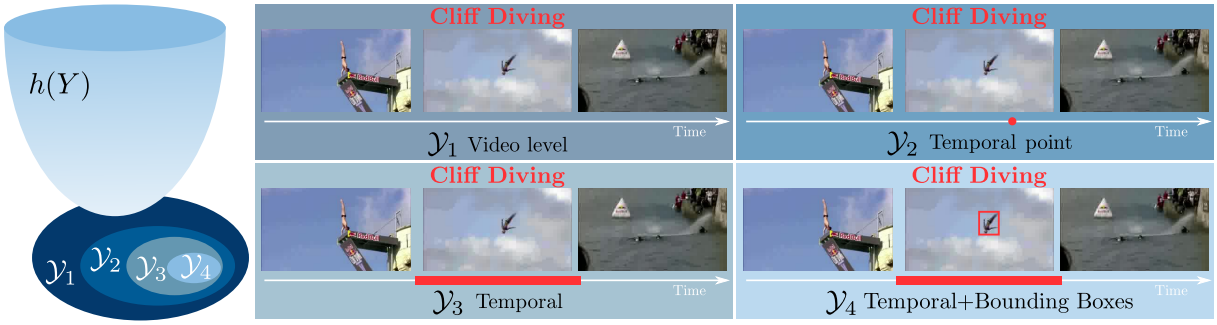


Figure 6.1: **A flexible method for handling varying levels of supervision.** Our method estimates a matrix Y assigning human tracklets to action labels in training videos by optimizing an objective function $h(Y)$ under constraints $\mathcal{Y}_s$. Different types of supervision define particular constraints $\mathcal{Y}_s$ and do not affect the form of the objective function. The increasing level of supervision imposes stricter constraints, e.g. $\mathcal{Y}_1 \supset \mathcal{Y}_2 \supset \mathcal{Y}_3 \supset \mathcal{Y}_4$ as illustrated for the Cliff Diving example above.

Related work. Our work is mostly related to spatio-temporal action localization, weakly-supervised learning for action understanding and discriminative clustering.

Spatio-temporal action localization consists in finding action instances in a video volume, i.e. both in space and time. Initial attempts [Ke et al., 2005; Laptev and Pérez, 2007] scanned the video clip with a 3D sliding window detector on top of volumetric features. Next, [Oneata et al., 2014; van Gemert et al., 2015] adapted the idea of object proposals [Arbeláez et al., 2014; Uijlings et al., 2013] to video action proposals. Currently, the dominant strategy [Gkioxari and Malik, 2015; Peng and Schmid, 2017; Saha et al., 2016; Singh et al., 2017; Weinzaepfel et al., 2015] is to obtain per-frame action detections and then to link them into continuous spatio-temporal tracks. The most recent methods [Gu et al., 2018; Hou et al., 2017; Kalogeiton et al., 2017a; Saha et al., 2017; Zolfaghari et al., 2017] operate on multiple frames and leverage temporal information to determine actions. Examples include stacking features from several frames [Kalogeiton et al., 2017a] and applying the I3D features [Carreira and Zisserman, 2017] to spatio-temporal volumes [Gu et al., 2018]. These methods are fully supervised and require a large quantity of annotations. Here, instead, we introduce a method that can leverage varying levels of supervision.

Weakly-supervised learning for action understanding is promising since it can potentially reduce the annotation effort significantly. Prior work has explored the use of readily available sources of information such as movie scripts [Bojanowski et al., 2013; Duchenne et al., 2009] to discover actions in clips using text analysis. Recent works have also explored more complex forms of weak supervision such as the ordering of actions [Bojanowski et al., 2014; Huang et al., 2016; Richard et al., 2017]. A number of approaches address temporal action detection with weak supervision. The UntrimmedNet [Wang et al., 2017] and Hide-and-Seek [Singh and Lee, 2017] methods are the current state of the art. In contrast, we seek to not only localize the action temporally but to also look for spatial localization from weak supervision. In [Soomro and Shah, 2017; Yang and Yuan, 2017], the authors propose unsupervised methods to localize actions by using graph clustering algorithms to first discover action classes and then localize actions within each cluster. Other works assume that clip level annotations are available [Chen and Corso, 2015; Siva and Xiang, 2011]. These methods often rely on action proposals, followed by a selection procedure to find the most relevant segment. Contrary to this line of work, we do not use action proposals. We rely instead on recent advances in off-the-shell human detectors [Girshick et al., 2018] and use human tracks. More recently [Mettes et al., 2017] makes use of pseudo annotation, to integrate biases (e.g. presence of objects, the fact that the action is often in the center of the video...) to further improve performance. Most of this work uses Multiple Instance Learning (MIL) in order to select discriminative instances from the set of all potential candidates. Here, we instead use discriminative clustering, which does not require EM optimization but instead relies on a convex relaxation with convergence guarantees.

Most related to us, are the works from [Mettes et al., 2016] and [Weinzaepfel et al., 2016] that also study the trade-off between annotation cost and final performance. Mettes et al. [2016] show that one can simply use spatial points instead of bounding boxes and

still obtain reasonable performance. Weinzaepfel et al. [2016] instead demonstrate that only a few frames with bounding box annotation are necessary for good performance. Here, we introduce a method that can leverage varying levels of supervision.

Discriminative clustering [Bach and Harchaoui, 2007; Xu et al., 2004] is a learning method that consists in clustering the data so that the separation is easily recoverable by a classifier. In this chapter, we employ the method proposed in [Bach and Harchaoui, 2007] which is appealing due to its simple analytic form, its flexibility and its recent successes for weakly supervised methods in computer vision [Alayrac et al., 2016; Bojanowski et al., 2013; Joulin et al., 2010]. To use this method, we rely on a convex relaxation technique which transforms the initial NP-hard problem into a convex quadratic program under linear constraints. The Frank-Wolfe algorithm [Frank and Wolfe, 1956] has shown to be very effective for solving such problems. Recently, [Miech et al., 2017] proposed a block coordinate [Lacoste-Julien et al., 2013] version of Frank-Wolfe for discriminative clustering. We leverage this algorithm to scale our method to hundreds of videos. We refer the reader to Chapter 3 for more details on discriminative clustering and associated optimization algorithms.

Outline. The chapter is organized as follows. Section 6.2 formulates our problem and introduces the proposed approach. Section 6.3 presents the implementation details, describes how we build the constraint sets according to the available supervison and evaluates our method. Qualitative results are shown in Section 6.4. Section 6.5 concludes the chapter.

6.2 Problem formulation and approach

We are given a set of N videos of people performing various actions. The total number of possible actions is K. Our goal is to learn a model for each action, in order to be able

to localize actions in time and space in *unseen* videos. Given the difficulty of manual action annotation, we investigate how different types and amounts of supervision affect the performance of action localization. To this end, we propose a model that can adopt and combine various levels of spatio-temporal action supervision such as (i) video-level only annotations, (ii) a single temporal point, (iii) a single bounding box, (iv) temporal bounds only, (v) temporal bounds with few bounding boxes and (vi) full supervision. In all cases, we are given tracks of humans, i.e. sequence of bounding box detections linked through time. These tracks are subdivided into short elementary segments in order to obtain precise action boundaries and to allow a same person to perform multiple actions. These segments are called *tracklet* and our goal is to assign them to action labels. In total, we have M such tracklets.

A single flexible model: one cost function, different constraints. We introduce a single model that can be trained with various levels and amounts of supervision. The general idea is the following. We learn a model for action classification on the train set from *weak supervision* using discriminative clustering [Bach and Harchaoui, 2007; Xu et al., 2004]. This model is used to predict spatio-temporal localization on test videos.

The idea behind discriminative clustering is to cluster the data (e.g. assign human tracklets to action labels) so that the clusters can be recovered by a classifier given some feature representation. The clustering and the classifier are simultaneously optimized subject to constraints that can regularize the problem and guide the solution towards a preferred direction. In this work we use constraints as a mean to encode different types of supervisory signals. More formally, we frame our problem as recovering an assignment matrix of tracklets to actions $Y \in \{0,1\}^{M \times K}$ which minimizes a clustering cost h under constraints:

$$\min_{Y \in \mathcal{Y}_s} h(Y), \tag{6.1}$$

where $\mathcal{Y}_s$ is a constraint set encoding the supervision available at training time. In this work we consider constraints corresponding to the following types of supervision.

(i) **Video level action labels**: only video-level action labels are known.

(ii) **Single temporal point**: we have a rough idea when the action occurs, but we do not have either the exact temporal extent or the spatial extent of the action. Here, we can for example build our constraints such that at least one human track (composed of several tracklets) should be assigned to the associated action class in the neighborhood of the temporal point.

(iii) **One bounding box (BB)**: we are given a human spatial location at a given time inside each action instance. Spatial constraints force tracks that overlap with the bounding box to match the action class around this time step.

(iv) **Temporal bounds**: we know *when* the action occurs but its spatial extent is unknown. We can constrain the labels so that at least one human track contained in the given temporal interval is assigned to that action. Everything outside is forced to be background.

(v) **Temporal bounds with bounding boxes (BBs)**: combination of (iii) and (iv).

(vi) **Fully supervised**: annotation is defined by the bounding box at each frame of an action.

All these constraints can be formulated under a common mathematical framework that we describe next. In this chapter we limit ourselves to the case where each tracklet should be assigned to only one action class (or the background). More formally, this can be written as follows:

$$\forall\ m \in [1\,..\,M], \quad \sum_{k=1}^{K} Y_{mk} = 1. \tag{6.2}$$

This does not prevent us from having multiple action classes for a video, or to assign different classes to tracklets composing a same human track (such cases will occur during the experiments). Also note that it is possible to handle the case where a single tracklet can be assigned to multiple actions by simply replacing the equality with, e.g., a greater or equal inequality. However, as the datasets considered in this work always satisfy (6.2), we keep that assumption for the sake of simplicity.

We propose to enforce the various levels of supervision with two types of constraints on the assignment matrix Y as described below.

Strong supervision with equality constraints. In many cases, even when dealing with weak supervision, the supervision may provide strong cues about a tracklet. For example, if we know that a video corresponds to the action 'Diving', we can assume that no tracklet of that video corresponds to the action 'Tennis Swing'. This can be imposed by setting to 0 the corresponding entry in the assignment matrix Y. Similarly, if we have a tracklet that is outside an annotated action interval, we know that this tracklet should belong to the background class. Such cues can be enforced by setting to 1 the matching entry in Y. Formally,

$$\forall (t,k) \in \mathcal{O}_s \quad Y_{tk} = 1, \quad \text{and} \quad \forall (t,k) \in \mathcal{Z}_s \quad Y_{tk} = 0, \tag{6.3}$$

with $\mathcal{O}_s$ and $\mathcal{Z}_s$ containing all the tracklet/action pairs that we want to match ($\mathcal{O}_s$) or dissociate ($\mathcal{Z}_s$).

Weak supervision with at-least-one constraints. Often, we are uncertain about which tracklet should be assigned to a given action k. For example, we might know when the action occurs, without knowing where it happens. Hence, multiple tracklets might overlap with that action in time but not all are good candidates. These multiple tracklets

compose a bag [Bojanowski et al., 2013], that we denote $\mathcal{A}_k$. Among them, we want to find at least one tracklet that matches the action, which can be written as:

$$\sum_{t \in \mathcal{A}_k} Y_{tk} \geq 1. \tag{6.4}$$

We denote by $\mathcal{B}_s$ the set of all such bags. Hence, $\mathcal{Y}_s$ is characterized by defining its corresponding strong supervision $\mathcal{O}_s$ and $\mathcal{Z}_s$ and the bags $\mathcal{B}_s$ that compose its weak supervision.

Discriminative clustering cost. As stated in Chapter 3, the intuition behind discriminative clustering is to separate the data so that the clustering is easily recoverable by a classifier over the input features. Here, we use the square loss and a linear classifier, which corresponds to the DIFFRAC setting [Bach and Harchaoui, 2007]:

$$h(Y) = \min_{W \in \mathbb{R}^{d \times K}} \frac{1}{2M} \|XW - Y\|_F^2 + \frac{\lambda}{2} \|W\|_F^2. \tag{6.5}$$

$X \in \mathbb{R}^{M \times d}$ contains the features describing each tracklets. $W \in \mathbb{R}^{d \times K}$ corresponds to the classifier weights for each action. $\lambda \in \mathbb{R}^+$ is a regularization parameter. $\|.\|_F$ is the standard Frobenius matrix norm. Following Bach and Harchaoui [2007], we can solve the minimization in W in closed form to obtain: $h(Y) = \frac{1}{2M} \mathrm{Tr}(YY^T B)$, where $\mathrm{Tr}(\cdot)$ is the matrix trace and B is a strictly positive definite matrix (hence h is strongly convex) defined as $B := I_M - X(X^T X + M\lambda I_d)^{-1} X^T$. I_d stands for the d-dimensional identity matrix.

Optimization. Directly optimizing the problem defined in (6.1) is NP hard due to the integer constraints. To address this challenge, we follow recent approaches such as [Bojanowski et al., 2014] and propose a convex relaxation of the constraints. The problem hence becomes $\min_{Y \in \bar{\mathcal{Y}}_s} h(Y)$, where $\bar{\mathcal{Y}}_s$ is the convex hull of $\mathcal{Y}_s$. To deal with

such constraints we use the Frank-Wolfe algorithm [Frank and Wolfe, 1956], which has the nice property of only requiring to know how to solve linear programs over the constraint set. Another challenge lies in the fact that we deal with large datasets containing hundreds of long videos. Using the fact that our set of constraints actually decomposes over the videos, $\bar{\mathcal{Y}}_s = \bar{\mathcal{Y}}_s^1 \times \cdots \times \bar{\mathcal{Y}}_s^N$, we use the block coordinate Frank-Wolfe algorithm [Lacoste-Julien et al., 2013] that has been recently adapted to the discriminative clustering objective [Miech et al., 2017]. This allows us to scale to several hundreds of videos.

6.3 Experiments

6.3.1 Implementation details

Person detector and tracker. Person boxes are obtained with Faster-RCNN [Ren et al., 2015] using the ResNet-101 architecture [He et al., 2016]. When no spatial annotation is provided, we use an off-the-shelf person detector trained on the COCO dataset [Lin et al., 2014]. Otherwise, action detections are obtained by training the detector on the available frames with bounding boxes starting from ImageNet [Russakovsky et al., 2015] pre-training. We use the implementation from the Detectron package [Girshick et al., 2018]. Detections from the human detector are linked into continuous tracks using KLT [Lucas et al., 1981] to differentiate between person instances. Class-specific detections are temporally aggregated by a simpler online linker [Kalogeiton et al., 2017a; Singh et al., 2017] based on action scores and overlap.

Feature representation. In our model, person tracks are divided into consecutive subtracks of 8 frames which we call *tracklets*. Due to its recent success for action recognition, we use the I3D [Carreira and Zisserman, 2017] network trained on the Kinetics dataset [Kay et al., 2017] to obtain tracklet features. More precisely, we extract video descriptors with I3D RGB and flow networks after the 7-th inception block, before max-pooling, to balance the depth and spatial resolution. The temporal receptive field is 63

frames and the temporal stride is 4 frames. The input frames are resized to 320×240 pixels, which results in feature maps of size 20×15 with 832 channels. We use ROI-pooling to extract a 832-dimensional descriptor for each human box at a given time step. The final 1664-dimensional representation is obtained by concatenating the descriptor from the RGB and the flow streams. Finally, a tracklet representation is obtained by averaging the descriptors corresponding to the detections spanned by this tracklet.

Optimization. For all types of supervision we run the block-coordinate Frank-Wolfe optimizer for 30k iterations (one iteration deals with one video). We use optimal line search and we sample videos according to the block gap values [Osokin et al., 2016] which speeds up convergence in practice.

Temporal localization. At test time, person tracks have to be trimmed to localize the action instances temporally. To do so, each of the T detections composing a track are scored by the learnt classifier. Similarly to Chapter 5, the scores are then smoothed out with a median filtering (with a window of size 25), giving, for an action $k \in [1\,..\,K]$, the sequence of detection scores $s^k = (s_1^k, ..., s_T^k)$. Within this track, temporal segmentation is performed by selecting consecutive person boxes with scores $s_t^k > \theta_k$, leading to subtrack candidates for action k. A single person track can therefore produce several spatio-temporal predictions at different time steps. A score for each subtrack is obtained by averaging corresponding detection scores. Finally, non-maximum-suppression (NMS) is applied to eliminate multiple overlapping predictions with spatio-temporal IoU above 0.2.

Hyperparameters. The regularization parameter λ (see equation (6.5)) is set to 10^{-4} in all of our experiments. To calibrate the model, the temporal localization thresholds θ_k are validated per class on a separate set corresponding to 10% of the training set.

6.3.2 Supervision as constraints

In this section, we describe in details the constraint sets $\mathcal{Y}_s$ for different levels of supervision considered in our work. Recall from Section 6.2 that $\mathcal{Y}_s$ is characterized by $\mathcal{O}_s$ and $\mathcal{Z}_s$, the sets of tracklets for which we have strong supervision, and by $\mathcal{B}_s$, the set of bags containing tracklets that are candidates to match a given action. Note that in all settings we have one class that corresponds to the background class. In the following, a *time unit* corresponds to one of the bins obtained after uniformly dividing a time interval. Its size is 8 frames.

Video level. Here, $\mathcal{B}_s$ contains as many bags as action *classes* occurring in the entire video. Every bag contains all the tracklets of the video. This constrains each annotated action class to be assigned to at least one tracklet from the video. We also construct $\mathcal{Z}_s$ making sure no tracklets are assigned to action classes not present in the video.

Shot level. This setting is identical to the '**Video level**' one, but we further decompose the video into clips and assume we have annotations at the clip level. These clips can be obtained by shot detection [Mathe, 2012] as is the case in the DALY dataset. This setting only makes sense when working with relatively long videos, in which the action instance is relatively short. Hence, we do not report results for such type of supervision for UCF101-24.

Temporal point. For each action instance, we are given a point in time which falls into the temporal interval of the action. In our experiments, we sample this time point uniformly within the ground truth interval. This corresponds to the scenario where an annotator would simply click one point in time where the action occurs, instead of precisely specifying the time boundaries. We then create a candidate interval around that time point with a fixed size of 50 frames (2 seconds). This interval is discretized into time units. For each of them, we impose that at least one tracklet should be assigned to the

action instance. Hence, we add $A \times U$ bags to $\mathcal{B}_s$, where A is the number of instances and U is the number of time units in a 50 frames interval. Finally, $\mathcal{Z}_s$ is built as before: no tracklet should match actions absent from the video.

One bounding box. At each location of the previous temporal points, we are now additionally given the corresponding action instance bounding box (a.k.a. keyframe in [Weinzaepfel et al., 2016]). Similarly, we construct 50 frames time intervals.We consider all tracklets whose original track has a temporal overlap with the annotated frame. Then, we compute the spatial IoU between the bounding box of the track at the time of the annotated frame and the bounding box annotation. If this IoU is less than 0.3 for all possible frames, we then construct $\mathcal{O}_s$ so that these tracklets are forced to the background class. Otherwise, if the tracklet is inside the 50 frames time interval, we force it to belong to the action class instance with the highest IoU by augmenting $\mathcal{O}_s$. Again, for each time unit of the interval, we construct $\mathcal{B}_s$ such that at least one tracklet matches the action instance. $\mathcal{Z}_s$ is built as previously.

Temporal bounds. If we know temporal boundaries for each action instance, we first impose through $\mathcal{O}_s$ that all tracklets that are outside these ground truth intervals are assigned to the background class. Then, for a given ground truth interval, we augment $\mathcal{B}_s$ with as many bags as its number of time units so that for each of them at least one tracklet should be assigned to the corresponding action class of the interval. The set $\mathcal{Z}_s$ is constructed as above.

Temporal bounds with bounding boxes. In addition to temporal boundaries of action instances, we are also given a few frames with spatial annotation. This is a standard scenario adopted in DALY [Weinzaepfel et al., 2016] and AVA [Gu et al., 2018]. In our experiments, we report results with 1 and 3 bounding boxes per action instance. $\mathcal{Z}_s$ and $\mathcal{B}_s$ are constructed as in the '**Temporal bounds only**' setting. $\mathcal{O}_s$ is initialized to assign all tracklets outside of the temporal intervals to the background class. Similarly to the

'**One bounding box**' scenario, we augment $\mathcal{O}_s$ in order to force tracklets to belong to either the corresponding action or the background based on a spatial overlap criterion. However, here an action instance has potentially several annotations, therefore the spatial overlap with the track is defined as the minimum IoU between the corresponding bounding boxes.

Fully supervised. Here, we enforce a hard assignment of all tracklets to action classes based on the ground truth through $\mathcal{O}_s$. When a tracklet has a spatial IoU greater than 0.3 with at least one ground truth instance, we assign it to the action instance with highest IoU. Otherwise, the tracklet is assigned to the background.

Temporal bounds with spatial points. In [Mettes et al., 2016], the authors introduce the idea of working with spatial points instead of bounding boxes for spatial supervision. Following [Mettes et al., 2016], we take the center of each annotated bounding box to simulate the spatial point annotation. Similarly to the '**Fully supervised**' setting, we build $\mathcal{O}_s$ to enforce a hard assignment for all tracklets, but we modify the action-tracklet matching criterion. When tracklet bounding boxes contain all the corresponding annotation points of at least one ground truth instance, we assign the tracklet to the action instance that has the lowest distance between the annotation point and the tracklet's bounding box center.

6.3.3 Results and analysis

In this section we present and discuss our experimental results. We evaluate our method on two datasets, UCF101-24 and DALY, for the following supervision levels: video level, temporal point, one bounding box, temporal bounds only, temporal bounds and one bounding box, temporal bounds and three bounding boxes. For UCF101-24, we also report temporal bounds and spatial points [Mettes et al., 2016] and the fully supervised settings (not possible for DALY as the spatial annotation is not dense). We evaluate the

additional 'shot-level' supervision setup on DALY. All results are given in Table 6.1. We compare to the state of the art whenever possible. To the best of our knowledge, for some levels of supervision no results are available. We provide an additional comparison to Weinzaepfel et al. [2016] by running our method on the same set of tracks used in their method in Table 6.2. We also report results for the mixture of different levels and amounts of supervision in Figure 6.3a. Qualitative results are illustrated in Figure 6.3b while more are provided in Section 6.4.

Comparison to the state of the art. We compare results to two state-of-the-art methods [Mettes et al., 2016; Weinzaepfel et al., 2016] that are designed to deal with weak supervision for action localization. Mettes et al. [2016] use spatial point annotation instead of bounding boxes. Weinzaepfel et al. [2016] compare various levels of spatial supervision (spatial points, few bounding boxes). Temporal boundaries are known in both cases.

UCF101-24 specificity. For UCF101-24, we noticed a significant drop of performance whenever we use an off-the-shelf detector versus a detector pretrained on UCF101-24 bounding boxes. We observe that this is due to two main issues: (i) the quality of images in the UCF101-24 dataset is quite low when compared to DALY which makes the human detection very challenging, and (ii) the bounding boxes have been annotated with a large margin around the person whereas a typical off-the-shelf detector produces tight detections (see Figure 6.2). Addressing the problem (i) is difficult without adaptive finetuning. Concerning (ii), a simple solution adopted in our work is to enlarge person detections with a single scaling factor (we use $\sqrt{2}$). However, we also observe that the size of boxes depends on action classes (e.g. the bounding boxes for the 'TennisSwing' contains the tennis racket), which is something we cannot capture without more specific information. To better highlight this problem we have included in Table 6.1 a baseline where we use the detections obtained after finetuning on spatial annotations even when it

is normally not possible ('temporal' and 'temporal with spatial points' annotations). The detector is the same as in the 'one bounding box' supervision setup. We enclose these baselines in parenthesis in Table 6.1. We can observe that the drop of performance is much higher on UCF101-24 (−22.2% mAP@0.2 for 'temporal') than on DALY (−1.9%), which confirms the hypothesis that the problem actually comes from the tracks rather than from our method. This observation will be analyzed next in Table 6.2, where we run our method with the same tracks as used in [Weinzaepfel et al., 2016] for all types of supervision on UCF101-24.

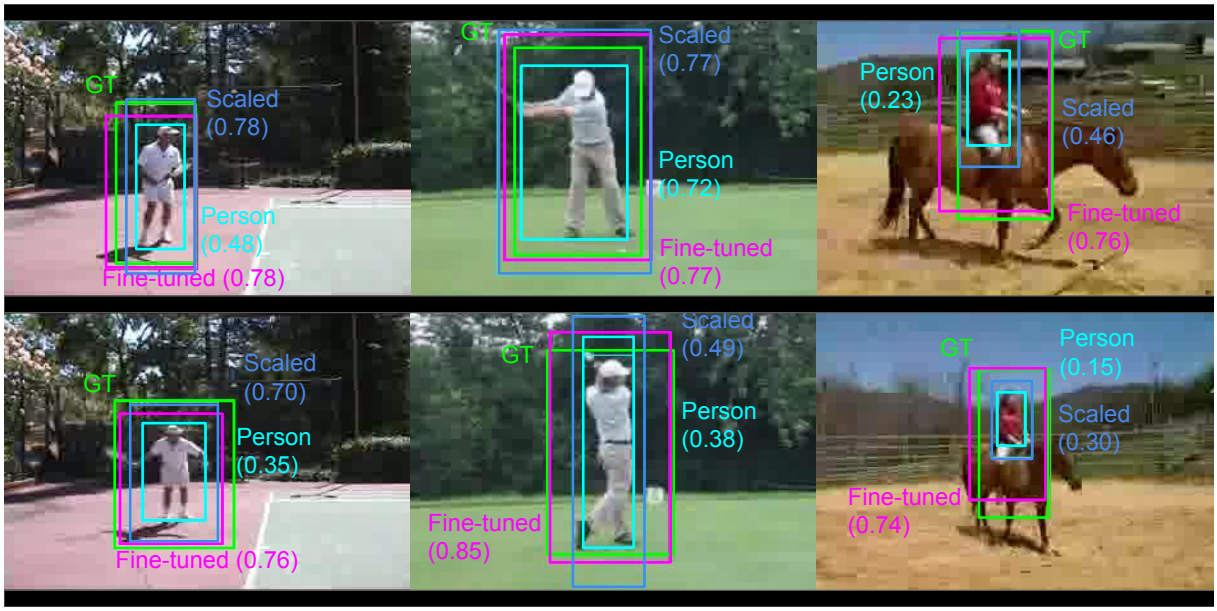


Figure 6.2: UCF101-24 specificity. The detections obtained from the off-the-shelf person detector (box in cyan) are tight to the human while the ground-truth annotations (green) are always large around the person and might contain an object like the tennis racket. This reduces significantly the IoU measure between predictions and ground truth (see numbers in parenthesis). These class-specific biases can be learnt by the fine-tuned detector (magenta) as explained in Section 6.3.3. The detection scaling procedure (blue) works in some cases but this simple linear expansion is not enough to reach satisfying precision. Also, in case of lower video quality the person detector can miss some body parts like the lower legs of the woman ridding the horse. The detection IoU with the ground truth is enclosed in brackets.

Supervision		Video level	Shot level	Temporal point	Temporal	Temporal + spatial points			1 BB	Temp. + 1 BB		Temp + 3 BBs		Fully Suppervised	
Method		Our	Our	Our	Our	Our	WMS16	MGS16	Our	Our	WMS16	Our	WMS16	Our	WMS16
UCF101-24	@0.2	43.9	-	45.5	47.3 (69.5)	49.1 (69.8)	57.5	34.8	66.8	70.6	57.4	74.5	57.3	76.0	58.9
(mAP)	@0.5	17.7	-	18.7	20.1 (38.0)	19.5 (39.5)	-	-	36.9	38.6	-	43.2	-	50.1	-
DALY	@0.2	7.6	12.3	26.7	31.5 (33.4)	No continuous			28.1	32.5	14.5	32.5	13.9	No full GT	
(mAP)	@0.5	2.3	3.9	8.1	9.8 (14.3)	spatial GT			12.2	13.3	-	15.0	-	available	

Table 6.1: Video mAP for varying level of supervision. We compare our method to WMS16 [Weinzaepfel et al., 2016] and MGS16 [Mettes et al., 2016].

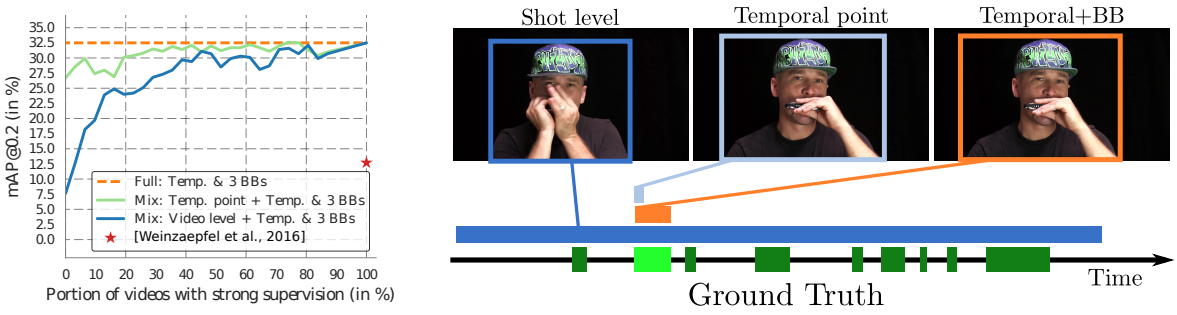
Results for varying levels of supervision. Results are reported in Table 6.1. As expected, the performance increases with access to more supervision. On the DALY dataset, when compared to the state-of-the-art method [Weinzaepfel et al., 2016], we are always superior by a significant margin (+18%). This highlights the strength of our model in addition to its flexibility. We first note that there is only a 1.0% difference between the 'temporal' and 'temp + 3 keys' levels of supervision. This shows that, with recent accurate human detectors, the spatial supervision is less important. Interestingly, we observe good performance when using one temporal point only (26.7%). This weak information is almost sufficient to match results of the precise 'temporal' supervision (31.5%). The corresponding performance drop is even smaller on the UCF101-24 dataset (−1.8%). This demonstrates the strength of the single 'temporal click' annotation which could be a good and cheap alternative to the precise annotation of temporal boundaries.

On the UCF101-24 dataset, we are always better compared to the state of the art when bounding box annotations are available, e.g., +17.2% in 'temp + 3 keys' setting. This demonstrates the strength of the sparse bounding box supervision which almost matches accuracy of the fully-supervised setup (−1.5%). We note that using only one bounding box already enables good performance (66.8%). Overall, this opens up the possibility to annotate action video datasets much more efficiently and at moderate cost.

However, as explained above, we observe a significant drop when removing the spatial supervision on the UCF101-24 dataset. The superiority of [Weinzaepfel et al., 2016] is mainly due to the fact that they are using more robust tracks obtained by a sophistic-

ated combination of detections and an instance specific tracker. We have then rerun our method with the same tracks as used in [Weinzaepfel et al., 2016] for all types of supervision on UCF101-24. These results are presented in Table 6.2 (our w. tracks of [Weinzaepfel et al., 2016]) and compared to [Weinzaepfel et al., 2016]. We observe that "Our w. tracks of [Weinzaepfel et al., 2016]" outperforms [Weinzaepfel et al., 2016] in *all settings*, and particularly for the "Temporal+spatial points". This confirms that the tracks of [Weinzaepfel et al., 2016] are better adapted to handle the specificity of the UCF101-24 annotation. This observation is reconfirmed in Table 6.2 where "Our w. tracks from [Weinzaepfel et al., 2016]" outperforms our method using a generic person detector in "Our" col. 2–5. When the bounding box supervision is available, however, the adjustment of our generic tracks for specific actions improves results beyond "Our w. tracks from [Weinzaepfel et al., 2016]" (col. 6–9). Finally, when using the same tracklets for all settings (Our w. tracks of [Weinzaepfel et al., 2016]), we observe only a small difference in performance between the "Temporal" and the "Temporal + 1 BB" setups. This suggests an interesting conclusion that the spatial supervision for action localization may not always be necessary.

Back to Table 6.1, note that, even if not the main focus of our work, our performance on the fully supervised setting is on par with the recent work [Kalogeiton et al., 2017a] which reports a mAP@0.2 of 76.5% on UCF101-24 (versus our 76.0%). This shows that our model is not only designed for weak supervision, and thus can be used to fairly compare all levels of supervision. At 'video level', we achieved better results than the model specifically designed for this supervision in [Mettes et al., 2017] (43.9% vs. 37.4% on UCF101-24) which is another cue demonstrating that our comparison of the different supervision levels is fair. This consolidates the conclusions drawn above about the alternative way of annotating videos ('temporal click') and what information is important (is the spatial supervision always necessary?). However, we are still below the current state of the art [Gu et al., 2018] on UCF101-24 with full supervision, which reports a mAP@0.5



(a) **Mixing supervision.** (b) **Qualitative results on DALY.**

Figure 6.3: **Left.** Mixing levels of supervision on the DALY dataset. **Right.** Matched predictions for the light green ground truth instance for various methods. We display the highest scoring detection for each method. When training with only 'shot-level' annotation it is hard for the method to discover precise boundaries of actions, as 'Playing Harmonica' could also consist in someone holding an harmonica. Only accessing a temporal point is enough to better detect the instance. Training with precise temporal boundaries further improves performance.

of 59.5% on UCF101-24 (versus our 50.1%). This can be explained by the fact that we are only learning a linear model for I3D features, whereas [Gu et al., 2018] train a non-linear classifier for the same features. Investigating how to build flexible deep (i.e. non-linear) models that can handle varying level of supervision is therefore an interesting avenue for future work.

Supervision	Video level	Temporal point	Temporal	Temporal + spatial points	1 BB	Temp. + 1 BB	Temp. + 3 BBs	Fully Supervised
[Weinzaepfel et al., 2016]	-	-	-	57.5	-	57.4	57.3	58.9
Our w. tracks of [Weinzaepfel et al., 2016]	**53.1**	**54.7**	**59.7**	**61.0**	61.2	58.9	62.1	61.2
Our	43.9	45.5	47.3 (69.5)	49.1 (69.8)	**66.8**	**70.6**	**74.5**	**76.0**

Table 6.2: Video mAP@0.2 on the UCF101-24 dataset for varying level of supervision with different set of tracks.

Mixing levels of supervision. To further demonstrate the flexibility of our approach, we conduct an experiment where we mix different levels of supervision in order to improve the results. As we observe a significant drop when going to the weakest form of supervision considered here, i.e. 'video-level' annotation, we report results on the DALY dataset. The experimental setup consists in constructing a training dataset with a portion of videos that have weak supervision (either 'video-level' or 'temporal point') and another set with

stronger supervision (temporal bounds and 3 bounding boxes). We vary the portions of videos that have stronger supervision available at train time (the rest having weak labeling), and we evaluate mAP@0.2 on the test set. We report results in Figure 6.3a. Tracks are obtained using the off-the-shelf person detector. With only 20 supervised videos (around 5% of the data) and the rest having only 'video level' label the performance goes from 7.6 to 18.2. We are on par with the performance of our fully supervised setting with only 40% of the fully annotated data on that same setting. This performance is reached even sooner when using 'Temporal point' weak labeling (with only 20% of fully annotated videos). This strongly encourages the use of semi-supervised methods to improve spatio-temporal action localization at a moderate cost.

6.4 Qualitative results

Some qualitative results are presented for DALY (Figure 6.4 and Figure 6.5) and for UCF101-24 (Figure 6.6 and Figure 6.7) datasets. For each video, we choose an action instance (shown in light green) and display action predictions that have been matched to this instance for three different models trained with varying levels of supervision. Other ground truth instances are displayed in dark green. For DALY, we report predictions made by models trained from 'Shot level' (dark blue), 'Temporal point' (light blue) and 'Temporal and bounding box (BB)' (orange) supervisions. For UCF101-24, we report 'Video level' (dark blue), 'Temporal and bounding box (BB)' (light blue) and 'Fully supervised' (orange). The image and the box displayed for each method are the ones that obtained the highest score of detection according to each model. On UCF101-24, the ground-truth bounding box is displayed in green.

6.5 Conclusion

This chapter presents a weakly-supervised method for spatio-temporal action localization which aims to address issues caused by fully-supervised learning. We propose a unifying framework that can handle and combine varying types of less-demanding *weak supervision*. The key observations are **(i)** the dense spatial annotation is not always needed due to the recent advances of human detection, **(ii)** the performance of 'Temporal point' supervision indicates that only annotating an action with a 'click' is very promising to decrease the annotation cost at a moderate performance drop, **(iii)** mixing levels of supervision (see Figure 6.3a) is also a powerful approach for reducing annotation efforts.

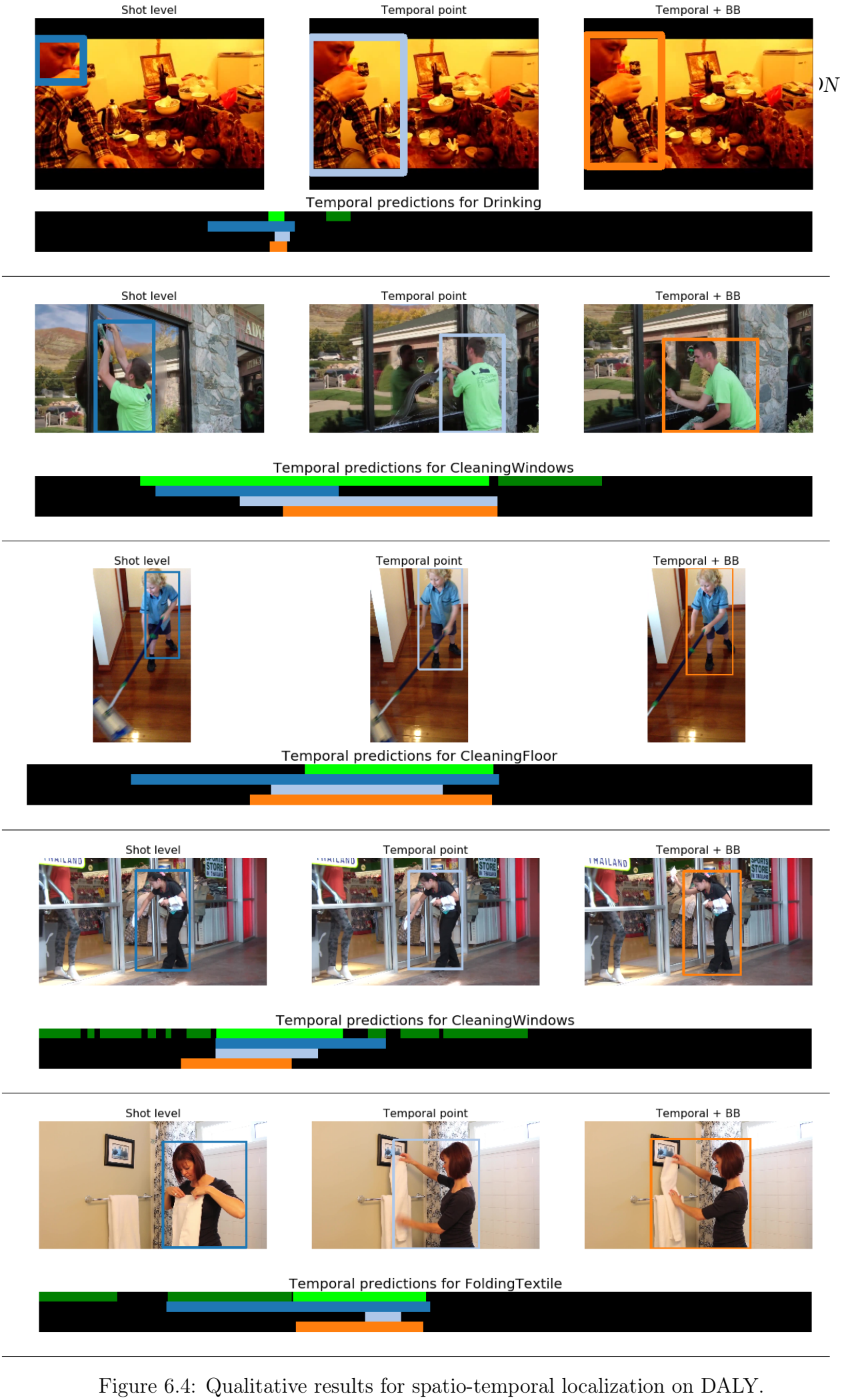


Figure 6.4: Qualitative results for spatio-temporal localization on DALY.

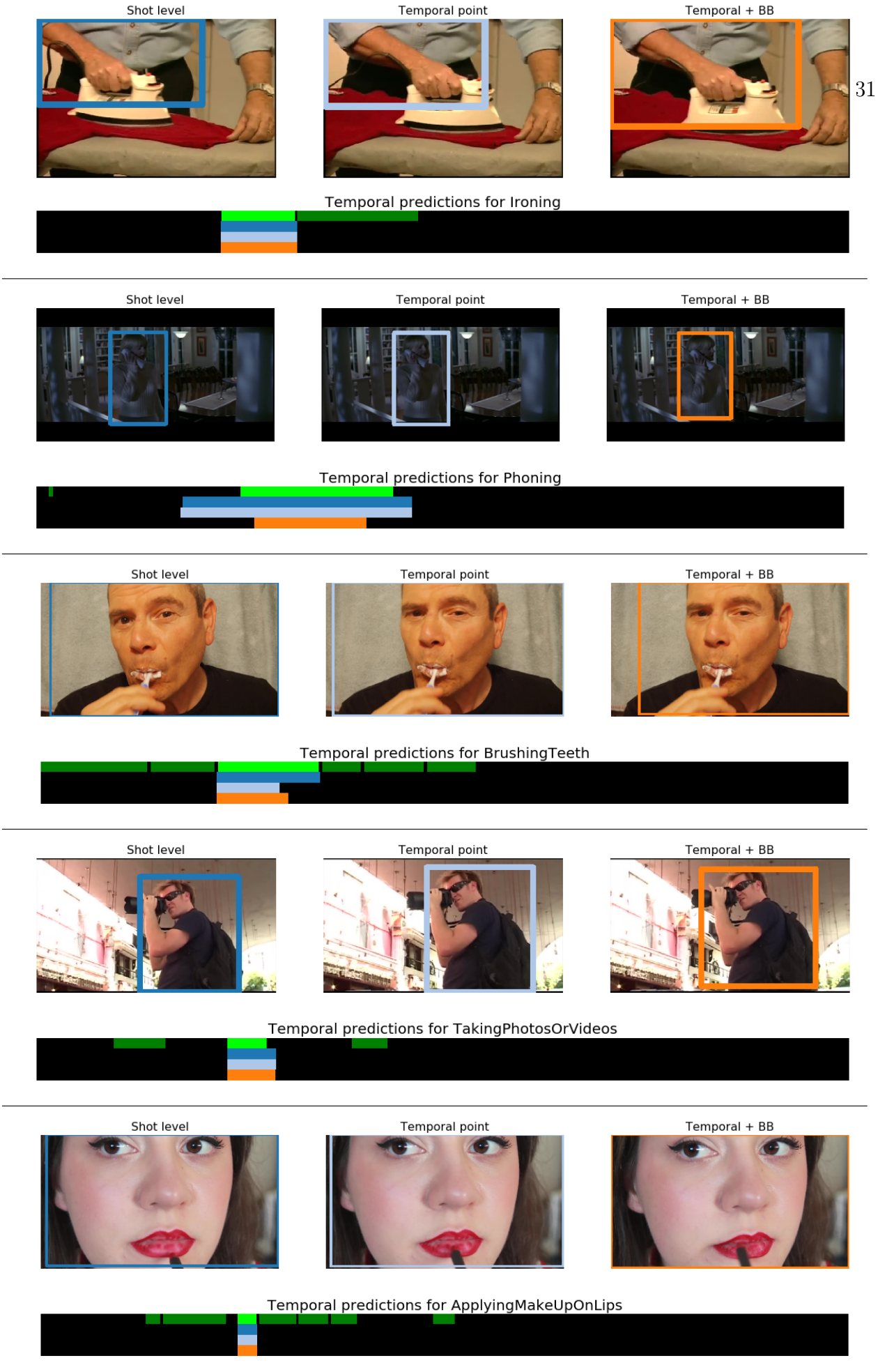


Figure 6.5: Qualitative results for spatio-temporal localization on DALY.

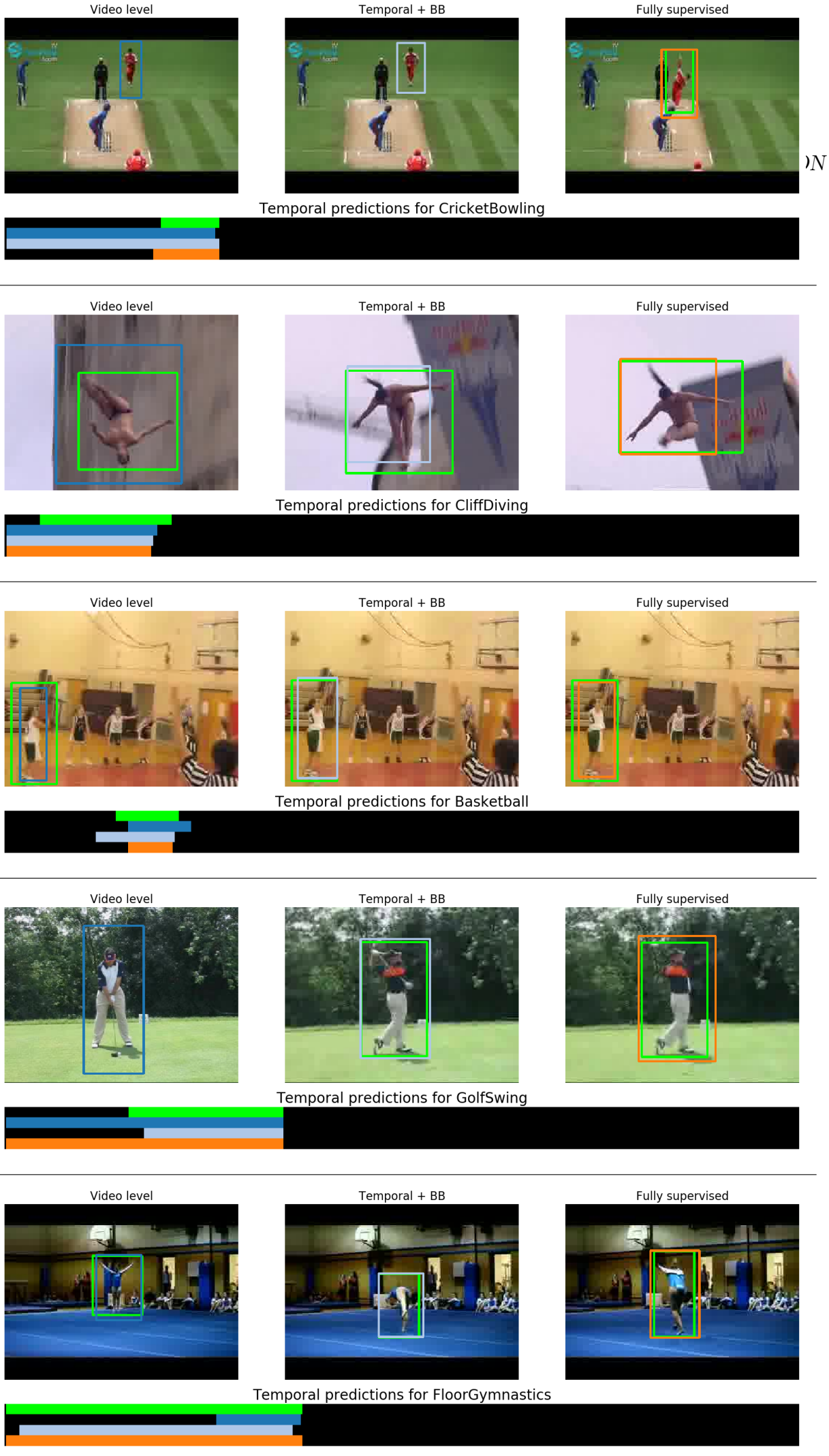


Figure 6.6: Qualitative results for spatio-temporal localization on UCF101-24.

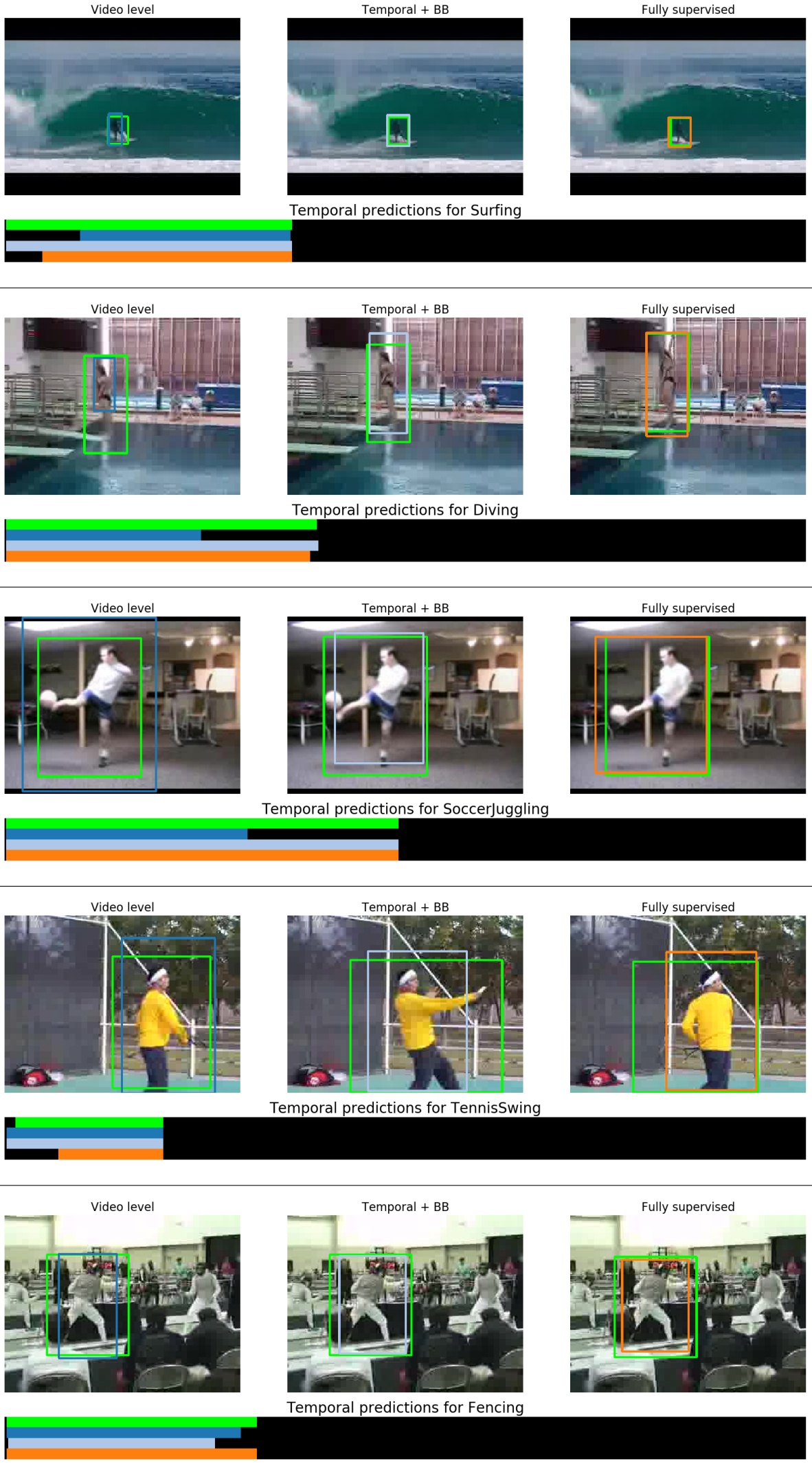


Figure 6.7: Qualitative results for spatio-temporal localization on UCF101-24.

Chapter 7

Conclusion

This thesis investigates two important tasks for video understanding, namely action classification and spatio-temporal action localization. In this chapter we summarize the contributions of our work in Section 7.1 and conclude the dissertation with directions for future research in Section 7.2.

7.1 Summary of contributions

In this thesis, we have presented three approaches, one addressing action classification and two for localizing actions in space and time. The first approach uses human pose to improve action classification performance in videos. While classic methods represent actions by local features statistics, we stress the importance of modeling the structure especially for fine-grained or sport-oriented action recognition. Our model pools local appearance and motion CNN descriptors at the spatial location of different body parts while performing temporal aggregations through the video. Thus, the introduced P-CNN features keep trace of the body configuration along the person track in a significantly more robust way than other pose-based descriptors [Jhuang et al., 2013]. Furthermore, this information has been shown to be complementary to typical classification approaches like dense trajectories [Wang et al., 2011].

Our second approach presented in Chapter 5 tackles action localization, *i.e.*, it simul-

taneously recognizes and localizes activities in space and time. Our method consists in modeling the temporal structure of actions with a recurrent network (RecLNet) trained on the level of person tracks. While classic approaches usually benefit from accurate spatial detection but achieve rough temporal localization (*e.g.* performing a pass of viterbi [Saha et al., 2016] or sliding window [Weinzaepfel et al., 2016] as a second step), our experimental results show that the performance gain is mainly due to the improved temporal localization. Indeed, after training our model to rescore human tracks generated by existing action localization methods we build on [Saha et al., 2016; Weinzaepfel et al., 2016] and supported by a filtering and thresholding post-processing, our method improves significantly the detection of action time boundaries hence the overall performance of spatio-temporal localization. This improvement allows our method not only to be complementary to the state of the art [Kalogeiton et al., 2017a] but also to outperform it. However, this model belongs to the family of fully-supervised methods and requires dense labeling, *i.e.*, manually annotated bounding boxes at each frame where people perform a specific activity. Such approaches bring substantial problems since this information is extremely costly to obtain and prohibits scalability. Among others, it is also hard to find a consensus between annotators leading to different labeling of temporal (*where does it start? where does it end?*) and spatial (*what should be included in the rectangle?*) action extents. As a next natural step, the third and last approach of this thesis presents a weakly-supervised method for spatio-temporal action localization. We proposed a unifying framework that can handle and combine varying types of less-demanding *weak supervision*. We leverage the flexibility of discriminative clustering and integrate different types of supervision as constraints on the optimization. Such supervisory signals range from video-level class labels over temporal points or sparse action bounding boxes to the dense per-frame annotation of action bounding boxes (as in a fully-supervised setting). While reaching competitive performance at a fraction of supervision used by previous methods, our model also allows us to

use and compare different types of annotation. Thanks to the flexibility of our approach, we can learn jointly from data with different levels of annotation. Experimental results are encouraging and demonstrate the usefulness of weakly-supervised methods to improve action localization by drastically reducing the labeling cost.

7.2 Perspectives for future research

This section introduces some possible directions for future research based on recent advances in the field of computer vision and machine learning.

7.2.1 Action classification

Specific datasets. Our model based on human pose exploits appearance and flow networks respectively trained on the ImageNet ILSVRC-2012 challenge [Deng et al., 2009] and UCF101 [Soomro et al., 2012] datasets. Pre-training networks on larger action classification datasets such as Kinetics [Kay et al., 2017] might increase the performance gain. Also, one interesting direction for future work would be to fine-tune one specialized network for each body part – potentially using larger external datasets – by training CNNs on image crops restricted to the corresponding body part.

Improved pose estimation. Huge progress has recently been made on human pose estimation [Toshev and Szegedy, 2014; Chen and Yuille, 2014; Newell et al., 2016; Wei et al., 2016]. We could increase the robustness of our P-CNN descriptor by taking advantage of these new methods, especially when running our model on the JHMDB [Jhuang et al., 2013] dataset where pose detection is challenging.

Modeling object interaction. Many actions involve an object of interest like a knife in a "cutting dice" task. When they are located close to the hands, information about

these objects is notably captured by the corresponding patches in our P-CNN descriptor. We can go further by modeling more sophisticated interactions as in [Prest et al., 2013; Sigurdsson et al., 2017; Kalogeiton et al., 2017b]. For example, Sigurdsson et al. [2017] propose to recognize actions with additional cues like "objects of interest", e.g., a cup for "drinking from a cup" class. Kalogeiton et al. [2017b] show promising results by jointly detecting actions and actors (*e.g.*, persons, animals or vehicules). Finally, modeling the trajectory of the object of interest would be helpful especially in the case of fine-grained action classification, *e.g.*, to distinguish "cutting dice" from "cutting slice".

Visual relations. Recent collaborations between the NLP and computer vision communities have led to very promising tasks like understanding visual relationships in still images [Lu et al., 2016; Peyre et al., 2017]. An interesting direction for future work would be to extend our P-CNN descriptor to tackle similar problems by analyzing spatial relations between people and objects but in videos.

7.2.2 Action localization

Human pose in action detection. Adapting our previous action classification model based on human pose [Chéron et al., 2015b] to action localization could improve detection. For example, Chesneau et al. [2017] use pose to obtain better spatial localization by robustly detecting people especially in the case of occlusions. Pose information could also be beneficial to structure and understand the action in order to improve classification and temporal localization performance. Indeed, pose coordinates analysis could help discovering a characteristic configuration corresponding to the beginning or end of an action. It could also help removing impossible configurations, *e.g.*, "playing flute" cannot happen if hands are far from the mouth. Moreover, modeling object interaction [Prest et al., 2013; Sigurdsson et al., 2017; Kalogeiton et al., 2017b] could help differentiating the action men-

tioned above with "apply make up". Similarly, analyzing the P-CNN hand patches could help disambiguating between these actions using the appearance of the involved object. Finally, tracking the evolution of pose coordinates to recognize static/moving configurations could be helpful to detect temporal boundaries for classes such as "brushing teeth".

Leveraging additional source of information. Recently, Miech et al. [2018] have shown promising results by introducing a model combining data from different modalities. Audio information would especially help segmenting actions temporally and understanding their content. Similar improvement could be achieved by analyzing text data from movie scripts [Duchenne et al., 2009; Bojanowski et al., 2013] or narrated instructional videos [Alayrac et al., 2016]. In the latter, the course of actions is naturally correlated with what is happening in the video.

7.2.3 Weakly-supervised action localization

Improving temporal selection. In our method for weakly-supervised action localization [Chéron et al., 2018a], temporal segmentation of human tracks is performed on the level of tracklets by assigning a prediction to each of them. For some constraints, *e.g.* temporal point, the tracklet corresponding to the annotation time is forced to belong to the labeled action together with the tracklets in the neighborhood. The latter is determined by a fixed-size interval centered at the annotation point inside which we impose strict constraints. One way to replace such interval would be to extract action proposals around its center by leveraging recent advances in this domain [Caba Heilbron et al., 2016; Escorcia et al., 2016; Shou et al., 2016; Buch et al., 2017]. Another idea is to add a contrastive cost. This idea has been inspired by Cho et al. [2015] who defined a standout scoring for unsupervised object discovery. Their score measures the maximum contrast of matching scores between a rectangular box and its surrounding boxes. Kantorov et al. [2016] have

recently used this concept for weakly-supervised object detection. The idea is to encourage the detector to select a predicted object region that is outstanding from its surrounding context region. In our case, we would define a feature representation $X_\Delta = \alpha(X - X_{out})$, where X describes the temporal segmentation candidates of the human tracks and X_{out} describes their surrounding context (*i.e.* the candidates slightly extended in time). Note that, contrary to our work [Chéron et al., 2018a] where human tracks were discretized into consecutive tracklets, here, X_i would describe the i-th segmentation candidate of a human track. Such temporal segmentation candidates would be predefined in advance, *e.g.*, the first half or the last third of the tracks. Then, we could modify the DIFFRAC objective by introducing a standout term $\mathrm{Tr}((X_\Delta W)^T Y)$ which tends to maximize the score difference between a positive action candidate and its surrounding region.

Replacing the linear classifier. A potential path for future work would be to investigate how to replace the linear classifier used in the discriminative clustering approach by a deep neural network in order to improve our model capacity [Chéron et al., 2018a]. In this method, the assignment matrix Y and the linear classifier W are updated after each iteration. Similarly to [Caron et al., 2018] who iteratively group the features with K-means and use the subsequent assignments as supervision to update the weights of a network, we could use Y as a labeling in order to train a classifier, *e.g.* a neural network, by assigning an action class to each tracklet as in a fully-supervised setting. Then, we would use the classifier to predict action scores related to the tracklets and obtain a new assignment. This can be done for example using a different linear oracle selecting the labeling that best matches the action scores while respecting the constraints. The next classifiers and assignments are then obtained iteratively.

Adding new constraints. Other constraints could be investigated in order to guide

the optimization in our weakly-supervised approach. For example, using an ordered list of actions as in [Bojanowski et al., 2014; Huang et al., 2016; Richard et al., 2017] would require even less annotation effort than a temporal point. Indeed, such annotation can be automatically extracted, *e.g.*, from a movie script without the need of looking at the video. An ordered list of actions would be a stronger constraint than an annotation containing only the set of actions performed in the video.

Bibliography

Alayrac, J.-B., Bojanowski, P., Agrawal, N., Laptev, I., Sivic, J., and Lacoste Julien, S. (2016). Unsupervised learning from narrated instruction videos. In *CVPR*. 43, 44, 48, 113, 139

Alayrac, J.-B., Sivic, J., Laptev, I., and Lacoste-Julien, S. (2017). Joint Discovery of Object States and Manipulation Actions. In *ICCV*. 48

Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*. 85

Arbeláez, P., Pont-Tuset, J., Barron, J. T., Marques, F., and Malik, J. (2014). Multiscale combinatorial grouping. In *CVPR*. 38, 111

Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., and Baskurt, A. (2011). Sequential deep learning for human action recognition. In *International Workshop on Human Behavior Understanding*. 34

Bach, F. and Harchaoui, Z. (2007). DIFFRAC: A discriminative and flexible framework for clustering. In *NIPS*. 42, 47, 48, 113, 114, 117

Bagautdinov, T., Alahi, A., Fleuret, F., Fua, P., and Savarese, S. (2017). Social scene understanding: End-to-end multi-person action localization and collective activity recognition. In *CVPR*. 38, 76

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. In *CoRR*. 79

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (SURF). *Comput. Vis. Image Underst.* 61

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. In *ECCV*. 27

Berg, T. and Belhumeur, P. N. (2013). Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*. 54

Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., and Torr, P. H. (2016). Fully-convolutional siamese networks for object tracking. In *BMVC*. 86

Bojanowski, P., Bach, F., Laptev, I., Ponce, J., Schmid, C., and Sivic, J. (2013). Finding actors and actions in movies. In *ICCV*. 42, 48, 112, 113, 117, 139

Bojanowski, P., Lajugie, R., Bach, F., Laptev, I., Ponce, J., Schmid, C., and Sivic, J. (2014). Weakly supervised action labeling in videos under ordering constraints. In *ECCV*. 43, 49, 50, 112, 117, 141

Bojanowski, P., Lajugie, R., Grave, E., Bach, F., Laptev, I., Ponce, J., and Schmid, C. (2015). Weakly-supervised alignment of video with text. In *ICCV*. 43

Bottou, L. (1998). Online learning and stochastic approximations. *On-line learning in neural networks*. 33

Brox, T., Bruhn, A., Papenberg, N., and Weickert, J. (2004). High accuracy optical flow estimation based on a theory for warping. In *ECCV*. 29, 57, 84

Buch, S., Escorcia, V., Shen, C., Ghanem, B., and Niebles, J. C. (2017). SST: Single-stream temporal action proposals. In *CVPR*. 139

Caba Heilbron, F., Carlos Niebles, J., and Ghanem, B. (2016). Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *CVPR*. 139

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In *ECCV*. 140

Carreira, J. a. and Zisserman, A. (2017). Quo vadis, action recognition? A new model and the Kinetics dataset. In *CVPR*. 34, 35, 39, 76, 97, 111, 118

Chatfield, K., Lempitsky, V., Vedaldi, A., and Zisserman, A. (2011). The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*. 59, 61

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*. 57

Chen, W. and Corso, J. J. (2015). Action Detection by implicit intentional Motion Clustering. In *ICCV*. 44, 112

Chen, X. and Yuille, A. (2014). Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*. 33, 54, 55, 71, 137

Cherian, A., Mairal, J., Alahari, K., and Schmid, C. (2014). Mixing body-part sequences for human pose estimation. In *CVPR*. 35, 55, 59, 60, 62, 63, 65, 66, 67, 68, 70

Chéron, G., Alayrac, J.-B., Laptev, I., and Schmid, C. (2018a). A flexible model for training action localization with varying levels of supervision. In *NIPS*. 20, 24, 139, 140

Chéron, G., Laptev, I., and Schmid, C. (2015a). http://www.di.ens.fr/willow/research/p-cnn/. 55

Chéron, G., Laptev, I., and Schmid, C. (2015b). P-CNN: Pose-based CNN features for action recognition. In *ICCV*. 20, 21, 138

Chéron, G., Osokin, A., Laptev, I., and Schmid, C. (2018b). Modeling spatio-temporal human track structure for action localization. In *CoRR*. 18, 20, 23

Chesneau, N., Rogez, G., Alahari, K., and Schmid, C. (2017). Detecting parts for action localization. In *BMVC*. 138

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *CoRR*. 75, 80

Cho, M., Kwak, S., Schmid, C., and Ponce, J. (2015). Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *CVPR*. 139

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *CoRR*. 80

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*. 30

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *ECCV workshop*. 29

Cula, O. G. and Dana, K. J. (2001). Compact representation of bidirectional texture functions. In *CVPR*. 29

Dahl, G. E., Yu, D., Deng, L., and Acero, A. (2012). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. In *IEEE Transactions on Audio, Speech, and Language Processing*. 73, 79

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *CVPR*. 27, 61

Dalal, N., Triggs, B., and Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. In *ECCV*. 29, 61

de Souza, C. R., Gaidon, A., Vig, E., and López, A. M. (2016). Sympathy for the details: dense trajectories and hybrid classification architectures for action recognition. In *ECCV*. 33, 73, 76

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*. 33, 57, 137

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*. 34, 71, 73, 74

Douze, M. and Jégou, H. (2014). The Yael library. In *ACM Multimedia*. 61

Duan, K., Parikh, D., Crandall, D., and Grauman, K. (2012). Discovering localized attributes for fine-grained recognition. In *CVPR*. 54

Duchenne, O., Laptev, I., Sivic, J., Bach, F., and Ponce, J. (2009). Automatic annotation of human actions in video. In *CVPR*. 42, 112, 139

Escorcia, V., Heilbron, F. C., Niebles, J. C., and Ghanem, B. (2016). Daps: Deep action proposals for action understanding. In *ECCV*. 139

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In *SCIA*. 61

Fei-Fei, L. and Perona, P. (2005). A bayesian hierarchical model for learning natural scene categories. In *CVPR*. 29

Feichtenhofer, C., Pinz, A., and Wildes, R. P. (2016a). Spatiotemporal residual networks for video action recognition. In *NIPS*. 33, 76

Feichtenhofer, C., Pinz, A., and Zisserman, A. (2016b). Convolutional two-stream network fusion for video action recognition. In *CVPR*. 33, 76

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*. 30, 61

Frank, M. and Wolfe, P. (1956). An algorithm for quadratic programming. In *Naval Research Logistics Quarterly*. 50, 113, 118

Gaidon, A., Harchaoui, Z., and Schmid, C. (2011). Actom sequence models for efficient action detection. In *CVPR*. 30

Girshick, R. (2015). Fast r-cnn. In *ICCV*. 75, 78, 85

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*. 33, 34

Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., and He, K. (2018). Detectron. `https://github.com/facebookresearch/detectron`. 44, 112, 118

Gkioxari, G. and Malik, J. (2015). Finding action tubes. In *CVPR*. 38, 57, 75, 76, 84, 109, 111

Gorban, A., Idrees, H., Jiang, Y.-G., Roshan Zamir, A. R., Laptev, I., Shah, M., and Sukthankar, R. (2015). THUMOS challenge: Action recognition with a large number of classes. `http://www.thumos.info/`. 40, 76

Graves, A., Mohamed, A.-r., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *ICASSP*. 81

Gu, C., Sun, C., Vijayanarasimhan, S., Pantofaru, C., Ross, D. A., Toderici, G., Li, Y., Ricco, S., Sukthankar, R., Schmid, C., et al. (2018). AVA: A video dataset of spatio-temporally localized atomic visual actions. In *CVPR*. 39, 98, 99, 100, 101, 109, 111, 121, 126, 127

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*. 27

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*. 97, 118

Heilbron, F. C., Escorcia, V., Ghanem, B., and Niebles, J. C. (2015). ActivityNet: A large-scale video benchmark for human activity understanding. In *CVPR*. 17, 35, 40

Hoai, M., Lan, Z.-Z., and De la Torre, F. (2011). Joint segmentation and classification of human actions in video. In *CVPR*. 38

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. In *Neural computation*. 79

Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. In *Artificial intelligence*. 29

Hou, R., Chen, C., and Shah, M. (2017). Tube convolutional neural network (T-CNN) for action detection in videos. In *ICCV*. 39, 99, 111

Huang, D.-A., Fei-Fei, L., and Niebles, J. C. (2016). Connectionist Temporal Modeling for Weakly Supervised Action Labeling. In *ECCV*. 43, 112, 141

Idrees, H., Zamir, A. R., Jiang, Y.-G., Gorban, A., Laptev, I., Sukthankar, R., and Shah, M. (2016). The THUMOS challenge on action recognition for videos "in the wild". In *Computer Vision and Image Understanding*. 40

Jain, M., Van Gemert, J., Jégou, H., Bouthemy, P., and Snoek, C. G. (2014). Action localization with tubelets from motion. In *CVPR*. 38

Jegou, H., Perronnin, F., Douze, M., Sánchez, J., Perez, P., and Schmid, C. (2012). Aggregating local image descriptors into compact codes. *TPAMI*. 29

Jhuang, H., Gall, J., Zuffi, S., Schmid, C., and Black, M. J. (2013). Towards understanding action recognition. In *ICCV*. 35, 36, 40, 54, 55, 59, 60, 64, 65, 66, 67, 71, 135, 137

Ji, S., Xu, W., Yang, M., and Yu, K. (2010). 3D convolutional neural networks for human action recognition. In *ICML*. 33

Joulin, A., Bach, F., and Ponce, J. (2010). Discriminative clustering for image co-segmentation. In *CVPR*. 48, 113

Kalogeiton, V., Weinzaepfel, P., Ferrari, V., and Schmid, C. (2017a). Action tubelet detector for spatio-temporal action localization. In *ICCV*. 39, 40, 75, 76, 77, 92, 93, 94, 96, 97, 99, 100, 109, 111, 118, 126, 136

Kalogeiton, V., Weinzaepfel, P., Ferrari, V., and Schmid, C. (2017b). Joint learning of object and action detectors. In *ICCV*. 138

Kantorov, V., Oquab, M., Cho, M., and Laptev, I. (2016). Contextlocnet: Context-aware deep network models for weakly supervised localization. In *ECCV*. 139

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *CVPR*. 74, 79

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *CVPR*. 35

Kay, W., ao Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., and Zisserman, A. (2017). The kinetics human action video dataset. In *CoRR*. 13, 16, 17, 34, 35, 98, 118, 137

Ke, Y., Sukthankar, R., and Hebert, M. (2005). Efficient visual event detection using volumetric features. In *ICCV*. 38, 111

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. In *CoRR*. 83

Klaser, A., Marszałek, M., and Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *BMVC*. 29

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *NIPS*. 32, 33, 54, 57

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., and Serre, T. (2011). HMDB: a large video database for human motion recognition. In *ICCV*. 13, 17, 19, 35, 36, 54

Lacoste-Julien, S., Jaggi, M., Schmidt, M., and Pletscher, P. (2013). Block-coordinate Frank-Wolfe optimization for structural SVMs. In *ICML*. 51, 113, 118

Laptev, I. (2005). On space-time interest points. In *IJCV*. 27, 28

Laptev, I. (2013). *Modeling and visual recognition of human actions and interactions*. Habilitation à diriger des recherches, Ecole Normale Supérieure de Paris - ENS Paris. 13

Laptev, I., Caputo, B., Schüldt, C., and Lindeberg, T. (2007). Local velocity-adapted motion events for spatio-temporal recognition. *Computer vision and image understanding*. 27, 28

Laptev, I. and Lindeberg, T. (2004). Local descriptors for spatio-temporal recognition. In *ECCV workshop*. 27

Laptev, I., Marszałek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *CVPR*. 19, 27, 29, 30, 53, 55, 61

Laptev, I. and Pérez, P. (2007). Retrieving actions in movies. In *ICCV*. 29, 38, 39, 111

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*. 29, 30

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998a). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. 33, 54

LeCun, Y., Bottou, L., Orr, G. B., and Müller, K.-R. (1998b). Efficient backprop. In *Neural networks: Tricks of the trade*. 33

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *ECCV*. 118

Liu, J., Shahroudy, A., Xu, D., and Wang, G. (2016a). Spatio-temporal LSTM with trust gates for 3D human action recognition. In *ECCV*. 74

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016b). Ssd: Single shot multibox detector. In *ECCV*. 39, 76, 97

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *ICCV*. 27

Lu, C., Krishna, R., Bernstein, M., and Fei-Fei, L. (2016). Visual relationship detection with language priors. In *ECCV*. 138

Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision. In *IJCAI*. 29, 118

Ma, S., Sigal, L., and Sclaroff, S. (2016). Learning activity progression in LSTMs for activity detection and early detection. In *CVPR*. 38, 74, 76

Marszałek, M., Schmid, C., Harzallah, H., and Van De Weijer, J. (2007). Learning object representations for visual object class recognition. In *ICCV workshop*. 32

Mathe, J. (2012). Shotdetect. `https://github.com/johmathe/Shotdetect`. 120

Mathias, M., Benenson, R., Pedersoli, M., and Van Gool, L. (2014). Face detection without bells and whistles. In *ECCV*. 85

Mettes, P., Snoek, C. G. M., and Chang, S.-F. (2017). Localizing Actions from Video labels and Pseudo-Annotations. In *BMVC*. 44, 112, 126

Mettes, P., van Gemert, J. C., and Snoek, C. G. M. (2016). Spot on: Action localization from pointly-supervised proposals. In *ECCV*. 45, 99, 110, 112, 122, 123, 125

Miech, A., Alayrac, J.-B., Bojanowski, P., Laptev, I., and Sivic, J. (2017). Learning from video and text via large-scale discriminative clustering. In *ICCV*. 51, 113, 118

Miech, A., Laptev, I., and Sivic, J. (2018). Learning a text-video embedding from incomplete and heterogeneous data. *CoRR*. 139

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *TPAMI*. 27

Newell, A., Yang, K., and Deng, J. (2016). Stacked hourglass networks for human pose estimation. In *ECCV*. 33, 54, 55, 137

Ng, J. Y.-H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., and Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. In *CVPR*. 34, 74, 81, 91

Ni, B., Paramathayalan, V. R., and Moulin, P. (2014). Multiple granularity analysis for fine-grained action detection. In *CVPR*. 35

Nowak, E., Jurie, F., and Triggs, B. (2006). Sampling strategies for bag-of-features image classification. In *ECCV*. 32

Oneata, D., Revaud, J., Verbeek, J., and Schmid, C. (2014). Spatio-temporal object detection proposals. In *ECCV*. 38, 111

Oneata, D., Verbeek, J., and Schmid, C. (2013). Action and event recognition with fisher vectors on a compact feature set. In *ICCV*. 30, 31, 61, 66

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*. 34

Osokin, A., Alayrac, J.-B., Lukasewitz, I., Dokania, P. K., and Lacoste-Julien, S. (2016). Minding the gaps for block Frank-Wolfe optimization of structured SVMs. In *ICML*. 119

Peng, X. and Schmid, C. (2017). Multi-region two-stream R-CNN for action detection. In *HAL*. 38, 75, 76, 77, 99, 100, 109, 111

Peng, X., Zou, C., Qiao, Y., and Peng, Q. (2014). Action recognition with stacked fisher vectors. In *ECCV*. 30

Perronnin, F., Liu, Y., Sánchez, J., and Poirier, H. (2010a). Large-scale image retrieval with compressed fisher vectors. In *CVPR*. 31

Perronnin, F., Sánchez, J., and Mensink, T. (2010b). Improving the fisher kernel for large-scale image classification. In *ECCV*. 30, 31, 61

Peyre, J., Laptev, I., Schmid, C., and Sivic, J. (2017). Weakly-supervised learning of visual relations. In *ICCV*. 17, 138

Pigou, L., van den Oord, A., Dieleman, S., Van Herreweghe, M., and Dambre, J. (2017). Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. In *IJCV*. 34, 76

Pirsiavash, H. and Ramanan, D. (2014). Parsing videos of actions with segmental grammars. In *CVPR*. 38

Pishchulin, L., Andriluka, M., Gehler, P., and Schiele, B. (2013). Poselet conditioned pictorial structures. In *CVPR*. 35, 55

Prest, A., Ferrari, V., and Schmid, C. (2013). Explicit modeling of human-object interactions in realistic videos. In *TPAMI*. 138

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*. 85, 97, 118

Richard, A., Kuehne, H., and Gall, J. (2017). Weakly supervised action learning with rnn based fine-to-coarse modeling. In *CVPR*. 43, 112, 141

Rodriguez, M. D., Ahmed, J., and Shah, M. (2008). Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*. 40

Rohrbach, M., Amin, S., Andriluka, M., and Schiele, B. (2012). A database for fine grained activity detection of cooking activities. In *CVPR*. 13, 36, 54, 68

Rohrbach, M., Rohrbach, A., Regneri, M., Amin, S., Andriluka, M., Pinkal, M., and Schiele, B. (2016). Recognizing fine-grained and composite activities using hand-centric features and script data. In *IJCV*. 40

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. In *Parallel distributed processing*. 33

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. In *IJCV*. 32, 118

Saha, S., Singh, G., and Cuzzolin, F. (2017). Amtnet: Action-micro-tube regression by end-to-end trainable deep architecture. In *ICCV*. 39, 99, 111

Saha, S., Singh, G., Sapienza, M., Torr, P. H., and Cuzzolin, F. (2016). Deep learning for detecting multiple space-time action tubes in videos. In *BMVC*. 38, 39, 41, 75, 76, 77, 78, 83, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 99, 100, 102, 103, 105, 109, 111, 136

Sapp, B. and Taskar, B. (2013). Modec: Multimodal decomposable models for human pose estimation. In *CVPR*. 35, 55, 59

Sapp, B., Weiss, D., and Taskar, B. (2011). Parsing human motion with stretchable models. In *CVPR*. 35, 55

Schmid, C. and Mohr, R. (1997). Local grayvalue invariants for image retrieval. *TPAMI*. 27

Schuldt, C., Laptev, I., and Caputo, B. (2004). Recognizing human actions: A local SVM approach. In *ICPR*. 27, 30, 53, 55

Shou, Z., Wang, D., and Chang, S.-F. (2016). Temporal action localization in untrimmed videos via multi-stage CNNs. In *CVPR*. 38, 139

Sigurdsson, G. A., Divvala, S. K., Farhadi, A., and Gupta, A. (2017). Asynchronous temporal fields for action recognition. In *CVPR*. 138

Simonyan, K., Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2013). Fisher vector faces in the wild. In *BMVC*. 29, 31

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *NIPS*. 33, 34, 35, 55, 56, 57, 73

Singh, B., Marks, T. K., Jones, M., Tuzel, O., and Shao, M. (2016). A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *CVPR*. 34, 38, 74, 76

Singh, G., Saha, S., Sapienza, M., Torr, P., and Cuzzolin, F. (2017). Online real time multiple spatiotemporal action localisation and prediction. In *CoRR*. 38, 39, 76, 77, 99, 109, 111, 118

Singh, K. K. and Lee, Y. J. (2017). Hide-and-Seek: Forcing a Network to be Meticulous for Weakly-supervised Object and Action Localization. In *ICCV*. 43, 112

Siva, P. and Xiang, T. (2011). Weakly Supervised Action Detection. In *BMVC*. 44, 112

Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *ICCV*. 29

Soomro, K. and Shah, M. (2017). Unsupervised action discovery and localization in videos. In *ICCV*. 44, 112

Soomro, K., Zamir, A. R., and Shah, M. (2012). UCF101: A dataset of 101 human actions classes from videos in the wild. In *CRCV-TR-12-01*. 13, 17, 35, 40, 57, 75, 76, 84, 137

Srivastava, N., Mansimov, E., and Salakhutdinov, R. (2016). Unsupervised learning of video representations using LSTMs. In *CoRR*. 34

Sutskever, I., Martens, J., and Hinton, G. E. (2011). Generating text with recurrent neural networks. In *ICML*. 73

Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In *CVPR*. 33

Taralova, E. H., De la Torre, F., and Hebert, M. (2014). Motion words for videos. In *ECCV*. 30

Taylor, G. W., Fergus, R., LeCun, Y., and Bregler, C. (2010). Convolutional learning of spatio-temporal features. In *ECCV*. 33

Tokmakov, P., Alahari, K., and Schmid, C. (2017). Learning video object segmentation with visual memory. In *ICCV*. 80, 89

Tompson, J. J., Jain, A., LeCun, Y., and Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*. 54, 55

Toshev, A. and Szegedy, C. (2014). DeepPose: Human pose estimation via deep neural networks. In *CVPR*. 33, 35, 54, 55, 137

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). Learning spatiotemporal features with 3D convolutional networks. In *ICCV*. 33

Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. In *IJCV*. 38, 111

van Gemert, J. C., Jain, M., Gati, E., and Snoek, C. G. M. (2015). APT: Action localization proposals from dense trajectories. In *BMVC*. 38, 111

Varol, G., Laptev, I., and Schmid, C. (2017). Long-term temporal convolutions for action recognition. In *TPAMI*. 33, 76

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell: A neural image caption generator. In *CVPR*. 74

Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2011). Action recognition by dense trajectories. In *CVPR*. 30, 31, 61, 135

Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2013). Dense trajectories and motion boundary descriptors for action recognition. *IJCV*. 29, 30, 31, 61

Wang, H. and Schmid, C. (2013). Action recognition with improved trajectories. In *ICCV*. 27, 30, 53, 54, 55, 59, 61, 66

Wang, L., Xiong, Y., Lin, D., and Gool, L. V. (2017). UntrimmedNets for Weakly Supervised Action Recognition and Detection. In *CVPR*. 43, 112

Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional pose machines. In *CVPR*. 33, 54, 55, 137

Weinzaepfel, P., Harchaoui, Z., and Schmid, C. (2015). Learning to track for spatio-temporal action localization. In *ICCV*. 38, 75, 76, 84, 99, 109, 111

Weinzaepfel, P., Martin, X., and Schmid, C. (2016). Human action localization with sparse spatial supervision. In *CoRR*. 14, 19, 38, 40, 41, 45, 75, 76, 77, 78, 84, 85, 88, 90, 99, 101, 102, 110, 112, 113, 121, 123, 124, 125, 126, 127, 136

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. (2013). Deepflow: Large displacement optical flow with deep matching. In *ICCV*. 29

Xu, C., Hsieh, S.-H., Xiong, C., and Corso, J. J. (2015). Can humans fly? action understanding with multiple classes of actors. In *CVPR*. 26

Xu, L., Neufeld, J., Larson, B., and Schuurmans, D. (2004). Maximum margin clustering. In *NIPS*. 43, 47, 113, 114

Yang, J. and Yuan, J. (2017). Common action discovery and localization in unconstrained videos. In *ICCV*. 44, 112

Yang, Y. and Ramanan, D. (2011). Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*. 35, 54, 55, 59, 65

Yeung, S., Russakovsky, O., Mori, G., and Fei-Fei, L. (2016). End-to-end learning of action detection from frame glimpses in videos. In *CVPR*. 38, 74, 76

Yuan, J., Ni, B., Yang, X., and A.Kassim, A. (2016). Temporal action localization with pyramid of score distribution features. In *CVPR*. 38, 74, 76

Yue-Hei, N. J., Matthew, H., Sudheendra, V., Oriol, V., Rajat, M., and George, T. (2015). Beyond short snippets: Deep networks for video classification. In *CVPR*. 33, 34, 35, 55

Zach, C., Pock, T., and Bischof, H. (2007). A duality based approach for realtime tv-l 1 optical flow. In *Joint Pattern Recognition Symposium*. 29

Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., and Lin, D. (2017). Temporal action detection with structured segment networks. In *ICCV*. 38

Zhou, Y., Ni, B., Hong, R., Wang, M., and Tian, Q. (2015). Interaction part mining: A mid-level approach for fine-grained action recognition. In *CVPR*. 35, 67, 68

Zhou, Y., Ni, B., Yan, S., Moulin, P., and Tian, Q. (2014). Pipelining localized semantic features for fine-grained action recognition. In *ECCV*. 68

Zolfaghari, M., Oliveira, G. L., Sedaghat, N., and Brox, T. (2017). Chained multi-stream networks exploiting pose, motion, and appearance for action classification and detection. In *ICCV*. 39, 99, 111

Résumé

La compréhension automatique de vidéos devrait impacter notre vie de tous les jours dans de nombreux domaines comme la conduite autonome, les robots domestiques, la recherche et le filtrage de contenu, les jeux vidéo, la défense ou la sécurité. Le nombre de vidéos croît plus vite chaque année, notamment sur les plateformes telles que YouTube, Twitter ou Facebook. L'analyse automatique de ces données est indispensable pour permettre à de nouvelles applications de voir le jour.

L'analyse vidéo, en particulier en environnement non contrôlé, se heurte à plusieurs problèmes comme la variabilité intra-classe (les échantillons d'un même concept paraissent très différents) ou la confusion inter-classe (les exemples provenant de deux activités distinctes se ressemblent). Bien que ces difficultés puissent être traitées via des algorithmes d'apprentissage supervisé, les méthodes pleinement supervisées sont souvent synonymes d'un coût d'annotation élevé. Dépendant à la fois de la tâche à effectuer et du niveau de supervision requis, la quantité d'annotations nécessaire peut être prohibitive. Dans le cas de la localisation d'actions, une approche pleinement supervisée nécessite les boîtes englobantes de l'acteur à chaque image où l'action est effectuée. Le coût associé à l'obtention d'un telle annotation empêche le passage à l'échelle et limite le nombre d'échantillons d'entraînement. Trouver un consensus entre les annotateurs est également difficile et mène à des ambiguïtés dans l'étiquetage (*Où commence l'action ? Quand se termine-t-elle ? Que doit inclure la boîte englobante ?* etc.).

Cette thèse adresse les problèmes évoqués ci-dessus dans le contexte de deux tâches, la *classification* et la *localisation* d'actions humaines. La *classification* consiste à reconnaître l'activité effectuée dans une courte vidéo limitée à la durée de l'action. La *localisation* a pour but de détecter en temps et dans l'espace des activités effectuées dans de plus longues vidéos. Notre approche pour la classification d'actions tire parti de l'information contenue dans la posture humaine et l'intègre avec des descripteurs d'apparence et de mouvement afin d'améliorer les performances. Notre approche pour la localisation d'actions modélise l'évolution temporelle des actions à l'aide d'un réseau récurrent entraîné à partir de suivis de personnes. Enfin, la troisième méthode étudiée dans cette thèse a pour but de contourner le coût prohibitif des annotations de vidéos et utilise le regroupement discriminatoire pour analyser et combiner différents types de supervision.

Mots Clés

vision par ordinateur, apprentissage statistique, reconnaissance d'action, apprentissage profond, compréhension des vidéos.

Abstract

Automatic video understanding is expected to impact our lives through many applications such as autonomous driving, domestic robots, content search and filtering, gaming, defense or security. Video content is growing faster each year, for example on platforms such as YouTube, Twitter or Facebook. Automatic analysis of this data is required to enable future applications.

Video analysis, especially in uncontrolled environments, presents several difficulties such as intra-class variability (samples from the same concept appear very differently) or inter-class confusion (examples from two different activities look similar). While these problems can be addressed with the supervised learning algorithms, fully-supervised methods are often associated with high annotation cost. Depending on both the task and the level of required supervision, the annotation can be prohibitive. For example, in action localization, a fully-supervised approach demands person bounding boxes to be annotated at every frames where an activity is performed. The cost of getting such annotation prohibits scalability and limits the number of training samples. Another issue is finding a consensus between annotators, which leads to labeling ambiguities (*where does the action start? where does it end? what should be included in the bounding box? etc.*).

This thesis addresses above problems in the context of two tasks, namely human action *classification* and *localization*. The former aims at recognizing the type of activity performed in a short video clip trimmed to the temporal extent of the action. The latter additionally extracts the space-time locations of potentially multiple activities in much longer videos. Our approach to action classification leverages information from human pose and integrates it with appearance and motion descriptors for improved performance. Our approach to action localization models the temporal evolution of actions in the video with a recurrent network trained on the level of person tracks. Finally, the third method in this thesis aims to avoid a prohibitive cost of video annotation and adopts discriminative clustering to analyze and combine different levels of supervision.

Keywords

computer vision, machine learning, action recognition, deep learning, video understanding.