Enhanced image and video representation for visual recognition

Mihir Jain

ANNÉE 2014

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

Ecole doctorale Matisse

présentée par

Mihir Jain

préparée au centre Inria Rennes - Bretagne Atlantique
Institut National de Recherche en Informatique et Automatique
(Université de Rennes 1)

Enhanced image and video representation for visual recognition

Thèse soutenue à Rennes
le date 9 Avril 2014

devant le jury composé de :

Prof. Arnold Smeulders
University of Amsterdam, rapporteur
Prof. Matthieu Cord
Université UPMC La Sorbonne, rapporteur
Dr. Cordelia Schmid
DR Inria Grenoble, examinateur
Prof. Eric Marchand
Université de Rennes 1, examinateur
Dr. Patrick Pérez
DR Technicolor, Rennes examinateur
Dr. Patrick Bouthemy
DR Inria Rennes, co-encadrant de thèse
Dr. Patrick Gros
DR Inria Rennes, directeur de thèse
Dr. Hervé Jégou
CR Inria Rennes, co-directeur de thèse

Acknowledgements

I am honored to have been given this opportunity to complete my PhD under the guidance of Hervé Jégou, Patrick Bouthemy and Patrick Gros. I thank them profusely for their oversight and constant interest towards my betterment. It has been my greatest pleasure to have met my ideal supervisor in Hervé. His constant counsel has been most valuable towards improving my ideas, writing skills and has influenced me to the core of my research culture. I have also learned a lot on a personal front due to his perfection towards work and infectious optimism. I will miss working with him, the discussions and especially the parties at his place.

I am also grateful to Patrick Bouthemy for his inputs and feedback on my thesis work. His expertise was pivotal to our contribution in action recognition, as were the detailed explanations and advice which helped during the analysis of our ideas and results. I thank Patrick Gros for his support and feedback on my research work, especially, his constructive comments on papers and presentations.

I am thankful to Cees Snoek and Jan van Gemert for their supervision and support during my visit to the University of Amsterdam. Thanks to Charlotte for her help in arranging this visit. I would also like to extend my thanks to all the reviewers and examiners for reading and examining my manuscript and for the thought-provoking discussions.

The support and assistance of my lab mates and friends from TEXMEX during this period has been invaluable. Special thanks to Giorgos and Josip for their valuable suggestions and understanding. Thanks to Hervé, Patrick, Arthur and Ioana for help with the French translation, "Résume" has been entirely translated by Hervé, Thanks! My thanks go to Deepak for corrections, Ahmet for proof reading and Petra for being such a fun office mate. I would also like to thank Wanlei, Raghav, Andrei, Jonathan, Miaojing, Ali, Sébastien, Jiangbo, Nghi and Rachid for being wonderful colleagues and making everyday-life better at the workplace. A special thanks to all my friends for their words of encouragement and support.

Most importantly, I want to thank my brother, Himalaya and my mother for their unconditional love, support and understanding. This thesis is dedicated to my late father who has always been an inspiration and who continues to look out for me.

CONTENTS

Résumé

Les images et vidéos sont devenues omniprésentes dans notre monde, en raison d'Internet et de la généralisation des appareils de capture à bas coût. Ce déluge de données visuelles requiert le développement de méthodes permettant de les gérer et de rendre leur exploitation possible pour des usages tant personnels que professionnels.

La compréhension automatique des images et vidéos permet de faciliter l'accès à ce type de contenu au travers d'information de haut niveau, appartenant au langage humain. Par exemple, une image peut être décrite par l'ensemble des objets qui y apparaissent, leur relations, et pour les vidéos par les actions des personnes qui s'y meuvent. Il est préférable, voire nécessaire, d'obtenir cette annotation de manière automatique, car l'annotation manuelle est limitée et coûteuse. Une autre contrainte est de pouvoir obtenir cette annotation de manière efficace, afin de pouvoir traiter les gros volumes de données générés par des ensembles importants d'utilisateurs.

La reconnaissance visuelle offre une variété d'applications dans plusieurs contextes, de l'intelligence artificielle à la recherche d'information. La reconnaissance efficace des images et/ou de leur contenu peut être utilisée pour organiser des collections de photos, pour identifier des lieux, rechercher des photos similaires, reconnaître des produits tels que des CD ou du vin, d'effectuer des recherches ciblées dans des vidéos, ou de permettre à des robots d'identifier des objets dans des contextes industriels. Un exemple d'une application populaire et utilisée massivement est Google Goggles, disponible sur téléphones mobiles pour la reconnaissance d'objets, de lieux, de code-bars, etc.

En vidéo, la reconnaissance et la localisation d'actions ou d'événements permet d'assister les systèmes de vidéo-surveillance, l'analyse automatique de séquences sportives, la recherche ciblée de lieux ou d'objets dans la vidéo, le résumé automatique, ou encore la reconnaissance gestuelle dans des contextes d'interface homme-machine.

Plus largement, une meilleure compréhension du contenu des images et des vidéos peut être vue comme un pas capital, voire un pré-requis, vers la reconnaissance intelligente de l'environnement basée sur la vision, avec des applications en particulier sur l'interaction homme-robot et la conduite automatique.

Motivation et objectifs

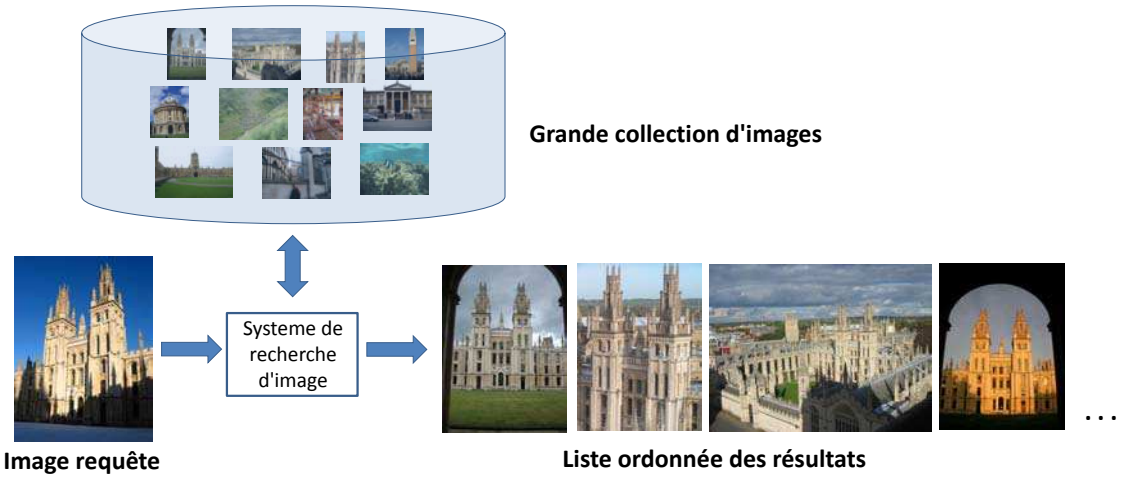
La compréhension du contenu visuel est le but fondamental de la vision par ordinateur. Plusieurs tâches types participent à cet objectif. Chacune de ces tâches prises individuellement est aussi associée à des applications.

En image, les tâches usuelles sont la recherche d'image, la classification d'objet, de catégorie d'objets ou de scènes, la détection d'objet et la segmentation d'image. De la même manière, on distingue en vidéo la reconnaissance de copies, la classification d'action et la localisation (temporelle et/ou spatio-temporelle) de ces actions.

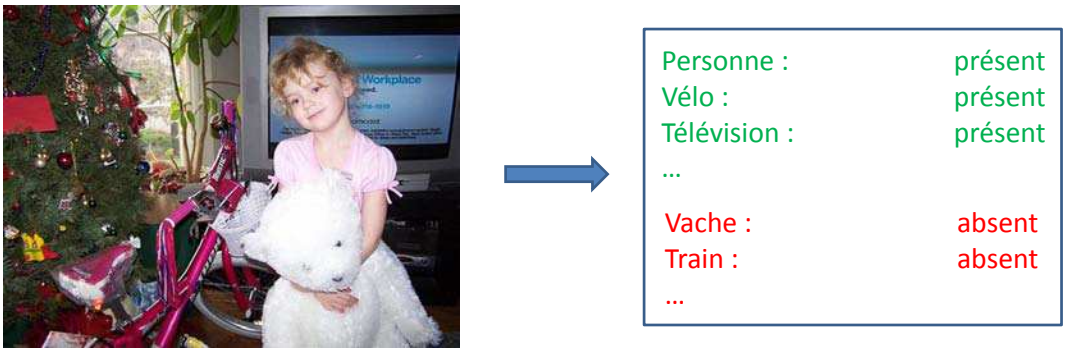
L'objectif de cette thèse est d'améliorer les représentations des images et des vidéos, dans le but d'obtenir une reconnaissance visuelle élaborée, tant pour des entités spécifiques que pour des catégories plus génériques. Les contributions de cette thèse portent, pour l'essentiel, sur les méthodes de génération de représentations abstraites du contenu visuel, à même d'améliorer l'état de l'art sur les tâches de reconnaissance suivantes :

- *Recherche d'image* : L'objectif de cette tâche est de retourner les images qui ressemblent le plus à une requête donnée, ou qui contiennent des objets ou lieux visuellement similaires. L'image requête est comparée efficacement aux images d'une grande collection. Un score de similarité est produit qui permet d'ordonner les images de la collection selon leur pertinence attendue. Un exemple est montré à la figure 1(a).
- *Classification d'image* : La tâche vise à déterminer si une scène ou un objet d'un certain type est présent dans une image. Un classifieur est appris par catégorie (ou un classifieur multi-classe) afin d'estimer si l'image contient un objet de la classe visée. La figure 1(b) illustre cette tâche. Sur cet exemple, le classifieur associé à la classe ≪personne≫ doit retourner un bon score tandis que la classe ≪vache≫ doit recevoir un score faible.
- *Classification d'action en vidéo* : De manière analogue à la classification d'image, l'objectif est de déterminer si une action d'un certain type apparaît ou non dans une vidéo donnée, comme sur l'exemple de la figure 1(c).
- *Localisation d'action en vidéos :* Cette tâche vise à identifier et à localiser une action d'intérêt. Elle s'apparente à la classification d'action, mais doit déterminer en plus où et quand l'action apparaît. La figure 1(c) est un exemple pour l'action ≪répondre au téléphone≫, où l'action est localisée par une boîte englobante verte (*où*) et où la flèche verte indique la localisation temporelle (*quand*).

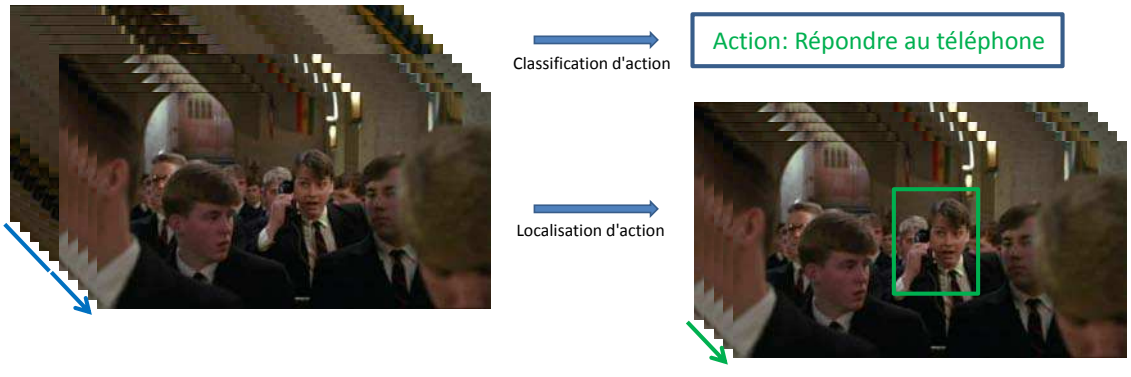
La représentation des images et des vidéos est souvent très liée : si l'on exclut le canal audio, les vidéos peuvent être vues comme des suites d'images, et il n'est donc pas étonnant que de nombreuses méthodes ont été étendues à la vidéo après avoir été introduites en image.



(a) Recherche d'image : un système type.



(b) Classification d'image : exemple pour la classification multi-classe.



(c) Classification et localisation d'action : en haut, l'action «répondre au téléphone» est reconnue, tandis qu'en bas elle est de plus localisée par une boîte englobante verte (elle représente en fait une suite temporelle de boîtes englobantes, qui n'est pas illustrée ici). Remarquez que la sortie a un nombre d'images plus faible, indiquant une restriction temporelle de la longueur de l'action.

Figure 1: Tâches de reconnaissance visuelle.

Dans ce manuscrit, nous commençons par considérer le problème le plus élémentaire, à savoir la recherche d'image, avant de considérer la classification d'image. Nous traiterons ensuite le cas des vidéos, en proposant en particulier plusieurs contributions visant à mieux exploiter le mouvement pour la classification d'action. Finalement, le dernier chapitre s'intéresse à la localisation d'actions au sein de vidéos.

Défis

Quelle que soit la tâche visuelle abordée, un des principaux défis consiste à décrire le contenu visuel de manière à ce que les caractéristiques importantes soient encodées et permettent de discriminer le contenu d'intérêt du contenu non pertinent. Parmi les autres étapes, on peut citer le besoin de méthodes d'appariement rapides ou de classification. Ci-dessous, nous détaillons quelques points relatifs à la description du contenu, principal objet de cette thèse.

Reconnaissance visuelle dans les images : De nombreux aspects rendent la recherche d'image difficile. Ils peuvent être catégorisés en deux types :

- Les variations d'apparence : la robustesse aux changements d'apparence des objets est requise tant pour la recherche que pour la classification d'image. De nombreuses sources de variation existent pour un type ou une classe d'objets, telles que les variations d'illumination, les changements d'échelle et de taille, les autres objets qui apparaissent dans les images, les changements de point de vue ou encore les occultations.
- Rapidité et passage à l'échelle : La recherche d'images similaires au sein d'une très grande base contenant des millions d'images est souvent associée à un contexte applicatif où les résultats doivent être présentés dans un délai très court à l'utilisateur. Cela requiert des solutions très rapides, mais aussi très précises afin de limiter le risque de retourner des faux positifs. L'efficacité est aussi critique pour la classification d'image, par exemple dans le cas d'une application de recherche par le contenu sémantique à partir d'une requête textuelle. La classification d'images devient coûteuse en ressources lorsque le nombre de classes devient important.

Reconnaissance d'actions dans les vidéos : Les variations d'apparence mentionnées ci-dessus sont également présentes pour les tâches de reconnaissance d'action. Sur des propriétés purement vidéos et liées à l'aspect temporel, mentionnons que les occultations changent en fonction du temps, par exemple des piétons qui se croisent ou occultent l'action d'intérêt pendant un certain laps de temps. Plus généralement, les changements de point de vue (ou changements de plan), d'échelle, d'éclairage et de fond peuvent apparaître au cours du temps. À cela s'ajoutent les mouvements de caméra, qui complexifient généralement à la description des actions.

Enfin, dans un contexte de détection, l'espace de recherche requis pour la localisation des actions est encore plus grand que dans un contexte de détection d'image, dans la mesure où il s'agit de déterminer à la fois *où* et *quand* l'action d'intérêt apparaît.

Les variations d'apparence au sein d'une classe d'objet ou d'action compliquent plus encore la classification. Si les facteurs de variations mentionnés ci-dessus causent des changements d'apparence, une difficulté encore plus grande vient de la variabilité au sein d'une classe. Par exemple, l'une des difficultés inhérente à la classification d'actions dans les vidéos est le fait que qu'il peut y avoir différentes manières d'opérer la même action ou activité humaine. À l'inverse, des classes différentes peuvent avoir des apparences similaires, comme deux races de chien, ou les actions ≪courir≫ et ≪marcher≫.

Contributions

Les contributions de cette thèse sont organisées en quatre chapitres, chacun traitant d'une tâche de reconnaissance visuelle différente. Nous les résumons ci-dessous.

Recherche d'image (Chapitre 3). Ce chapitre présente une méthode de plongement de Hamming asymétrique pour la recherche d'image à grande échelle à partir de descripteurs locaux. La comparaison de deux descripteurs repose sur une mesure vecteur-à-code, ce qui permet de limiter l'erreur de quantification associée à la requête par rapport à la méthode de plongement de Hamming originale (symétrique). L'approche est utilisée en combinaison avec une structure de fichier inversé qui lui offre une grande efficacité, comparable à celle d'un système à base de sacs de mots.

La comparaison avec l'état de l'art est effectuée sur deux jeux de données, et montre que l'approche proposée améliore la qualité de la recherche par rapport à la version symétrique. Le compromis mémoire-qualité est evalué, et montre que la méthode est particulièrement utile pour des signatures courtes, offrant une amélioration de 4% de précision moyenne.

Classification d'image (Chapitre 4). Ce chapitre décrit un nouveau cadre de classification d'image à partir d'appariement de descripteurs locaux. Plus précisément, nous adaptons la méthode de plongement de Hamming, introduite initialement dans un contexte de recherche d'image, à un contexte de classification. La technique d'appariement repose sur la comparaison rapide des signatures binaires associées aux descripteurs locaux. Ces vecteurs binaires permettent de raffiner la recherche par rapport à une méthode utilisant uniquement des mots visuels, ce qui limite le bruit de quantification. Ensuite, afin de permettre l'utilisation de noyaux linéaires efficaces de type machine à vecteur support, nous proposons un plongement des votes dans un espace de scores, alimenté par les appariements produits par le plongement de Hamming.

Des expériences effectuées sur les jeux d'évaluation PASCAL VOC'2007 et Caltech 256 montrent l'intérêt de notre approche, qui obtient de meilleurs résultats que toutes les méthodes à base d'appariement de descripteurs locaux, et produit des résultats compétitifs par rapport aux approches de l'état de l'art à base de techniques d'encodage.

Classification d'action (Chapitre 5). Plusieurs travaux récents sur la reconnaissance d'action ont montré l'importance d'intégrer explicitement les caractéristiques de mouvement dans la description des vidéos. Ce chapitre montre l'intérêt d'une décomposition adéquate du mouvement visuel en un mouvement dominant et un mouvement résiduel, c'est-à-dire essentiellement entre un mouvement de caméra et le mouvement des éléments mobiles de la scène. Cette décomposition est utile tant lors de l'extraction de trajectoires spatio-temporelles que pour le calcul des descripteurs associés au mouvement, et offre un gain de performance conséquent pour la reconnaissance d'action.

Ensuite, nous introduisons un nouveau descripteur de mouvement, le descripteur DCS, qui exploite les quantités scalaires différentielles de mouvement, à savoir les propriétés de divergence, de vorticité et de cisaillement. Ces informations ne sont pas capturées par les descripteurs de mouvement de l'état de l'art et notre descripteur apporte donc une complémentarité qui, en combinaison avec les descripteurs usuels, améliore les résultats. Finalement, pour la première fois, nous utilisons la méthode de codage VLAD initialement introduite en recherche d'image dans un contexte de reconnaissance d'action.

Ces trois contributions offrent des gains complémentaires et apportent un gain substantiel à l'état de l'art sur les jeux de données Hollywood 2, HMDB51 et Olympic Sports.

Localisation d'action (Chapitre 6). Ce chapitre considère le problème de la localisation d'action, où l'objectif est de déterminer où et quand une action d'intérêt apparaît dans la vidéo. Nous introduisons une stratégie d'échantillonnage de volumes spatio-temporels, sous la forme de séquences de boîtes englobantes 2D+t, appelées tubelettes. Par rapport aux stratégies de l'état de l'art, cela réduit drastiquement le nombre d'hypothèses à tester lors de la phase de classification des zones spatio-temporelles.

Notre méthode s'inspire d'une technique récente d'échantillonnage introduite en localisation d'image. Notre contribution ne se limite pas à l'adapter pour la reconnaissance d'actions. D'une part, nous utilisons une partition de la vidéo en super-voxels. D'autre part, nous introduisons un critère d'échantillonnage utilisant le mouvement et permettant d'identifier comment le mouvement lié à l'action dévie du mouvement du fond associé à la caméra.

L'intérêt de notre approche est démontré par des expériences effectuées sur deux jeux importants d'évaluation pour cette tâche, à savoir UCF Sports et MSR-II. Notre approche améliore nettement l'état de l'art, tout en limitant la recherche des actions à une fraction des séquences de boîtes englobantes possibles.

Abstract

The objective of this work is to improve image and video representation for visual recognition, where high level information such as objects contained in the images or actions in the videos are automatically extracted. There are many visual recognition tasks that can lead to better management of and systematic access to visual data. This manuscript, specifically investigates (i) Image Search (for both image and textual query) and (ii) Action Recognition (both classification and localization).

In image retrieval, images similar to the query image are searched from a large dataset, typically with more than a million images. On this front, we propose an asymmetric version of Hamming Embedding method, where the comparison of query and database descriptors relies on a vector-to-binary code comparison. This limits the quantization error on the query side while having a limited impact on efficiency. Then we consider image search with textual query, *i.e.*, image classification, where the task is to identify if an image contains any instance of the queried category. Our contribution is to propose a novel approach based on match kernel between images, more specifically based on Hamming Embedding similarity. As a secondary contribution we present an effective variant of the SIFT descriptor, which leads to a better classification accuracy.

Handling camera motion and using flow information is a crucial aspect of video representation for action recognition. We improve action classification by proposing the following methods: (i) dominant motion compensation, which generates improved trajectories and better motion descriptors; (ii) a novel descriptor based on kinematic features of flow, namely diversion, curl and shear. We depart from *Bag-of-words* and for the first time in action recognition use other higher-order encoding, namely VLAD.

The last contribution and chapter is devoted to action localization. The objective is to determine where and when the action of interest appears in the video. Most of the current methods localize actions as a cuboid, while we do it spatio-temporally, *i.e.*, as a sequence of bounding boxes, which is more precise. We propose a selective sampling strategy to produce 2D+t sequences of bounding boxes, which drastically reduces the candidate locations. Our sampling strategy advantageously exploits a criterion that takes in account how motion related to actions deviates from the background motion.

We thoroughly evaluated all the proposed methods on real world images and videos from challenging benchmarks. Our methods outperform the previously published related state of the art and at least remain competitive with the subsequently proposed methods.

CHAPTER

ONE

Introduction

Visual data in form of images and videos have become ubiquitous, thanks to Internet and improved accessibility to an exploding amount of video and image data, which in due, in particular, to the proliferation of cheap digital cameras and smartphones. Be it a holiday trip or a party or an event, loads of pictures are taken and instantly shared through social networking sites such as Facebook, Flickr, Pinterest, Instagram etc. Recently Facebook revealed that users have uploaded more than 250 billion photos to the site, and currently, on an average 350 million photos are uploaded per day. Instagram in its 3rd year has reached over 150 million users and 16 billion photos shared. A billion likes happen each day on just Instagram, which gives an idea of how widely these images are viewed.

Video sharing sites are not lagging behind with over 60 hours of videos uploaded each minute on YouTube. These are disseminated at an amazing rate: 700 YouTube videos are shared on Twitter every minute and 500 years of YouTube videos are watched on Facebook every day. The omnipresence of photos and videos on the internet is evident from these statistics. This explosion of visual data naturally calls for methods to manage and make it usable. Image and video understanding methods can make images and videos searchable, explorable by extracting high-level information from them. It could be recognizing objects or scenes contained in the images or actions performed in the videos. This has to be based on visual content to avoid expensive and manual tagging. It is also very important to do these visual recognition tasks efficiently, in order to deal with the large volumes and to meet the user demands.

Visual recognition has a variety of potential applications with scope in areas of artificial intelligence and information retrieval. Efficient recognition of image and its content can be used for organizing photo collections, identification of places on maps, content-based image search, finding similar products (*e.g.*Google Goggle), video data mining, object identification for mobile robots. Video-based action recognition, event detection and locating actions and events can further assist in video surveillance, sports play analysis, web-based video retrieval, video summarization, gesture-based human computer interfaces and vision-based intelligent environments. Better understanding of both images and videos would advance towards human-robot interaction and assisted driving.

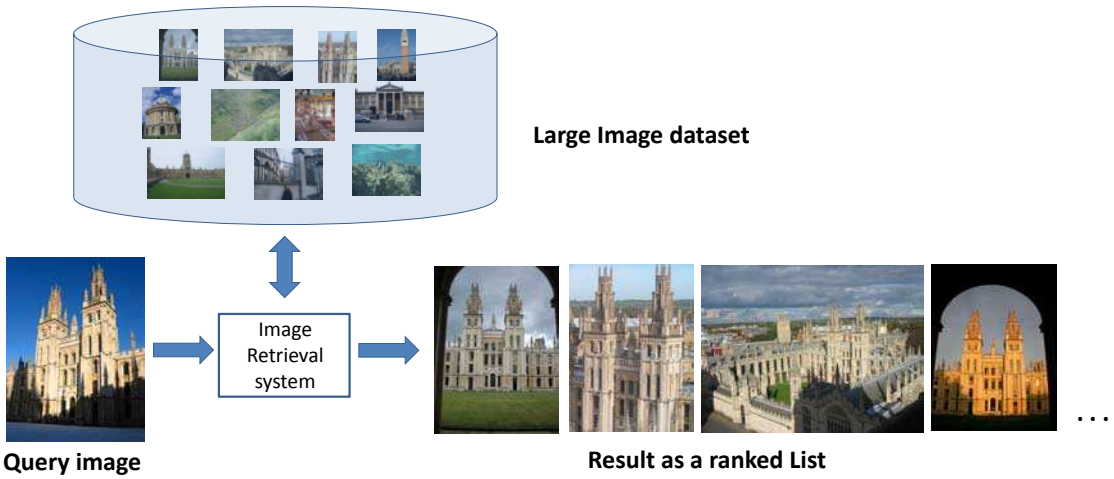
1.1 Motivation and objectives

Understanding visual content of images and videos is the fundamental goal of Computer Vision. Several tasks contribute and facilitate this goal, while being themselves useful in various applications. In images, typical tasks are image retrieval, image (scene or object) classification, object detection, image segmentation. Image retrieval or image search aims at finding from a large set of images the ones that best resemble the query image. The task of image classification is to find if a particular scene or object is present in the given image. Object detection or localization additionally requires to localizing the identified object it with a bounding box. In semantic segmentation, the given image is divided into coherent regions, which are categorized simultaneously.

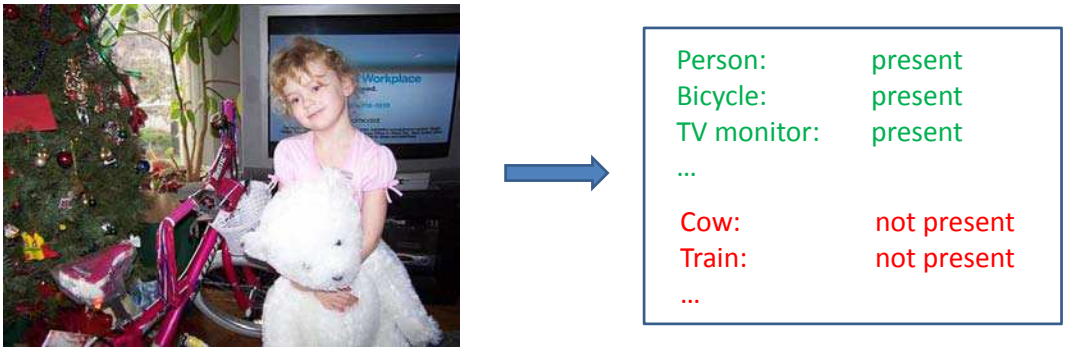
Similarly in videos, typical tasks are video copy detection, action classification and action localization (temporal or spatio-temporal). Object classification and detection can also be done in videos. Content-based Copy Detection methods match duplicates (*i.e.*, exact copies) or near-duplicates (*i.e.*, some noise or a few changes) to the original image or video. The goal of action classification is to determine *which* action appears in the video, while action localization additionally finds *when and where* action of interest appears.

The aforementioned tasks are regarded as standard in visual recognition, and may be categorized in two types. The first is specific instance recognition, in which the aim is to identify instances of a particular object or entity; for example the Eiffel Tower or a given painting. Some sample tasks in this category are image retrieval, content based copy detection, etc. Typically, these tasks rely on matching of images, frames or local features. The second is generic category recognition, where the objective is to recognize different instances belonging to a given conceptual class, such as 'car', 'table', 'hug' or 'diving'. In such cases, models for categories are learned from training examples. In all of the recognition tasks of either case, one very important aspect is the representation of images or videos. Our objective in this dissertation is to enhance image and video representations to achieve improved visual recognition of both specific entities and generic categories. Conforming to this goal, we contribute mainly towards how to produce abstract representations of the visual content amenable to improve the following four recognition tasks:

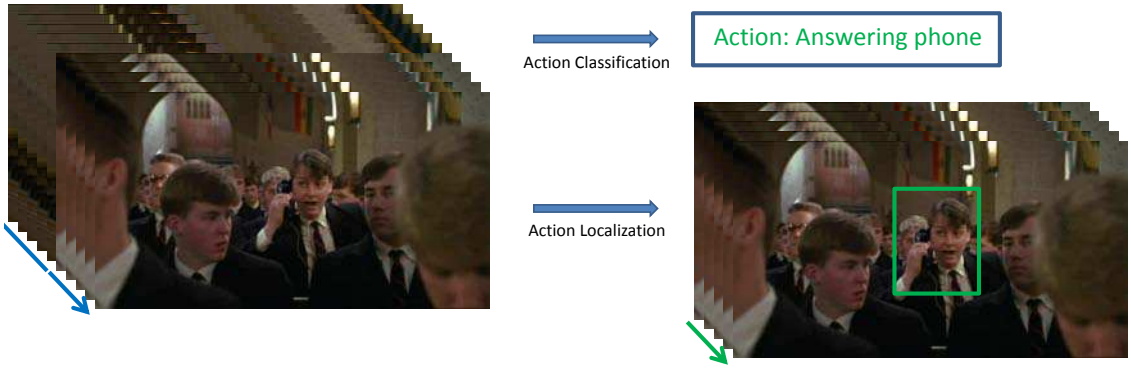
- Image retrieval: A query image is efficiently matched with a large database of more than a million images to obtain a list of similar images ranked according to similarity score. An example is shown in Figure 1.1(a).
- Image classification: A classifier is learned for each category (or a multi-class classifier is learned), given an image it identifies if the object is present or not. For example in Figure 1.1(b), a classifier for 'person' should find it or have a high classification score while the one for 'cow' should have a low score.
- Action classification in videos: Similar to image classification, the goal is to find if



(a) Image retrieval, a typical setup.



(b) Image classification, an illustration of multi-class classification.



(c) Action classification and localization: On top action "answering phone" is recognized while in the bottom it is shown to be localized by a green box (actually sequence of boxes which is not shown). Not that the output has less number of frames and green arrow highlighting temporal localization.

Figure 1.1: Visual recognition tasks.

the action of interest is present in the given video or not (Figure 1.1(c)).

- Action localization in videos: Identify the action of interest and then locate where and when it happens. Figure 1.1(c), shows the action "answering phone" is located by a green bounding box (*where*) along with green arrow to signify temporal localization (*when*).

Representing images and videos are very connected operations. Videos are sequences of images so it is not surprising that many methods are analyzed and adapted from images to videos. We start with the most fundamental problem of these four, image retrieval and then move to recognizing object categories in images, image classification. Then we proceed to videos and enhance the motion descriptors for action classification, in particular by better exploiting the motion. Finally, we go a step further to localize actions in videos.

1.2 Challenges

Take any of the visual recognition tasks, the challenge lies often in describing the visual content adequately such that the characteristics are encoded and the representation is discriminative. Definitely other stages such as matching, classification are also critically involved. We here mention some of the prominent types of challenges for visual recognition in images and videos.

Visual recognition in images:
First we discuss factors that make image search and classification challenging problems.

- Variations in Appearance: Robustness to object appearance changes is key to both image search and classification. Some examples of appearance variations are shown in Figure 1.2, and are briefly described here:
  - Illumination variation: Change in lightening condition causes large variation in intensity values of pixels and thus has a major influence on the appearance.
  - Scale and size variation: Such variations can significantly influence the inter and intra class similarity. Also when the size of the object of interest is very small it becomes difficult to recognize it as its share to the image representation can be severely limited.
  - Background clutter / Environment variations: Highly complex background leads to confusion between foreground objects and the background and thus to false positives in both in image search and classification.
  - Occlusion and truncation: Visibility of the object of interest is hampered when occluded by some other objects or truncated by image borders. This obviously

Figure 1.2: Examples of variations in appearance

affects the recognition task as the representation or the model learned has to be sufficiently robust to handle such cases.

- Viewpoint variations: Viewpoint position of camera relative to the object and the objects appearing in various poses can significantly change the appearance.

- Speed and scalability: Searching for similar images in a huge database containing millions of images needs to be near real time, so that user can browse the result at the same time. This requires highly efficient solutions and not to mention accurate also to limit the false positives. As accessing hard drive is slow, with compact representation [59, 64] of images, it is possible to load the database in RAM and search can be carried on a single machine for millions of images. Speed helps for scalability, but apart from that, searching a large number of database images means searching a needle in a haystack. The method needs to be robust to such distractors. The speed is also critical for image classification, for instance in the case of search based on semantic using textual query. For example, finding instances of "boat" from the image database. Classification of images gets more demanding as number of classes increases. The image representation needs to be rich enough to encode the relevant visual information to distinguish between all possible classes.

Visual recognition in videos:

The above variations in appearance also affect videos at the frame level. With the additional dimension of *time*, the occlusions change their positions, *e.g.*, pedestrians walking and occluding action of interest. Similarly, viewpoint (shot change), scale, lightening condition, background can vary over time. Another source of variation is the diverse ways of performing the same type of action. We now discuss the challenges inherent

(a) Camera motion: Only person is moving but there is motion induced due to camera motion, which can be seen as optical flow in the background.

(b) Different ways of 'situps' and 'brushing-hair'.

(c) Different types of human activities (from left to right): 'smile' is subtle while 'cartwheel' involves lot of movement, 'kiss' could be of either type. Action of 'kicking' here involves interaction with an object, ball.

Figure 1.3: Challenges in action recognition (frames are from Hollywood2 [89], HMDB51 [75] and UCF Sports [118] datasets)

to the complexity of both actions and videos that are faced by action recognition under uncontrolled conditions.

- Camera motion:
  Motion is at the core of video representation for action recognition, however very often it includes motion induced by moving camera when dealing with uncontrolled and realistic situations. An example of motion induced by camera is shown in Figure 1.3(a). It does not necessarily mean that the camera motion is nuisance, it gives useful information in many scenarios such as in sports videos. Appropriately separating the motion from the camera and that related to the action improves action recognition, as shown in Chapter 5.
- Articulation and actor movement variations:
  Different actors perform actions at different speed and with varying execution style. Action representation and learned models need to be robust to such variations. Figure 1.3(b) gives two examples of actors performing same action in different ways.

- Types of human activities:
  Some human activities can be distinguished with subtle difference, *e.g.*, *eating*, *smiling*, *smoking*, etc. These are gesture like actions and involve little movement. Other actions are more evident in the video, such as *running*, *cartwheeling*, *riding horse* etc. There are actions that vary a lot in execution style, such as *kissing*, *hugging*, which can be very subtle and at times more animated. Then, there are activities involving interactions among humans, objects, and scenes. Figure 1.3(c) gives examples of types of human activities. Certain activities (sports actions) are sequential, but more often actions are set of atomic actions rather than a sequence, that is why Bag-of-Words like approaches have done better.
- Huge search space when localizing actions:
  Action localization is much more difficult task than action classification, simply because it involves finding *where* and *when* action happens in the video addition to classifying it. Typically localization is done by determining a cuboid or subvolume [19, 129, 160] containing the action. The search space here is huge, in $\mathcal{O}(h^2w^2n^2)$, where the video size is $h \times w \times n$. Another way adopted recently is to localize action as a sequence of bounding boxes, also called spatio-temporal localization [77, 132]. In this case the search space for possible spatio-temporal paths in the video space is exponential [132]. This motivates the need for sampling the potential spatio-temporal paths in a different manner. We propose one such an approach in Chapter 6.

Intra class variations and inter class similarity complicate the classification and therefore recognition. Above factors do cause intra class variation because of appearance changes. But apart from that, some instances of same class look very different, for example two breeds of dog. Similarly high inter class similarity is also observed, 'walking'/'running' is an example of instances of different classes that can look very similar.

1.3 Contributions

The contributions of this dissertation are organized into four main chapters, each focused on a different recognition task. Here, we briefly describe each of them.

Image retrieval (Chapter 3). This chapter presents an asymmetric Hamming Embedding scheme for large scale image search based on local descriptors. The comparison of two descriptors relies on a vector-to-binary code comparison, which limits the quantization error associated with the query compared with the original Hamming Embedding method. The approach is used in combination with an inverted file structure that offers high efficiency, comparable to that of a regular bag-of-features retrieval systems. The comparison is performed on two popular datasets. Our method consistently improves the search quality over the symmetric version. The trade-off between memory usage

and precision is evaluated, showing that the method is especially useful for short binary signatures, typically giving improvement of 4% of mean average precsion (mAP). This work was published in [57].

Image classification (Chapter 4). In this chapter, we present a novel image classification framework based on patch matching. More precisely, we adapt the Hamming Embedding technique, first introduced for image search to improve the bag-of-words representation. This matching technique allows the fast comparison of descriptors based on their binary signatures, which refines the matching rule based on visual words and thereby limits the quantization error. Then, in order to allow the use of efficient and suitable linear kernel-based SVM classification, we propose a mapping method to cast the scores output by the Hamming Embedding matching technique into a proper similarity space. Comparative experiments of our proposed approach and other existing encoding methods on two challenging datasets PASCAL VOC 2007 and Caltech-256, report the interest of the proposed scheme, which outperforms all methods based on patch matching and even provides competitive results compared with the state-of-the-art coding techniques. This work was published in [55].

Action classification (Chapter 5). Several recent works on action classification have attested the importance of explicitly integrating motion characteristics in the video description. This chapter establishes that adequately decomposing visual motion into dominant and residual motions, *i.e.*, into camera and scene motion, both in the extraction of the space-time trajectories and for the computation of descriptors, significantly improves action recognition algorithms. Then, we design a new motion descriptor, the DCS descriptor, based on differential motion scalar quantities, divergence, curl and shear features. It captures additional information on the local motion patterns enhancing results. Finally, applying the recent VLAD coding technique proposed in image retrieval provides a substantial improvement for action recognition. Our three contributions are complementary and lead to outperform all the previously reported results by a significant margin on three challenging datasets, namely Hollywood 2, HMDB51 and Olympic Sports. Our work was published in [56]. More recently, some of the approaches [103, 147, 164] further improved the results, one of the main reasons being use of the Fisher vector encoding. Therefore, in this chapter we also employ Fisher vector. Additionally, we further enhance our approach by combining trajectories from both optical flow and compensated flow.

Action localization (Chapter 6). This chapter considers the problem of action localization, where the objective is to determine when and where certain actions appear. We introduce a sampling strategy to produce 2D+t sequences of bounding boxes, called tubelets. Compared to state-of-the-art techniques, this drastically reduces the number of hypotheses that are likely to include the action of interest. Our method is inspired

by a recent technique introduced in a context of image localization. Beyond considering this technique for the first time for videos, we revisit this strategy for 2D+t sequences obtained from super-voxels. Our sampling strategy advantageously exploits a criterion that reflects how action-related motion deviates from background motion.

We demonstrate the interest of our approach by extensive experiments on two public datasets: UCF Sports and MSR-II. Our approach significantly outperforms the state-of-the-art on both datasets, while restricting the search of actions to a fraction of possible bounding box sequences. This work has been accepted to be published in [58].

CHAPTER

TWO

Visual representation methods

In this chapter, we discuss image and video representation methods for visual recognition. The purpose is to provide the pre-requisites related to the core contributions of this thesis. The literature reviews for the specific visual recognition tasks we address are covered separately in the respective chapters. We do not use any supervised learning for optimizing visual representation in this thesis. K-means clustering is employed for unsupervised learning. For classification tasks, we use support vector machine (SVM) [23]. We give more details in the related chapters.

The goal of visual representation is to convert image or video into a mathematical representation such that "similar" images (or videos) have similar representation and "dissimilar" images have dissimilar representations, with respect to some comparison metrics. The representation has to be informative and robust to different types of variations as discussed in Section 1.2. At the same time, its computation, storing and processing have to be efficient. These are three conflicting requirements and the trade-off is application dependent. One way is to compute a single global descriptor per image or video. This approach, though highly scalable, is sensitive to background clutter, truncation/occlusion, scale or viewpoint change, camera motion, duration variations of actions. To handle these variations more local approach is adopted, *i.e.*, many local regions are extract and descriptors are computed for them.

There are primarily two ways to use such collections of local descriptors for recognition. The first is to match local descriptors to compute a similarity score between two images or videos, for instance by counting the number of inliers. Another way aggregating the local descriptors of an image to produce a global representation as a single vector. Section 2.1 briefly discusses the methods for extracting and describing local features from images and videos. In subsequent sections, we discuss various methods for matching and aggregating descriptors.

2.1 Local features from images and videos

2.1.1 Local features for image search

Selecting image patches and describing them in a meaningful way is a cornerstone of most image representations. In general, this is carried out in two steps: (i) detecting interest points/regions in the image; (ii) extracting a local descriptor for each region. Since only a subset of image regions are represented by these feature descriptors, this provides a sparse representation of the image.

Feature detectors
The detectors basically locate stable keypoints (and their supporting regions), which allows matching the same image-region found in two images despite variations in viewpoint or scale. Some of the popular feature detectors include Harris-Affine and Hessian-Affine [93], Difference of Gaussians (DoG) [85], Laplacian-of-Gaussian (LoG) [85, 93], Maximally Stable Extremal Regions (MSER) [90]. The interest point detectors can be categorized as: corner detector, blob detector and region detector. Details and comparisons can be found in survey by Mikolajczyk *et al.* [95].

Another option is to densely sample feature points from a regular grid instead of interest point detection. This has been mainly shown useful in classification [68, 21, 81, 83, 100], where almost always better results are obtained with dense sampling. It is somewhat equivalent to giving more data to the classifier and letting it decide what is more discriminative and informative. Recently dense sampling has been also shown to be useful in image/scene/object retrieval [46].

Feature descriptors
Once a set of feature points is obtained, the next step is to choose a region around each point and compute a vector that describes it. These descriptors characterize the visual pattern of the local patches supporting the feature points, hence also known as local descriptors. They must be distinctive and robust to image transformations to ensure that similar local patches extracted from different images with different transformations are similarly represented. Arguably the most popular feature descriptor is SIFT (scale invariant feature transform) [85] and then there are its variants SURF [11], Daisy [130], GLOH [94] that are also prominent among local descriptors. Since SIFT is the only type of descriptor used in this thesis, we review it in more details.

SIFT describes a feature point by a "histogram"[1] of image gradient orientation and location. For each keypoint an orientation histogram with 36 bins weighted by magnitude

[1]Formally, it is not a histogram because the accumulation is done with gradient magnitudes and not by counting occurrences.

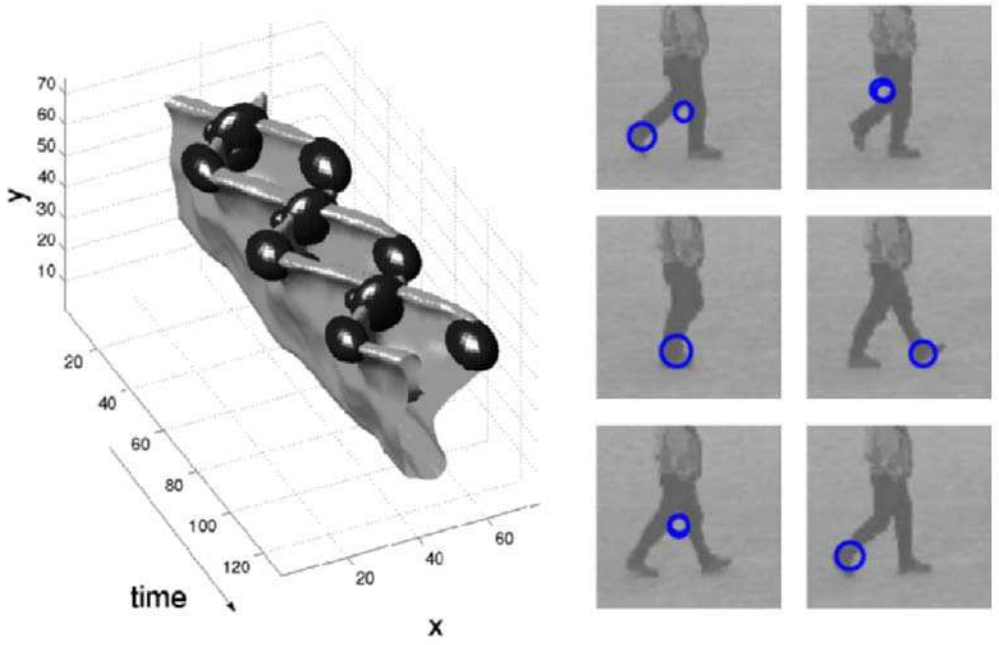


Figure 2.1: Detected spatio-time interest points (the ellipsoids on right) for a walking person correspond to the upside-down level surface of a leg pattern (left). Image courtesy of [78].

and Gaussian-mask is computed. The dominant gradient orientation is detected and assigned for each keypoint. Before the descriptor computation, the local patch around each keypoint is rotated to its dominant orientation. This alignment gives rotation invariance. To compute descriptor the local patch is partitioned into a grid, [85] reports 4×4 partitioning. The gradient orientations are quantized into 8 bins for each spatial cell, which leads to 16 8-bin histogram; *i.e.*, a 128 dimensional SIFT descriptor. As gradients are used, the descriptor is invariant to additive intensity changes. Also due to the use of spatial binning, the descriptor is robust to some level of geometric distortion and noise. In [95], the SIFT descriptor has been shown to outperform other descriptors and always offers satisfactory performance under different contexts. Recently, two variants of SIFT descriptor were proposed in [6, 55] that yield superior performance for many visual recognition tasks. Both are similar and involve component wise square-rooting of each descriptor.

2.1.2 Local features for action recognition

Most of the state-of-the-art action recognition methods represent video as a collection of local space-time features. These features and their local descriptors capture shape and motion information for a local region in a video. Local features provide a representation that is robust to background clutter, multiple motions, spatio-temporal shifts of actions and are useful for action recognition under uncontrolled video conditions. In the last few

years, action recognition methods based on local features have shown excellent results on a wide range of challenging data [45, 89, 98, 118, 145, 148].

In videos, there are two types of "interesting locations" which can be used for local description: (i) spatio-temporal features and (ii) point trajectories. In the following, we first discuss these two and then we briefly review popular descriptors computed around the feature points or along the trajectories.

Spatio-temporal feature detectors

Similar to images, feature detectors select characteristic locations both in space and time. These spatio-temporal features are efficiently extracted to represent the visual content of local sub-volumes of video. Many of the detectors are 3-D generalizations of image feature detectors. One of the most popular ones is the space-time interest points (STIP) detector proposed in Laptev *et al.* [78] as an extension of the Harris cornerness criterion. Figure 2.1 illustrates an example of STIPs. Dollár *et al.* [30] noted that true spatio-temporal corner points are relatively rare in some cases and proposed their interest point detector to yield denser coverage in videos. Another instance is the Hessian3D detector proposed by Willems *et al.* [151], which is a spatio-temporal extension of the Hessian blob detector.

Trajectories

It is more intuitive to treat 2D space domain and 1D time domain in video individually as they have different characteristics. A straightforward alternative to detecting features in a joint 3D space is to track the spatial feature points through the video. This is an efficient way to encode the local motion consistent with actions. There are methods employing long-term trajectories in literature that aim to associate every scene entity to the same motion track. Approaches based on long term trajectories , e.g., [3, 67, 92, 115] have shown to recognize certain actions using only trajectory and velocity information. But tracking a feature point [18, 82] for many frames is very challenging and faces difficulties due to occlusion, fast and articulated human motion, appearance variations, etc. Problem of drifting while tracking is another factor in long range trajectories.

Another way is to extract shorter trajectories of fixed length less than 15–20 frames. These are more like local features as they combine advantages of both long-term trajectories and local spatio-temporal features. Similar to local features, (short-term) trajectories can be extracted easily and reliably [145], for instance by tracking points in optical flow field computed using dependable algorithms such as proposed by Farnebäck [40]. Due to the limited and fixed durations of the trajectories, many methods designed for spatio-temporal features have been easily adapted for trajectories. Trajectons of Matikainen *et al.* [91], motion trajectories of Wang *et al.* [145] and *tracklets* of Gaidon *et al.* [44] are some examples.

Dense Sampling of features or trajectories at regular positions in space and time is another option. Analogous to images, in videos dense sampling has been shown to perform better than interest points [146, 148]. Wang *et al.* [148] show that dense sampling outperforms state-of-the-art spatio-temporal interest point detectors. Wang *et al.* [145, 146] sample feature points on a dense grid and track them to obtain dense trajectories, which perform significantly better than sparse tracking techniques, such as the KLT tracker. They report excellent results on many action recognition datasets. A dense representation better captures surrounding context along with foreground motion.

Feature descriptors

Spatio-temporal features and trajectories are described by robust feature descriptors. The descriptors capture the motion and shape information in the local neighborhood of the feature points and trajectories. An image feature point is simply a location at a certain scale, whereas a video feature point = $(x, y, t, \sigma, \tau)^T$ is located at $(x, y, t)^T$ in the video sequence with σ and τ as the spatial and temporal scales, respectively. The support region is now a cuboid instead of a rectangle, which is subdivided into a set of $M \times M \times N$ cells. Each cell is represented by a histogram to encode some types of visual information such as gradient orientation, optical flow, etc. Not surprisingly and analogous to feature detectors, many feature descriptors are spatio-temporal extensions of their 2D counterparts. For instance, an extension of image SIFT descriptors to 3D was proposed by Scovanner *et al.* [122]. Each pixel from 3D patch based on its gradient orientation votes into $M \times M \times M$ grid of local histograms. Similarly, an extended SURF (ESURF) was proposed by Willems *et al.* [151].

Kläser *et al.* [70] proposed histograms of 3D gradient orientations, HOG3D, an extension of Histograms of Oriented Gradients (HOG) [25] to the space-time domain. They developed a quantization method for 3D orientation based on regular polyhedrons. The number of bins of the cell histogram depends on the type of polyhedron chosen. The descriptor for a feature point concatenates 3D gradient histograms of all cells which are normalized separately. Laptev *et al.* [79] also introduced HOG for videos, which differs from HOG3D. The main difference is that it uses 2D gradient orientations, but since the histograms are computed for 3D cells, the temporal information is also included in the descriptor. They also proposed Histograms of Optical Flow (HOF) to capture local motion. The authors [79] used 4 bins for HOG histograms and 5 bins for HOF histograms. This parameter can vary depending on the application or dataset.

Trajectory shape and velocities have also been used as descriptors [44, 66, 145] for action recognition with impressive results especially with trajectory shape descriptors. Recently, Motion boundary histogram (MBH) of Dalal *et al.* [26] was extended for action recognition by Wang *et al.* [145]. Since then it has come out as one of the best descriptors contributing to the state-of-the-art performances of many methods [5, 44, 56, 66, 146, 147].

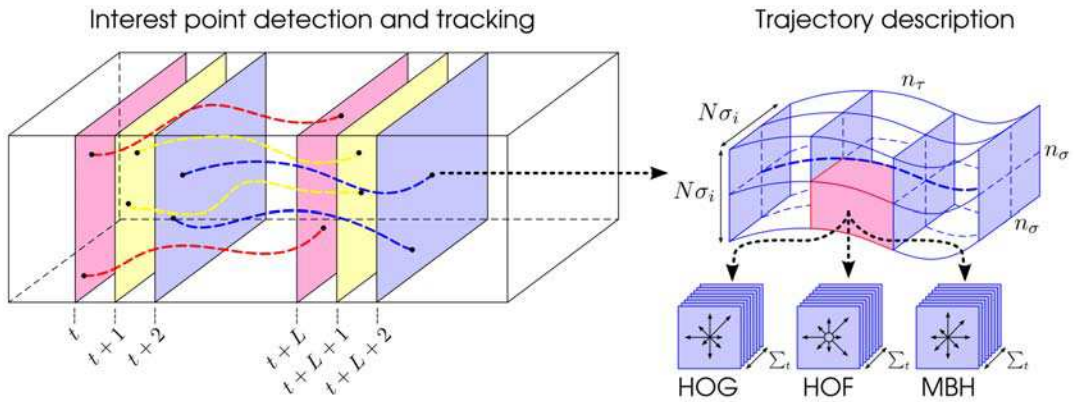


Figure 2.2: On left trajectories are shown passing through frames, supporting region for a trajectory and its cell histogram computation are displayed on the right. Image courtesy of [145].

Trajectories are also used like local features and descriptors such as HOG, HOF and MBH are computed along them. Support region of a trajectory (sequence of points) was defined as sequence of 2D patches in [145] and followed in many other works [44, 56, 66, 145, 147]. Figure 2.2 illustrates how a descriptor is computed for the supporting region of a trajectory.

2.2 Image and video representation for recognition

Any visual recognition task typically involves either matching or learning a classifier or sometimes both. Matching could be between local descriptors or global vectors, leading to a similarity score. Such an approach is often called matching-based method. On the other hand, a global representation is preferred for learning a classifier. The choice of matching or learning is made typically based on the application and the visual representation used. In this section, we discuss the representations based on local features that are employed for recognition. The two possible types of such representations are: (i) Global Representation (Aggregated) and (ii) Set of local features (Non-aggregated).

Global representation (aggregated) are obtained by first encoding and then aggregating local descriptors into a single vector. It is convenient to use such a representation for classification [24, 109, 139] as many classifiers, such as SVM, can be trained to discriminate the positive and negative vectors. A global representation is also suitable for retrieval [64, 65] as images/videos can be directly compared. It improves the scalability as there is no need to save information related to descriptor matches.

Set of local features (non-aggregated): In voting-based approaches, the local descriptors are matched to compute the similarity scores between two images or videos. Here,

instead of aggregating the local features to produce the representation, it is the matching scores that are aggregated for similarity computation. It is a natural choice for retrieval and many of the state-of-the-art methods [123, 60] can be seen as voting-based. Although this choice is more prevalent in retrieval area [22, 60, 62, 63, 111, 123], it has also been used for image classification [14, 33, 69] and action recognition [116]. With such a representation, usually matching-based approach is used. Of course, a global representation can also be obtained, *e.g.*, from matching scores.

First, we discuss the global representation Bag of Words (BOW) [24, 123] in Section 2.3; Vector of locally aggregated descriptors (VLAD) [64] and Fisher Vector [107, 109] in Section 2.4. These have been used for classification as well as retrieval. Then, voting-based or non-aggregated representations are reviewed in Section 2.5. This includes Hamming Embedding [60] and a few matching-based methods for classification.

2.3 Bag of visual words

Bag of visual words was introduced in retrieval by Sivic and Zisserman in their seminal "Video-Google" [123] work, where they presented image/video retrieval system inspired by text retrieval. Concurrently, Csurka *et al.* [24] showed the interest of this approach for image classification. Since then, over the years it has been one of the most popular and successful approaches proposed in the field of visual recognition, which has been extended and improved in many ways for image retrieval [60, 64] and classification [107, 163]. It has been also employed by many methods for object localization [76, 137, 142] and action classification [34, 44, 79, 146].

From local descriptors to global representation is a 3-step process: (1) Visual Codebook Creation, (2) Encoding and (3) Aggregation. These are common for BOW and other related global representations. To our knowledge, the NeTra toolbox [86] is the first work to follow this pipeline.

Visual vocabulary creation: The first step is to build a codebook or a visual vocabulary. It consists of a set of reference vectors known as "visual words". These are created from a large set of local descriptors often using an unsupervised clustering approach to partition the descriptor space into, say, K clusters. They are associated with K representative vectors $(\mu_1, \mu_2, ..., \mu_K)$, referred to as visual words. A visual vocabulary or codebook was initially built by using k-means clustering [24, 123, 152]. Later its variants were introduced such as more efficient hierarchical k-means [99] or Gaussian Mixture Model (GMM) [107] to obtain a probabilistic interpretation of the clustering. It is well known that the classification accuracy improves when increasing the size of the codebook, as experimentally shown in some works [21, 140]. There are other approaches to learn a codebook such as meanshift-clustering [68] and sparse dictionary learning [87, 149].

Also methods with supervised dictionary learning have been proposed [73, 96, 108, 152].

Encoding local descriptors: The local descriptors are encoded using the visual vocabulary. The simplest way is hard assignment where each descriptor is assigned to its nearest visual word. This vector quantization (VQ) is the simplest and probably the most popular encoding. Though this representation is very compact, VQ is lossy and leads to quantization errors. Choosing the vocabulary size (K) is a trade-off between quantization noise and descriptor noise. Depending on the size (for large K or small clusters) we may have very similar descriptors assigned to different visual words. Conversely, with larger clusters very different descriptors may get assigned to the same cluster.

Several better encoding techniques have been proposed to limit quantization errors. For instance, coding techniques based on soft assignment such as [108, 152], Kernel codebook [112, 141] or sparse-coding [15, 87, 149, 158]. In Kernel codebook, each descriptor is softly assigned to all the visual words with weights proportional to $exp(-\frac{d^2}{2\sigma^2})$, where d is the distance between cluster center and descriptor. Soft-assignment has also been used with a sparsity constraint on the reconstruction weight in Mairal *et al.* [87] and Yang *et al.* [158]. Locality-constrained linear coding by Wang *et al.* [149] reconstructs a local descriptor by using a soft-assignment over a few of the nearest visual words from the codebook. Another coding technique, namely Super-vector coding [163], is similar to Fisher and VLAD, which we discuss in the next sub-sections.

Aggregating local descriptors: The final step is to aggregate the encoded features into a global vector representation. The features are pooled in one of the two ways: sum pooling or max pooling. In sum-pooling, the encoded features are additively combined into a bin or a part of the final vector. For example, in case of BOW histogram it is simply adding the count of features belonging to bin corresponding to each visual word. For max pooling, the highest value across the features is assigned for each bin, as done in Yang *et al.* [158]. Avila *et al.* [9] propose an improved pooling approach called BOSSA (Bag Of Statistical Sampling Analysis). In this work, the local descriptors are pooled (sum pooling) per cluster based on their distances from the cluster center and the resulting histograms from all the clusters are concatenated. Recently, this approach is further improved in an approach coined BossaNova [8].

The bag-of-words is an order-less representation and thus invariant to the layout of the given image or video. A standard way of incorporating weak geometry is to divide the image into spatial grid (or spatio-temporal in case of video) and aggregate each spatial cell separately. This was first introduced by Lazebnik *et al.* [81] as spatial pyramids consisting of several layers, where in layer l the image is divided in $2^l \times 2^l$, each such cell being described by a histogram. Since then, many different types of spatial grids have been used, e.g., 3×1 and various spatio-temporal grids for videos. The concept can be used for any of the encodings mentioned here, including VLAD and Fisher, by

computing one encoding for each spatial region and then stacking the results.

In the case of image classification, the aforementioned global vectors are used to train a model. For retrieval with BOW, the following process is followed.

BOW in retrieval: matching with Visual Words
BOW histogram or a vector of visual words frequencies is $L2$ normalized. The rare visual words are assumed to be more discriminative and are assigned higher weights; this is done according to the *tf-idf* (term frequency–inverse document frequency) scheme [123]. The similarity measure between two BOW vectors is, most often, cosine similarity. Vocabulary size (K) is a parameter which is usually very large for image retrieval (typically $20,000$ or larger). With around few thousand descriptors per image the BOW histograms are very sparse, which enables very efficient retrieval, thanks to inverted file indexing. Inverted file is a set of inverted lists, where each list is associated with a visual word. All the local descriptors corresponding to the same visual word are stored in the same list with their image ids. Another option is to store image id and number of descriptors in the image belonging to that particular visual word.

2.4 Higher-order representations

BOW only counts the number of local descriptors assigned to each visual word or a set of visual words (in case of soft-assignment). Including higher-order statistics, that is mean and covariance of local descriptors, lead to much improved representations. The objective is to model the approximate distribution of descriptors in each cluster and aggregate these higher-order statistics.

2.4.1 Fisher vectors

Perronnin *et al.* [107] introduced the Fisher Vector for image representation that employs Gaussian Mixture Model (GMM) for vocabulary building. Fisher encoding captures both first and second order statistics by aggregating all residuals (vector differences between descriptors and Gaussian means), normalized by the variance of the corresponding Gaussian component. The key idea is based on the Fisher kernel of Jaakkola *et al.* [54].

Fisher kernel: The Fisher kernel is based on the gradient of the log-likelihood of a generative probabilistic model with respect to its parameters. Given a likelihood function u_λ with parameters λ, the score function of a sample with T observations, $X = \{x_t, t = 1...T\}$, is given by:

$$G_\lambda^X = \bigtriangledown_\lambda \log\ u_\lambda(X) \tag{2.1}$$

Intuitively, this gives the direction in which the parameters λ should be moved to better fit the data. As the gradient is with respect to the model parameters, the dimension of G_λ^X does not depend on the dimension of x_t or T and is equal to the dimension of λ. The Fisher kernel is given by the normalized inner product of two Fisher score vectors:

$$\mathcal{K}_{FK}(X, Y) = G_\lambda^X I_F^{-1} G_\lambda^Y \tag{2.2}$$

where $I_F = E_{x\sim u_\lambda}[G_\lambda^X G_\lambda^X]$ denotes the Fisher information matrix. As I_F is positive-semi-definite, so is its inverse, which can be decomposed as: $I_F^{-1} = L_\lambda' L_\lambda$. Now the Fisher Kernel can be rewritten as a dot product between two Fisher Vectors, which for sample X is given as:

$$\mathcal{G}_\lambda^{\mathcal{X}} = L_\lambda G_\lambda^X = L_\lambda \bigtriangledown_\lambda \log\ u_\lambda(X) \tag{2.3}$$

Fisher Vector for visual representation: To represent visual data, a Gaussian Mixture Model (GMM) is used as the probabilistic generative model of the local descriptors [107, 109]. The GMM defines the visual vocabulary, whose parameters are $\lambda = \{\alpha_k, \mu_k, \Sigma_k\}_{k=1}^K$, *i.e.*, the mixture weights, means and covariances (diagonal). Assuming that the local descriptors are independent, we can write:

$$G_\lambda^X = \frac{1}{T}\sum_{t=1}^{T} \bigtriangledown_\lambda \log\ u_\lambda(x_t) \tag{2.4}$$

where $u_\lambda(x_t)$ is a GMM of K Gaussians.

For the weight parameters, the soft-max formalism of Krapac *et al.* [74] can be used, $\pi_k = \frac{exp(\alpha_k)}{\sum_j exp(\alpha_j)}$. The posterior probability of Gaussian k for descriptor x_t is given as:

$$q_{tk} = \frac{p(x_t|\mu_k, \Sigma_k)\pi_k}{\sum_{j=1}^K p(x_t|\mu_j, \Sigma_j)\pi_j}$$

and with the new mixing weights π_k, GMM $u_\lambda(x_t)$ is given by:

$$u_\lambda(x) = \sum_{k=1}^{K} \pi_k u_k(x) \tag{2.5}$$

To compute the Fisher Vector, an analytically closed-form approximation of the Fisher information matrix is proposed [107]. In this case, the normalization of the gradient by L_λ is simply a whitening w.r.t. mixture weights, means and covariance. The gradients are given by:

$$\bigtriangledown_{\alpha_k} G_\lambda^X = \frac{1}{T\sqrt{\pi_k}}\sum_t (q_{tk} - \pi_k), \tag{2.6}$$

$$\bigtriangledown_{\mu_k} G_\lambda^X = \frac{1}{T\sqrt{\pi_k}}\sum_t q_{tk}\Sigma_k^{-\frac{1}{2}}(x_t - \mu_k), \tag{2.7}$$

$$\bigtriangledown_{\Sigma_k} G_\lambda^X = \frac{1}{T\sqrt{2\pi_k}}\sum_t q_{tk}(\Sigma_k^{-1}(x_t - \mu_k)^2 - 1). \tag{2.8}$$

Perronnin *et al.* [109] observe that derivatives w.r.t. mixture weights (α_k) do not add much information and can be safely discarded. Consequently, the final Fisher Vector is obtained as the concatenation of the whitened gradients for the mean and standard deviation, $\mathcal{G}^{\mathcal{X}}_{\mathcal{FK}} = [(\bigtriangledown_{\mu_k} G^X_\lambda) \ (\bigtriangledown_{\Sigma_k} G^X_\lambda)]$. Therefore the final dimension is $2KD$, where D is the dimension of local descriptor.

Power Normalization: It was noted in [109] that as the number of Gaussians increases, Fisher Vectors become more and more sparse. This is because descriptors x_t are assigned with a significant weight q_{tk} to each Gaussian. To unsparsify the Fisher Vector, a power normalization [61, 109] is applied on each element of the vector:

$$f(z) = sign(z)|z|^\alpha \tag{2.9}$$

where $0 \leq \alpha \leq 1$. This is followed by $L2$ normalization. A commonly used value for α is 0.5 [65, 109], though in some cases cross-validating it on train data can boost results as we observe in Sections 4.2.4 and 5.5.

Fisher vector representation has not only been used in image classification but also in image retrieval [65, 110] and recently for action recognition [104, 147]. For each of these tasks, Fisher vector has been shown as one of the best global representations.

2.4.2 VLAD: Vector of locally aggregated descriptors

VLAD was introduced by Jégou *et al.* [64] as a compact image representation for retrieval. It encodes the descriptor positions in each cluster by computing their differences from the centroid. The residuals are aggregated per cluster to obtain the corresponding sub-vectors. Finally, all the sub-vectors are concatenated and the resulting vector is $L2$ normalized. Given a codebook, $\{\mu_i, i = 1...K\}$ and a set of local descriptors $X = \{x_t, t = 1...T\}$, VLAD is computed as follows:

1. Assign to nearest visual word: $NN(x_t) = arg\ min_{\mu_i} ||x_t - \mu_i||$
2. Compute the sub-vector: $v_i = \sum_{x_t : NN(x_t) = \mu_i} x_t - \mu_i$
3. Concatenate v_i's ($i = 1...K$) to obtain the $D \times K$ dimensional VLAD, where D is the dimension of local descriptor.

The VLAD can be seen as a special case of the Fisher Vector with only first order moments and hard assignments of local descriptors. Due to its simplicity and hard-assignment, VLAD is faster to compute than FV. For image retrieval, it was found [64, 65] that in Fisher Vector, the gradients with respect to the variances also do not provide much information. So, being more efficient VLAD is a popular choice in image retrieval. In [59], Jégou *et al.* introduced power normalization for VLAD and since, there have been a few other extensions or variants [7, 27, 162]. Though originally proposed for image retrieval, this representation is general and can be used for classification also. It has been used for

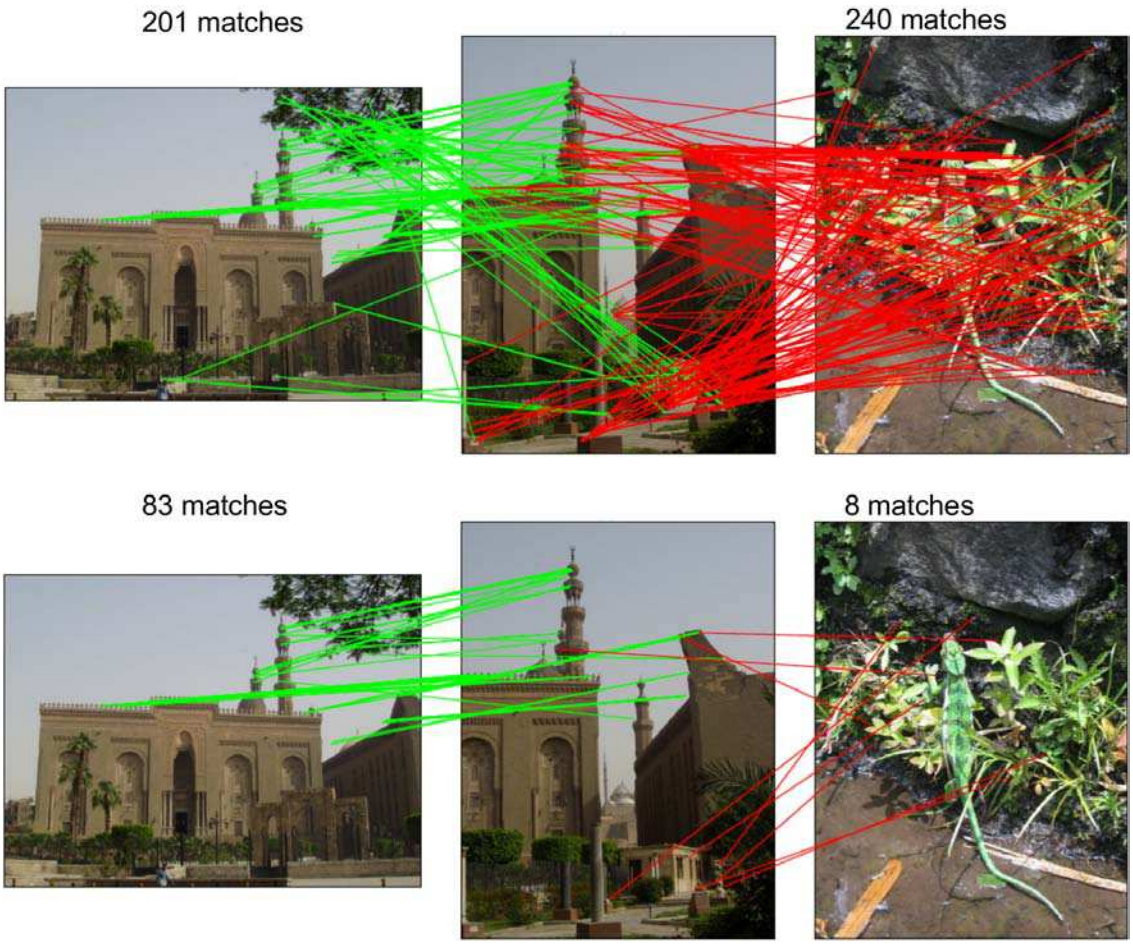


Figure 2.3: Top: BOW with $K = 20,000$ has many matches with the non-corresponding image. Bottom: Hamming Embedding with $K = 20,000$ has 10 times more matches for corresponding image. Image courtesy of [62].

video retrieval [117] and its extension VLAT for image classification [97]. We use it for the first time in action recognition in Section 5.5.

2.5 Voting based representations

2.5.1 Hamming Embedding

Hamming Embedding (HE) was introduced by Jégou *et al.* [60] as an extension of BOW. In this approach, the descriptor space is also partitioned by k-means, but in addition a binary code is also computed for each descriptor. This signature refines the descriptor representation based on the visual word, and therefore leads to an improved image representation. In BOW, the descriptors belonging to the same cluster are assumed to match. With HE, the matching is made more selective: Two descriptors x and y are assumed to match if they belong to the same cluster and if the Hamming distance between

Figure 2.4: Features assigned to the most bursty visual words highlighted. Courtesy of [61].

their binary signature is less than a predefined threshold. Figure 2.3 illustrates how the descriptor matching is drastically improved by limiting the quantization error in this manner. There are many more true matches and only few false matches with HE in comparison to BOW. We provide more details about HE, *i.e.*, the process of binary signature generation and computation of image matching score, in Section 3.2.2. Now we discuss in brief the burstiness phenomenon and how it is handled, in particular in conjunction with HE.

Visual burstiness handling: Jégou *et al.* [61] observed that when a visual word appears in an image, it is more likely that it will appear again. In other words, a given visual element appears more times than a statistically independent model, such as *tf-idf* would predict. Figure 2.4 illustrates this phenomenon: many detected features belong to the same visual word. This means every added feature to the same visual word adds progressively less information. By not taking this effect into account, bursty instances contribute equally to the match scores, which corrupts the similarity measure.

Three strategies were proposed to handle the burstiness. The first is to remove multiple matches, *i.e.*, only the best match is considered, in case of HE it is the one with minimum Hamming distance. The Second is to handle intra-image burstiness, if there are several descriptors in the query image assigned to the same visual word, their scores are penalized. Similarly to handle inter-image burstiness, the match scores of descriptors that vote for many images in the database are penalized.

2.5.2 Matching for classification

Matching approaches have not only been employed for image retrieval but also for image classification. Boiman *et al.* [14] proposed a NBNN (Naive-Bayes Nearest-Neighbor) classifier, which does not require any training. NBNN employs NN-distances in the space of the local image descriptor instead of in the space of images. It computes direct 'Image-to-Class' distances without descriptor quantization. Tuytelaars *et al.* [135] introduced a kernelized version of NBNN which allows learning the classifier in a discriminative setting. It also becomes easy to combine it with other kernels as they did

by combining with bag-of-features based kernels. Kim *et al.* [69] proposed a region-to-image matching scheme that involves matching features from segmented region of one image to another unsegmented image. The final matching score between two images is computed as a summation of match scores between all their corresponding points obtained from each region-to-image match. A graph based matching between images for classification is proposed by Duchenne *et al.* [33]. They model an image as a graph with a dense set of regions as nodes and the underlying grid structure of the image as edges. A fast approximate algorithm is presented to match these graphs associated with two images.

CHAPTER

THREE

Asymmetric Hamming Embedding

Large scale image search is still a very active domain. The task consists in finding in a large set of images the ones that best resemble the query image. Typical applications include finding searching images on web [154], location [111] or particular object [99] recognition, or copy detection [80].

Earlier approaches were based on global descriptors such as color histograms or GIST [102]. These are sufficient in some contexts [32], such as copy detection, where most of the illegal copies are very similar to the original image. However, global description suffer from well-known limitations, in particular they are not invariant to significant geometrical transformations such as cropping. That is why we focus here on the bag-of-words (BOW) framework [123] and its extension [61], where local descriptors are extracted from each image [85] and used to compare images.

The BOW representation of images was proved be very discriminant and efficient for image search on millions of images [60, 99]. Different strategies have been proposed to improve it. For instance, [99] improves the efficiency in two ways. Firstly, the assignment of local descriptors to the so-called visual words is much faster thanks to the use of a hierarchical vocabulary. Secondly, by considering large vocabularies (up to 1 million visual words), the size of the inverted lists used for indexing is significantly reduced. Accuracy is improved by a re-ranking stage performing spatial verification [85], and by query expansion [22], which exploits the interaction between the relevant database images.

Another way to improve accuracy consists in incorporating additional information on descriptors directly in the inverted file. This idea was first explored in [60], where a richer descriptor representation is obtained by Hamming Embedding (HE) and weak geometrical consistency [60]. HE, in particular, was shown successful in different contexts [154], and improved in [61, 62]. However, this technique has a drawback: each local descriptor is represented by relatively large signatures, typically ranging from 32 [154] to 64 bits [61].

In this chapter, we propose to improve HE in order to better exploit the information con-

veyed by the binary signature. This is done by exploiting the observation, first made in [31], that the query should not be approximated. We therefore adapt the voting method to better exploit the precise query location instead of the binarized query vector. This requires, in particular, two regularization steps used to adjust the dynamic of the local query distribution. This leads to an improvement over the reference symmetric HE scheme. As a complementary contribution, we evaluate how our approach trades accuracy against memory with smaller number of bits. The work presented in this chapter was published in [57].

The chapter is organized as follows. The datasets representing the application cases and the evaluation protocol are introduced in Section 3.1. Section 3.2 briefly describes the most related works: BOW and HE. Our asymmetric method is introduced in Section 3.3. Finally, experiments in Section 3.4 compare the performance of our asymmetrical method with the original HE, and provides a comparison with the state of the art on image search. It shows a significant improvement: we obtain a mean average precision of 70.4% on the Oxford5K Building dataset before spatial verification, *i.e.*, +4% compared with the best concurrent method.

3.1 Evaluation datasets

This section introduces the datasets used in our experiments, as well as the measures of accuracy used to evaluate the different methods. These datasets reflect two application use-cases for which our method is relevant, namely place and object recognition. They are widely used to evaluate image search systems.

Oxford5K and Paris These two datasets of famous building in Oxford and Paris contain 5,062 and 6,412 images, respectively. We use Paris as an independent learning set to estimate the parameters used by our method. The quality is measured on Oxford5K by mean average precision (mAP), as defined in [111]: for each query image we obtain a precision/recall curve, and compute its average precision (the area under the curve). The mAP is then the mean for a set of queries.

INRIA Holidays This dataset contains 1491 images of personal holiday photos, partitioned into 500 groups, each of which represents a distinct scene, location or object. The first image of each group is the query image and the correct retrieval results are the other images of the group. Again, the search quality is measured by the mAP, see [111, 60] for details. A set of images from Flickr is used for learning the vocabulary, as done in [60].

Flickr1M In order to evaluate the behavior of our method on a large scale, we have used a set of up to one million images. More precisely, we have used the descriptors

shared online[1] by Jégou et al., which were downloaded from Flickr and described by the same local descriptor generation procedure as the one used for Holidays. This dataset is therefore merged with Holidays and the mAP is measured using the Holidays ground-truth, as in [62].

The recall@R measure is used for this large scale experiment. It measures, at a particular rank R, the ratio of relevant images ranked in top R positions. [32] states that it is a good measure to evaluate the filtering capability of an image search system, in particular if the large scale image search is followed by a precise spatial geometrical stage, as classically done in the literature.

3.2 Related work

3.2.1 Bag-of-features representation

The BOW framework [123] is based on local invariant descriptors [93, 85] extracted from covariant regions of interest [93]. It matches small parts of images and can cope with many transformations, such as scaling, local changes in illumination and cropping.

The feature extraction is performed in two steps: detecting regions of interest with the Hessian-Affine detector [93], and computing SIFT descriptors for these regions [85]. We have used the features provided by the authors for all the aforementioned datasets. For Holidays and Flickr1M, the features are rotation-invariant, while for Oxford5K and Paris they are not because the images are all in up-right orientation.

The fingerprint of an image is obtained by quantizing the local descriptors using a nearest-neighbor quantizer, produced the so-called *visual words* (VW). The image is represented by the histogram of VW occurrences normalized with the L2 norm. A *tf-idf* weighting scheme is applied [123] to the k components of the resulting vector. The similarity measure between two BOW vectors is, most often, cosine similarity. The visual vocabulary of the quantizer is produced using k-means. It contains a large number k of visual words. In this work, we set $k = 20,000$ for the sake of consistency with [60] and [61]. Therefore, the fingerprint histograms are sparse, making queries in the inverted file efficient.

3.2.2 Hamming Embedding

The Hamming Embedding method of [60] is a state of the art method extension of BOF, where a better representation of the images is obtained by adding a short signature that refines the representation of each local descriptor. In this approach, a descriptor x is

[1] `http://lear.inrialpes.fr/people/jegou/data.php`

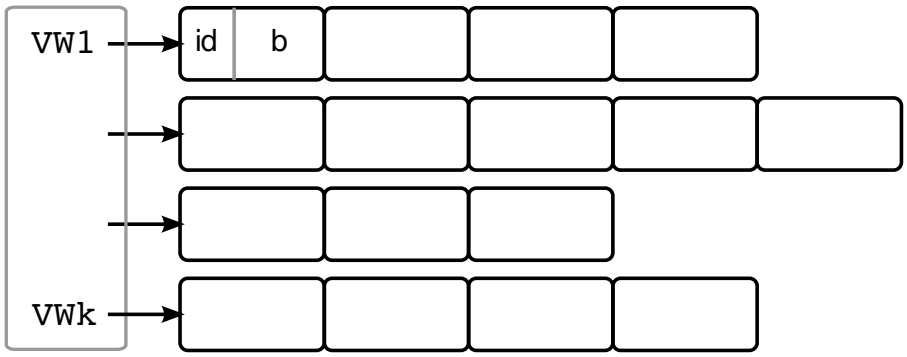


Figure 3.1: Overview of the HE indexing structure: it is a modified inverted file. Each inverted list is associated with a visual word. Each local descriptor is stored in a cell containing both the image identifier and a binary signature (from 4 to 64 bits in our work). This indexing structure is also used in our AHE method: only the score similarity is modified.

represented by a tuple $(q(x), b(x))$, where $q(x)$ is the visual word and $b(.)$ is a binary signature of length m computed from the descriptors to refine the information provided by $q(x)$. Two descriptors are assumed to match if

$$\begin{cases} q(x) = q(y) \\ h\left(b(x), b(y)\right) = \sum_{i=1..m} |b_i(x) - b_i(y)| \leq h_t \end{cases}, \tag{3.1}$$

where $h(b, b')$ is the Hamming distance between binary vectors $b = [b_1, .., b_m]$ and $b' = [b'_1, .., b'_m]$, and h_t is a fixed threshold. The image score is obtained as the sum [60] or weighted sum [62] of the distances of the matches satisfying (3.1), then normalized as in BOF.

Similar to that in BOF, the method uses an inverted file structure, which is modified to incorporate the binary signature, as illustrated by Figure 3.1. The matches associated with the query local descriptors are retrieved from the structure and used as follows.

- Find the nearest centroid of the query descriptor x, producing quantized indexes $q(x)$, *i.e.*, the visual word (VW). The entries of the inverted list associated with $q(x)$ are visited.
- A given descriptor x is projected by a rotation matrix to an m-dimensional feature space: $\mathbf{Q} \times x = b^* = [b^*_1(x), \ldots, b^*_m(x)]^\top$. To generate $\mathbf{Q}$, a matrix of Gaussian values is randomly drawn and QR factorization is applied on it. The first m rows of the orthogonal matrix obtained by this decomposition form the matrix $\mathbf{Q}$.
- The binary signature is obtained by comparing each component $b^*_i, i = 1..m$ with a threshold $\tau_{q(x),i}$. This amounts to selecting $b_i = 1$ if $b^*_i - \tau_{q(x),i} > 0$, else $b_i = 0$. The thresholds $\tau_{c,i}$ are the median values of b^*_i measured on an independent learning set for all VWs c and all bit components i.
- Only the database descriptors satisfying Equation 3.1 make a vote for the corresponding image, *i.e.*, they vote only if their Hamming distance is below a pre-defined

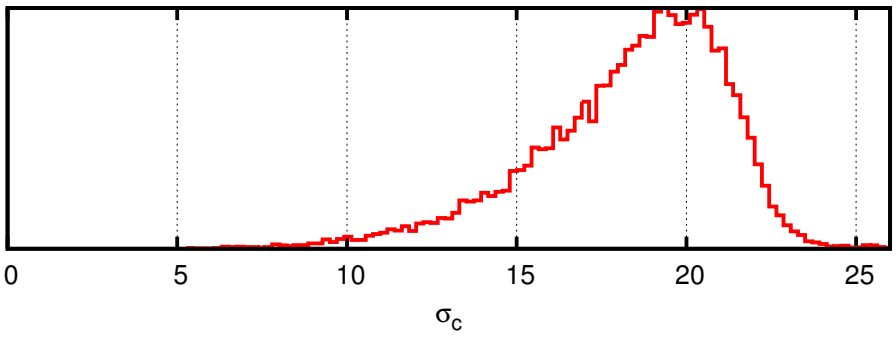


Figure 3.2: Empirical probability distribution function of σ_c measured for all visual words. The large variation of density across cells shows the need for a per-cell variance regularization.

threshold h_t. The vote's score is 1 in [60]. Scoring with a function of the distance improves the results [61]. We therefore adopt this choice.

- All images scores are finally normalized.

Additionally, we consider two techniques [61] that improve the results. First, multiple assignment (MA) reduces the number of matches that are missed due to incorrect quantization indexes. Second, the so-called *burstiness* (denoted by "burst") handling method regularizes the score associated with each match, to compensate the bursty statistics of regular patterns in images.

3.3 Asymmetric Hamming Embedding

This section introduces our approach. It is inspired by the work of [31], where the use of asymmetric distances was investigated in the context of Locality Sensitive hashing. This method has to be significantly adapted in our context. Using the distance to hyperplanes may suffice for pure nearest neighbor search, where the objective is the find the Euclidean k-nearest neighbor of a given query [31]. However, in our case, this is not sufficient, because computed distances are used as match quality measurements. Our goal is therefore to provide a soft weighting strategy that better exploits the confidence measures of all matches to produce the aggregated image scores.

Intra-cell distance regularization We first adapt the local distances so that they become more comparable for different visual words. This is done, in our AHE scheme, by computing the standard deviation σ_c of the distance $b_i^* - \tau_{c,i}$ for each visual word c, *i.e.*, the distance from the separating hyperplanes. This estimation is carried out using a large set of vectors from an independent learning set. We used 50M Flickr descriptors for Holidays and all the descriptors from Paris for Oxford5K. The standard deviation is either computed component-wise (one per bit dimension) of for the whole cell. In our case

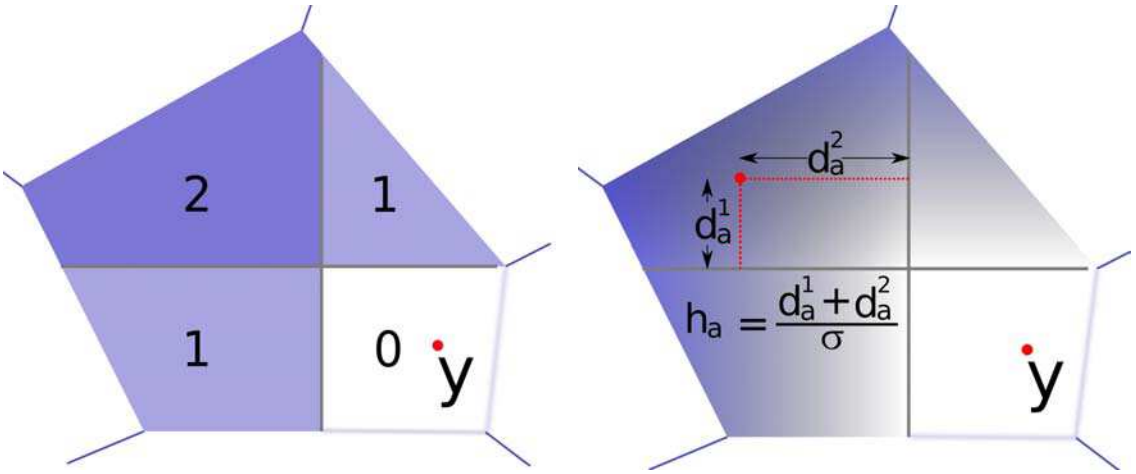


Figure 3.3: Illustration of HE and AHE for binary signatures of length 2. In the symmetric case, only three distances are possible (0, 1 or 2) between query and database descriptor y. AHE gives a continuous distance (reflected by the intensity of blue).

we chose the simple choice of estimating a single parameter per cell used for all bits (isotropic assumption).

As observed in Figure 3.2, the standard deviations significantly vary from one cell to another. It is then worth obtaining more comparable values when using distances as quality measurements.

Distance to hyperplanes In the symmetric version, the query projected by $\mathbf{Q}$ is binarized and compared with the database descriptors. We instead compute the distance between the projected query ($b^*(x) = \mathbf{Q} \times x$) and the database binary vectors that lie in the same cell (associated with $q(x)$). The "distance" between the i^{th} component of $b^*(x)$ and the binary vector $b(y)$ is given by

$$d_a^i\left(b_i^*(x), b_i(y)\right) = |b_i^*(x) - \tau_{q(x),i}| \times |b_i(x) - b_i(y)|\,. \tag{3.2}$$

This quantity is zero when x is on the same side of the hyperplane associated with the i^{th} component. The distances are added for all the m components to get an asymmetric "distance" between a query descriptor x and a database descriptor y, defined as

$$h_a\left(b^*(x), b(y)\right) = \frac{1}{\sigma_{q(x)}} \times \sum_{i=1..m} d_a^i\left(b_i^*(x), b_i(y)\right). \tag{3.3}$$

The descriptors are assumed to match if $h_a\left(b^*(x), b(y)\right) \leq h_t$, as for the symmetric version. For a given query x, the values $|b_i^*(x) - \tau_{q(x),i}|$ are precomputed before being compared to the database vectors. The similarity is penalized according to the distance from the hyperplane in the embedded Hamming space, providing improved measurements, as illustrated by Figure 3.3. In the symmetric case, it does not matter how far b_i^* lies from $\tau_{q(x),i}$. In contrast, the distance is a continuous function in our method.

Score weighting Similar to what is done in [61] for the symmetric case, the distance obtained by Equation 3.3 is used to weight the voting score. In [61] weights are obtained as a Gaussian function of Hamming distance. Here the weights are simply the difference $h_t - h_a\left(b^*(x), b(y)\right)$ between the threshold and the normalized "distance". We also apply the burstiness regularization method of [61]. As we will show in Section 3.4, its impact is very important in our case because the aforementioned variance regularization does not sufficiently balance the amount of score received by the different *query* vectors, leading individual descriptors from the query image to have a very different impact in the final score. The burstiness regularization effectively addresses this issue.

3.4 Experiments

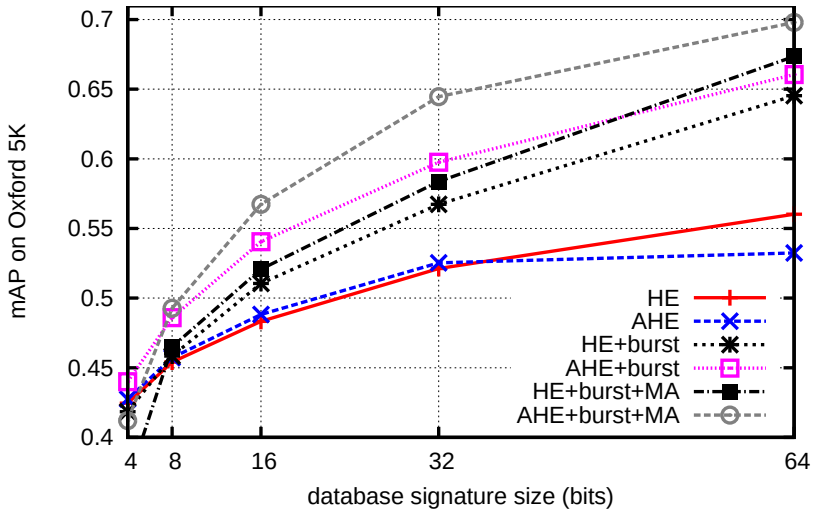
Search quality: HE *vs* AHE Figure 3.4 evaluates our AHE method introduced in Section 3.3 against the original HE one, for varying numbers of bits. For both HE and AHE, we report the results obtained with the best threshold. As shown by Figure 3.5, the performance is stable around this best value.

Using the asymmetric version significantly improves the results, especially for short signatures. As stated in Section 3.3, the burstiness regularization of [61] is important in our case: without it AHE only achieves a slight improvement for short signatures.

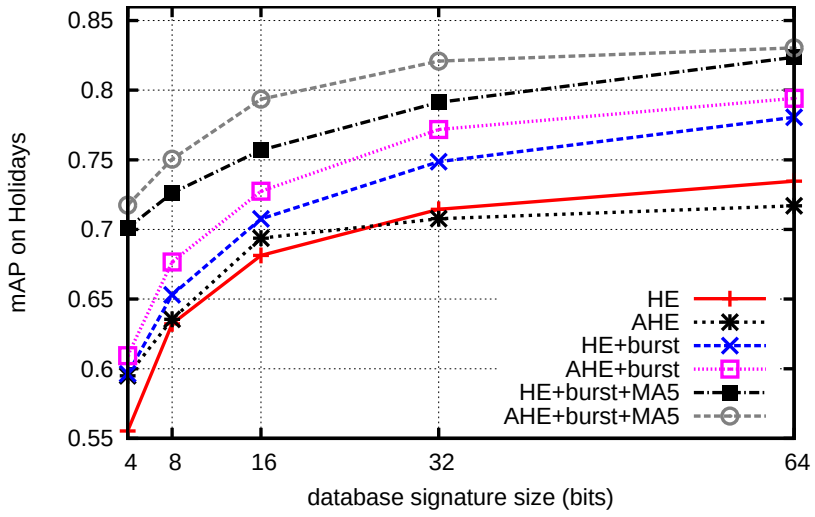
Observe the important trade-off between the search accuracy and the signature length: using more bits clearly helps. However it is important to keep this signature short so that database images remain indexed in memory. Multiple Assignment helps in both cases, at the cost of increased query time. As advocated by a previous work [61], we perform MA on the query side only, with 5 or 10 nearest visual words, denoted by MA5 and MA10 respectively.

Comparison with the state of the art

Table 3.1 compares our results with, to our knowledge, the best ones reported in the literature. We also report the thresholds (h_t) used to obtain the results. Our approach clearly outperforms the state of the art on both Holidays and Oxford5K. Interestingly, for symmetric HE, our results (*in italics*) on Oxford5K are better than those reported in [61] with a geometry check. This is because the rotation invariant features used in [112] are more discriminative for these datasets (only images in upright orientation) without spatial verification. We only include in our comparison the results reported with learning done on an independent dataset itself. Some papers show that learning on the test set itself improves the results, as to be expected. But as stated in [62] such results do not properly reflect the expected accuracy when using the system on a large scale. Some visual examples of results are shown for Oxford5K in Figure 3.7 and for Holidays in Figure 3.8.



(a) Oxford building dataset



(b) INRIA Holidays dataset

Figure 3.4: HE vs AHE: Trade-off between memory usage (per descriptor) and search quality.

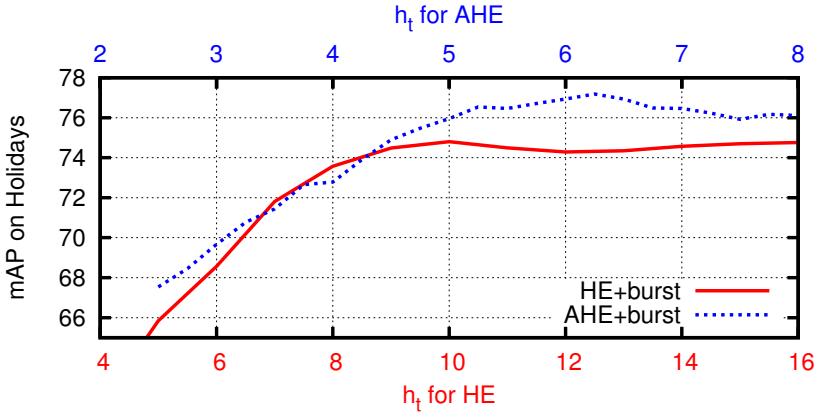


Figure 3.5: Impact of the threshold on accuracy ($m = 32$ bits): Note that the ranges for h_t differ for HE (Hamming distance) and AHE (derived from normalized distance to hyperplanes).

	Oxford5K	h_t	Holidays	h_t
BOF [112]	40.3	-	-	-
BOF+soft MA [112]	49.3	-	-	-
HE+MA5 [62]	61.5	-	77.5	-
HE+burst [61]	*64.5*	31	*78.0*	21
HE+burst+MA5 [61]	*67.4*	22	*82.4*	23
AHE+burst	66.0	18.5	79.4	15.5
AHE+burst+MA5	69.8	17	83.0	17
AHE+burst+MA10	**70.4**	16	**83.4**	16.5

Table 3.1: Comparison with the state of the art: For [61], we report *in italics* the results obtained by our implementation, [61] reports inferior results with different descriptors and with geometrical information only. Results are shown for 64 bits.

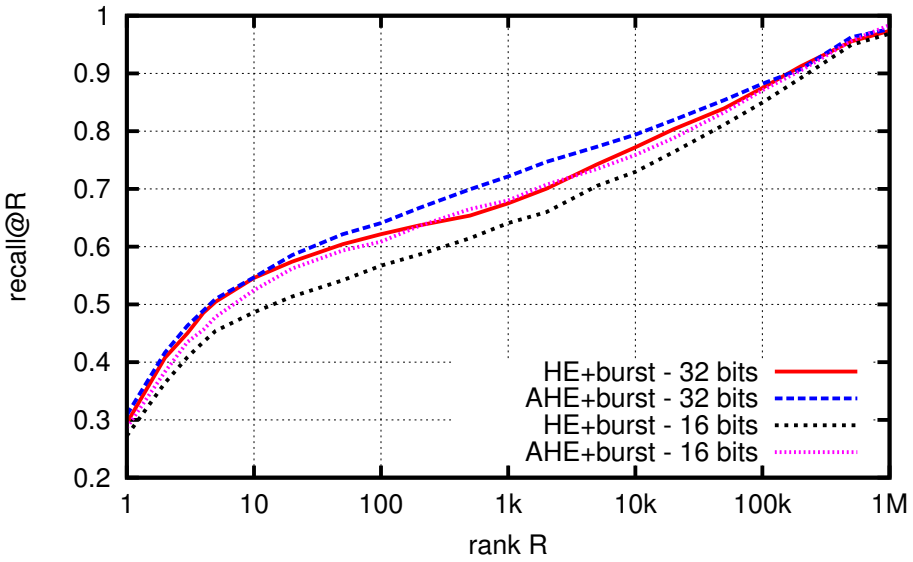


Figure 3.6: Search quality on a large scale (1 million images): Holidays merged with Flickr1M.

Large scale experiments As shown in Figure 3.6, the filtering capability of AHE is better than HE: the recall@R measure is almost as good for AHE with 16 bits as HE with 32 bits. Equivalently, the performance is much better for a given memory usage.

The complexity of the method is increased compared to the original symmetric method. In both HE and AHE, the vector has to be projected. The main difference appears in the similarity computation, which is a simple `XOR` operation following by a bit count in HE, while we need to add pre-computed floating point values to get h_a in Equation 3.3. As a result, on 1 million images the search speed is roughly 1.7 times slower in the asymmetric case, on average, compared to [61]: on average searching in one million images (using 64 bits) with AHE it takes 2.9s on one processor core, against 1.7s for HE.

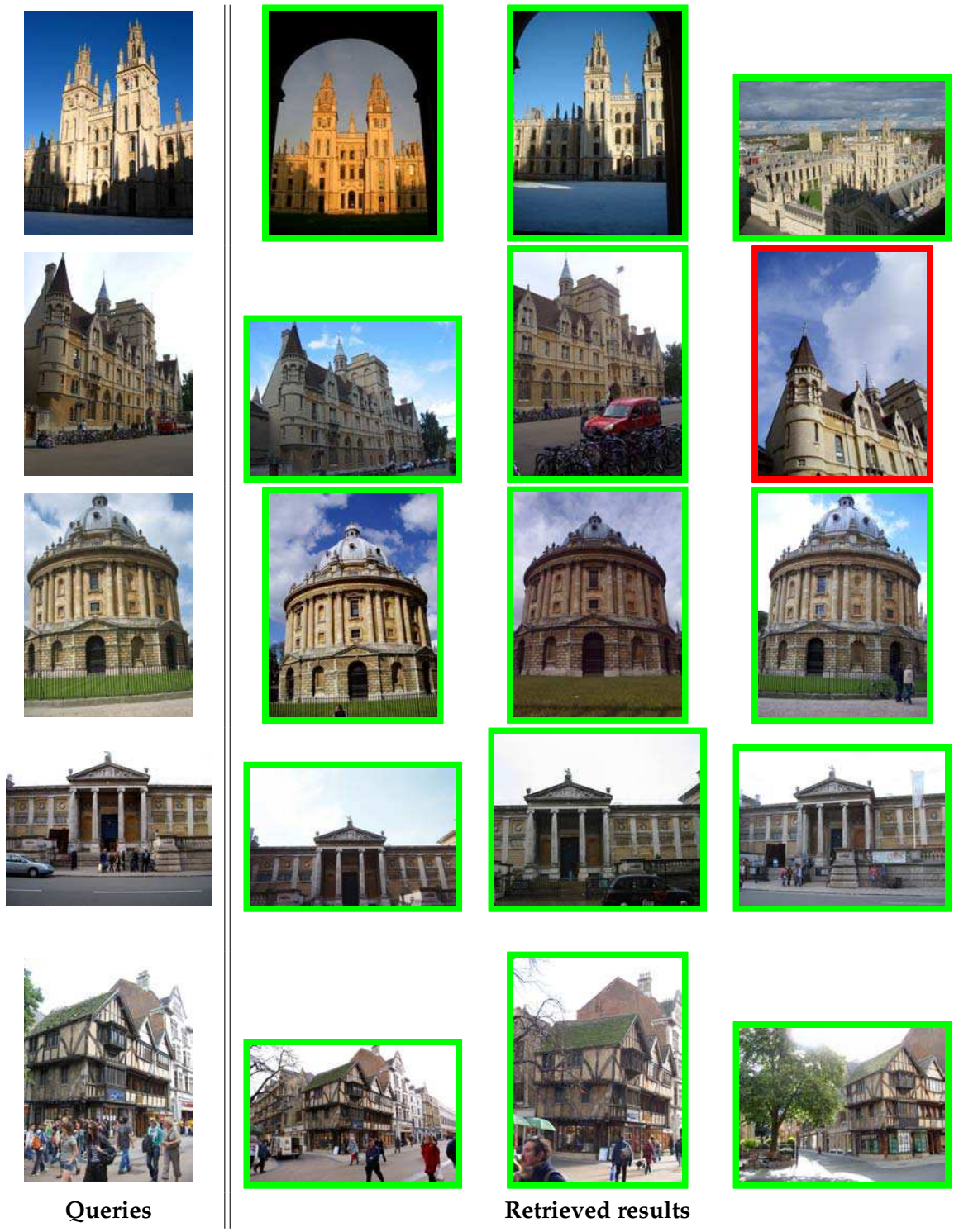


Figure 3.7: Results on Oxford5K dataset: Each row shows a query and the top three retrieved images from the dataset. True-positives and false-positives are shown with green and red borders respectively.

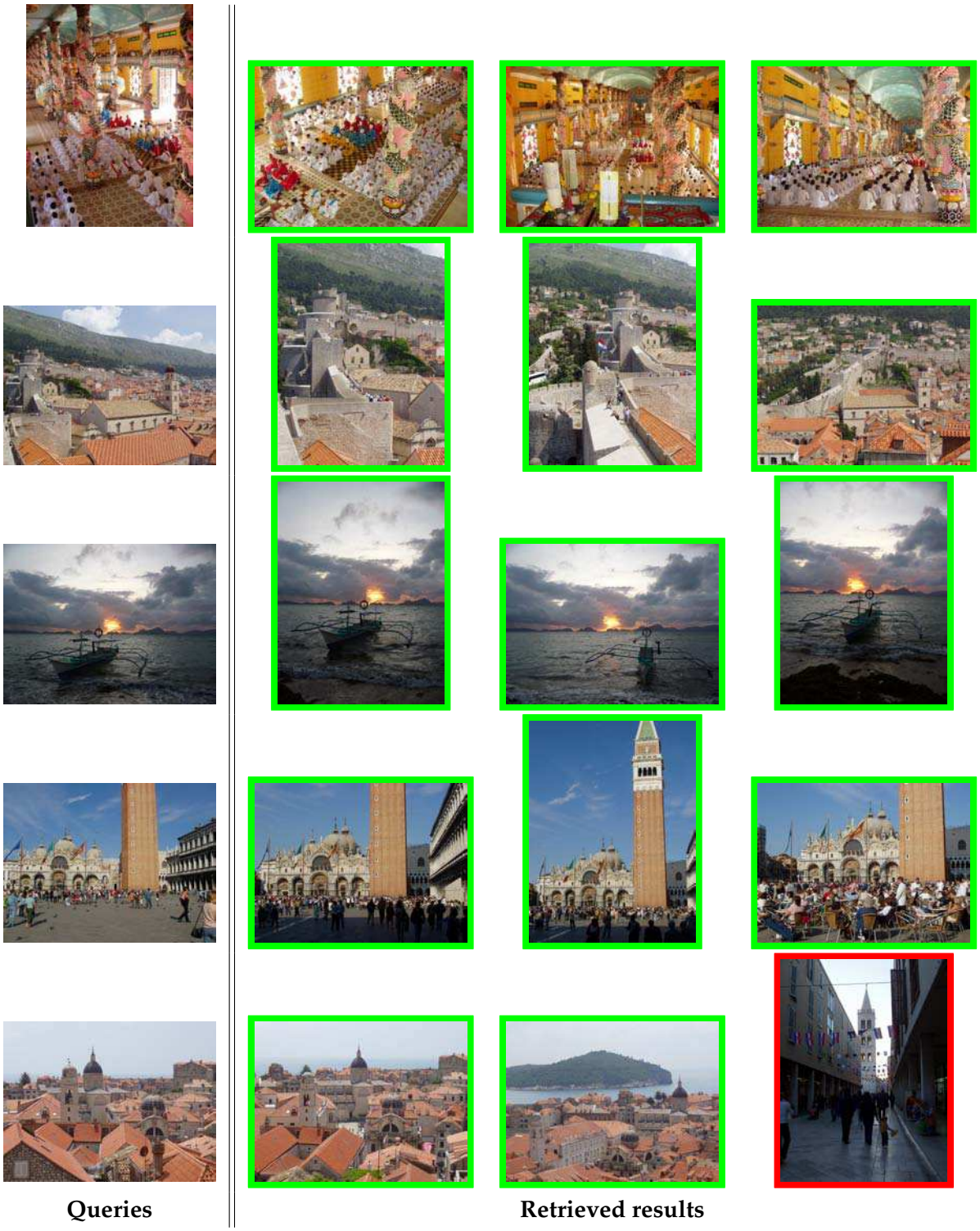


Figure 3.8: Results on INRIA Holidays dataset: Each row shows a query and the top three retrieved images from the dataset. True-positives and false-positives are shown with green and red borders respectively.

3.5 Conclusion

This chapter shows that a vector-to-binary code comparison significantly improves the state-of-the-art Hamming Embedding technique by reducing the approximation made on the query. This is done by exploiting the vector-to-hyperplane distances. The improvement is obtained at no additional cost in terms of memory. As a result, we improve the best results ever reported on two popular image search benchmarks before geometrical verification.

CHAPTER

FOUR

Hamming Embedding similarity-based image classification

Image classification is a challenging problem and the key technology for many existing and potential mining applications. It has attracted a large interest from the Multimedia and Computer Vision communities in the past few decades, due to the ever increasing digital image data generated around the world. It is defined as the task of assigning one or multiple labels corresponding to the presence of a category in the image.

Recently, machine learning tools have been widely used to classify images into semantic categories. Local features combined with the bag-of-visual-words representation of images demonstrate decent performance on classification tasks [161]. The idea is to characterize an image with the number of occurrences of each visual word [123]. However, it is generally admitted that this setup is sub-optimal, as the discriminative power of the local descriptors is considerably reduced due to the coarse quantization [14] operated by the use of a pre-defined visual vocabulary. To address this problem, several encodings have been proposed such as locality-constrained linear [149], super vector [64, 163], kernel codebook [141], and the Fisher Kernel [107, 109]. These coding schemes are compared by Chatfield et al. [21] on the popular PASCAL'07 and Caltech-101 benchmarks. Considering dense SIFT sampling, which are shown to outperform interest points for classification, they use a linear classifier for better efficiency and confirm that these new coding schemes indeed achieve better classification accuracy than the spatial histogram of visual-words baseline. The superiority of the improved Fisher Kernel [109] is evidenced among all these schemes.

Another way to limit the quantization error introduced by the use of visual words instead of full descriptors consists in adopting a matching approach [14, 135]. These schemes require the use of full raw descriptors, which is not feasible when considering large learning sets such as those considered in large image databases like ImageNet [28]. Moreover, to our knowledge these methods have not been shown to exhibit a classification accuracy as good as those reported with the aforementioned new coding schemes.

Besides, in the context of image search, some solutions have been proposed to dramati-

cally improve the matching quality while keeping a decent efficiency and memory usage. In particular, improved accuracy is achieved by incorporating additional information, jointly with the descriptors, directly in the inverted file. This idea was first explored by Jégou *et al.* [60] for image search, where a richer descriptor representation is obtained by using binary codes in addition to visual words and weak geometrical consistency. In our work, we will mainly focus on the interest of the complementary information provided by the binary vectors, using the so-called Hamming Embedding method [62, 61]. We push this idea further in Chapter 3 and show that a vector-to-binary code comparison improves the Hamming Embedding baseline by limiting the approximation made on the query, leading to state-of-the-art results for the image search problem.

In this chapter, in the spirit of recent works that have shown the interest of patch-based techniques for classification, we propose to adopt the state-of-the-art Hamming Embedding method for category-level recognition. This produces a representation which is more efficient and compact in memory than the solutions based on exact patch matching. However, the original Hamming Embedding technique can not be used off-the-shelf, since the similarity output by this technique is not a Mercer Kernel. A naive option would be to adopt instead a k-nearest neighbor classifier, but from our preliminary experiments the resulting classification accuracy is then low. To address this problem, we adopt a kernelization technique on top of our matching-based solution, which enables the use of support vector machines and thereby allows good generalization properties even when using a linear classifier. As a result, Hamming Embedding classification is efficient in both training and testing stages, and provides better performance and efficiency than the recently proposed concurrent matching-based classification techniques [14, 135].

Last but not least, the proposed approach is shown to outperform the most recent coding schemes benchmarked in [21]. The only noticeable exception is the latest improvement of the Fisher Kernel [109], which still remains competitive. Beside, we show that the combination of Hamming Embedding similarity with Fisher Kernel is complementary and achieves the current state-of-the-art performance. Most importantly and as noticed in [135], the high flexibility offered by a matching-based framework is likely to pave the way to several extensions. The work presented in this chapter was published in [55].

The rest of the chapter is organized as follows: Section 2 describes the most related works. Section 3 presents our system architecture. It includes the feature extraction procedure, where an improved SIFT descriptor is introduced, and the Hamming Embedding similarity-based representation. Section 4 reports the experimental results conducted on the PASCAL VOC 2007 and Caltech-256 collections and compare them to the main state-of-the-art methods discussed in Section 2. Section 5 concludes the chapter.

4.1 Related work

The bag of visual-words (BOW) is one of the most popular image representations, due to its conceptual simplicity, computational efficiency and discriminative power stemming for the use of local image information. It represents each local feature with the closest visual word and counts the occurrence frequencies in the image. The length of the histogram is given by the number of visual words of a codebook dictionary. Van Gemert *et al.* [141] introduced an uncertainty model based on kernel density estimation to smooth the hard assignment of image features to codewords. However, BOW discards the spatial order of local descriptors, which severely limits the descriptive power of the image representation. To take into account the rough geometry of a scene, the spatial pyramid matching (SPM) proposed by Lazebnik *et al.* [81] divides the image into blocks and concatenates all the histograms to form a vector descriptor which incorporates the spatial layout of the visual word. The BOW and this SPM extension are generally used in conjunction with non-linear classifiers. In this case, the computational complexity is $O(N^3)$ and the memory complexity is $O(N^2)$ in the training phase, where N is the size of the training dataset. This complexity limits the scalability of BOW- and SPM-based non-linear SVM methods.

In order to limit the quantization error, Yang *et al.* [158] propose a linear spatial pyramid matching method based on sparse coding (ScSPM). A *max* pooling spatial pooling replaces the average pooling method for improved robustness to local spatial translations. A very successful method is the improved Fisher Kernel (FK) proposed by Perronnin *et al.* [109] in the context of image categorization. The idea of FK [53, 107] is to characterize an image with the gradient vector of the parameters associated with a pre-defined generative probability model (a gaussian mixture in [107]). This representation is subsequently fed to a linear discriminative classifier and can be used jointly with other techniques such as the SPM representation, power-law [109] and L_2 normalizations. The FK boosts the classification accuracy, at the cost of a high descriptor dimensionality, which is two orders of magnitude larger than BOW for the same vocabulary size. However, the FK is classified using a linear SVM, which counter-balance the higher cost of the non-linear classifiers involved in BOW-based classification. Other improvements are achieved by combining different types of local descriptors or by integrating objects localization task [50].

This chapter follows another line of research on building a kernelized efficient matching system. Various similarity matching methods are proposed in the literature. Some of these use feature correspondences to construct an image comparison kernel which is compatible with SVM-based classification [20]. Pyramid match kernel [48] represents a bag of features as a distribution of prototypes of points of interest, whereas Gosselin *et al.* [47] work with regions. Bo and Sminchisescu [13] propose an effective kernel computation through low-dimensional projection. Duchenne *et al.* [33] extend the region-to-

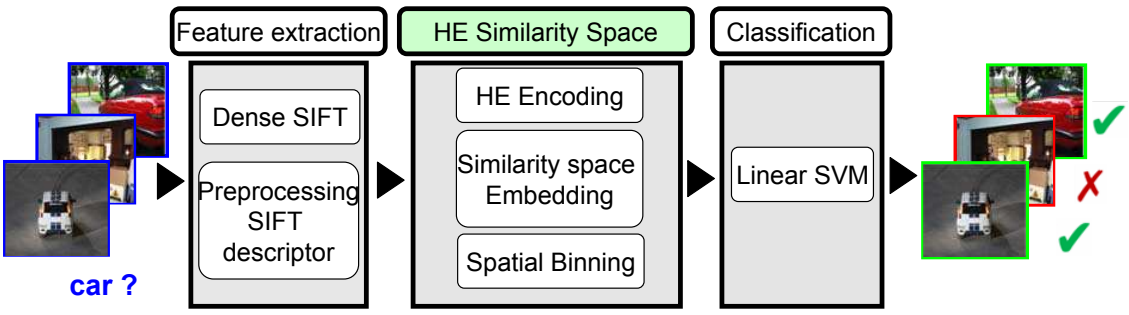


Figure 4.1: Proposed image classification system architecture.

image matching method of Kim *et al.* [69], and formulate image graph matching as an energy optimization problem, whose nodes and edges represent the regions associated with a coarse image grid and their adjacency relationships.

A very simple matching-based method is the one proposed by Boiman *et al.* [14], who use a Nearest Neighbor (NN) as a nonparametric model, with does not require any training phase. Two approaches are considered: NN Images-to-Images and NN Images-to-Classes. For the first approach, each test image is compared to all known images and the class of the closest image is chosen and assigned to queried image. The second approach pools all descriptors of all the images belonging to each class to form a single representation of that class. A given image is then compared to all the classes. NN Images-to-Classes achieves good results on standard benchmarking datasets. Tuytelaars *et al.* [135] exploited the kernel complementarity by combining NN and BOW. However, as shown in our experimental section, our matching-based method is the first to report competitive results against the best encoding method, namely the FK.

4.2 Proposed approach

Figure 4.1 gives an overview of our method. We first extract local features on a dense grid. For this purpose, we propose an improved variant of the SIFT [85] descriptor. It better takes into account the contrast information than the original one. These descriptors are *individually* encoded using the Hamming Embedding technique [62], which represents each descriptor by a visual word and a short binary signature. This binary vector is shown to integrate some residual information about the class which is not captured by the visual word. At this stage, the similarity between two images is done by computing the score produced by HE. More precisely, we used the extended HE method [61] by Jégou et al., which integrates a regularization technique to address the visual burstiness phenomenon encountered in images. The images are then described in a *similarity space*, which amounts to constructing a vector whose components correspond to a similarity to

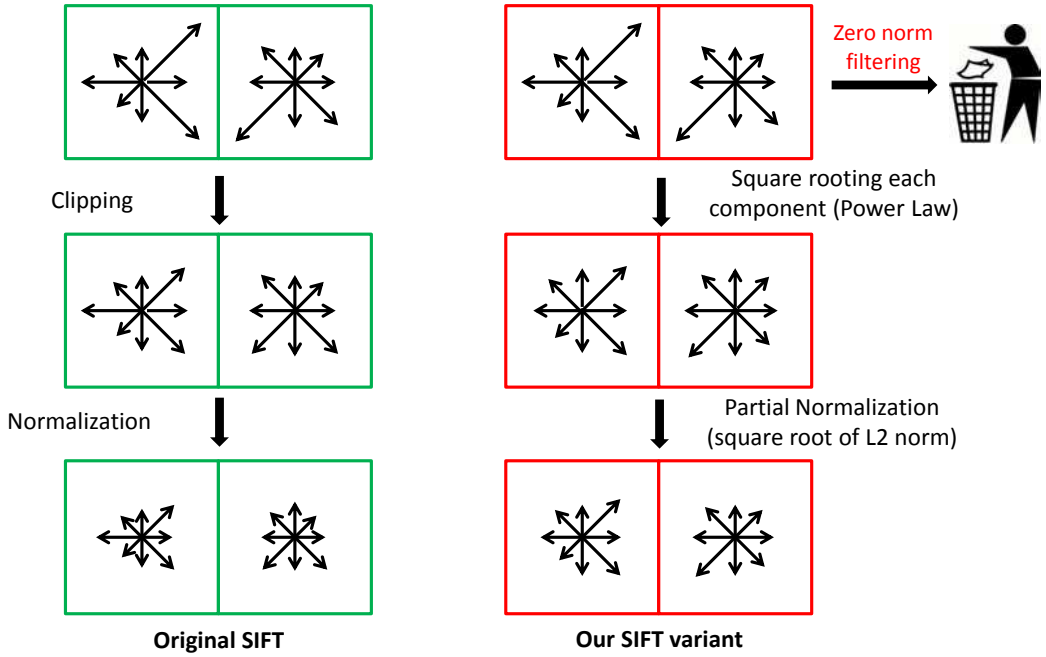


Figure 4.2: Proposed variant of SIFT

a fixed set of training images. A linear SVM classifier is then learned in this similarity space.

4.2.1 Feature extraction

We extract SIFT [85] descriptors from a dense grid. More precisely, we adopt the same grid parameters as used in [21], *i.e.*, a spatial stride of 3 pixels with multiple resolutions. These extracted patches are described using a local descriptor derived from the original SIFT descriptor [85]. The proposed variant of SIFT aims at addressing the following issues:

- A strong gradient, such as generated by a boundary, gives an overwhelming importance to a few components in the SIFT descriptor. Lowe proposes a solution [85] to address this problem by clipping the components whose value is larger than 20% of the whole energy. However, this solution is not satisfactory since it does not correct the components which magnitude is lower than this threshold.
- The SIFT descriptor are L2-normalized[1], in order to ensure invariance to intensity changes. However, this solution completely discards the absolute value of the gradient, which is a meaningful information.
- Dense patch sampling produces many uniform patches which are not very infor-

[1]In typical implementations, they are finally multiplied by a constant such that the components lie in the range [0..255], in order to encode each component with 1 byte.

mative. Worst, uniform patches have a low signal to noise ratio. Consequently, the normalization that is performed to achieve intensity-invariance magnifies the noise.

To address these issues, the SIFT generation procedure is modified as follows. Starting with the SIFT descriptor before clipping and normalization,

1. The descriptors with zero norm are filtered out, which amounts to removing the uniform patches.
2. Instead of the clipping procedure, each component is square rooted. This power-law component-wise regularization is similar to the one performed in the improved Fisher Kernel [109], but here applied directly on the local descriptor.
3. Finally, instead of using the L2 normalization, the final vector is normalized by the square root of the L2 norm of the transformed descriptor. This gives a better trade-off between full invariance to intensity change and keeping the information about the absolute intensity measure.

The above explained SIFT variant is compared with SIFT in Figure 4.2. We have evaluated the interest of this SIFT variant on the PASCAL VOC 2007 classification benchmark. For this we use our proposed approach described in Section 4.2.4 as well as the improved FK method [109]. We observe gain of around 2% of mAP when this SIFT variant is used with our approach for classification. Fisher Kernel with this variant achieves an mAP of 59.8% and 62.2% without and with spatial grid, respectively. These results are 0.5 to 1.5% better than the regular SIFT descriptor in the same setup.

4.2.2 Hamming Embedding

The Hamming Embedding method of [60] is a state of the art method for image retrieval. It provides an accurate way of computing the similarity between two images based on the distance between their local descriptors. It can be seen as an extension of BOW, where a better representation of the images is obtained by adding, to the visual word, a short binary signature that refines the representation of each local descriptor.

To generate the binary signature, each descriptor is first projected onto a m-dimensional space by a fixed random rotation matrix. Each projected component is compared with a median value learned, in an unsupervised way, on an independent dataset. This comparison produces either a **0** or a **1** per component, producing a bit-vector of length m. This binary signature gives a better localization of the local descriptor in the Voronoi cell associated with the visual word. Figure 4.3 illustrates this method for a 2-dimensional feature space ($m = 2$). Each red dot shows a descriptor. The cluster centers (visual words) are represented by the blue dots. For a given cell, the hyperplanes or axes (shown in dashed lines) represent the decision boundaries associated with the comparison of the projected

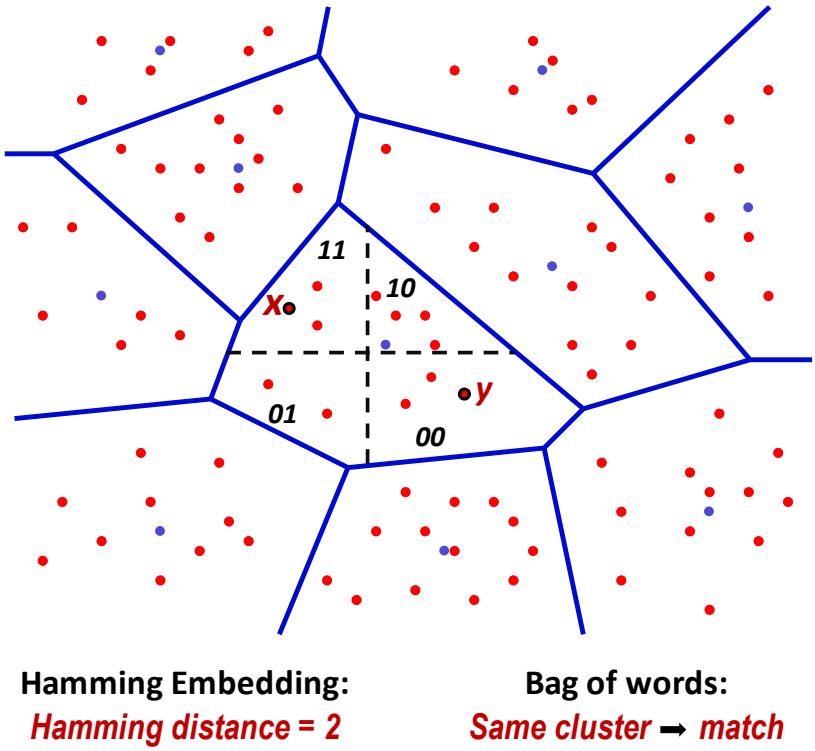


Figure 4.3: Hamming Embedding matching: In case of bag of words, being in the same cluster descriptors x and y match. While with HE matching depends on the relative location of the descriptors.

components with the median values. As a result, the cell is partitioned into 2^m sub-cells, each of which being associated with a given binary signature.

Two descriptors assigned to the same visual word are compared with the Hamming distance between their binary signatures. They are said to be matched only if the distance is less than a fixed threshold h_t. This provides a more accurate comparison between descriptors than in the BOW model, where the descriptors are assumed to match if they are assigned to the same visual word. In the example of Figure 4.3, the descriptors **x** and **y** belong to the same cluster, but have binary signatures **00** and **11**, respectively, which means that they are not similar.

Each successful matching pair votes, which increases the similarity score by a quantity that depends on the Hamming distance between the binary vectors. The final image similarity score is computed as the sum of voting scores and then normalized as in BOW. For the sake of efficiency, the method uses a modified inverted file structure which incorporates the binary signature. As in the original work of Jegou et al. [60] we use $m = 64$, and consider Hamming thresholds between $h_t = 20$ and $h_t = 24$.

Score weighting As mentioned above, the Hamming distance between two descriptors is used to weight the voting score. This was first done by considering a Gaussian func-

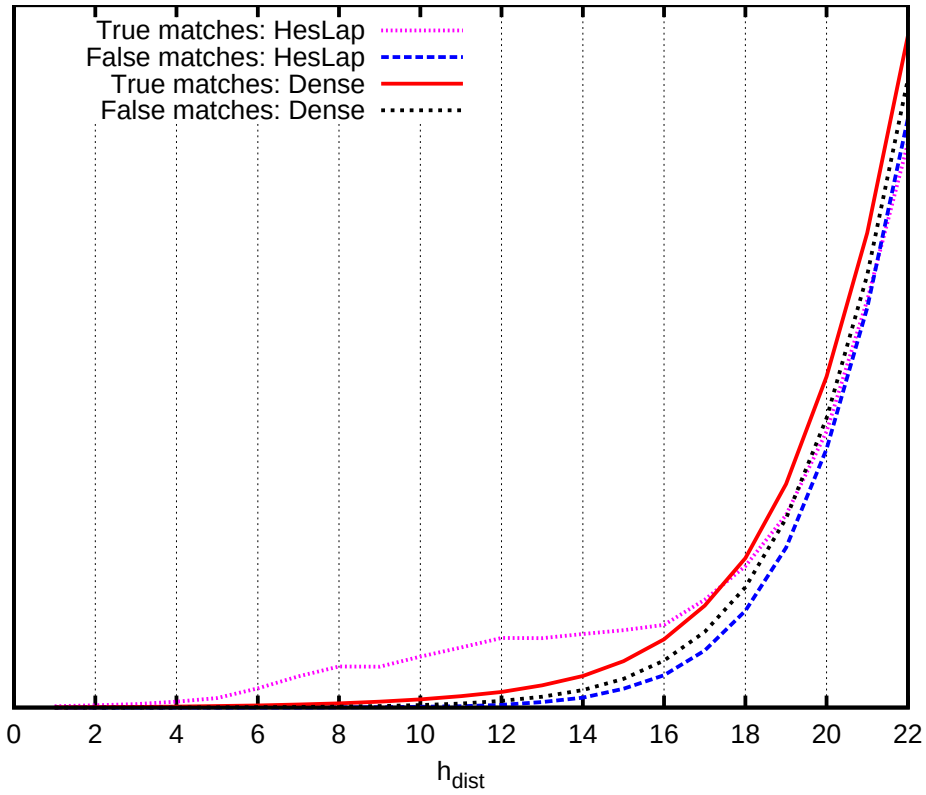


Figure 4.4: Empirical distribution of true matches and false matches as a function of the Hamming distance (h_{dist}). We only show a zoomed version for $h_{\text{dist}} = 0$ to 22. Measurements are performed on PASCAL VOC 2007 dataset (category *boat*).

tion [61] of this distance. In this work, we adopt a more simple choice in order to remove the parameter σ associated with the Gaussian function. More precisely, we use the linear scoring function $\frac{h_t - h}{h_t}$, where h is the Hamming distance between the binary signatures. From our preliminary experiments, the results obtained by this linear weighting scheme are comparable to the original Gaussian weighting function, which requires to optimize σ by cross-validation.

Burstiness Regularization In [61], a Burstiness regularization procedure is proposed to achieve improved results in image search. The so-called *burstiness* handling method regularizes the score associated with each match, to compensate the bursty statistics of regular patterns in images. Following these guidelines, we also apply this regularization to obtain better similarity scores.

4.2.3 HE for classification: motivation

Compared to the BOW representation, the main interest of HE is the additional information provided by the binary signature. We have conducted an analysis to evidence

that the Hamming distances between local descriptors provide a complementary and discriminative information for image classification. This analysis is performed on the PASCAL VOC 2007 dataset. SIFT descriptors are extracted from a dense grid as well as from the Hessian Laplace interest points. Any pair of descriptors is referred to as a *true match* if the two descriptors come from same object category, otherwise it is considered as a *false match*.

Figure 4.4 gives the empirical distribution, zoomed on small distances, of the true and false matches, as a function of the Hamming distance h_{dist}, for the category *boat*. One can observe that the Hamming distance provides a strong prior about the class: the expectation of false matches grows faster than that of true matches with h_{dist}, which confirms that low Hamming distances are more often related to true matches.

Note that all the false matches are accepted in the case of the BOW framework, that only uses vector quantization. Hamming Embedding based matching is able to filter out many false matches by choosing a proper threshold h_t. Moreover, the contribution of the matches are advantageously weighted based on the Hamming distance, in order to reflect the true/false match prior, and therefore to achieve better image classification. Note that setting a high threshold h_t would allow many false-matches to vote (as in BOW). On the other hand, a very low value is not satisfactory because too few matches are kept. It is therefore important to choose a threshold in an appropriate range, which is done by cross validation for a given dataset, see Section 4.3.1.

4.2.4 Hamming Embedding similarity space

We propose to apply HE to represent images in a similarity space. The idea is to represent an image by its similarity, as output by a strong matching system, to a set of sample images. There are few methods in the literature that employ such a similarity space for classification. These include nearest neighbors based approaches like NBNN [14] and its variations [12, 135] or graph based matching methods [33], see Section 4.1 for a short survey. However, none of these works is able to compete with the state-of-the-art Fisher kernel [109].

Similarity space image representation Unlike NBNN, which relies on pure NN classification, our motivation is to produce an image representation that can be fed to a strong classifier such as an SVM. This is more similar to [135] and [33], which use NBNN and graph-matching in their matching system. In our case, the HE similarity between a given image and the training images is obtained as the sum of voting scores, with burstiness regularization. Based on the analysis in Section 4.2.3, such a *similarity space embedding* is expected to be more discriminative than BOW. The image is represented by an N dimensional vector, where N is the number of training images. Each of its component is a similarity score to one of the training images. A given image I is therefore represented

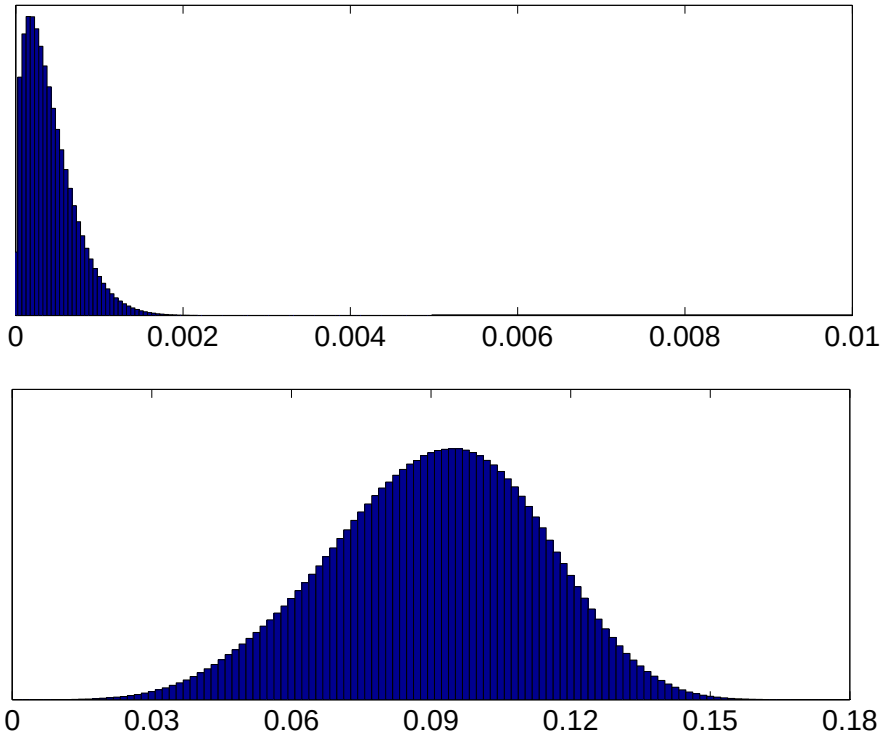


Figure 4.5: Distribution of similarity scores (a) before and (b) after power normalization (with $\alpha = 0.3$). Note the change in the scales. The scores are obtained for *trainval* set of PASCAL VOC 2007 dataset.

as:

$$I_{HE} = [\mathrm{HE}_{\mathrm{sim}}(I, I_1) \;\; \mathrm{HE}_{\mathrm{sim}}(I, I_2) \;\; \ldots \;\; \mathrm{HE}_{\mathrm{sim}}(I, I_N)] \tag{4.1}$$

where $\mathrm{HE}_{\mathrm{sim}}(I, I_i)$ is the similarity computed by HE between images I and I_i.

Remark: One of the key advantage of HE over NBNN based methods [14, 135] is that it does not need to compute the Euclidean nearest neighbors of the descriptors, which is costly both in terms of memory (to store the raw SIFT descriptors) and efficiency. In contrast, HE efficiently achieves accurate matching based on the binary signatures, which in addition are compact in memory (8 bytes per descriptor).

Normalization of similarity scores Figure 4.5 (top) shows the distribution of the scores produced by HE. One can observe that most of the scores are low, while few are high, due to the high discriminative power of this matching method. A similar observation was done for the Fisher Kernel [109]. Such a score distribution is not desirable for classification, because large values may generate some artifacts. In order to distribute the scores more evenly, we therefore adopt the power normalization method proposed to improve the Fisher Kernel [109]. It consists in applying the following component-wise function: $f(x) = x^{\alpha}$. Note that, in our case, the scores are all non-negative.

When optimizing the parameter α by cross-validation, we consistently obtain a value between 0.2 and 0.35. The values in this range provide comparable results, which suggests that this parameter can be set to a constant, e.g., $\alpha = 0.3$. Figure 4.5 (bottom) shows the distribution of images scores after power normalization with $\alpha = 0.3$, again computed on PASCAL VOC 2007 dataset. As one may observe, the power-law emphasizes the relative importance of low scores. Finally and similar to the Fisher Kernel, the vector is L2-normalized, producing the following final image representation:

$$\hat{I}_{\mathrm{HE}} = \frac{[\mathrm{HE}_{\mathrm{sim}}(I, I_1)^{\alpha} \;\; \mathrm{HE}_{\mathrm{sim}}(I, I_2)^{\alpha} \;\; \ldots \;\; \mathrm{HE}_{\mathrm{sim}}(I, I_N)^{\alpha}]}{\sqrt{\Sigma_{i=1}^{N} \mathrm{HE}_{\mathrm{sim}}(I, I_i)^{2\alpha}}}, \tag{4.2}$$

where the denominator is computed such that the Euclidean norm of the final vector is 1.

Spatial Grid The spatial pyramid matching proposed in [81] is a standard way to introduce some partial geometrical information in a bag-of-words representation. It consists in subdividing the image in a spatial grid and in computing histograms separately for each of the spatial regions thus defined. These spatial histograms are weighted according to the size of the region, normalized separately and then concatenated together to produce the final representation.

This idea is adapted to our HE-based representation. It is done by computing the HE similarities between each of the spatial regions and the training images. A noticeable difference is that the full training images are used to compute the similarity and not just the associated regions, because we observed that larger region gives slightly better results. The image is represented as 1×1 and 1×3 (three horizontal stripes) grids, that is 4 regions in total. Other methods usually draw 8 or 21 regions (add 2×2 or 4×4).

Another difference w.r.t. the method of [81] is that we train a linear SVM separately for each grid. Two SVMs are trained, one for 1×1 grid and another for 1×3 grid. The similarity scores of the three regions of 1×3 grid are stacked together to make $3N$ dimensional representation for training. The final classification scores are obtained as a weighted sum of the scores from both the classifiers. These weights are learned by cross-validating on the validation data. With the proposed image representation, training a SVM separately for each grid performs better than one SVM for both the grids.

4.3 Experiments and results

In this section, we first present some implementation details and then evaluate the proposed method on two challenging datasets for image classification: PASCAL VOC 2007 [37] and Caltech-256 [49].

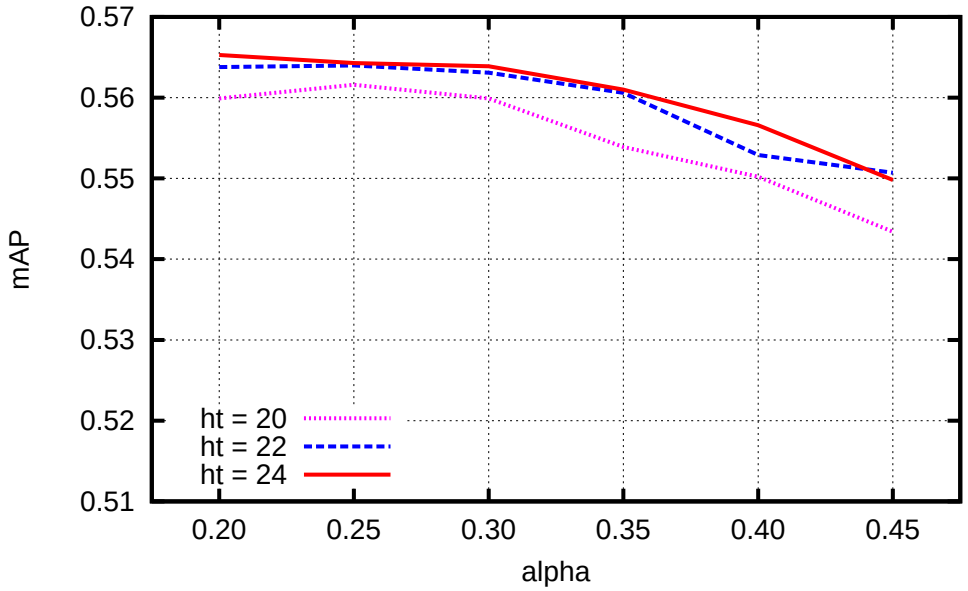


Figure 4.6: Impact of HE threshold (h_t) and α on mAP for test set of PASCAL VOC 2007.

4.3.1 Implementation Details

Only one type of feature is used in all our experiments, namely the SIFT descriptor computed on a dense grid. The descriptors are extracted from patches densely located with a spatial stride of 3 pixels on the image, under five scales. In [21], it is observed that such a dense sampling has a positive impact on classification accuracy. Also, it allows us to provide a consistent comparison of our method with several recent encodings evaluated in [21], and shows the interest of the variant of the SIFT descriptor introduced in Section 4.2.1. As we use dense features, *burstiness* handling [61] becomes more important as visual burst increases. Therefore in all the experiments we use burstiness regularization. For the sake of consistency, the vocabulary size is set $K = 4096$ for all our experiments with BOW and HE.

Key parameters There are two important parameters in our method, namely the HE threshold (h_t) and the parameter α involved in the power-law component-wise normalization. The impact of these parameters on the performance is shown in Figure 4.6, on the PASCAL VOC 2007 benchmark. The best choice of h_t, as obtained by cross-validation for binary signatures of length 64, is a threshold between 20 to 24. Interestingly, these values are consistent with those used with HE [62] in an image retrieval context. As suggested in Section 4.2, the parameter α is not very sensitive in the range [0.2,0.35], and is therefore set to the constant $\alpha = 0.3$ for all the experiments.

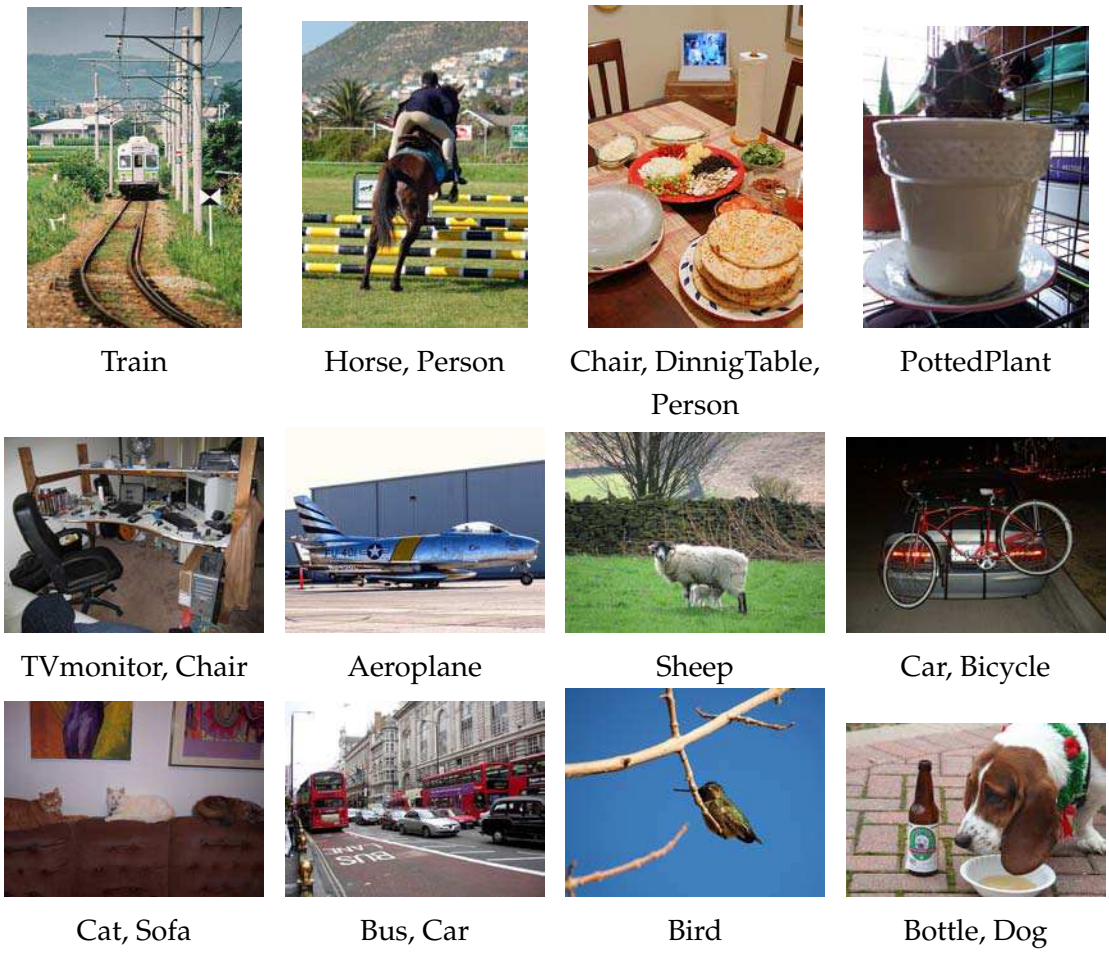


Figure 4.7: Examples from VOC-Pascal 2007 [37] dataset.

4.3.2 PASCAL VOC 2007

Evaluation protocol The PASCAL VOC 2007 [37] dataset contains about 10,000 images split into *train*, *validation* and *test* sets. It has 20 object categories and is considered a challenging dataset because of significant variations in appearances and poses with frequent occlusions. Some examples are shown in Figure 4.7, many images in the dataset contain objects of more than one class. For classification, a 1-versus-rest linear SVM classifier is trained for each category and the performance is evaluated in terms of average precision (AP) for each class. The overall performance is measured as the average of these APs, *i.e.*, it is the mean average precision (mAP). We follow the standard practice of training on *train+validation* and testing on *test*. The cross-validation of the different parameters, in particular the threshold h_t and the C parameter of the SVM (regularization-loss trade off), is performed by training on the train set and testing on the validation set. The C parameter is validated for each class whereas h_t is validated for the dataset and finally fixed to $h_t = 22$.

Methods	Codebook	Spat grid	SVM	mAP
FK	256	yes	Lin	61.69
FK*	256	yes	Lin	62.22
SV	1k	yes	Lin	58.13
BOW	4k	yes	Lin	46.54
BOW	4k	yes	Chi	53.42
LLC	4k	yes	Lin	53.79
LLC	4k	yes	Chi	53.47
LLC-F	4k	yes	Lin	55.87
KCB	4k	yes	Chi	54.60
HE	4k	no	Lin	**53.98**
HE	4k	yes	Lin	**56.68**
HE*	4k	no	Lin	**56.31**
HE*	4k	yes	Lin	**58.34**
HE* + FK*	4k, 256	no	Lin	**60.84**
HE* + FK*	4k, 256	yes	Lin	**62.78**

Table 4.1: Image classification results using PASCAL VOC 2007 [37] dataset with consistent setting of parameters. [FK: Fisher Kernel, FK*: Fisher Kernel with our SIFT variant, SV: super vector coding, BOW: bag of words, LLC: locally constrained linear coding, LLC-F: LLC with with original+left-right flipped training images, KCB: Kernel codebook, HE: Hamming Embedding similarity, HE*: HE with our SIFT variant; Lin/Chi: linear/χ^2 Kernel map].

The state of the art The best results reported on this dataset using only the SIFT feature were obtained with the Fisher Kernel [109]. They report 58.3% with grid and 55.3% without grid. Their descriptor dimensionality is typically about $32K$ (or more) without spatial grid, and 8 times more with it. In our case, the final representation is equal to the number of training images (*i.e.*, 5011 here) and 4 times more when using the spatial grid. Classification results are boosted by improved pooling of Avila *et al.* [8] (58.5%) and Krapac *et al.* [74] (56.7%).

Better results have been reported using more than one feature channel. For instance, the best classification method (by INRIA) in the original competition [37] obtained 59.4% using multiple feature channels and costly non-linear classifiers. Similarly, Kernel Codebook [141] and Yang et al [157] use many channels with soft assignment or sophisticated multiple Kernel learning to achieve mAP=60.5% and 62.2% respectively. The best results on this dataset have been obtained either by using costly object localization [35, 50, 126] or by using extra data [105].

Since different methods employ varying experimental settings (single/multiple feature, various sampling density and codebook sizes), it is difficult to have a consistent comparison. This issue is addressed by Chatfield et al. [21], who perform an independent evaluation of the recent encoding methods. With a consistent setting of parameters for all methods, the Fisher Kernel improves to 61.69% and the super vector coding [163]

Method/Class	HE*	FK	FK*	HE* + FK*
Aeroplane	76.50	78.97	**80.92**	80.75
Bicycle	62.70	57.43	67.39	**67.70**
Bird	50.23	51.94	**57.10**	56.77
Boat	68.62	**70.92**	69.01	69.86
Bottle	28.40	30.79	33.17	**33.77**
Bus	63.35	**72.18**	69.08	69.68
Car	79.37	79.94	80.42	**81.27**
Cat	61.20	61.35	61.51	**62.71**
Chair	52.52	55.98	55.43	**56.26**
Cow	45.24	49.61	49.89	**51.67**
DiningTable	52.85	58.40	58.71	**59.22**
Dog	47.11	44.77	48.98	**49.96**
Horse	77.06	78.84	79.56	**80.12**
Motorbike	64.27	**70.81**	70.02	70.27
Person	83.18	84.96	84.56	**85.20**
PottedPlant	32.46	31.72	34.65	**37.00**
Sheep	41.29	**51.00**	49.80	46.71
Sofa	50.15	**56.41**	55.08	55.54
Train	77.02	80.24	81.36	**81.81**
TVmonitor	53.24	57.46	57.76	**57.80**
mAP	**58.34**	**61.69**	**62.22**	**62.78**

Table 4.2: Image classification results per class using PASCAL VOC 2007 [37] dataset. Again, recall that FK* is the improved Fisher Kernel [109] combined with our better SIFT variant.

achieves a score of 58.13% (reported 64.0%). In Table 4.1, we refer to those results for a fair comparison. All the results reported in this table are, except for HE, HE* and FK*, from this paper [21]. Other elements like vocabulary size, classifiers used, spatial grid are mentioned in the table.

Impact of our SIFT's variant The method denoted by HE* is our Hamming Embedding similarity approach combined with the proposed SIFT variant detailed in Section 4.2.1. Similarly, FK* represents the Fisher Kernel combined with our SIFT variant. A considerable improvement of around 2% is observed by using HE* over HE both with and without spatial grid. The difference is only that HE uses original SIFT descriptors. As one can observe the variant also improves in case of Fisher Kernel, FK (original SIFT) and FK* use exactly the same parameters otherwise.

HE classification Our method, HE* with spatial grid, performs better than all the methods except the improved Fisher Kernel. With original sift descriptors (HE) mAP of 56.68% is obtained, which again compares favorably to most of the methods. Even without spatial grid HE* achieves a competitive mAP of 56.31%, while relying on a matching-based method. To our knowledge, it is the first method of that kind that approaches the

Figure 4.8: Examples from Caltech-256 [49] dataset.

best coding method FK on the PASCAL VOC 2007 benchmark.

Moreover, one would expect such a matching based method to be complementary with the coding based methods, even by using the same local descriptors in input. To confirm this, we combine our Hamming Embedding method (HE*) with the Fisher Kernel (FK*), using a late fusion of confidence scores. Doing so, we obtain an mAP of 60.84% and 62.78% without and with spatial grid respectively. Class-wise APs are reported in Table 4.2 for our approach and its combination with the Fisher Kernel. This combination improves the results for most of the categories.

4.3.3 Caltech-256

Evaluation protocol The Caltech-256 [49] dataset contains approximately 30K images falling into 256 categories. Each category contains at least 80 images. Figure 4.8 shows a few examples from the dataset. There is no provided division of dataset into train and test though. However, the standard practice is to split the dataset into train and test sets and repeat each experiment multiple times with different splits. We run experiments

	Methods/ntrain	15	30	45	60
Coding methods	**Baseline [49]**	-	34.10	-	-
	Kernel Codebook [141]	-	27.17	-	-
	EMK [13]	23.20	30.50	34.40	37.60
	SCSPM [158]	27.70	34.02	37.50	40.14
	Standard FK [107]	25.60	29.00	34.90	38.50
	Improved FK [109]	**34.70**	40.80	45.00	47.90
	LLC [149]	-	41.19	-	47.68
Matching-based	**Kim et al. [69]**	-	36.30	-	-
	NBNN [14]	-	38.00	-	-
	Duchenne et al. [33]	-	38.10	-	-
	HE*	32.49	**41.80**	**46.69**	**49.83**

Table 4.3: Comparison of HE similarity-based representation with the state-of-the-art on Caltech-256 [49].

with different numbers of training images per category: $ntrain = 15, 30, 45, 60$. The remaining images are used for testing. Validation is done on 5 images from train set by training on $ntrain - 5$ images. The validated h_t is equal to 20 for this dataset. We run experiments for five random splits for each *ntrain*. Again a 1-vs-rest linear SVM is trained for each class. We report the average classification accuracy (standard practice) across all classes.

Results Table 4.3 compares our results with the best reported ones. We divide the methods as matching or coding based, all of them use only SIFT feature. Compared to PASCAL VOC, matching-based methods perform comparatively better on Caltech-256, outperforming many coding approaches such as Kernel-Codebook [141], Sparse-Coding [158], Standard FK [107] and the baseline by the authors of Caltech 256 dataset [49]. Overall, our method outperforms all the matching and coding based approaches. Only the improved Fisher Kernel [109] and LLC [149] perform better in the case of 15 training images. This is not surprising, because in our case the dimensionality of the final representation is equal to the number of training images. With more training images, the dimensionality of our descriptor increases and leads to the best results.

4.4 Conclusions

In this chapter, we have presented a novel approach to image classification based on a matching technique. It consists in combining the Hamming-Embedding similarity-based matching method with a similarity space encoding, which subsequently allows the use of a linear SVM. This method is efficient and achieves state-of-the-art classification results on two reference image classification benchmarks: the PASCAL VOC 2007 and Caltech-256 datasets. Our approach is one of the very few methods that obtain best results on

both these datasets. To our knowledge, this method is the first matching-based approach to provide such competitive results. Furthermore, it is shown to be complementary with the other best classification method, namely the Fisher kernel.

We believe that the flexibility offered by this framework is likely to be extended, in particular for a better integration of the geometrical constraints. As a secondary contribution, we have proposed an effective variant of the SIFT descriptor, which gives a slight yet consistent improvement on classification accuracy. Its interest has been validated with the proposed Hamming-Embedding similarity-based matching as well as the Fisher Kernel.

CHAPTER

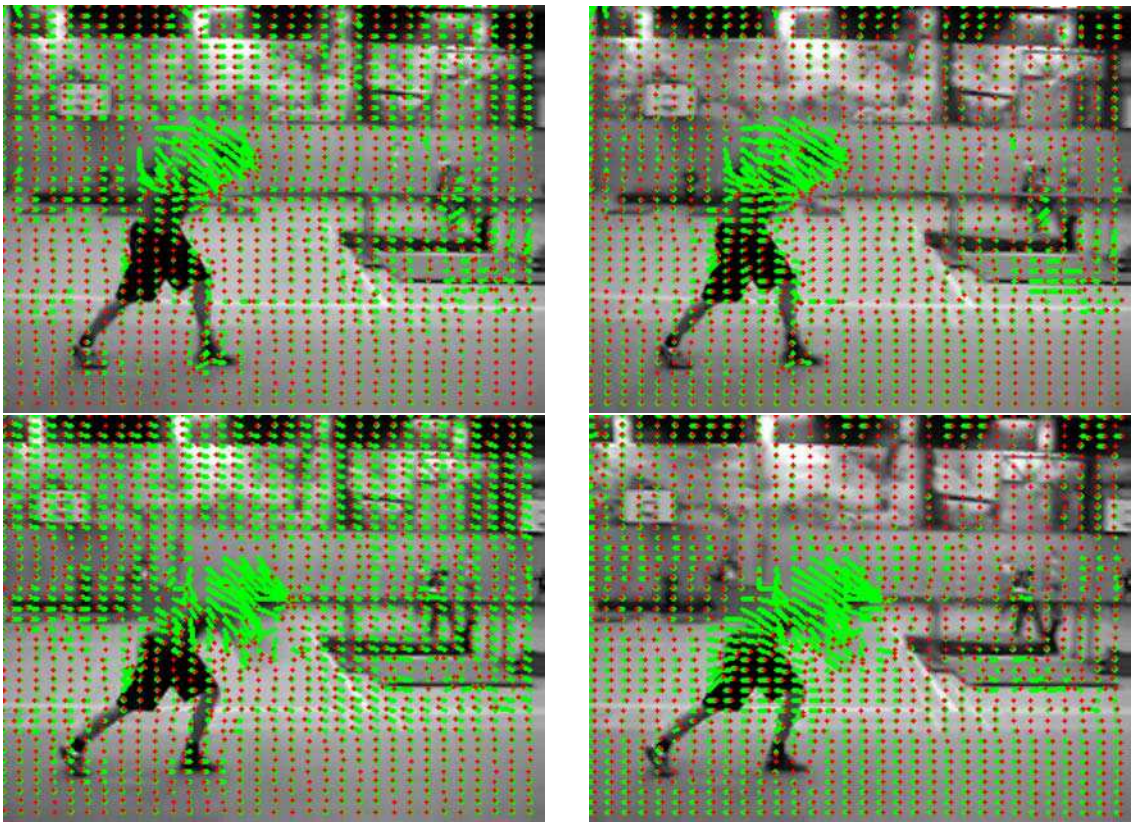
FIVE

Improved motion description for action classification

Human actions often convey the essential meaningful content in videos. Yet, recognizing human actions in unconstrained videos is a challenging problem in Computer Vision which receives a sustained attention due to the potential applications. In particular, there is a large interest in designing video-surveillance systems, providing some automatic annotation of video archives as well as improving human-computer interaction. The solutions proposed to address this problem inherit, to a large extent, from the techniques first designed for the goal of image search and classification. The successful local features developed to describe image patches [85, 121] have been translated in the 2D+t domain as spatio-temporal local descriptors [79, 148] and now include motion clues [145]. These descriptors are often extracted from spatial-temporal interest points [78, 151]. More recent techniques assume some underlying temporal motion model involving trajectories [18, 44, 51, 91, 92, 128, 145, 147, 153].

Most of these approaches produce large set of local descriptors which are in turn aggregated to produce a single vector representing the video, in order to enable the use of powerful discriminative classifiers such as support vector machines (SVMs). This is usually done with the bag-of-words technique [123], which quantizes the local features using a k-means codebook. Thanks to the successful combination of this encoding technique with the aforementioned local descriptors, the state of the art in action recognition is able to go beyond the toy problems of classifying simple human actions in controlled environment and considers the detection of actions in real movies or video clips [75, 89]. Despite these progresses, the existing descriptors suffer from an uncompleted handling of motion in the video sequence.

Motion is arguably the most reliable source of information for action recognition, as often related to the actions of interest. However, it inevitably involves the background or camera motion when dealing with uncontrolled and realistic situations. Although some attempts have been made to compensate camera motion in several ways [72, 113, 136, 145, 153], how to separate action motion from that caused by the camera, and how to reflect it in the video description remains an open issue. The motion



Optical flow vectors | Compensated flow vectors

Figure 5.1: Optical flow field vectors (green vectors with red end points) before and after dominant motion compensation. Most of the flow vectors due to camera motion are suppressed after compensation. One of the contributions of this work is to show that compensating for the dominant motion is beneficial for most of the existing descriptors used for action recognition.

compensation mechanism employed in [72] is tailor-made to the Motion Interchange Pattern encoding technique. The Motion Boundary Histogram (MBH) [145] is a recent appealing approach to suppress the constant motion by considering the flow gradient. It is robust to some extent to the presence of camera motion, yet it does not explicitly handle the camera motion. Another approach [136] uses a sophisticated and robust (RANSAC) estimation of camera motion. It first segments the color image into regions corresponding to planar parts in the scene and estimates the (three) dominant homographies to update the motion associated with local features. A rather different view is adopted in [153] where the motion decomposition is performed at the trajectory level. All these works support the potential of motion compensation.

As the first contribution of this chapter, we address the problem in a way that departs from these works by considering the compensation of the dominant motion in *both* the tracking stages and encoding stages involved in the computation of action recognition descriptors. We rely on the pioneering works on motion compensation such as the technique proposed in [101], that considers 2D polynomial affine motion models for estimating the dominant image motion. We consider this particular model for its robustness and

its low computational cost. It was already used in [113] to separate the dominant motion (assumed to be due to the camera motion) and the residual motion (corresponding to the independent scene motions) for dynamic event recognition in videos. However, the statistical modeling of both motion components was global (over the entire image) and only the normal flow was computed for the latter.

Figure 5.1 shows the vectors of optical flow before and after applying the proposed motion compensation. Our method successfully suppresses most of the background motion and reinforces the focus towards the action of interest. We exploit this compensated motion *both* for descriptor computation and for extracting trajectories. However, we also show that the camera motion should not be thrown as it contains complementary information that is worth using to recognize certain action categories. The most similar to our dominant motion compensation approach [56] is the very recently proposed method of Wang *et al.* [147]. They also adopt this idea of handling camera motion for improving motion trajectories and descriptors, but propose different means to achieve it. We further discuss and compare with their approach in Section 5.6.

Then, we introduce the Divergence-Curl-Shear (DCS) descriptor, which encodes scalar first-order motion features, namely the motion divergence, curl and shear. It captures physical properties of the flow pattern that are not involved in the best existing descriptors for action recognition, except in the work of [4] which exploits divergence and vorticity among a set of eleven kinematic features computed from the optical flow. Our DCS descriptor provides a good performance recognition performance on its own. Most importantly, it conveys some information which is not captured by existing descriptors and further improves the recognition performance when combined with the other descriptors.

As a last contribution, we bring an encoding technique known as VLAD (vector of local aggregated descriptors) [65] to the field of action recognition. This technique is shown to be better than the bag-of-words representation for combining all the local video descriptors we have considered. We also employ another higher-order encoding technique, Fisher vector, which has been used in many recent works on action recognition [5, 103, 125, 127, 147]. The work presented in this chapter was published in [56]. Here we have also added a few minor extensions to the paper.

The organization of the chapter is as follows. Section 5.1 introduces the motion properties that we will consider through this chapter. Section 5.2 presents the datasets and classification scheme used in our different evaluations. Section 5.3 details how we revisit several popular descriptors of the literature by the means of dominant motion compensation. Our DCS descriptor based on kinematic properties is introduced in Section 5.4. In Section 5.5, VLAD and Fisher encoding techniques are presented that lead to improved performance. Section 5.6 provides a comparison with the state-of-the-art. Finally, Section 5.7 concludes the chapter.

5.1 Motion Separation and Kinematic Features

In this section, we describe the motion clues we incorporate in our action recognition framework. We separate the dominant motion and the residual motion. In most cases, this will account to distinguishing the impact of camera movement and independent actions. Note that we do not aim at recovering the 3D camera motion: The 2D parametric motion model describes the global (or dominant) motion between successive frames. We first explain how we estimate the dominant motion and employ it to separate the dominant flow from the optical flow. Then, we will introduce kinematic features, namely divergence, curl and shear for a more comprehensive description of the visual motion.

5.1.1 Affine motion for compensating camera motion

Among polynomial motion models, we consider the 2D affine motion model. Simplest motion models such as the 4-parameter model formed by the combination of 2D translation, 2D rotation and scaling, or more complex ones such as the 8-parameter quadratic model (equivalent to a homography), could be selected as well. The affine model is a good trade-off between accuracy and efficiency which is of primary importance when processing a huge video database. It does have limitations since strictly speaking it implies a single plane assumption for the static background. However, this is not that penalizing (especially for outdoor scenes) if differences in depth remain moderated with respect to the distance to the camera. The affine flow vector at point $p = (x, y)$ and at time t, is defined as

$$w_{\text{aff}}(p_t) = \begin{bmatrix} c_1(t) \\ c_2(t) \end{bmatrix} + \begin{bmatrix} a_1(t) & a_2(t) \\ a_3(t) & a_4(t) \end{bmatrix} \begin{bmatrix} x_t \\ y_t \end{bmatrix}. \tag{5.1}$$

$u_{\text{aff}}(p_t) = c_1(t) + a_1(t)x_t + a_2(t)y_t$ and $v_{\text{aff}}(p_t) = c_2(t) + a_3(t)x_t + a_4(t)y_t$ are horizontal and vertical components of $w_{\text{aff}}(p_t)$ respectively. Let us denote the optical flow vector at point p at time t as $w(p_t) = (u(p_t), v(p_t))$. We introduce the flow vector $\omega(p_t)$ obtained by removing the affine flow vector from the optical flow vector

$$\omega(p_t) = w(p_t) - w_{\text{aff}}(p_t). \tag{5.2}$$

The dominant motion (estimated as $w_{\text{aff}}(p_t)$) is usually due to the camera motion. In this case, Equation 5.2 amounts to canceling (or compensating) the camera motion. Note that this is not always true. For example in case of moving camera with close-up on a moving actor, the dominant motion will be the affine estimation of the combination of the apparent actor motion and the camera motion. The interpretation of the motion compensation output will not be that straightforward in this case. However, the resulting ω-field will still exhibit different patterns for the foreground action part and the background part. Even when the camera is static, the affine model cannot completely account for actor's

complex motion, so there is no major depletion of the action of interest in the residual or compensated motion. In the remainder, we will refer to the "compensated" flow as *ω-flow*.

Figure 5.1 displays the computed optical flow and the ω-flow. We compute the affine flow with the publicly available Motion2D software[1] [101] which implements a real-time robust multiresolution incremental estimation framework. The affine motion model has correctly accounted for the motion induced by the camera movement which corresponds to the dominant motion in the image pair. Indeed, we observe that the compensated flow vectors in the background are close to null and the compensated flow in the foreground, *i.e.*, corresponding to the actors, is conversely inflated. The experiments presented along this chapter will show that effective separation of dominant motion from the residual motions is beneficial for action recognition. As explained in Section 5.3, we will compute local motion descriptors, such as HOF, on both the optical flow and the compensated flow (ω-flow), which allows us to explicitly and directly characterize the scene motion.

5.1.2 Local kinematic features

By kinematic features, we mean local first-order differential scalar quantities computed on the flow field. We consider the divergence, the curl (or vorticity) and the hyperbolic terms. They inform on the physical pattern of the flow so that they convey useful information on actions in videos. They can be computed from the first-order derivatives of the flow at every point p at every frame t as

$$\begin{cases} \mathsf{div}(p_t) & = \frac{\partial u(p_t)}{\partial x} + \frac{\partial v(p_t)}{\partial y} \\ \mathsf{curl}(p_t) & = \frac{-\partial u(p_t)}{\partial y} + \frac{\partial v(p_t)}{\partial x} \\ \mathsf{hyp}_1(p_t) & = \frac{\partial u(p_t)}{\partial x} - \frac{\partial v(p_t)}{\partial y} \\ \mathsf{hyp}_2(p_t) & = \frac{\partial u(p_t)}{\partial y} + \frac{\partial v(p_t)}{\partial x} \end{cases} \tag{5.3}$$

The divergence is related to axial motion, expansion and scaling effects, the curl to rotation in the image plane. The hyperbolic terms express the shear of the visual flow corresponding to more complex configuration. We take into account the shear quantity only:

$$\mathsf{shear}(p_t) = \sqrt{\mathsf{hyp}_1^2(p_t) + \mathsf{hyp}_2^2(p_t)}. \tag{5.4}$$

In Section 5.4, we propose the DCS descriptor that is based on the kinematic features (divergence, curl and shear) of the visual motion discussed in this subsection. It is computed on either the optical or the compensated flow, ω-flow.

[1] http://www.irisa.fr/vista/Motion2D/

Figure 5.2: Examples from Hollywood2 [89] dataset, one for each of the twelve classes.

5.2 Datasets and evaluation

This section first introduces the datasets used for the evaluation. Then, we briefly present the bag-of-feature model and the classification scheme used to encode the descriptors which will be introduced in Section 5.3.

Hollywood2. The Hollywood2 [89] dataset contains 1,707 video clips from 69 movies representing 12 action classes. In Figure 5.2, one example is shown for each class. The dataset is divided into train set and test set of 823 and 884 samples respectively. Following the standard evaluation protocol of this benchmark, we use average precision (AP) for each class and the mean of APs (mAP) for evaluation.

HMDB51. The HMDB51 [75] dataset is a large dataset containing 6,766 video clips extracted from various sources, ranging from movies to YouTube. It consists of 51 action classes, each having at least 101 samples. Many action categories of different types are covered including the regular day-to-day actions, sports activities and the subtle ones with very less amount of motion. Some of examples are shown in Figure 5.3. We follow the evaluation protocol of [75] and use three train/test splits, each with 70 training and 30 testing samples per class. The average classification accuracy is computed over all classes. Out of the two released sets, we use the original set as it is more challenging and used by most of the works reporting results in action recognition.

Olympic Sports. The third dataset we use is Olympic Sports [98], which again is obtained from YouTube. This dataset contains 783 samples with 16 sports action classes.

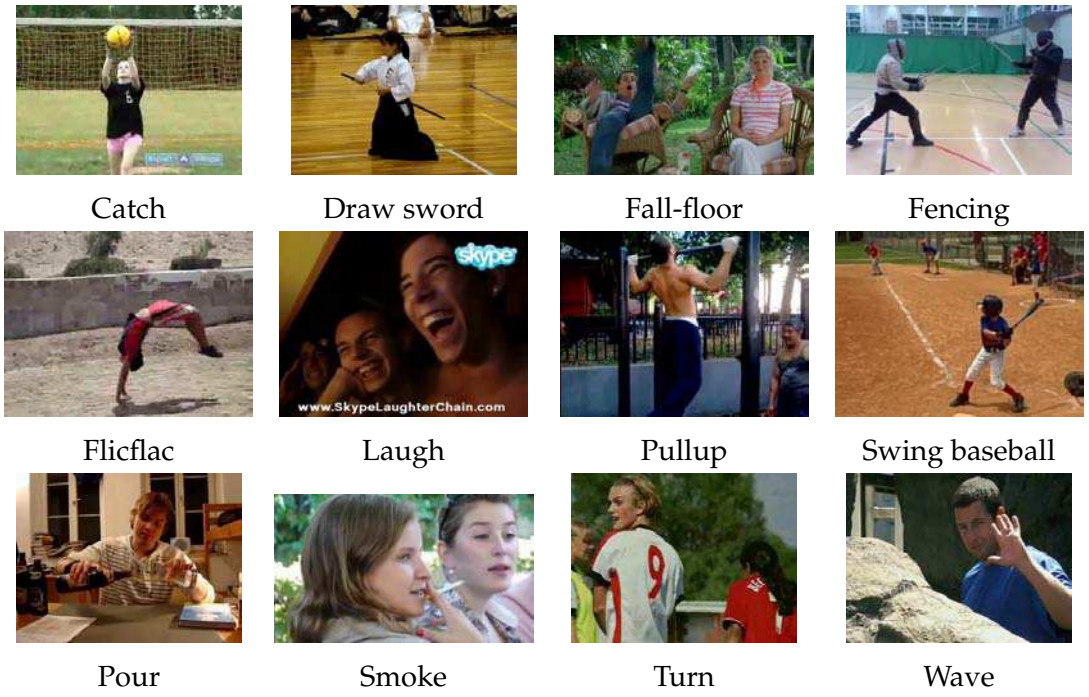


Figure 5.3: Examples from HMDB51 [75] dataset for a few of 51 classes.

Figure 5.4: Examples from Olympic Sports [98] dataset, one for each of the sixteen classes.

One example from each class is shown in Figure 5.4. We use the provided[2] train/test split, there are 17 to 56 training samples and 4 to 11 test samples per class. Mean AP is used for the evaluation, which is the standard choice.

Bag of features and classification setup. We first adopt the standard BOF [123] approach to encode all kinds of descriptors. It produces a vector that serves as the video representation. The codebook is constructed for each type of descriptor separately by the k-means algorithm. Following a common practice in the literature [138, 145, 148], the codebook size is set to k=4,000 elements. Note that Section 5.5 will consider encoding technique for descriptors.

For the classification, we use a non-linear SVM with χ^2-kernel. When combining different descriptors, we simply add the kernel matrices, as done in [138]:

$$K(x_i, x_j) = \exp\left(-\sum_c \frac{1}{\gamma^c} D(x_i^c, x_j^c)\right), \tag{5.5}$$

where $D(x_i^c, x_j^c)$ is χ^2 distance between video x_i^c and x_j^c with respect to *c-th* channel, corresponding to *c-th* descriptor. The quantity γ^c is the mean value of χ^2 distances between the training samples for the *c-th* channel. The multi-class classification problem that we consider is addressed by applying a *one-against-rest* approach.

5.3 Compensated descriptors

This section describes how the compensation of the dominant motion is exploited to improve the quality of descriptors encoding the motion and the appearance around spatio-temporal positions, hence the term "compensated descriptors". First, we briefly review the local descriptors [30, 79, 89, 145, 148] used here along with dense trajectories [145]. Second, we analyze the impact of motion flow compensation when used in two different stages of the descriptor computation, namely in the tracking and the description part.

5.3.1 Dense trajectories and local descriptors

Employing dense trajectories to compute local descriptors is one of the state-of-the-art approaches for action recognition. It has been shown [145] that when local descriptors are computed over dense trajectories the performance improves considerably compared to when computed over spatio temporal features [148].

Dense Trajectories [145]: The trajectories are obtained by densely tracking sampled points using optical flow fields. For optical flow computation an efficient algorithm by

[2] http://vision.stanford.edu/Datasets/OlympicSports/

Farnebäck [40] is used. First, feature points are sampled from a dense grid, with step size of 5 pixels and over 8 scales. Each feature point $p_t = (x_t, y_t)$ at frame t is then tracked to the next frame by median filtering in a dense optical flow field $F = (u_t, v_t)$ as follows:

$$p_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * F)|_{(\bar{x}_t, \bar{y}_t)}, \tag{5.6}$$

where M is the kernel of median filtering and $(\bar{x}_t, \bar{y}_t)$ is the rounded position of (x_t, y_t). The tracking is limited to L (=15) frames to avoid any drifting effect. Excessively short trajectories and trajectories exhibiting sudden large displacements are removed as they induce some artifacts. Trajectories must be understood here as tracks in the space-time volume of the video.

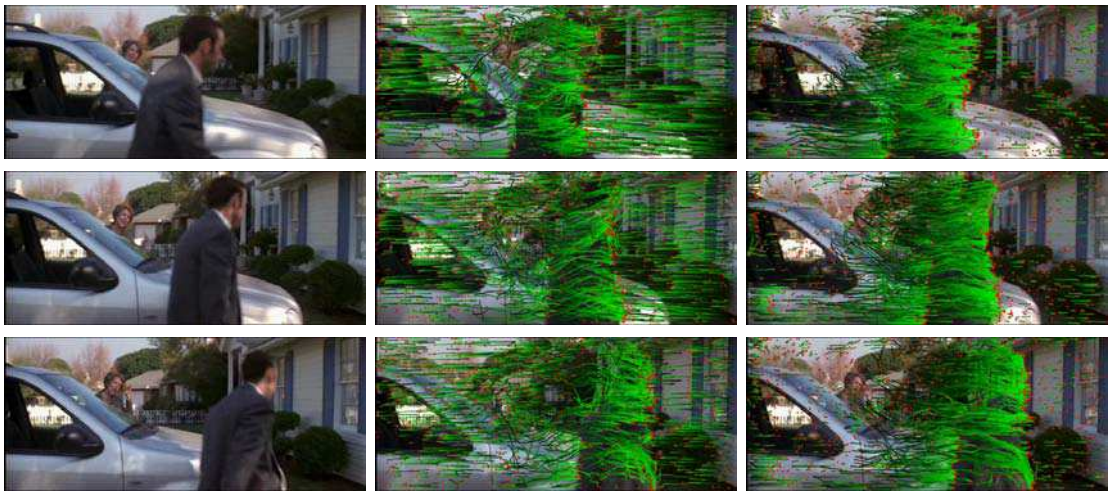
Local descriptors: The descriptors are computed within a space-time volume centered around each trajectory. Four types of descriptors are computed to encode the shape of the trajectory, local motion pattern and appearance, namely Trajectory [145], HOF (histograms of optical flow) [79], MBH [26] and HOG (histograms of oriented gradients) [25]. All these descriptors depend on the flow field used for the tracking and as input of the descriptor computation:

1. The **Trajectory** descriptor encodes the shape of the trajectory represented by the normalized relative coordinates of the successive points forming the trajectory. It directly depends on the dense flow used for tracking points.
2. **HOF** is computed using the orientations and magnitudes of the flow field.
3. **MBH** is designed to capture the gradient of horizontal and vertical components of the flow. The motion boundaries encode the relative pixel motion and therefore suppress camera motion, but only to some extent.
4. **HOG** encodes the appearance by using the intensity gradient orientations and magnitudes. It is formally not a motion descriptor. Yet the position where the descriptor is computed depends on the trajectory shape.

As in [145], volume around a feature point is divided into a $2 \times 2 \times 3$ space-time grid. The orientations are quantized into 8 bins for HOG and 9 bins for HOF (with one additional zero bin). The horizontal and vertical components of MBH are separately quantized into 8 bins each.

5.3.2 Impact of motion compensation

The optical flow is simply referred to as *flow* in the following, while the compensated flow (see subsection 5.1.1) is denoted by *ω-flow*. Both of them are considered in the tracking and descriptor computation stages. The trajectories obtained by tracking with the ω-flow are called *ω-trajectories*. Figure 5.5 comparatively illustrates the ω-trajectories and the trajectories obtained using the flow. The input video shows a man moving away from the car. In this video excerpt, the camera is following the man walking to the



(a) Consecutive frames (b) Optical flow trajectories (c) ω-trajectories

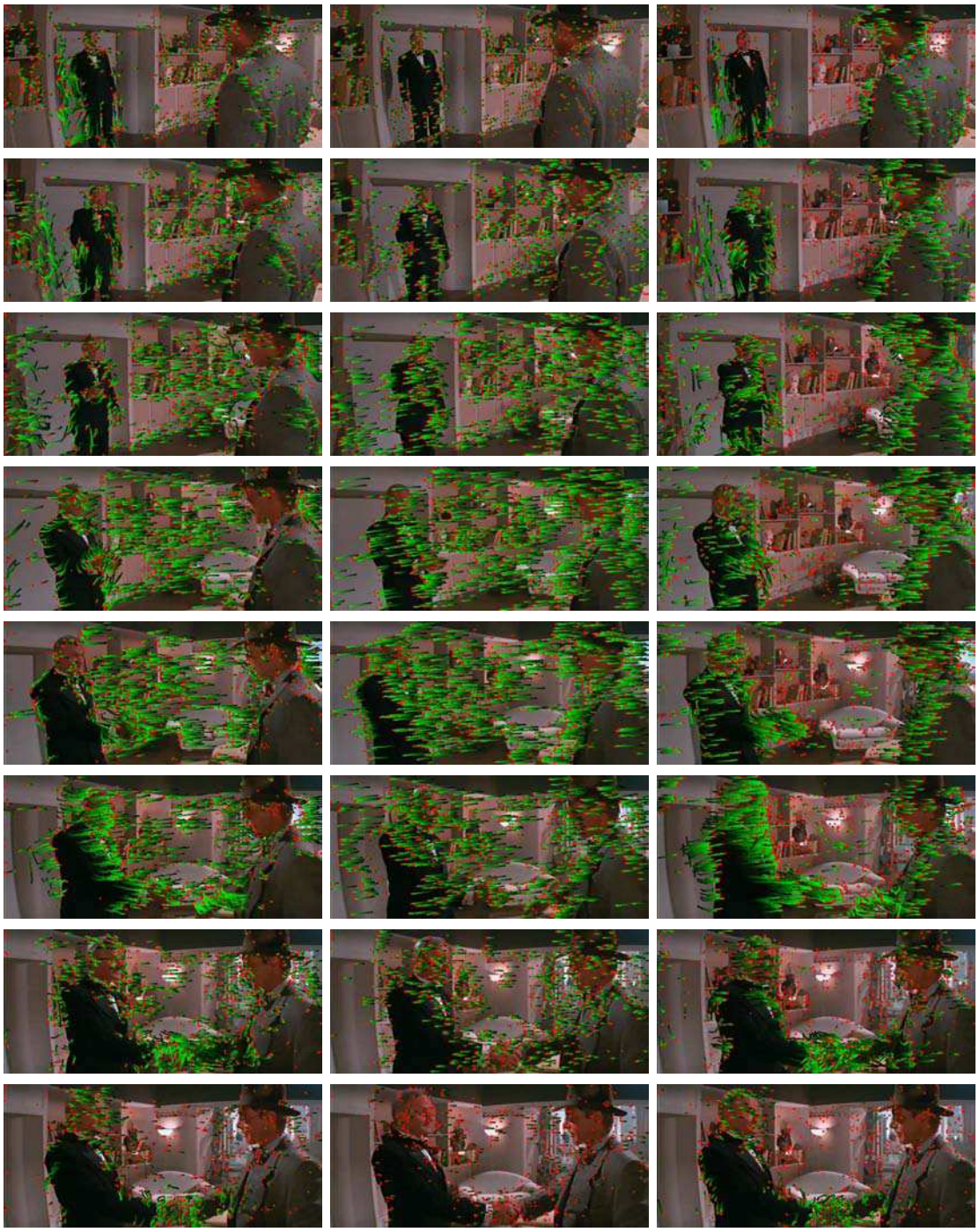
Figure 5.5: Trajectories obtained from optical and compensated flows. The green tail is the trajectory over 15 frames with red dot indicating the current frame. The trajectories are sub-sampled for the sake of clarity. The frames are shown at an interval of 5 frames.

right, thus inducing a global motion to the left in the video. When using the flow, the computed trajectories reflect the combination of these two motion components (camera *and* scene motion) as depicted by Subfigure 5.5(b), which hampers the characterization of the current action. In contrast, the ω-trajectories plotted in Subfigure 5.5(c) are more active on the actor moving on the foreground, while those localized in the background are now parallel to the time axis enhancing static parts of the scene. The ω-trajectories are therefore more relevant for action recognition, since they are more regularly and more exclusively following the actor's motion.

Another example is shown in Figure 5.6, a sequence of *HandShake* action. Camera is following the person on the left (in black dress), who is moving towards another person on the right. This induces global motion towards left as displayed by the trajectories from affine flow in the center of the figure. As a result, there are many trajectories from flow between the two persons shaking hands, *i.e.*, in the background. After motion compensation, most of the trajectories in the background are suppressed and the resulting ω-trajectories are more exclusively following the action of interest.

Impact on Trajectory and HOG descriptors. Table 5.1 reports the impact of ω-trajectories on Trajectory and HOG descriptors, which are both significantly improved by 3%-4% of mAP on the two datasets. When improved by ω-flow, these descriptors will be respectively referred to as ω-*Trajdesc* and ω-*HOG* in the rest of the chapter.

Although the better performance of ω-Trajdesc versus the original Trajectory descriptor was expected, the one achieved by ω-HOG might be surprising. Our interpretation is that HOG captures more context with the modified trajectories. More precisely, the orig-



(a) Optical flow trajectories (b) Affine flow trajectories (c) ω-trajectories

Figure 5.6: Frame sequence of action *HandShake* is shown with trajectories obtained from optical, affine and compensated flows. The green tail is the trajectory over 15 frames with red dot indicating the current frame. The trajectories are sub-sampled for the sake of clarity. The frames are shown at an interval of 5 frames. Note the many trajectories in the background from optical flow are suppressed in ω-trajectories.

Descriptor	Hollywood2	HMDB51
Trajectory [145]	47.7%	–
Baseline (reproduced)	47.7%	28.8%
ω-Trajdesc	**51.4%**	**32.9%**
HOG [145]	41.5%	–
Baseline (reproduced)	41.8%	26.3%
ω-HOG	**45.6%**	**29.1%**

Table 5.1: ω-Trajdesc and ω-HOG: Impact of compensating flow on Trajectory descriptor and HOG descriptors.

Method		Hollywood2	HMDB51
HOF [145]		50.8%	–
HOF (Tracking flow)	flow	50.8%	30.8%
	ω-flow	52.4%	**36.8%**
	both	**54.1%**	**37.7%**
HOF (Tracking ω-flow)	flow	50.2%	33.0%
	ω-flow	52.5%	**37.1%**
	both: ω-HOF	**53.9%**	**38.6%**

Table 5.2: Impact of using ω-flow on HOF descriptors: mAP for Hollywood2 and average accuracy for HMDB51. The ω-HOF is used in subsequent evaluations.

inal HOG descriptor is computed from a 2D+t sub-volume aligned with the corresponding trajectory and hence represents the appearance along the trajectory shape. When using ω-flow, we do not align the video sequence. As a result, the ω-HOG descriptor is no more computed around the very same tracked physical point in the space-time volume but around points lying in a patch of the initial feature point, whose size depends on the affine flow magnitude. ω-HOG can be viewed as a "patch-based" computation capturing more information about the appearance of the background or of the moving foreground. As for ω-trajectories, they are closer to the real trajectories of the moving actors as they usually cancel the camera movement, and so, more easier to train and recognize.

Impact on HOF. The ω-flow impacts both the trajectory computation used as an input to HOF and the descriptor computation itself. Therefore, HOF can be computed along both types of trajectories (ω-trajectories or those extracted from flow) and can encode both kinds of flows (ω-flow or flow). For the sake of completeness, we evaluate all the variants as well as the combination of both flows in the descriptor computation stage.

The results are presented in Table 5.2 and demonstrate the significant improvement obtained by computing the HOF descriptor with the ω-flow instead of the optical flow. Note that the type of trajectories which is used, either "Tracking flow" or "Tracking ω-flow", has a limited impact in this case. From now on, we only consider the "Tracking

Method		Hollywood2	HMDB51
MBH [145]		54.2%	–
MBH (Tracking *flow*)	flow	**54.2%**	39.7%
	ω-flow	54.0%	39.3%
MBH (Tracking ω-flow)	flow	52.7%	**40.9%**
	ω-flow	52.5%	40.6%

Table 5.3: Impact of using ω-flow MBH descriptors: mAP for Hollywood2 and average accuracy for HMDB51.

ω-flow" case where HOF is computed along ω-trajectories.

Interestingly, combining the HOF computed from the flow and the ω-flow further improves the results. This suggests that the two flow fields are complementary and the affine flow that was subtracted from ω-flow brings in additional information. For the sake of brevity, the combination of the two kinds of HOF, *i.e.*, computed from the flow and the ω-flow using ω-trajectories, is referred to as the *ω-HOF* descriptor in the rest of this chapter. Compared to the HOF baseline, the ω-HOF descriptor achieves a gain of $+3.1\%$ of mAP on Hollywood 2 and of $+7.8\%$ on HMDB51.

Impact on MBH. Since MBH is computed from gradient of flow and cancels the constant motion, there is practically no benefit in using the ω-flow to compute the MBH descriptors, as shown in Table 5.3. However, by tracking ω-flow, the performance improves by around 1.3% for HMDB51 dataset and drops by around 1.5% for Hollywood2. This relative performance depends on the encoding technique. We will come back on this descriptor when considering higher-order encoding schemes in Section 5.5.

5.3.3 Summary of compensated descriptors

Table 5.4 summarizes the refined versions of the descriptors obtained by exploiting the ω-flow, and both ω-flow and the optical flow in the case of HOF. The revisited descriptors considerably improve the results compared to the original ones, with the noticeable exception of ω-MBH which gives mixed performance with a bag-of-features encoding scheme.

Another advantage of tracking the compensated flow is that fewer trajectories are produced. For instance, the total number of trajectories decreases by about 9.16% and 22.81% on the Hollywood2 and HMDB51 datasets, respectively. Note that exploiting both the flow and the ω-flow do not induce much computational overhead, as the latter is obtained from the flow and the affine flow which is computed in real-time and already used to get the ω-trajectories. The only additional computational cost that we introduce by using the descriptors summarized in Table 5.4 is the computation of a second HOF

Descriptor	Tracking with	Computing descriptor with	ω-flow descriptor
Trajectory	ω-flow	N/A	ω-Trajdesc
HOG	ω-flow	N/A	ω-HOG
HOF	ω-flow	ω-flow + flow	ω-HOF
MBH	ω-flow	ω-flow	ω-MBH

Table 5.4: Summary of the updated *ω-flow* descriptors

descriptor, but this stage is relatively efficient and not the bottleneck of the extraction procedure.

5.4 Divergence-Curl-Shear descriptor

This section introduces a new descriptor encoding the kinematic properties of motion discussed in Section 5.1.2. It is denoted by DCS in the rest of this chapter.

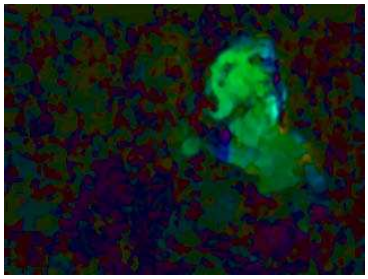
Combining kinematic features. The spatial derivatives are computed for the horizontal and vertical components of the flow field, which are actually horizontal (MBHx) and vertical (MBHy) parts of MBH descriptor. The input frame and the computed optical flow are shown with these two gradients in Figure 5.7(a). These gradients are in turn used to compute the divergence, curl and shear scalar values as given by Equation 5.3. Figure 5.7(b) shows these three kinematic features computed for the input frame.

We consider all possible pairs of kinematic features, namely (div, curl), (div, shear) and (curl, shear). At each pixel, we compute the orientation and magnitude of the 2-D vector corresponding to each of these pairs. Figure 5.7(c) illustrates the information captured by these three pairs. The orientation is quantized into histograms and the magnitude is used for weighting, similar to SIFT. Our motivation for encoding pairs is that the joint distribution of kinematic features conveys more information than exploiting them independently. Another example is shown in Figure 5.8 to illustrate the information captured by our descriptor.

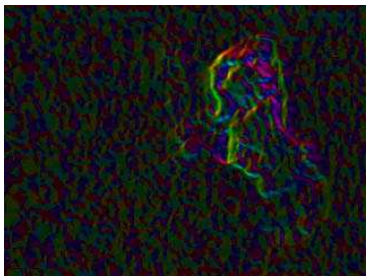
Implementation details. The descriptor computation and parameters are similar to HOG and other popular descriptors such as MBH, HOF. We obtain 8-bin histograms for each of the three feature pairs or components of DCS. The range of possible angles is 2π for the (div,curl) pair and π for the other pairs, because the shear is always positive.

The DCS descriptor is computed for a space-time volume aligned with a trajectory, as done with the four descriptors mentioned in the previous section. In order to capture the spatio-temporal structure of kinematic features, the volume (32×32 pixels and $L = 15$ frames) is subdivided into a spatio-temporal grid of size $n_x \times n_y \times n_t$, with $n_x = n_y = 2$

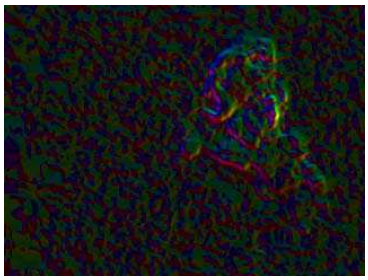
Frame



Optical flow

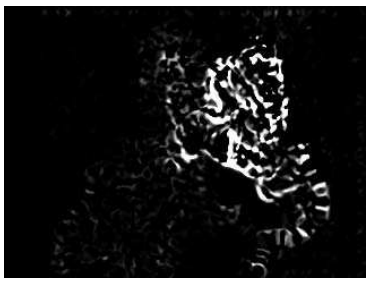


Horizontal gradient



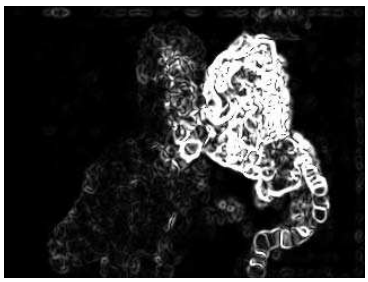
Vertical gradient

(a) Input frame, optical flow and the horizontal (MBHx) and vertical (MBHy) gradients of the optical flow.



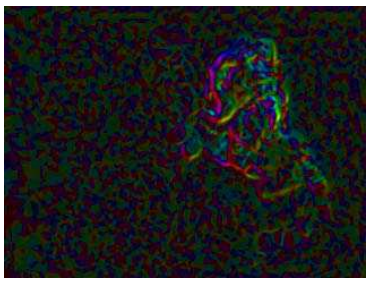
Divergence

Curl

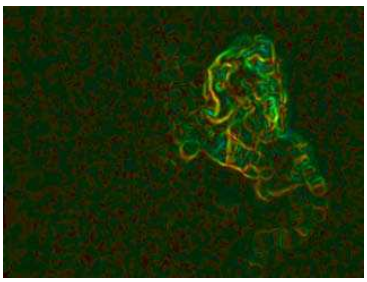


Shear

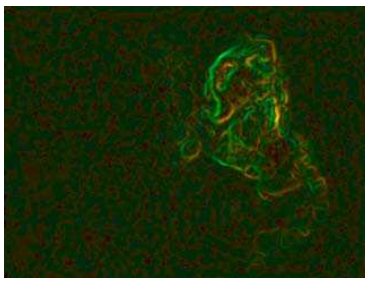
(b) Divergence, curl and shear computed from the same optical flow above.



Div-curl



Div-shear



Curl-shear

(c) Joint information captured by each of the 3 possible pairs of kinematic features.

Figure 5.7: Illustration of the information captured by the kinematic features and their 3 possible pair combinations. Divergence, curl and shear are scalar quantities, for rest, flow/orientation is indicated by color and magnitude by saturation. The example (*wave* action) is from HMDB51 [75].

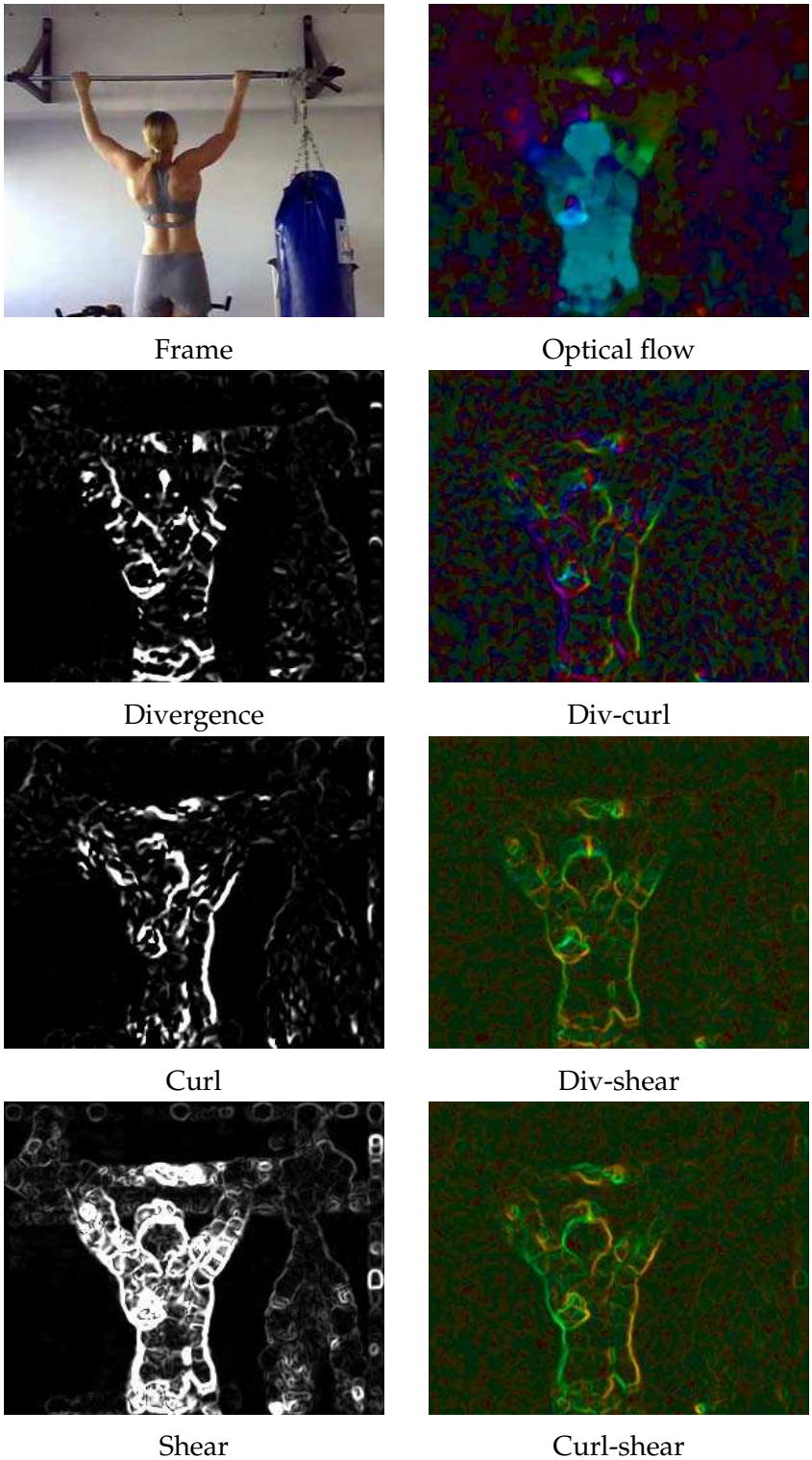


Figure 5.8: An example of *pullup* action from HMDB51 [75] to illustrate DCS. Divergence, curl and shear are scalar quantities, for images in the right column, flow/orientation is indicated by color and magnitude by saturation.

and $n_t = 3$. These parameters have been fixed for the sake of consistency with the other descriptors. For each pair of kinematic features, each cell in the grid is represented by a histogram. The resulting local descriptors have a dimensionality equal to $288 = n_x \times n_y \times n_t \times 8 \times 3$. At the video level, these descriptors are encoded into a single vector representation using either BOF or the higher-order encoding schemes introduced in the next section.

5.5 Higher-order representations: VLAD and Fisher Vector

In this section, we employ two higher-order encodings for aggregation of local features: VLAD [65] and Fisher vector [107, 109]. Below, we briefly introduce them and give the performance achieved for all the descriptors introduced along the previous sections.

VLAD. It is a descriptor encoding technique that aggregates the descriptors based on a locality criterion in the feature space. To our knowledge, this technique is first time considered for action recognition in our work [56]. Similar to BOF, VLAD relies on a codebook $C = \{c_1, c_2, ...c_k\}$ of k centroids learned by k-means. The representation is obtained by summing, for each visual word c_i, the differences $x - c_i$ of the vectors x assigned to c_i, thereby producing a vector representation of length $d \times k$, where d is the dimension of the local descriptors. We use the codebook size, $k = 256$.

Fisher vector. This encoding uses Gaussian Mixture Models (GMM) for vocabulary building. It captures the first and second order differences between the image descriptors and the centers of a GMM. We use the same codebook size as used for VLAD, *i.e.*, 256 Gaussians. We apply Principal Component Analysis (PCA) on the local descriptors and reduce the dimensionality by factor of two, as done in [109] . Fisher has extra d dimensions per Gaussian to add second order moments, therefore, the final representation is of $2 \times d/2 \times k$ dimensions.

Despite this large dimensionality, these representations are efficient because they are effectively compared with a linear kernel. Both of them are post-processed using a component-wise power normalization, which dramatically improves its performance [65]. While cross validating the parameter α involved in this power normalization, we consistently observe, for all the descriptors, a value between 0.15 and 0.3. Therefore, this parameter is set to $\alpha = 0.2$ in all our experiments. For classification, we use a linear SVM and *one-against-rest* approach everywhere, unless stated otherwise.

Impact on existing descriptors. These higher-order representations encode more information and hence are less sensitive to quantization parameters. This property is interesting in our case, because the quantization parameters involved in the local descriptors have been used unchanged in Section 5.3 for the sake of direct comparison. They might

Descriptor	Hollywood2			HMDB51		
	Fisher	VLAD	BOF	Fisher	VLAD	BOF
ω-Trajdesc	50.3%	45.5%	**51.4%**	**33.0%**	27.8%	32.9%
ω-HOG	**50.5%**	44.1%	45.6%	**37.4%**	28.9%	29.1%
ω-HOF	**57.7%**	53.9%	53.9%	**47.1%**	41.3%	38.6%
ω-MBH	**59.0%**	55.5%	52.5%	**48.0%**	43.3%	40.6%
ω-DCS	**56.5%**	52.5%	50.2%	**42.3%**	39.1%	35.8%
ω-DCS + ω-MBH	**59.3%**	56.1%	53.1%	**49.7%**	45.1%	41.2%
ω-Trajdesc + ω-HOG + ω-HOF	**61.9%**	59.6%	58.5%	**52.6%**	47.7%	45.6%

Table 5.5: Performance of VLAD with ω-Trajdesc, ω-HOG, ω-HOF, ω-DCS and ω-MBH descriptors and their combinations.

be suboptimal when using the ω-flow instead of the optical flow on which they have initially been optimized [145].

In Table 5.5, we compare these encodings with BOF. For all the descriptors VLAD improves over BOF and Fisher further improves over VLAD, with exception of ω-Trajdesc and ω-HOG. BOF performs better than VLAD for these two descriptors on both the datasets, while it just exceeds Fisher for ω-Trajdesc on Hollywood2. For all other cases, these encodings significantly outdo BOF, especially Fisher with boost of up to 7%.

Another thing to observe is that the gain is more for the descriptors having larger dimensionality. This is beneficial when combining different descriptors. Consequently, for the two combinations considered: (a) ω-MBH + ω-DCS and (b) ω-Trajdesc + ω-HOG + ω-HOF, VLAD beats BOF, even though BOF did better individually with lower dimensional descriptors. Improvement obtained by Fisher for these combinations is even larger, ranging +7-9% over BOF and around +4-5% over VLAD on HMDB51. We also observe that ω-DCS is complementary to ω-MBH and adds to the performance. Still DCS is probably not best utilized in the current setting of parameters.

5.5.1 Combining Trajectories

We have seen that with ω-descriptors results are boosted, this is due to effective separation of dominant motion and residual motion, *i.e.*, camera motion and action-related motion. However, as we already mentioned before, the camera motion also contains useful information and should not be thrown away. Here, we use this complementary information by combining trajectories from optical flow with ω-trajectories. Table 5.6 reports the results for Hollywood2 when: (i) optical flow is used for trajectory extraction and descriptor computation, (ii) ω-flow is used for description along ω-trajectories and (iii) the combination of the two. The results are reported for both VLAD and Fisher vector; Table 5.7 reports the same for HMDB51. The performance for each descriptor improves by combining the two types of trajectories, with both the encodings and on both

Descriptor	flow trajectories + flow		ω-trajectories + ω-flow		Combination	
	VLAD	Fisher	VLAD	Fisher	VLAD	Fisher
Trajectory	40.2%	44.5%	45.5%	50.3%	**48.2%**	**52.7%**
HOG	40.2%	48.4%	44.1%	50.5%	**44.5%**	**51.8%**
HOF	47.8%	52.2%	51.8%	56.3%	**54.2%**	**58.1%**
MBH	55.1%	58.5%	55.5%	59.0%	**56.8%**	**59.6%**
DCS	53.1%	55.3%	52.5%	56.5%	**54.7%**	**57.3%**
All five	59.6%	60.6%	62.0%	63.9%	**62.9%**	**64.6%**

Table 5.6: Combination of trajectories from optical flow and ω-trajectories with VLAD and Fisher aggreagation on Hollywood2 dataset.

Descriptor	flow trajectories + flow		ω-trajectories + ω-flow		Combination	
	VLAD	Fisher	VLAD	Fisher	VLAD	Fisher
Trajectory	24.6%	27.7%	27.8%	33.0%	**31.6%**	**35.6%**
HOG	27.0%	37.9%	28.9%	37.4%	**31.2%**	**41.4%**
HOF	33.7%	41.8%	38.5%	46.4%	**40.5%**	**47.8%**
MBH	43.4%	49.3%	43.3%	48.0%	**47.0%**	**50.6%**
DCS	39.0%	44.4%	39.1%	42.7%	**41.9%**	**45.6%**
All five	49.2%	52.9%	52.0%	55.4%	**52.6%**	**56.0%**

Table 5.7: Combination of trajectories from optical flow and ω-trajectories with VLAD and Fisher aggreagation on HMDB51 dataset.

the datasets. This shows the importance of the camera motion that is integrated with the optical flow.

5.6 Comparison with the state of the art

This section reports our results with all descriptors combined and compares our method with the state of the art.

Descriptor combination. Table 5.8 reports the results obtained when the descriptors are combined. Since we use Fisher, our baseline is updated that is combination of Trajectory, HOG, HOF and MBH with Fisher vector representation. When DCS is added to the baseline there is an improvement of 0.6% and 1.1% for Hollywood2 and HMDB51 respectively. With combination of all five compensated descriptors we obtain 63.8% and 54.8% on the two datasets. This is a large improvement even over the updated baseline, which shows that the proposed motion compensation and the way we exploit it are significantly important for action recognition. When descriptors computed using both types of trajectories are combined as explained in the subsection 5.5.1, there is further increase. Finally, we reach **64.6%** and **56.0%** with all five descriptors (64.2% and 55.4% without DCS descriptor).

Combination	Hollywood2	HMDB51
Trajectory+HOG+HOF+MBH	60.0%	51.8%
Trajectory+HOG+HOF+MBH+DCS	60.6%	52.9%
ω-Trajdesc +ω-HOG +ω-HOF +ω-MBH +ω-DCS	63.8%	54.8%
All but DCS descriptor with combination of trajectories	64.2%	55.4%
All 5 descriptors with combination of trajectories	**64.6%**	**56.0%**

Table 5.8: Different combinations descriptors and trajectories using Fisher representation.

The comparison with the state of the art is shown in Table 5.9. In Jain *et al.* [56], our approach outperformed all the previously reported results in the literature. In particular, on the HMDB51 dataset, the improvement over the best reported results till then was more than 11% in average accuracy. More recently, new methods were proposed [103, 147, 164], which yielded even better results. Approach of Wang *et al.* [147] is based on the same notion as our ω-trajectories and ω-flow, *i.e.*, to compensate for camera motion. Their camera motion estimation is based on estimating homography using RANSAC between two consecutive frames. To match feature points they use SURF descriptors in addition to dense optical flow. The inconsistent matches due to human motion are removed by human detection for better camera motion estimation. They use Fisher vector to aggregate local descriptors.

In this chapter, with Fisher vector representation and combination of both optical flow trajectories and ω-trajectories, we have further improved our results. Both these additions to our approach [56] have boosted our results to match the best published result till date of Wang *et al.* [147] on this two datasets. We include one extra descriptor, DCS and at the same time do not use human detection. So if we compare our approach without DCS and method in [147] without human detection, there is not much difference between the two. Our method leads by 1.2% on Hollywood and trails by 0.5% on HMDB51.

On Olympic Sports dataset we obtain mAP of **85.2%**. The best reported mAPs on this dataset are by Liu *et al.* [84] (74.4%), Jiang *et al.* [66] (80.6%) and recently by Wang *et al.* [147] (**91.1%**). Brendel *et al.* [17] and Gaidon *et al.* [44] obtained average accuracy of 77.3% and 82.7% respectively. Our method performs better than all these methods with notable exception of improved trajectories of Wang *et al.* [147]. The main reason is that their motion compensation involves warping the second frame according to the camera motion estimation and then recomputing the optical flow for each pair of consecutive frames. This is better suited for MBH descriptor as it is computed from gradient of flow where the constant motion is canceled. As a result, our approach of direct canceling of dominant motion is not as effective though it is more efficient as we do not have to compute optical flow again.

Hollywood2		HMDB51	
Ullah *et al.* [138]	55.7%	Kuehne *et al.* [75]	22.8%
Wang *et al.* [145]	58.3%	Sadanand *et al.* [120]	26.9%
*Vig *et al.* [143]	60.0%	Orit *et al.* [72]	29.2%
Jiang *et al.* [66]	59.5%	*Jiang *et al.* [66]	40.7%
Jain *et al.* [56]	62.5%	Jain *et al.* [56]	52.1%
Zhu *et al.* [164]	61.4%	Zhu *et al.* [164]	54.0%
Oneata *et al.* [103]	63.3%	Oneata *et al.* [103]	54.8%
Wang (w/o HD) *et al.* [147]	63.0%	Wang (w/o HD) *et al.* [147]	55.9%
Wang (with HD) *et al.* [147]	64.3%	Wang (with HD) *et al.* [147]	**57.2**%
Our Method (w/o DCS)	64.2%	Our Method (w/o DCS)	55.4%
Our Method (with DCS)	**64.6%**	Our Method (with DCS)	56.0%

Table 5.9: Comparison with the state of the art on Hollywood2 and HMDB51 datasets. *Vig *et al.* [143] gets 61.9% by using external eye movements data. *Jiang *et al.* [66] used one-vs-one multi class SVM while our and other methods use one-vs-rest SVMs. With one-against-one multi class SVM we obtain 45.1% for HMDB51. 'HD' is for human detection.

5.7 Conclusions

This chapter first demonstrates the interest of canceling the dominant motion (predominantly camera motion) to make the computed image motion truly related to actions, for both the trajectory extraction and descriptor computation stages. It produces significantly better versions (called compensated descriptors) of several state-of-the-art local descriptors for action recognition. The simplicity, efficiency and effectiveness of this motion compensation approach make it applicable to any action recognition framework based on motion descriptors and trajectories. The second contribution is the new DCS descriptor derived from the first-order scalar motion quantities specifying the local motion patterns. It captures additional information which is proved complementary to the other descriptors. Finally, we show that VLAD and Fisher encoding techniques instead of bag-of-words boost action descriptors, and overall exhibit a significantly better performance when combining different types of descriptors and trajectories. Our contributions are all complementary and lead to the state-of-the-art results when combined, as demonstrated by our extensive experiments on the Hollywood 2, HMDB51 and Olympic Sports datasets.

CHAPTER

SIX

Action localization with tubelets from motion

Recognizing actions in videos is an active area of research in computer vision. Because of the many fine grained spatio-temporal variations in action appearance the recognition performance of existing systems is far from that achieved in other visual tasks such as image search, face detection, or object recognition. The goal of action classification is to determine *which* action appears in the video. Temporal action detection estimates, additionally, *when* it occurs by providing a temporal interval. This chapter specifically considers the problem of action localization: the objective is to detect when *and* where action of interest occurs.

The expected output of such an action localization system is typically a subvolume encompassing the action of interest. This task can be seen as the counterpart of detection in images. Since a localized action only covers a fraction of the spatio-temporal volume in a video, the task is considerably more challenging than action classification and temporal detection.

One application of action localization is video surveillance, which considers controlled environments such as the airport videos considered in the Trecvid video-surveillance task [124]. In contrast, we consider action localization in uncontrolled environments as in sports videos, sitcoms, etc.

Making the parallel with object detection in images, we notice that there is a large body of literature that aims at bypassing the costly sliding window approach [144]. The general strategy is to limit the set of tested windows to an acceptable number by varying optimization strategies such as efficient sub-window search [76] (branch and bound search), objectness [2] and, more recently, a "Selective Search" strategy [137]. The latter generates a set of category-independent candidate windows by iteratively agglomerating superpixels based on various similarity criterions. It achieves, on average, a similar accuracy as that obtained by Deformable Part Models [41] (DPM), while drastically reducing the number of box hypotheses to be tested.

Most action localization systems are inspired by the aforementioned object detection strategies. For instance, Yuan *et al.* have extended the branch and bound approach to

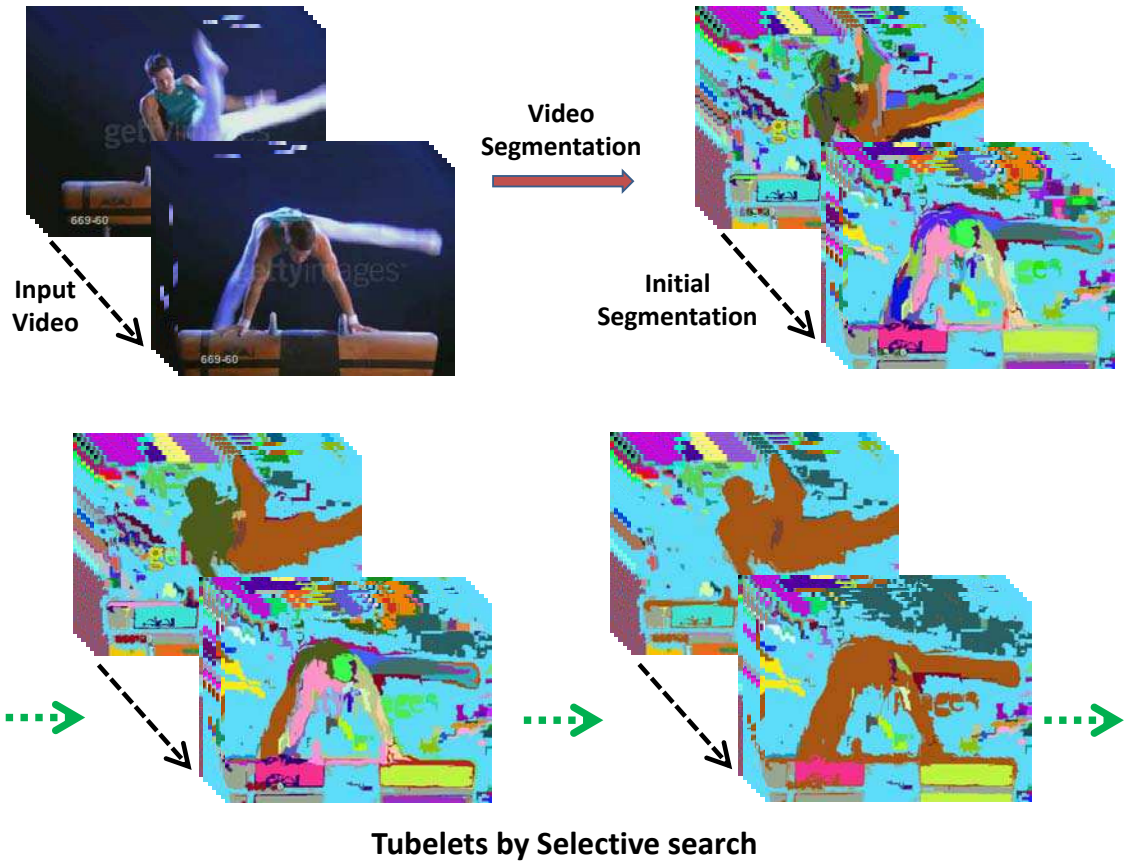


Figure 6.1: Overview of tubelets from motion: From an initial spatio-temporal segmentation in super-voxel, such as the one we propose based on motion, we produce additional super-voxels by merging them based on a criterion capturing the motion similarity. This produces a small set of tubelets, which is fed to a classifier.

videos [160], while Tian *et al.* [129] have proposed spatial-temporal DPM (SDPM). A noticeable exception is selective search [137]: To the best of our knowledge and despite its amenability to handle varying aspect ratios (in this respect, better than DPM), it has never been explored for videos.

Our first contribution is to investigate the selective search sampling strategy for videos. We adopt the general principle and extend it. First, we consider super-voxels instead of super-pixels to produce spatio-temporal shapes. This directly gives us 2D+t sequences of bounding boxes, referred to as *tubelets* in this chapter, without the need to address the problem of linking boxes from one frame to another, as required in other approaches [131, 132].

Our second contribution is explicitly incorporating motion information in various stages of the analysis. We introduce *independent motion evidence* as a feature to characterize how the action motion deviates from the background motion. By analogy to image descriptors such as the Fisher vector [107], we encode the singularity of the motion in a feature

vector associated with each super-voxel. First, motion is used as a merging criterion in the agglomerative stage of our sampling strategy. Second, motion is used as an independent cue to produce super-voxels partitioning the video.

Our approach offers several advantages. Firstly, we produce a small set of candidate tubelets. For a given complexity, this allows us to describe them with more expensive representations. Secondly, the bounding boxes are tailored to super-voxel shapes, which tends to improve the spatio-temporal adjustment of our bounding box sequences. As a result, we observe a consistent and significant gain over concurrent approaches for action localization. This is not surprising, as the image detection counterpart was recently shown to outperform DPM, as demonstrated in the VOC'2012 challenge [38]. However, our motion-based adaptation brings a large benefit, as shown by a comparison with a more naive adaptation of "selective search" to videos. The work presented in this chapter has been accepted to be published in [58].

This chapter is organized as follows. First, we give more details about the scientific context in Section 6.1. Section 6.2 describes the general principle of our tubelet sampling, while Section 6.3 describes how we incorporate motion. The experiments are reported in Section 6.4, and demonstrate the interest of our method on two public datasets, namely UCF Sports and MSR-II.

6.1 Related work

In this section, we present existing works into more details, in order to position our method with respect to the literature. Most references address recognition tasks in videos, but our work is also related to papers on object recognition, in particular object localization.

Action classification and localization. Current action recognition methods determine which action occurs in a video with good accuracy [17, 39, 56, 120, 145]. The task of localization is more demanding as it also requires to specify where the action happens in the video. This "location" is often expressed as a cuboid referred to as 'subvolume' [19, 129, 160]. Subvolume-based detection is inadequate in the case of complex actions, when the actor moves spatially or when the aspect ratio varies a lot like the one in Figure 6.2. Recently, "location" is more precisely defined as a sequence of bounding boxes [77, 132, 133]. The corresponding 2D+t volume, which we refer to as tubelet, tightly bounds the actions in the video space and provides a more accurate spatio-temporal localization of actions. However, the method considering this definition are more costly: The search space is significantly larger [132] than in subvolume-based localization. Therefore, it is critical to have a sub-sampling strategy for tubelets, as we propose in this chapter.

We have recently witnessed a trend for methods aiming at providing a more precise

localization, for instance for obtaining generic spatio-temporal human tracks [71] using a human detector and tracker. In another work [77], the detector and tracker are avoided by treating the actor location as a latent variable. Raptis *et al.* [116] selects trajectory groups that serve as candidates for the parts of an action. This localizes only parts of actions but this mid-level representation assists classification. In [131, 132], candidate bounding boxes are generated for each frame separately and then the optimal spatio-temporal path is found by Max-Path Search. One disadvantage of this approach is that it requires training a sliding window object detector, which is not only impractical on larger video datasets but is also unsuitable for very articulated poses with varying aspect ratios. Rather than considering a video as a set of images, we prefer to consider it as a spatio-temporal source from the very beginning.

The only methods we are aware of that uses representation similar to tubelets are by Trichet *et al.* [134] and Wang *et al.* [150]. Spatio-temporal tubes are proposed for video segmentation in [134]. Wang *et al.* [150] presented a representation for action localization based on the mutual information of feature trajectories towards the action class. They modeled human action as spatio-temporal tube of maximum mutual information.

Extensions from object localization. Many action localization approaches are inspired by box sampling strategies adapted from the object detection literature. The most popular is the sliding-window approach, extended to sliding-subvolume for actions [129]. Due to its considerable computational cost in object localization, not to mention its temporal extension to video, many works have attempted to circumvent sliding windows: Efficient subwindow search [76] finds the optimal bounding box in an image by a branch-and-bound strategy. It has inspired a spatio-temporal variant for action localization [160]. DPM [41] is another state-of-art approach that is extended to spatio-temporal action localization [129]. All these approaches or extensions only detect subvolumes and are not as precise as the tubelets considered here in our work. Moreover, these samplings need to be repeated for every new action considered in the video.

Rather than reducing the number of sliding windows, category-independent object detection has been proposed for object localization [1, 36, 88, 114]. The ”proposals” produced by these methods are 2D-locations likely to contain an object. This class of approaches was shown successful for salient object detection [42], weakly supervised object localization [29], and fully supervised object detection [137]. Category-independent proposals generate high-quality object proposals on static images. Finally and relatively different from the aforementioned methods, the object localization technique by Uijlings *et al.* [137] is suited for object categories with many articulated poses. It outperforms DPM for flexible categories such as cat, cow, dog, etc.

In this work, the goal is to generate flexible tubelets that are independent of the action category. Our approach is inspired by the object sampling of Uijlings [137], yet specifically considers the context of spatio-temporal localization in videos, *i.e.*, of action se-

quences (and not objects). In this context and as shown in later in the chapter, motion is a key feature and our method explicitly takes it into account when generating tubelets. Since actions are highly non-rigid, we use a flexible over-segmentation of the video into super-voxels. Super-voxels give excellent boundary recall [16, 155, 156] for nonrigid objects. Thus, in analogy of the 2D super-pixel methods used for static object proposals [1, 88, 137], we use super-voxels as the main mechanism to build video tubelets.

6.2 Action sequence hypotheses: Tubelets

This section describes our approach for iteratively sampling a set of candidate box sequences or *tubelets*. We generalize the Selective search [137] method from images to videos to delineate spatio-temporal action sequences. This generalization from 2D to 2D+t is not straight forward and involves updating certain aspects of the image-based techniques, such as relying on super-voxels instead of super-pixels. The extra dimension brings its own challenges, *e.g.*, due to camera-motion, scalability issues etc.

We first give a brief overview of the action localization pipeline. Then, we describe how tubelets are sampled iteratively. Finally, we focus on an important aspect of the technique, *i.e.*, the merging criteria and the video features upon which they are built. Later in Section 6.3, we further extend this approach by incorporating motion in two stages of the processing pipeline.

6.2.1 Overview of the action localization pipeline

1. **Super-voxel segmentation.** To generate the initial set of super-voxels, we first rely on a third-party Graph-based (GB) video segmentation method [155]. We choose GB over other segmentation methods in [155] because it is more efficient w.r.t. time and memory, *i.e.*, about 13 times faster than a slightly more accurate hierarchical version (GBH) [155]. This step produces n super-voxels, to which we associate n initial tubelets, obtained as the sequences of bounding boxes that tightly encompass the super-voxels.
2. **Iterative generation of additional tubelets.** This critical stage is detailed in Subsection 6.2.2. It consists of $n - 1$ iterations. Each merges two super-voxels into a new one. The choice of the two super-voxels to be merged in a given iteration depends on a similarity criterion that is specifically discussed in Subsection 6.2.3.
3. **Descriptor computation.** This step computes a bag-of-words (BOW) representation for each tubelet produced by the previous steps. As local descriptor we employ MBH [26] computed along the ω-trajectories [56].
4. **Classification step.** BOW histograms of tubelets are used for training a classifier per class.

6.2.2 Hierarchical sampling of tubelets

In this section, our objective is to produce additional tubelets from successive mergings of the super-voxels produced by the initial spatio-temporal segmentation. The algorithm is inspired by the selective search method proposed for image localization [137].

Super-voxel generation. We iteratively merge super-voxels in an agglomerative manner. Starting from the initial set of super-voxels, we hierarchically group them until the video becomes a single super-voxel. At each iteration, a new super-voxel is produced from two super-voxels, which are then not considered anymore in subsequent iterations.

Formally, we produce a hierarchy of super-voxels that are represented as a tree: The leaves correspond to the initial super-voxels while the internal nodes are produced by the merge operations. The root node is the whole video and the corresponding super-voxel is produced in the last iteration. Since this hierarchy of super-voxels is organized as a binary tree, it is straightforward to show that $n - 1$ additional super-voxels are produced by the algorithm.

Tubelets. In each frame where it appears, a super-voxel is tightly bounded by a rectangle. The temporal sequence of bounding boxes forms a tubelet. The hierarchical algorithm samples tubelets with spatial boxes at all scales and sequences of all possible lengths in time. Note that a tubelet is a more general shape than the cuboids considered in earlier works on action localization [77, 132, 133]. As the output of the algorithm, we have $2n - 1$ tubelets, $n - 1$ of which obtained from the new super-voxels and n from the initial segmentation.

The merge operation starts by selecting the two super-voxels to be merged. For this purpose, we rely on similarities computed between all the neighboring super-voxels that are still active. The similarity measure is discussed in Subsection 6.2.3. After the merge, we compute the new similarities between the resulting super-voxel and its neighbors.

Figure 6.2 illustrates the method on a sample video. Each color represents a super-voxel and after every iteration a new entry is added and two are removed. After 1000 iterations, observe that two tubelets (blue and dark green) emerge around the action of interest in the beginning and the end of the video, respectively. At iteration 1720, the two corresponding super-voxels are merged. The novel tubelet (dark green) resembles the ground-truth yellow tubelet. This exhibits the ability of our method to group tubelets both spatially and temporally. As importantly, it shows the capability to sample a tubelet with boxes having very different aspect ratios. This is unlikely to be coped by sliding-subvolumes or even approaches based on efficient sub-window search.

Figure 6.3 depicts another example, with a single frame considered at different stages of the algorithm. Here the initial super-voxels (second image) are spatially more decom-

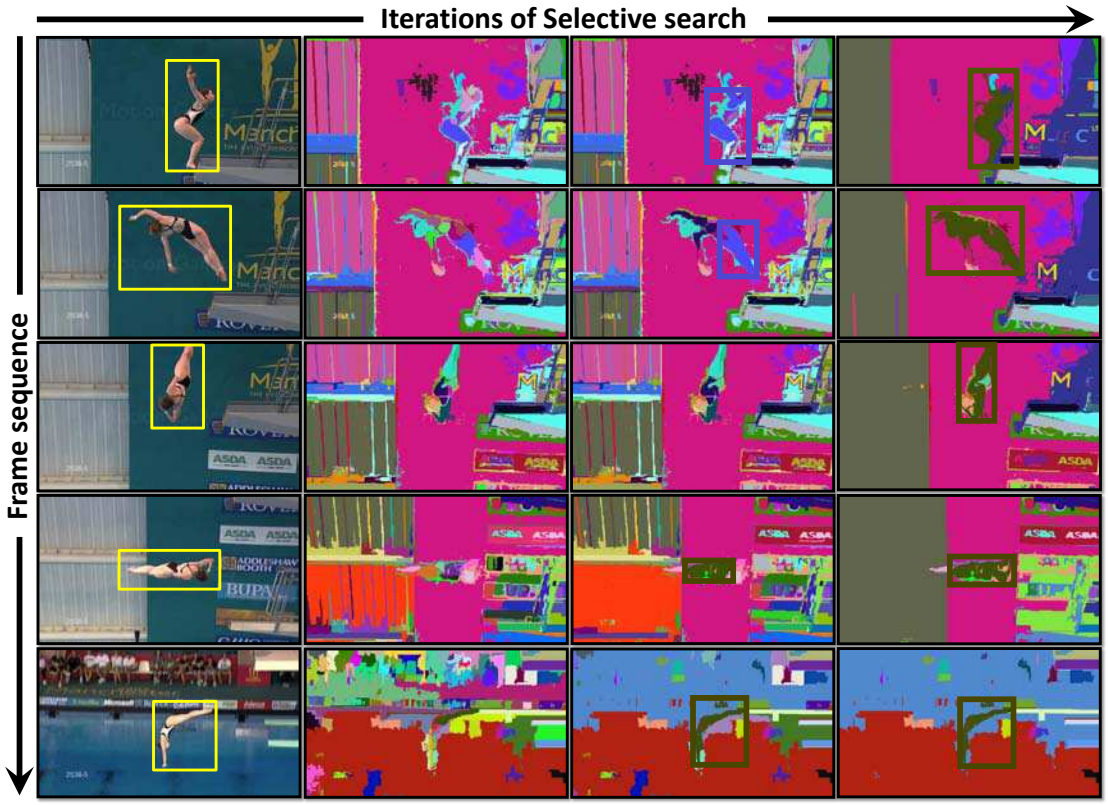


Figure 6.2: Illustration of hierarchical sampling of tubelets. *Left most.* A sampled sequence of frames (1^{st}, 15^{th}, 25^{th}, 35^{th}, 50^{th}) associated with action 'diving' from UCF-Sports dataset. The yellow bounding boxes represent the ground-truth tubelet. *Column 2* shows the initial video segmentation used as input to our method. *The last two columns* show two stages of the hierarchical grouping algorithm. A tubelet close to the action is also represented by bounding boxes in each column. Observe how close it is to the ground-truth tubelet in the last column despite the varying aspect ratios in different frames.

posed because the background is cluttered both in appearance and in motion (spectators cheering). Even in such a challenging case our method is able to group the super-voxels related to the action of interest.

6.2.3 Merging criteria: Similarity measures

We employ five complementary similarity measures to compare super-voxels, in order to select which should be merged. They are fast to compute. Four of these measures are adapted from selective search in image: The measures based on Color, Texture, Size and Fill were computed for super-pixels [137]. We revise them for super-voxels. As our objective is not to segment the objects but to delineate the action or actors, we additionally employ a motion-based similarity measure, encoding *independent motion evidence* (IME) to characterize a super-voxel.

Merging with color, texture and motion: s_C, s_T, s_M. These three similarity measures are computed in a similar manner: They describe each super-voxel with a histogram and for comparison between two super-voxels histogram intersection is used. Though the method of similarity computation is the same, they differ in the way the histograms are computed from different characteristics of a given super-voxel:

- The color histogram h_C captures the HSV components of the pixels included in a super-voxel;
- h_T encodes the texture or gradient information of a given super-voxel;
- The motion histogram h_M is computed from our IME feature, which is detailed in Section 6.3 devoted to motion.

Apart from being computed on super-voxels instead of super-pixels, h_C and h_T are identical to the histograms considered for selective search in images [137]. Please refer to this prior work for details.

As the process of merging is the same for the histograms, let us generically denote one of them by h. We compute a ℓ_1-normalized histogram h_i for each super-voxel r_i in the video. Two histograms h_i and h_j are compared with histogram intersection, $s = \delta_1(h_i, h_j)$. The histograms are efficiently propagated through the hierarchy of super-voxels. Denoting $r_t = r_i \cup r_j$, the super-voxel obtained by merging the super-voxels r_i and r_j, we have

$$h_t = \frac{\Gamma(r_i) \times h_i + \Gamma(r_j) \times h_j}{\Gamma(r_i) + \Gamma(r_j)} \tag{6.1}$$

where $\Gamma(r)$ denotes the number of pixels in super-voxel r. The size of the new super-voxel r_t is $\Gamma(r_t) = \Gamma(r_i) + \Gamma(r_j)$.

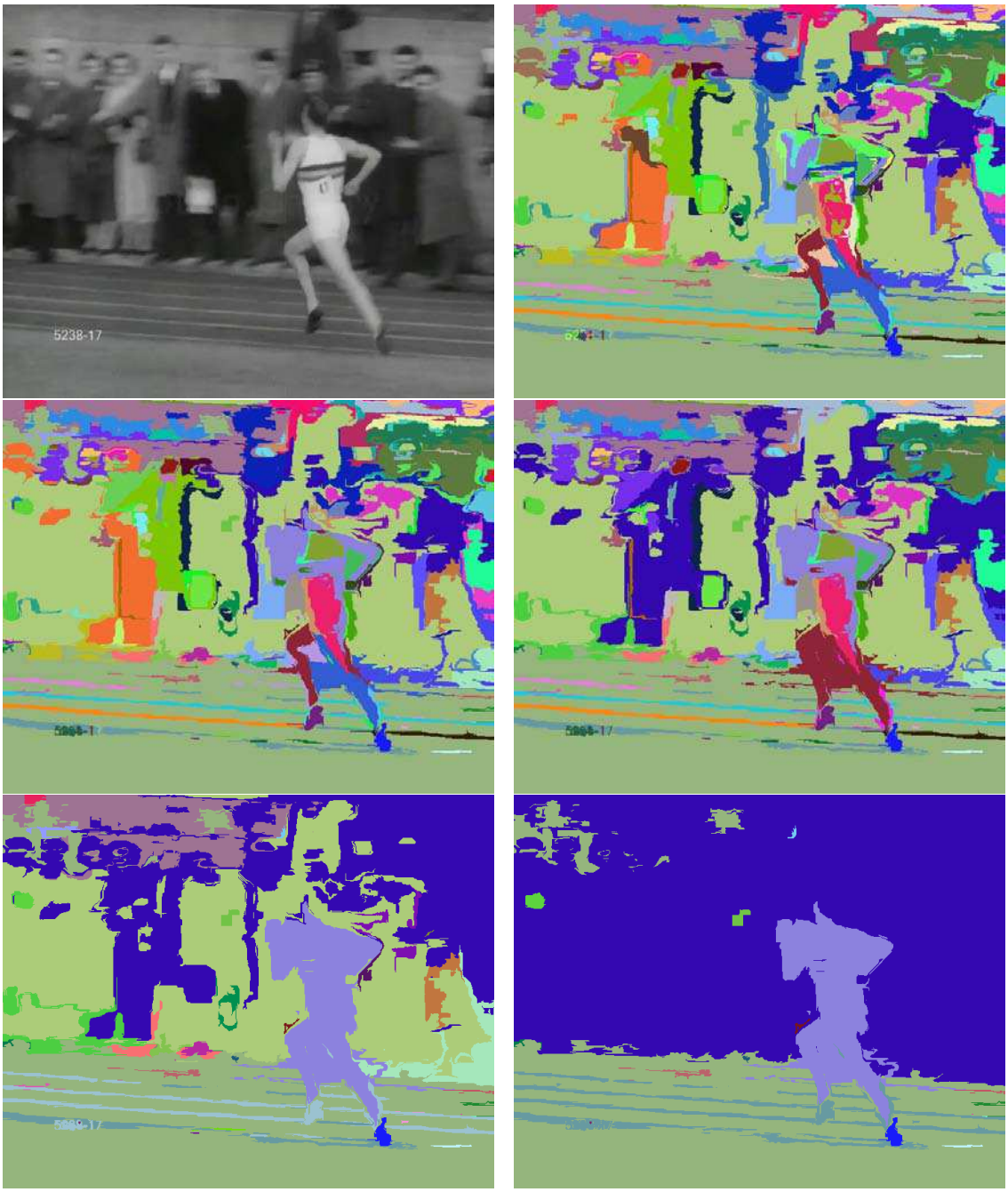


Figure 6.3: Example from action 'Running': The two images on the top depict a video frame and the initial super-voxel segmentation used as input of our approach. The next four images represent the segmentation after a varying number of merge operations.

Merging criterions based on size and fill: s_Γ, s_F. The similarity $s_\Gamma(r_i, r_j)$ aims at merging smaller super-voxels first:

$$s_\Gamma(r_i, r_j) = 1 - \frac{\Gamma(r_i) + \Gamma(r_j)}{\Gamma(\text{video})} \tag{6.2}$$

where $\Gamma(\text{video})$ is the size of the video (in pixels). This tends to produce super-voxels and therefore tubelets of varying sizes in all parts of the video (recall that we only merge contiguous super-voxels).

The last merging criterion s_F measures how well super-voxels r_i and r_j fit into each other. We define $B_{i,j}$ to be the tight bounding cuboid enveloping r_i and r_j. The similarity is given by

$$s_F(r_i, r_j) = \frac{\Gamma(r_i) + \Gamma(r_j)}{\Gamma(B_{i,j})}. \tag{6.3}$$

Merging strategies

Merging strategy can be one of the above discussed merging criteria (or similarity measures) or it can be a combination of them. For instance, merging can be done based on only color similarity (s_C) or motion similarity (s_M); or it can be done using sum of color, motion and fill similarities ($s_C + s_M + s_F$). Each merging strategy has a corresponding hierarchy, starting from n super-voxels, it leads to a set of new $n - 1$ super-voxels. In Section 6.4.1, we experiment with various strategies, *i.e.*, combinations of the merging criteria discussed. The best strategies that yield quality hypotheses using reasonable number of tubelets are selected. The combined hierarchies of tubelets from these strategies are used as the final hypotheses.

6.3 Motion features

Different ways of exploiting motion information could be envisaged. Since we are concerned with action localization, we need to aggregate super-voxels corresponding to the action of interest, *i.e.*, points that deviate from the background motion due to camera motion. We can assume that usually later is dominant motion in the image frame. The dominant (or global) image motion can be represented by a 2D parametric motion model. Typically, an affine motion model of parameters $\theta = (a_i)$, $i = 1...6$, or a quadratic model (equivalent to homography) with 8 parameters can be used, depending on the type of camera motion and of the scene layout likely to occur:

$$\begin{aligned} w_\theta(p) =& (a_1 + a_2 x + a_3 y, a_4 + a_5 x + a_6 y) \\ \text{or } w_\theta(p) =& (a_1 + a_2 x + a_3 y + a_7 x^2 + a_8 xy, \\ & a_4 + a_5 x + a_6 y + a_7 xy + a_8 y^2), \end{aligned}$$

where $w_\theta(p)$ is the velocity vector supplied by the motion model at point $p = (x, y)$ in the image domain Ω. In this chapter, we use affine motion model for all the experiments.

6.3.1 Evidence of independent motion

First, we formulate the evidence that a point $p \in \Omega$ undergoes an independent motion (*i.e.*, an actor related motion) at time step t. Let us introduce the displaced frame difference at point p and at time step t for the motion model of parameter θ : $r_\theta(p,t) = I(p+w_\theta(p), t+1) - I(p,t)$. Here, $r_\theta(p,t)$ represents the background motion due to camera motion. To simplify notation, we drop t when there is no ambiguity. At every time step t, the global parametric motion model can be estimated with a robust penalty function as

$$\hat{\theta} = \arg\min_\theta \sum_{p\in\Omega} \rho(r_\theta(p,t)), \tag{6.4}$$

where ρ is the robust function. To solve (6.4), we use the publicly available software Motion2D [101], where $\rho(.)$ is defined as the Tukey function. $\rho(r_\theta)$ produces a maximum likelihood type estimate: the so-called M-estimate [52]. Indeed, if we write $\rho(r_\theta) = -\log f(r_\theta)$ for a given function f, $\rho(r_\theta)$ supplies the usual maximum likelihood (ML) estimate. Since we are looking for action related moving points in the image, we want to measure *the deviation* to the global (background) motion. This is in spirit of the Fisher vectors [107], where the deviation of local descriptors from a background GMM model is encoded to produce an image representation.

Let us consider the derivative of the robust function $\rho(.)$. It is usually denoted as $\psi(.)$ and corresponds to the influence function [52]. More precisely, the ratio $\psi(r_\theta)/r_\theta$ accounts for the influence of the residual r_θ in the robust estimation of the model parameters. The higher the influence, the more likely the point conforms to the global motion. Conversely, the lower the influence, the less likely the point approves to the global motion. This leads to define the *independent motion evidence* (IME) as

$$\xi(p,t) = 1 - \varpi(p), \tag{6.5}$$

where $\varpi(p)$ is the ratio $\frac{\psi(r_{\hat{\theta}}(p,t))}{r_{\hat{\theta}}(p,t)}$ normalized within $[0,1]$.

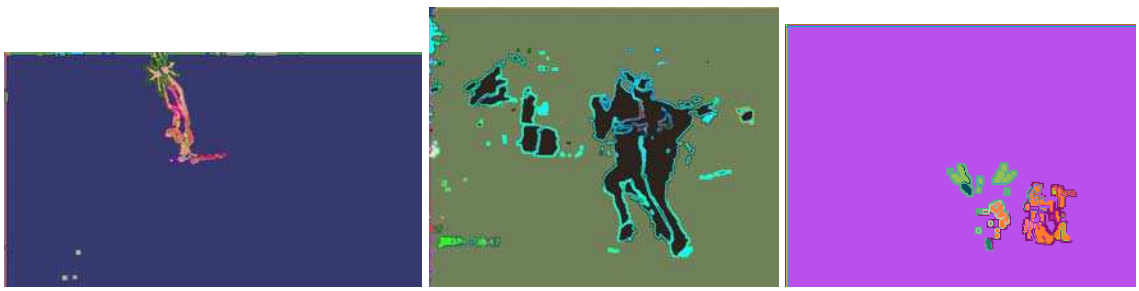
6.3.2 Motion for segmentation

Each frame can be represented with the IME of pixels, $\xi(p,t)$. Figure 6.4(b) shows IMEs of frames in Figure 6.4(a). In practice, we scale the range of values from $[0,1]$ to $[0,255]$ and quantize the values into integer values. We post-process these IMEs of frames by applying morphological operations to obtain binary images. These binary images are applied as masks on the corresponding IME frames to obtain denoised IME maps, displayed in Figure 6.4(c). Applying GB video segmentation on sequences of these denoised maps partitions the video into super-voxels with independent motion. Therefore, we use it as an alternative for producing our initial super-voxels (Step 1 in Section 6.2). A few examples of results obtained by applying GB on IME maps are shown in Figure 6.4(d).

(a) Video frames.

(b) IMEs of frames.

(c) IME maps



(d) IME segmentation (using GB)

Figure 6.4: IME maps for motion feature and segmentation: First two rows show the original frames and their IMEs. The IME maps and the result of applying GB video segmentation on them are shown in third and fourth rows.

Thus resulting tubelets are more adapted to the action sequences than the ones obtained by applying GB on the original frames, as evaluated in Section 6.4.1.

Figure 6.4 illustrates the process with three examples of different types of action from UCF-Sports and MSR-II datasets. The first column shows a frame from action "Swing-Bench". Here the action of interest is highlighted by IME map itself and then clearly delineated by segmenting IME maps. Second column shows an example from action "Running", though the segmentation does not give an ideal set of initial super-voxels but the IME map has useful information to be exploited by our motion feature based merging criterion. An example of "HandWaving" from MSR-II dataset is shown in the last column. In spite of clutter and illumination variations IME map successfully highlights the action.

6.3.3 Motion feature as merging criterion

In this subsection, we define a super-voxel representation for IME maps capturing the relevant information and efficient enough. This representation is the histogram h_{M} involved in the merging criterion s_{M} mentioned in Section 6.2. We consider the binarized version of IME maps, *i.e.*, the binary images that resulted from morphological operations. At every point p, we evaluate the number of points q (including p) in its 3D neighborhood that are set to one. In a subvolume of $5 \times 5 \times 3$ pixels, this count value ranges from 0 to 75. The motion histogram $h_{\mathsf{M}i}$ of these values is computed over the super-voxel r_i. Intuitively, this histogram captures both the density and the compactness of a given region with respect to the number of points belonging to independently moving objects.

6.4 Experiments

We evaluate our approach on benchmarks: UCF-Sports [118] and MSR-II [19]. The first dataset consists of sports broadcasts with realistic actions captured in dynamic and cluttered environments. This dataset is challenging due to many actions with large displacement and intra-class variability. MSR-II contains videos of actors performing actions (handwaving, handclapping and boxing) in complex environments. It is suitable for cross-dataset experiment. As a standard practice, we use the KTH dataset for training.

We first evaluate the quality of tubelet hypotheses generated by our approach. Then, Section 6.4.2 details our localization pipeline and compares our results with the state-of-the-art methods on the two datasets.

Merging Strategy	Video Segmentation			IME Segmentation		
	MABO	MaxRecall ($\sigma = 0.6$)	# Tubelet	MABO	MaxRecall ($\sigma = 0.6$)	# Tubelet
Initial voxels	0.362	0.044	862	0.486	0.280	118
M (s_M)	0.562	0.432	299	0.529	0.357	90
C (s_C)	0.473	0.2428	483	0.511	0.351	93
T (s_T)	0.446	0.2336	381	0.512	0.388	81
S (s_Γ)	0.478	0.2352	918	0.522	0.352	158
F (s_F)	0.509	0.3073	908	0.527	0.388	155
M+S+F	0.572	0.4979	719	0.542	0.403	129
T+S+F	0.526	0.3402	770	0.539	0.463	145
C+T+S+F	0.534	0.3844	672	0.545	0.452	127
M+C+T+S+F	0.581	0.4860	656	0.551	0.415	122
Strategy set I	0.615	0.5821	2346	0.566	0.483	469
Strategy set II	0.620	0.5888	3253	0.568	0.495	625

Table 6.1: Mean Average Best Overlap for tubelet hypotheses using variety of segmentation strategies from UCF-Sports train set. [M:Independent motion evidence, C: Color, T: Texture, S: Size, F: 3D Fill, Strategy set I: {M, M+S+F, C+T+S+F, M+C+T+S+F}, Strategy set II: {M, F, M+S+F, C+T+S+F, M+C+T+S+F}].

6.4.1 Evaluation of tubelet quality

To evaluate the quality of our tubelet hypotheses, we compute the upper bound on the localization accuracy, as previously done to evaluate the quality of object hypotheses [137], by the Mean Average Best Overlap (MABO) and maximum possible recall. In this subsection, we extend these measures to videos. This requires measuring the overlap between two sequences of boxes instead of boxes.

Localization score. In a given video V of F frames comprising m instances of different actions, the i^{th} groundtruth sequence of bounding boxes is given by $gt^i = (B_1^i, B_2^i, ...B_F^i)$. If there is no action of i^{th} instance in frame f, then $B_1^i = \emptyset$. From the tubelet hypotheses, the j^{th} tubelet formed by a sequence of bounding boxes is denoted as, $dt^j = (D_1^j, D_2^j, ...D_F^j)$. Let $OV_{i,j}(f)$ be the overlap between the two bounding boxes in frame, f, which is computed as "intersection-over-union". The localization score between groundtruth tubelet gt^i and a tubelet dt^j is given by:

$$S(gt^i, dt^j) = \frac{1}{|\Gamma|} \sum_{f \in \Gamma} OV_{i,j}(f), \tag{6.6}$$

where Γ is the set of frames where at least one of B_f^i, D_f^j is not empty. This criterion generalizes the one proposed by Lan *et al.* [77] by taking into account the temporal axis. An instance is considered as localized or detected if the action is correctly predicted by the classifier and also the localization score is enough, *i.e.*, $S(gt^i, dt^j) > \sigma$, the threshold for localization score.

Mean Average Best Overlap (MABO). The Average best overlap (ABO) for a given class c is obtained by computing, for each groundtruth annotation $gt^i \in G^c$, the best localization from the set of tubelet hypotheses $T = \{dt^j | j = 1 \dots m\}$:

$$\text{ABO} = \frac{1}{|G^c|} \sum_{gt^i \in G^c} \max_{dt^j \in T} S(gt^i, dt^j). \tag{6.7}$$

The mean ABO (MABO) synthesizes the performance over all the classes. Note however that adding more hypotheses necessarily increases this score. So, MABO must be considered jointly with the number of hypotheses.

Maximum possible recall (*MaxRecall*). Another measure for quality of localization used for images is maximum possible recall. It is an upper bound on the recall with the given tubelet hypotheses. Along with MABO, we also compare different merging strategies using MaxRecall with a very stringent localization threshold, $\sigma = 0.6$.

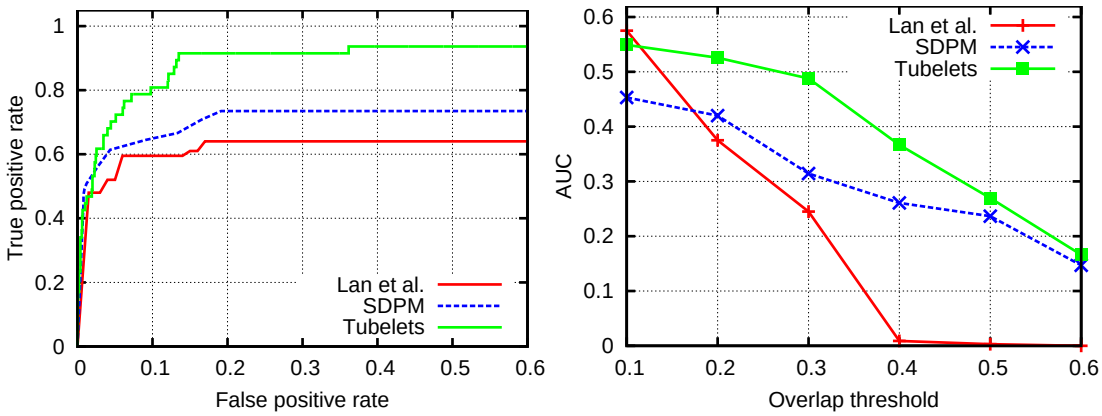
Table 6.1 reports the MABO, MaxRecall (at $\sigma = 0.6$) and the average number of tubelets for the train-set of UCF-Sports dataset. Different strategies are compared for the two methods considered for initial segmentation (regular GB, and GB on IME). With regular GB segmentation, the best hypotheses are clearly produced by the strategies that include our s_M merging criterion: they attain the highest MABO and MaxRecall with the small number of tubelets. Many combinations of strategies were tried and the two best sets of strategies were chosen (described in Table 6.1). For the first chosen set, we achieve MABO=0.615 and MaxRecall=58% with only 2346 tubelets per video. Considering that the localization score threshold (σ) used in literature is 0.2, these MABO values are very promising.

The GB segmentation applied on our IME de-noised maps (See Section 6.3) generates a very good initial set (MABO = 0.486). The MABO and specially MaxRecall further improve for all the strategies. Although the best values obtained, MABO=0.568 and MaxRecall=0.495, are lower than those for the original video segmentation, the number of tubelets is only 625 on an average. This is very useful for large videos where the number of samples, by sliding-subvolume or even by segmentation, is substantially higher.

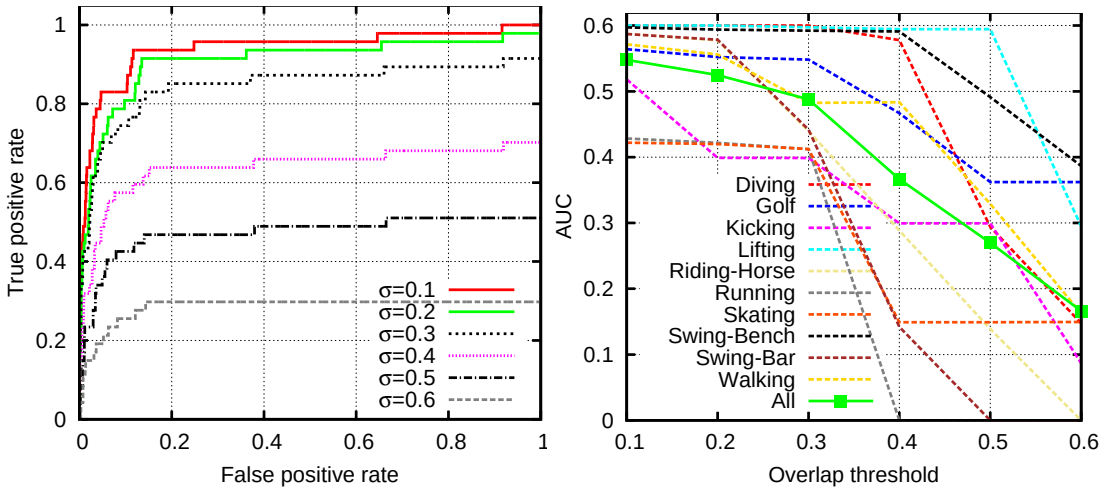
The combinations considered in the rest of this section are following. For regular GB segmentation, MABO and MaxRecall are similar for both the sets, so we choose strategy set I as it needs lesser number of tubelets. With segmentation of IME maps, we choose strategy set II for its higher MaxRecall.

6.4.2 Action localization

We, now, evaluate our tubelet hypotheses for action localization. With a relatively small number of candidate locations, our approach enables the use of expensive and powerful Bag-of-words based representation with large vocabulary sizes. We first extract



(a) Comparison with concurrent techniques [77, 129] on UCF-Sports. Left: ROC at σ=0.2, Right: AUC for σ from 0.1 to 0.6.



(b) Left: complete ROCs are shown for σ from 0.1 to 0.6. Right: class-wise performance for each class of UCF-Sports is shown, solid green curve shows AUC on an average.

Figure 6.5: Evaluation of UCF-Sports dataset.

state-of-art MBH and HOF descriptors computed along ω-trajectories [56][1]. Recently proposed, ω-flow is obtained by compensating dominant motion from optical flow, and ω-trajectories are computed using ω-flow. We prefer using ω-trajectories over trajectories from optical flow [145] because they are more active on the actors, and also fewer trajectories are produced with ω-flow. To represent a tubelet, we aggregate all the visual words corresponding to the trajectories that pass through it. For training, we use a one-vs-rest SVM classifier with Hellinger (square-rooting+linear) kernel.

Experiments on UCF-Sports. This dataset consists of 150 videos with actions extracted from sports broadcasts. 10 action categories are represented, for instance "diving", "swinging-bench", "horse-riding", etc. We use the disjoint train-test split suggested by

[1]We use the code available online:
`http://www.irisa.fr/texmex/people/jain/w-Flow`

Lan *et al.* in [77]. The ground truth is provided as sequences of bounding boxes enclosing the actors. For training, we use the groundtruth tubelets and the tubelets provided by our method that have localization score greater than 0.7 with the groundtruth. Negative samples are randomly selected by considering tubelets whose overlap with ground truth is less than 0.2. We set the vocabulary size to $K = 500$ for Bag-of-words and use spatial pyramid [81] with five cells (1x1+2x2). Initial super-voxels are obtained by the GB segmentation performed on the original videos and strategy set I is used (Table 6.1).

For evaluating the quality of action localization, we follow the criteria explained in [77] and described in Section 6.4.1. Following previous works, we compare using the ROC curves and its AUC in Figure 6.5(a). On the left, we plot the ROC curve with $\sigma = 0.2$. In order to be consistent with SDPM and Lan *et al.*, we stop at FPR=0.6 and compute the AUC only for this part. We report AUCs for thresholds ranging from 0.1 to 0.6 on right of Figure 6.5(a).

As can be seen from these figures, our approach significantly outperforms both the methods. Figure 6.5(b), on left, shows the complete ROC curves with different thresholds. We have almost total recall for $\sigma \leq 0.2$ and even for $\sigma = 0.5$ (a more strict criterion commonly used for object localization), our recall is around 50%. We also report AUC for each action class from UCF-Sports on the right of Figure 6.5(b).

Method	Boxing	Handclapping	Handwaving
Cao *et al.* [19]	0.1748	0.1316	0.2671
SDPM [129]	0.3886	0.2391	0.4470
Tubelets	0.4600	0.3141	**0.8579**

Table 6.2: Average precisions for MSR-II

Experiments on MSR-II. This dataset consists of 54 videos recorded in crowded environment, with many people moving in the background. Each video may contain one or more of three types of actions: boxing, handclapping and handwaving. An actor appears, performs one of these actions, and walks away. A single video has multiple actions (5-10) of different types, making the temporal localization challenging. Bounding subvolumes or cuboids are provided in the ground-truth. Since the actors do not change their location, it is as good as a sequence of bounding boxes. The localization criterion is subvolume-based, so we follow Cao *et al.* [19] and use the tight subvolume or cuboid enveloping tubelet. Precision-recall curves and average precision (AP) is used for evaluation [19]. Since MSR-II videos are much larger than UCF-Sports videos, to keep the number of tubelets low, we use the initial super-voxels from the GB segmentation of the IME maps along with strategy set II.

This dataset is designed for cross-dataset evaluation. Following standard practice, we train on KTH dataset and test on MSR-II. While training for one class, the videos from other two classes are used as negative set. We compare with Cao *et al.* [19] and

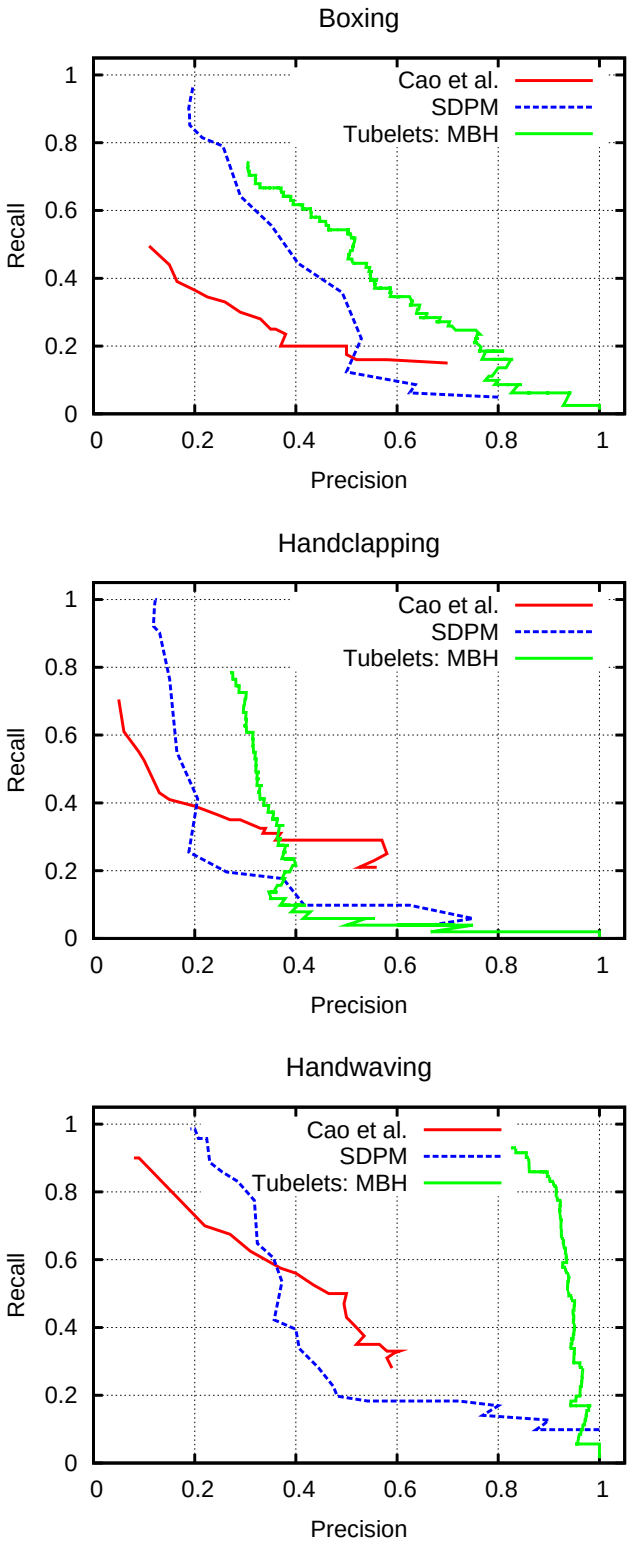


Figure 6.6: Precision/recall: Comparison [19, 129] for the 3 classes on MSR-II.

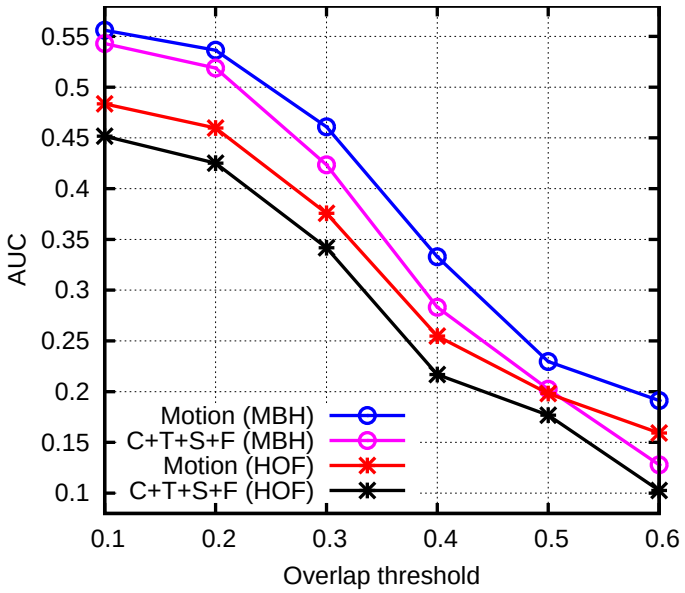


Figure 6.7: Motion Vs Color+Texture+Size+Fill (UCF-Sports): Comparison using MBH and HOF descriptors, performance measured by AUC for σ from 0.1 to 0.6.

Strategy (Descriptor)	Boxing	Handclapping	Handwaving
Motion (MBH)	0.3738	0.2579	**0.8759**
C+T+S+F (MBH)	0.3953	0.2402	0.8416
Motion (HOF)	**0.4046**	0.3280	0.8077
C+T+S+F (HOF)	0.4033	**0.4320**	0.7907

Table 6.3: Motion Vs Color+Texture+Size+Fill (MSR-II): Comparison using MBH and HOF descriptors, and average precision as measure. Note since MSR videos are much larger compared to UCF-Sports, IME maps are used for segmentation which involve motion information also. The average number of tubelets per video for Motion and C+T+S+F are 402 and 506 respectively.

SDPM [129] in Figure 6.6. Table 6.2 shows that our tubelets significantly outperform the two other methods for the three classes.

Impact of sampling strategies and descriptors

Quality of tubelets from various strategies is evaluated using MABO measure in Table 6.1. Here we compare, for localization performance, strategy with only motion as a merging criterion against strategy that combines the other four criteria: Color+Texture+Size+Fill. We also use HOF descriptors along with MBH to show that the conclusions drawn with MBH descriptors are also valid with HOF. The results for UCF-Sports are shown in Figure 6.7 and for MSR-II in Table 6.3 and Figure 6.8. Motion based criterion performs better than the other 4 combined for UCF-Sports (Figure 6.7). In case of MSR-II, motion-based strategy is not better, this is because for MSR-II IME maps are used for segmentation and hence motion information is already included.

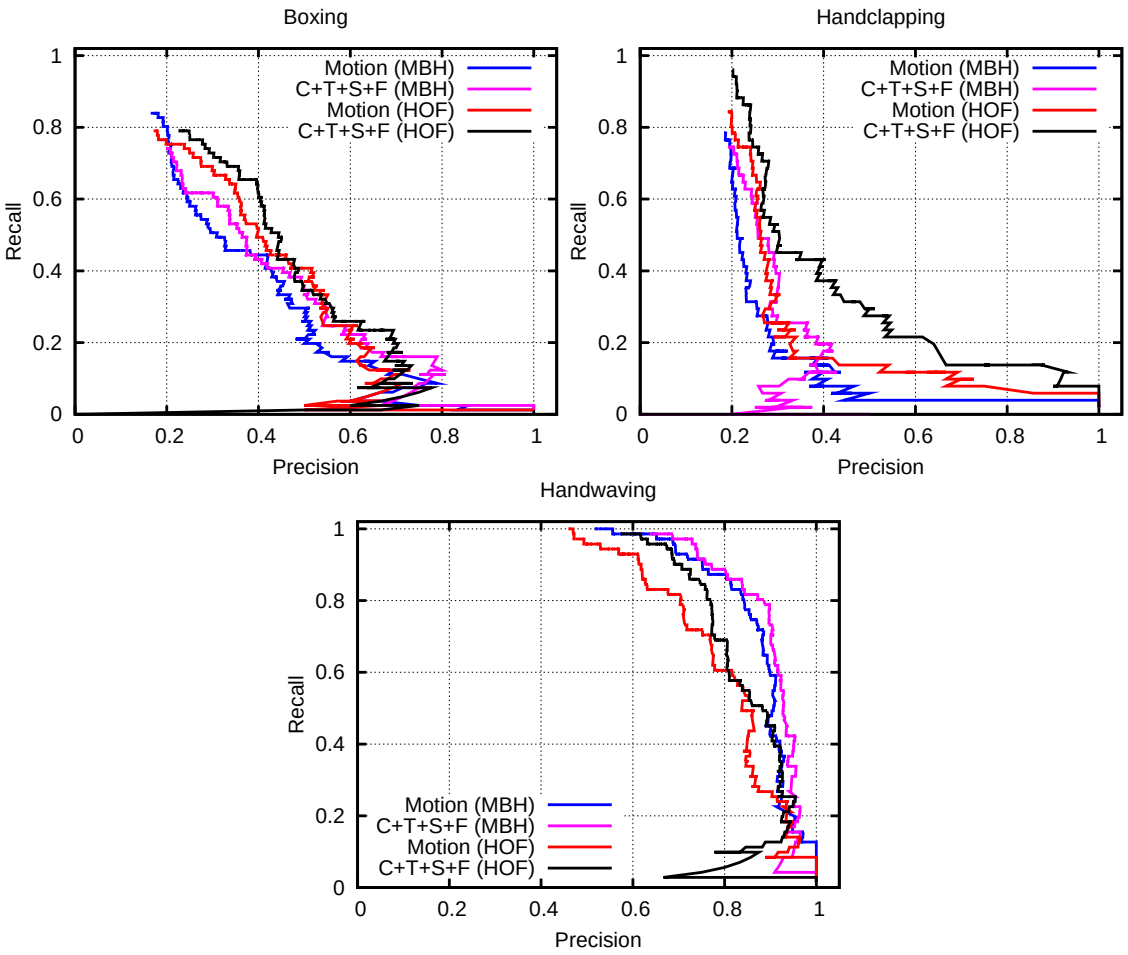


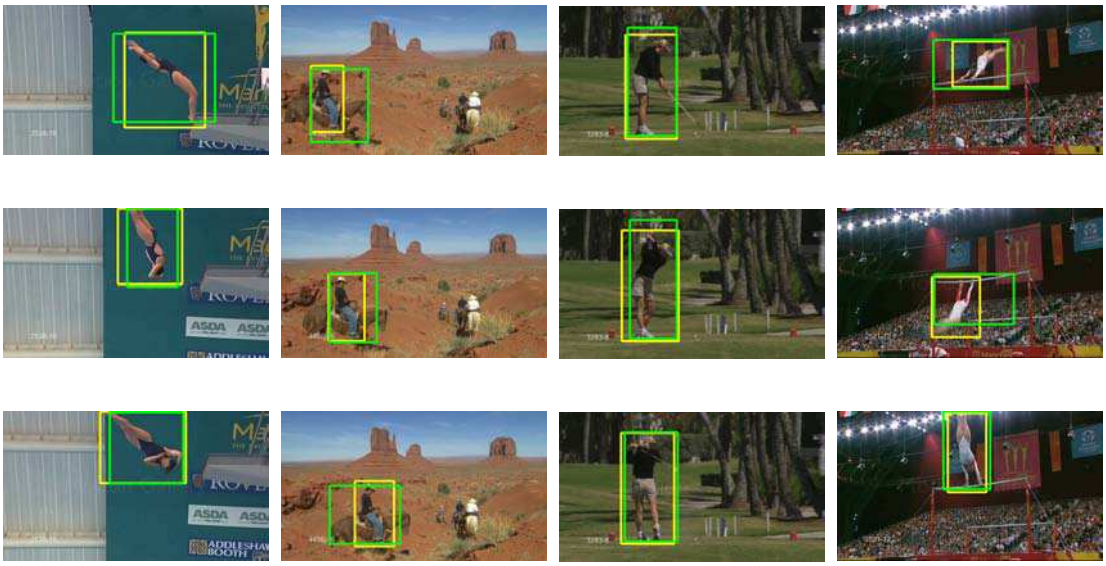
Figure 6.8: Motion Vs Color+Texture+Size+Fill (MSR-II): Precision/recall curves for the three classes.

Localization Examples

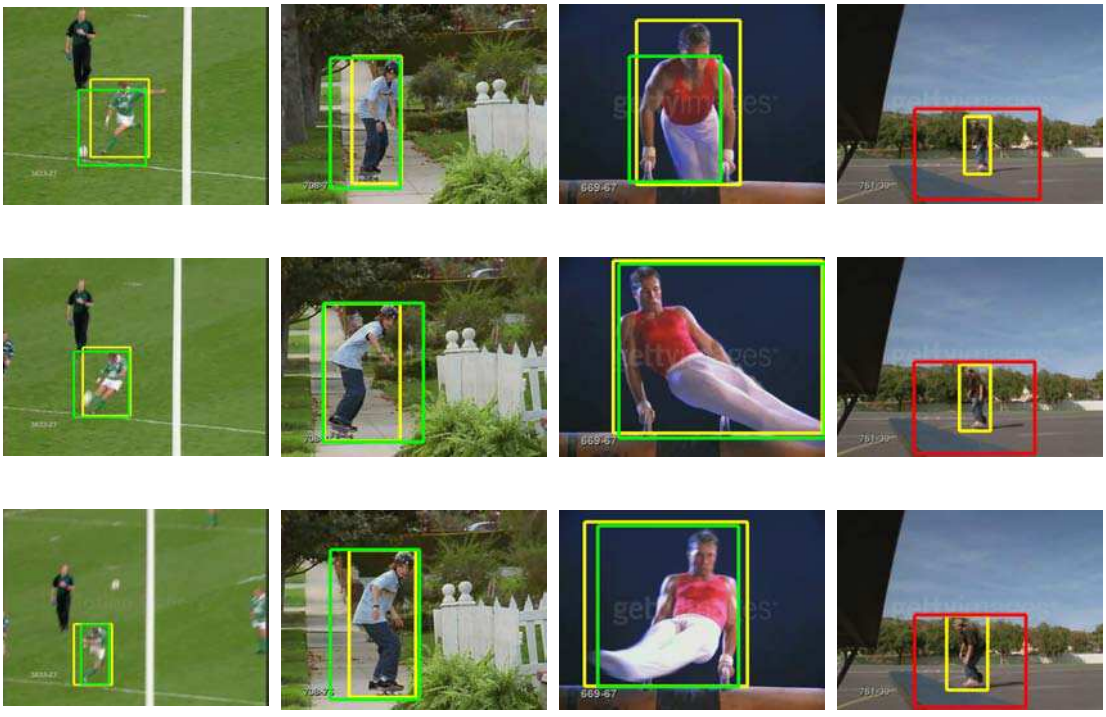
In Figures 6.9 and 6.10, several examples of localizations from UCF-Sports and MSR-II datasets are shown in diverse challenging settings. As we localize actions spatio-temporally, *i.e.*, as tubelets, we express them as bounding boxes from an ordered subsequence of 3 frames from the videos rather than showing just one frame.

6.5 Conclusions

We show, for the first time, the effectiveness of selective sampling for action localization in videos. To this end, we have revisited this concept for videos, by employing motion for producing super-voxels. Such hierarchical sampling produces category-independent proposals for action localization (not per class) and implicitly covers variable aspect ratios and temporal lengths. Our independent motion evidence (IME) based representation of video provides a more efficient alternative for segmentation. The IME motion



(a) Diving (b) Riding-Horse (c) Golf (d) Swinging-Bar



(e) Kicking (f) Skating (g) Swinging-Bench (h) Skating

Figure 6.9: Localization results shown as a sequence of bounding boxes (UCF-Sports): Groundtruth is shown in yellow, correctly localized tubelets in green and false positives, missed detections (or poorly localized ones) in red. Caption below each sequence reports the class detected.

(a) Boxing (b) HandClapping (c) HandWaving (d) HandClapping

(e) Boxing (f) HandClapping (g) HandWaving (h) Boxing

Figure 6.10: Localization results shown as a sequence of bounding boxes (MSR-II): Groundtruth is shown in yellow, correctly localized tubelets in green and false positives, missed detections (or poorly localized ones) in red. Caption below each sequence reports the class detected.

feature expresses both the individual density and the compactness of the action-related moving points in the super-voxel. It allows us to build a histogram-based motion similarity for merging super-voxels, supplying a parsimonious tubelet hierarchy. An analysis shows that the proposed tubelet sampling method heavily benefits from our motion features. Overall, our approach outperforms the state of the art for action localization on two public benchmarks.

Importantly, our method considers a relatively small number of candidate volumes at test time, *i.e.*, orders of magnitude smaller than in other concurrent approaches. For this reason, we believe that our method will enable the use of more effective but also more costly representations of spatio-temporal volumes in future works.

CHAPTER

SEVEN

Conclusions

This thesis has investigated different aspects of image and video representations, and contributed in many ways to improve the state-of-the-art performance for several recognition tasks. In images, we improved the matching of local descriptors, and presented a representation based on Hamming Embedding similarities amenable to classification. In videos, we have considered the importance of better utilization of motion. With this notion, we proposed methods to properly distinguish between camera motion and scene motion for action modeling. First, by improving the description of trajectories and second, by revisiting existing descriptors and introducing a new descriptor for action classification. Lastly, we proposed a hierarchical sampling of videos as super-voxels for action localization.

In the following, we summarize the main achievements of this thesis and discuss possible future research directions.

7.1 Summary of contributions

Asymmetric (vector-to-binary code) matching
Our asymmetric version of Hamming Embedding compares the query descriptor, not binarized, to the binary descriptors coded on the database side. Such an approach leads to more accurate comparison because the query does not suffer any quantization loss, while keeping the storage requirements identical since the database vectors are still binarized. This strategy only slightly increases the query processing time.

Hamming Embedding similarity based image representation
A novel image representation based on Hamming Embedding similarities between the given image and the training images is presented. This is the first time that Hamming Embedding is used, as a matching-based approach, in the context of image classification. In contrast to most approaches based on matching, such as NBNN, this approach offers competing results on PASCAL VOC 2007 benchmark, along with Caltech-256.

Sub-normalized variant of SIFT
A variant of SIFT is proposed that handles the dominance of strong gradients by square rooting the descriptors component-wise. This is followed by mild normalization, with square root of L2 norm, which offers a trade-off between (i) the invariance to intensity changes and (ii) preserving some information about the absolute values of gradients. A similar variant of SIFT, *RootSIFT*, was concurrently proposed by Arandjelvoić *et al.* [6].

Dominant motion compensation
The dominant motion is approximated using an affine motion model estimated for each pair of consecutive frames in the video. Cancelling this camera-induced motion component from optical flow significantly improves the description for action recognition. This is because the resulting trajectories and motion descriptors better capture the action-related motion when using the residual motion or ω-flow. Our results were much ahead of the state-of-the-art for the three used benchmark datasets at the time of publication. This approach was later followed by Wang *et al.* [147], who realized it differently and additionally employed human detection to further improve the camera motion estimation and hence the results.

Divergence-Curl-Shear (DCS) descriptor
A new descriptor that captures kinematic features of motion, namely divergence, curl and shear, is introduced. Since it encodes the local properties of motion not captured by other popular descriptors, it is complementary to other descriptors and leads to improved accuracy when combined with them.

Selective sampling of super-voxels
We presented a selective sampling strategy of super-voxels by hierarchical grouping. It is the first time that category-independent sampling is proposed for action localization. Apart from generating hypotheses for all classes at once, it implicitly covers variable aspect ratios and temporal lengths and samples relatively small number of candidate locations, thereby drastically reducing the computational cost compared to traditional video localization approaches.

Independent motion evidence (IME)
A novel evidence measure of action-related motion is presented. It is based on IME maps, which provide a faster alternative for generating initial super-voxels from video. A new motion feature computed from neighborhoods in the IME map expresses both the individual density and the compactness of the action-related moving points in the super-voxel. It allows us to build a histogram-based motion similarity for merging super-voxels, supplying a parsimonious tubelet hierarchy.

7.2 Future research

In this section we detail potentially promising directions for future research inspired by our experiments and the recent progress in the field of computer vision.

Hamming Embedding for classification
Robust descriptor matching of Hamming Embedding has been proved very effective in image retrieval. We employed Hamming Embedding similarity for image classification; it can be further explored for other visual classification tasks in images as well as in videos. Relatively very few matching-based approaches have been proposed for classification and Hamming Embedding can potentially change this. A possible direction could be to extend this representation for matching not only descriptors or low-level features but also mid-level features such as super-pixels in images and super-voxels or 3D regions in videos.

Improving motion compensation for action recognition
In Chapter 5, our dominant motion compensation produced significantly better versions of several state-of-the-art local descriptors computed along the ω-trajectories that are more related to actions. Though the performance was good, we observed that certain aspects of our method can be improved. First, the motion estimation does not conform to camera motion when there is close-up on the actors. By detecting whenever there is close-up by using, e.g., face or human detection, such cases can be handled differently. Second, we choose the affine motion model as it is a good trade-off between accuracy and efficiency. But there are many other promising options to be considered, such as quadratic motion model or using multiple motion models, that may be better adapted to action recognition. Similarly, there are other advanced algorithms [10] for optical flow computation.

Another important aspect that needs to be analyzed is the use of dominant motion or camera motion. We mentioned before that the camera motion is not a nuisance and can add important information in certain scenarios such as sports videos. We observed that when trajectories from the optical flow are also used along with ω-trajectories, the results are boosted. This confirms the utility of camera motion. However, we did not use the camera motion explicitly, which could provide additional information. For instance, separating it to create exclusive representations such as affine trajectories and then combine it with ω-trajectories.

Divergence-Curl-Shear descriptor for action recognition
Our DCS descriptor encodes kinematic properties of motion and is complementary to other descriptors. For the sake of comparison, we presented it with a representation

similar to MBH with same parameters. We believe that it can be expressed in a better way by optimizing the design parameters or even by reconsidering the descriptor computation procedure. For instance, instead of encoding three pairs separately, there may be a way to exploit the joint distribution of all three - divergence, curl and shear. The quantization strategy, mainly inherited from HOG and SIFT, could be probably extended with the quantization of 3D orientations using regular polyhedrons by Kläser *et al.* [70]. It would also be interesting to see the impact of the way in which gradients are computed, as these kinematic features rely upon the gradients of the flow field.

Improving tubelets for action recognition

The tubelets obtained from our hierarchical grouping of super-voxels cover many spatio-temporal scales and aspect ratios, and is therefore able to locate very flexible actions with few hypotheses. However, we found that the temporal localization needs to be enhanced. To handle partial temporal overlaps, top detected tubelets can be merged and appraised collectively to obtain a more compact tubelet. For this reassessment, ideas can be drawn from temporal localization methods like Actom Sequence Models (ASM) of Gaidon *et al.* [43].

Tubelets are also a kind of mid-level representation that could be used for action classification or semi-supervised action localization. One obvious direction is to use it for *classification-by-detection*, which is often done in images for object localization [50, 126]. Video as a collection of tubelets can be a promising catalyst for semi-supervised localization. Again, some inspiration can be drawn from the object localization literature [119, 159].

Towards efficient action localization

Current methods for action localization are not efficient, as the task is only an emerging one: The researchers have mainly focused on improving the accuracy. Action classification, albeit not real-time, is relatively more efficient and scalable. It has been applied on large datasets of TRECVID [106] for Multimedia Event Detection (MED) [5, 125] by resizing videos and sampling frames. This is expected as localization is always more computationally intensive than classification because large number of possible locations are evaluated.

Our selective sampling approach, which results in a small number of tubelets, certainly provides an alternative that can limit the complexity. Nevertheless, the video segmentation (GB) used to obtain initial super-voxels is the bottleneck. By applying GB on IME maps, we achieved to make it faster. However, we believe that there is still room for improvement on this front.

Publications

International conferences

- Mihir Jain, Hervé Jégou and Patrick Gros.
  Asymmetric Hamming Embedding.
  In ACM International conference on Multimedia, November 2011.
- Mihir Jain, Rachid Benmokhtar, Patrick Gros and Hervé Jégou.
  Hamming Embedding Similarity-based Image Classification.
  In ACM International Conference on Multimedia Retrieval (ICMR), Hong Kong, June 2012
- Mihir Jain, Hervé Jégou and Patrick Bouthemy.
  Better exploiting motion for better action recognition.
  In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June 2013.
- Mihir Jain, Jan van Gemert, Patrick Bouthemy, Hervé Jégou and Cees G. M. Snoek.
  Action localization by tubelets from motion.
  In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June 2014.

Other publications

- C. G. M. Snoek, K. E. A. van de Sande, D. Fontijne, A. Habibian, M. Jain, S. Kordumova, Z. Li, M. Mazloom, S. L. Pintea, R. Tao, D. C. Koelma, and A. W. M. Smeulders.
  Mediamill at trecvid: Searching concepts, objects, instances and events in video.
  In Proceedings of TRECVID, 2013.

LIST OF FIGURES

LIST OF TABLES

BIBLIOGRAPHY

[1] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[2] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[3] S. Ali, A. Basharat, and M. Shah. Chaotic invariants for human action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2007.

[4] S. Ali and M. Shah. Human action recognition in videos using kinematic features and multiple instance learning. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 32(2):288–303, Feb. 2010.

[5] R. Aly, R. Arandjelovic, K. Chatfield, M. Douze, B. Fernando, Z. Harchaoui, K. Mcguiness, N. O'Connor, D. Oneata, O. Parkhi, D. Potapov, J. Revaud, C. Schmid, J.-L. Schwenninger, D. Scott, T. Tuytelaars, J. Verbeek, H. Wang, and A. Zisserman. The AXES submissions at TrecVid 2013. In *TRECVID Workshop*, Nov. 2013.

[6] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[7] R. Arandjelović and A. Zisserman. All about VLAD. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013.

[8] S. Avila, N. Thome, M. Cord, E. Valle, and A. De A. AraúJo. Pooling in image representation: The visual codeword point of view. *Computer Vision and Image Understanding*, 117(5):453–465, 2013.

[9] S. Avila, N. Thome, M. Cord, E. Valle, and A. de Albuquerque AraúJo. Bossa: Extended bow formalism for image classification. In *International Conference on Image Processing (ICIP)*, 2011.

[10] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. Black, and R. Szeliski. `vision.middlebury.edu/flow/`.

[11] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *Proceedings of the European Conference on Computer Vision*, May 2006.

[12] R. Behmo, P. Marcombes, A. Dalalyan, and V. Prinet. Towards optimal naive bayes nearest neighbors. In *Proceedings of the European Conference on Computer Vision*, Sep. 2010.

[13] L. Bo and C. Sminchisescu. Efficient match kernels between sets of features for visual recognition. In *Advances in Neural Information Processing Systems*, 2009.

[14] O. Boiman, E. Shechman, and M. Irani. In defense of nearest neighbor based image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008.

[15] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2010.

[16] W. Brendel and S. Todorovic. Video object segmentation by tracking regions. In *Proceedings of the IEEE International Conference on Computer Vision*, 2009.

[17] W. Brendel and S. Todorovic. Learning spatiotemporal graphs of human activities. In *Proceedings of the IEEE International Conference on Computer Vision*, Nov. 2011.

[18] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *Proceedings of the European Conference on Computer Vision*, Sep. 2010.

[19] L. Cao, Z. Liu, and T. S. Huang. Cross-dataset action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2010.

[20] B. Caputo and L. Jie. A performance evaluation of exact and approximate match kernels for object recognition. *Electronic Letters on Computer Vision and Image Analysis*, 8(3):15–26, 2009.

[21] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *Proceedings of the British Machine Vision Conference*, Sep. 2011.

[22] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2007.

[23] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[24] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV Workshop Statistical Learning in Computer Vision*, 2004.

[25] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2005.

[26] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *Proceedings of the European Conference on Computer Vision*, May 2006.

[27] J. Delhumeau, P. H. Gosselin, H. Jégou, and P. Pérez. Revisiting the vlad image representation. In *ACM International conference on Multimedia*, Oct. 2013.

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

[29] T. Deselaers, B. Alexe, and V. Ferrari. Weakly supervised localization and learning with generic knowledge. *International Journal of Computer Vision*, 100(3):275–293, 2012.

[30] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, Oct. 2005.

[31] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *SIGIR*, Jul. 2008.

[32] M. Douze, H. Jégou, H. Singh, L. Amsaleg, and C. Schmid. Evaluation of GIST descriptors for web-scale image search. In *Proceedings of the International Conference on Image and Video Retrieval*, July 2009.

[33] O. Duchenne, A. Joulin, and J. Ponce. A graph-matching kernel for object categorization. In *Proceedings of the IEEE International Conference on Computer Vision*, Sep. 2011.

[34] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce. Automatic annotation of human actions in video. In *Proceedings of the IEEE International Conference on Computer Vision*, Sep. 2009.

[35] T. Durand, N. Thome, M. Cord, and S. Avila. Image classification using object detectors. In *International Conference on Image Processing (ICIP)*, Sep. 2013.

[36] I. Endres and D. Hoiem. Category independent object proposals. In *Proceedings of the European Conference on Computer Vision*, 2010.

[37] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, 2007.

[38] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, Jun. 2010.

[39] I. Everts, J. van Gemert, and T. Gevers. Evaluation of color stips for human action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[40] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, 2003.

[41] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[42] J. Feng, Y. Wei, L. Tao, C. Zhang, and J. Sun. Salient object detection by composition. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

[43] A. Gaidon, Z. Harchaoui, and C. Schmid. Actom sequence models for efficient action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011.

[44] A. Gaidon, Z. Harchaoui, and C. Schmid. Recognizing activities with cluster-trees of tracklets. In *Proceedings of the British Machine Vision Conference*, Sep. 2012.

[45] A. Gaidon, M. Marszalek, and C. Schmid. Mining visual actions from movies. In *Proceedings of the British Machine Vision Conference*, Sep. 2009.

[46] A. Gordo, J. A. Rodríguez-Serrano, F. Perronnin, and E. Valveny. Leveraging category-level labels for instance-level image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012.

[47] P. H. Gosselin, M. Cord, and S. Philipp-Foliguet. Kernels on bags for multi-object database retrieval. In *Proceedings of the International Conference on Image and Video Retrieval*, Jul. 2007.

[48] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2005.

[49] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[50] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *Proceedings of the IEEE International Conference on Computer Vision*, Sep. 2009.

[51] A. Hervieu, P. Bouthemy, and J.-P. Le Cadre. A statistical video content recognition method using invariant features on object trajectories. IEEE *Transactions on Circuits and Systems for Video Technology*, 18(11):1533–1543, 2008.

[52] P. Huber. *Robust statistics*. Wiley, New York, 1981.

[53] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, 1998.

[54] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, Dec. 1999.

[55] M. Jain, R. Benmokhtar, P. Gros, and H. Jégou. Hamming embedding similarity-based image classification. In *ACM International conference on Multimedia Retrieval*, Jun. 2012.

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

[57] M. Jain, H. Jégou, and P. Gros. Asymmetric hamming embedding: taking the best of our bits for large scale image search. In *ACM International conference on Multimedia*, Nov. 2011.

[58] M. Jain, J. van Gemert, P. Bouthemy, H. Jégou, and C. G. M. Snoek. Action localization by tubelets from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014.

[59] H. Jégou and O. Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *Proceedings of the European Conference on Computer Vision*, Oct. 2012.

[60] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proceedings of the European Conference on Computer Vision*, Oct. 2008.

[61] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, Jun. 2009.

[62] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87(3):316–336, Feb. 2010.

[63] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 33(1):117–128, Jan. 2011.

[64] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2010.

[65] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local descriptors into compact codes. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716, 2012.

[66] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. Trajectory-based modeling of human actions with motion reference points. In *Proceedings of the European Conference on Computer Vision*, Oct. 2012.

[67] I. N. Junejo, E. Dexter, I. Laptev, and P. Pérez. View-independent action recognition from temporal self-similarities. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 33(1):172–185, 2010.

[68] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2005.

[69] J. Kim and K. Grauman. Asymmetric region-to-image matching for comparing images with generic object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2010.

[70] A. Kläser, M. Marszalek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *Proceedings of the British Machine Vision Conference*, Sep. 2008.

[71] A. Kläser, M. Marszałek, C. Schmid, and A. Zisserman. Human focused action localization in video. In *Trends and Topics in Computer Vision*, pages 219–233, 2012.

[72] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion interchange patterns for action recognition in unconstrained videos. In *Proceedings of the European Conference on Computer Vision*, Oct. 2012.

[73] J. Krapac, F. Jurie, and B. Triggs. Learning tree-structured descriptor quantizers for image categorization. In *Proceedings of the British Machine Vision Conference*, Aug. 2011.

[74] J. Krapac, J. J. Verbeek, and F. Jurie. Modeling spatial layout with fisher vectors for image categorization. In *Proceedings of the IEEE International Conference on Computer Vision*, Nov. 2011.

[75] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, Nov. 2011.

[76] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008.

[77] T. Lan, Y. Wang, and G. Mori. Discriminative figure-centric models for joint action localization and recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, Nov. 2011.

[78] I. Laptev and T. Lindeberg. Space-time interest points. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2003.

[79] I. Laptev, M. Marzalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008.

[80] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *Proceedings of the International Conference on Image and Video Retrieval*, pages 371–378, New York, NY, USA, 2007. ACM.

[81] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2006.

[82] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011.

[83] F.-F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2005.

[84] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011.

[85] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[86] W. Y. Ma and B. S. Manjunath. Netra: a toolbox for navigating large image databases. *Multimedia Systems*, 7(3):184–198, 1999.

[87] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce. Discriminative sparse image models for class-specific edge detection and image interpretation. In *Proceedings of the European Conference on Computer Vision*, Oct. 2008.

[88] S. Manen, M. Guillaumin, and L. Van Gool. Prime Object Proposals with Randomized Prim's Algorithm. In *Proceedings of the IEEE International Conference on Computer Vision*, 2013.

[89] M. Marzalek, I. Laptev, and C. Schmid. Actions in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009.

[90] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, pages 384–393, Sep. 2002.

[91] P. Matikainen, M. Hebert, and R. Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *Workshop on Video-Oriented Object and Event Classification, ICCV*, Sep. 2009.

[92] R. Messing, C. J. Pal, and H. A. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *Proceedings of the IEEE International Conference on Computer Vision*, Sep. 2009.

[93] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[94] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[95] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1/2):43–72, 2005.

[96] F. Moosmann, B. Triggs, and F. Jurie. Randomized clustering forests for building fast and discriminative visual vocabularies. In *Advances in Neural Information Processing Systems*, pages 985–992, 2007.

[97] R. Negrel, D. Picard, and P. H. Gosselin. Using spatial pyramids with compacted vlat for image categorization. In *ICPR*, Nov. 2012.

[98] J. C. Niebles, C.-W. Chen, and F.-F. Li. Modeling temporal structure of decomposable motion segments for activity classification. In *Proceedings of the European Conference on Computer Vision*, Sep. 2010.

[99] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, Jun. 2006.

[100] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Proceedings of the European Conference on Computer Vision*, May 2006.

[101] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Jal of Vis. Comm. and Image Representation*, 6(4):348–365, Dec. 1995.

[102] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[103] D. Oneata, J. Verbeek, and C. Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2013.

[104] D. Oneata, J. Verbeek, and C. Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2013.

[105] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014.

[106] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, W. Kraaij, A. F. Smeaton, and G. Quéenot. Trecvid 2013 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2013*. NIST, USA, 2013.

[107] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[108] F. Perronnin, C. R. Dance, G. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. In *Proceedings of the European Conference on Computer Vision*, May 2006.

[109] F. Perronnin, J.Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Proceedings of the European Conference on Computer Vision*, Sep. 2010.

[110] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed Fisher vectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2010.

[111] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2007.

[112] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008.

[113] G. Piriou, P. Bouthemy, and J.-F. Yao. Recognition of dynamic video contents with global probabilistic models of visual motion. IEEE *Transactions on Image Processing*, 15(11):3417–3430, 2006.

[114] E. Rahtu, J. Kannala, and M. Blaschko. Learning a category independent object detection cascade. In *Proceedings of the IEEE International Conference on Computer Vision*, 2011.

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

[116] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012.

[117] J. Revaud, M. Douze, C. Schmid, and H. Jégou. Event retrieval in large video collections with circulant temporal encoding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013.

[118] M. D. Rodriguez, J. Ahmed, and M. Shah. Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008.

[119] C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2005.

[120] S. Sadanand and J. J. Corso. Action bank: A high-level representation of activity in video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012.

[121] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, May 1997.

[122] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *ACM International conference on Multimedia*, Sep. 2007.

[123] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1470–1477, Oct. 2003.

[124] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and Trecvid. In *MIR*, 2006.

[125] C. G. M. Snoek, K. E. A. van de Sande, D. Fontijne, A. Habibian, M. Jain, S. Kordumova, Z. Li, M. Mazloom, S. L. Pintea, R. Tao, D. C. Koelma, and A. W. M. Smeulders. Mediamill at trecvid 2013: Searching concepts, objects, instances and events in video. In *Proceedings of TRECVID*, 2013.

[126] Z. Song, Q. Chen, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011.

[127] C. Sun and R. Nevatia. ACTIVE: Activity Concept Transitions in Video Event Classification. In *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2013.

[128] J. Sun, X. Wu, S. Yan, L. F. Cheong, T.-S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009.

[129] Y. Tian, R. Sukthankar, and M. Shah. Spatiotemporal deformable part models for action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013.

[130] E. Tola, V.Lepetit, and P. Fua. A Fast Local Descriptor for Dense Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008.

[131] D. Tran and J. Yuan. Optimal spatio-temporal path discovery for video event detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011.

[132] D. Tran and J. Yuan. Max-margin structured output regression for spatio-temporal action localization. In *Advances in Neural Information Processing Systems*, Dec. 2012.

[133] D. Tran, J. Yuan, and D. Forsyth. Video event detection: From subvolume localization to spatio-temporal path search. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[134] R. Trichet and R. Nevatia. Video segmentation with spatio-temporal tubes. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2013.

[135] T. Tuytelaars, M. Fritz, K. Saenko, and T. Darrel. The NBNN kernel. In *Proceedings of the IEEE International Conference on Computer Vision*, Sep. 2011.

[136] H. Uemura, S. Ishikawa, and K. Mikolajczyk. Feature tracking and motion compensation for action recognition. In *Proceedings of the British Machine Vision Conference*, Sep. 2008.

[137] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[138] M. M. Ullah, S. N. Parizi, and I. Laptev. Improving bag-of-features action recognition with non-local cues. In *Proceedings of the British Machine Vision Conference*, Sep. 2010.

[139] J. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In *Proceedings of the European Conference on Computer Vision*, Oct. 2008.

[140] J. van Gemert, C. G. M. Snoek, C. J. Veenman, A. W. M. Smeulders, and J.-M. Geusebroek. Comparing compact codebooks for visual categorization. *Computer Vision and Image Understanding*, 114(4):450–462, 2010.

[141] J. van Gemert, C. Veenman, A. Smeulders, and J. Geusebroek. Visual word ambiguity. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1271–1283, Jul. 2010.

[142] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[143] E. Vig, M. Dorr, and D. Cox. Saliency-based space-variant descriptor sampling for action recognition. In *Proceedings of the European Conference on Computer Vision*, Oct. 2012.

[144] P. A. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[145] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011.

[146] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, 2013.

[147] H. Wang and C. Schmid. Action Recognition with Improved Trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2013.

[148] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *Proceedings of the British Machine Vision Conference*, Sep. 2009.

[149] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2010.

[150] T. Wang, S. Wang, and D. Xiaoqing. Detecting human action as the spatio-temporal tube of maximum mutual information. IEEE *Transactions on Circuits and Systems for Video Technology*, 24(2):277–290, 2014.

[151] G. Willems, T. Tuytelaars, and L. J. V. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proceedings of the European Conference on Computer Vision*, Oct. 2008.

[152] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2005.

[153] S. Wu, O. Oreifej, and M. Shah. Action recognition in videos acquired by a moving camera using motion decomposition of lagrangian particle trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, Nov. 2011.

[154] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling features for large scale partial-duplicate web image search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 25–32, 2009.

[155] C. Xu and J. Corso. Evaluation of super-voxel methods for early video processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[156] C. Xu, C. Xiong, and J. Corso. Streaming hierarchical video segmentation. In *Proceedings of the European Conference on Computer Vision*, 2012.

[157] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao. Group sensitive multiple kernel learning for object categorization. In *Proceedings of the IEEE International Conference on Computer Vision*, Sep. 2009.

[158] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801, 2009.

[159] M. S. Yang Yang, Guang Shu. Semi-supervised learning of feature hierarchies for object detection in a video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013.

[160] J. Yuan, Z. Liu, and Y. Wu. Discriminative video pattern search for efficient action detection. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1728–1743, 2011.

[161] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73:213–238, Jun. 2007.

[162] W.-L. Zhao, H. Jégou, and G. Gravier. Oriented pooling for dense and non-dense rotation-invariant features. In *Proceedings of the British Machine Vision Conference*, Sep. 2013.

[163] X. Zhou, K. Yu, T. Zhang, and T. S. Huang. Image classification using super-vector coding of local image descriptors. In *Proceedings of the European Conference on Computer Vision*, Sep. 2010.

[164] J. Zhu, B. Wang, X. Yang, W. Zhang, and Z. Tu. Action Recognition with Actons. In *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2013.

Enhanced image and video representation for visual recognition

Résumé en Francais. L'objectif de cette thèse est d'améliorer les représentations des images et des vidéos dans le but d'obtenir une reconnaissance visuelle accrue, tant pour des entités spécifiques que pour des catégories plus génériques. Les contributions de cette thèse portent, pour l'essentiel, sur des méthodes de description du contenu visuel. Nous proposons des méthodes pour la recherche d'image par le contenu ou par des requêtes textuelles, ainsi que des méthodes pour la reconnaissance et la localisation d'action dans des vidéos.

En recherche d'image, les contributions se fondent sur des méthodes à base de plongements de Hamming. Tout d'abord, une méthode de comparaison asymétrique vecteur-à-code est proposée pour améliorer la méthode originale, symétrique et utilisant une comparaison code-à-code. Une méthode de classification fondée sur l'appariement de descripteurs locaux est ensuite proposée. Elle s'appuie sur une classification opérée dans un espace de similarités associées au plongement de Hamming.

En reconnaissance d'action, les contributions portent essentiellement sur des meilleures manières d'exploiter et de représenter le mouvement. Finalement, une méthode de localisation est proposée. Elle utilise une partition de la vidéo en super-voxels, qui permet d'effectuer un échantillonnage 2D+t de suites de boîtes englobantes autour de zones spatio-temporelles d'intérêt. Elle s'appuie en particulier sur un critère de similarité associé au mouvement.

Toutes les méthodes proposées sont évaluées sur des jeux de données publics. Ces expériences montrent que les méthodes proposées dans cette thèse améliorent l'état de l'art au moment de leur publication.

Résumé en Anglais. The subject of this thesis is about image and video representations for visual recognition. This thesis first focuses on image search, both for image and textual queries, and then considers the classification and the localization of actions in videos.

In image retrieval, images similar to the query image are retrieved from a large dataset. On this front, we propose an asymmetric version of the Hamming Embedding method, where the comparison of query and database descriptors relies on a vector-to-binary code comparison. For image classification, where the task is to identify if an image contains any instance of the queried category, we propose a novel approach based on a match kernel between images, more specifically based on Hamming Embedding similarity. We also present an effective variant of the SIFT descriptor, which leads to a better classification accuracy.

Action classification is improved by several methods to better employ the motion inherent to videos. This is done by dominant motion compensation, and by introducing a novel descriptor based on kinematic features of the visual flow. The last contribution is devoted to action localization, whose objective is to determine where and when the action of interest appears in the video. A selective sampling strategy produces 2D+t sequences of bounding boxes, which drastically reduces the candidate locations. The method advantageously exploits a criterion that takes in account how motion related to actions deviates from the background motion.

We thoroughly evaluated all the proposed methods on real world images and videos from challenging benchmarks. Our methods outperform the previously published related state of the art and remains competitive with the subsequently proposed methods.