Universitat Autònoma de Barcelona

Domain Adaptation of Deformable Part-based Models

许娇龙

A dissertation submitted by **Jiaolong Xu** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor of Philosophy**.

Bellaterra, March 2, 2015

Director | **Dr. Antonio López Peña**
Dept. Ciències de la computació & Centre de Visió per Computador
Universitat Autònoma de Barcelona

This document was typeset by the author using $\LaTeX\,2_\varepsilon$.

The research described in this book was carried out at the Centre de Visió per Computador, Universitat Autònoma de Barcelona.

ISBN: 978-84-940530-1-6

Printed by Ediciones Gráficas Rey, S.L.

To my parents...

The journey of a thousand miles begins with one step.
Lao Tzu (604 BC - 531 BC)

Acknowledgements

First and foremost, I must thank my PhD advisor Dr. A. M. López, for his mentorship and friendship over these years. His conscientious, meticulous, discipline and commitment to hard work, attitude to life, all had a great influence on me. I consider him one of the best advisor in the world, not just on research topics, but also in life. I am also impressed by his talent in water-color painting. It is hard to describe either my gratitude or my good luck in having Antonio as my advisor.

I would also like to thank my committee members, and the European mention evaluators for accepting the task of evaluating my PhD work. I also want to thank the anonymous journal and conference reviewers for their enriching comments during my PhD.

I sincerely thank Dr. J. Marín and Dr. D. Vázquez. Without their previous hard work on creating virtual world and opening the direction of domain adaptation for pedestrian detection, I would never get such a great chance to work on this exciting topic. Thanks for their kind help for answering any of my questions even when they were very busy. Especially to Dr. D. Vázquez, thanks for his guidance in programming and reviewing the papers.

Many thanks to Dr. K. Mikolajczyk for offering me the great opportunity of doing a research stay at the CVSSP of Surrey University. Thanks for his advises during the stay. I would also like to take this opportunity to acknowledge the friends I met in Surrey University, including M. Su, G. Hu, Z. Feng, D. Jing and P. Huber.

Thanks must be given to all the Computer Vision Center (CVC) fellows. Thanks to the technical staff and administrative stuff for their patience and support over these years. To the members of ADAS group, including Dr. J. Serrat, Dr. A. Sappa, Dr. D. Ponsa, G. Ros, S. Ramos, Dr. D. Gerónimo, A. González, Y. Socarrás. To the fellows from other groups, including Dr. J. V. Weijer, Dr. A. D. Bagdanov, M. Serra and P. Marquez. Special thanks to H. Gao, N. Cirera and L. Yu, who I consider my best friends in CVC. Working in CVC would not have been the same without them.

I would like to thank X. Hu from Oregon State University for many fruitful discussions during my PhD study. Thanks for his sharing of interesting papers and thoughts, which broadened my view and gave me inspirations. I sincerely appreciate his guidance in graphical models during google summer of code 2014.

I am also grateful to my friends living in Cerdanyola. Thanks Dr. W. Guo and P. Niu for being great flatmates. Thanks T. Liu for offering the room when I came back from Surrey and having trouble in finding a short term flat. Thanks to my basketball team mates, W. Qiu, Y. Xin, Y. Wang, Z. Shao, S. Yu, J. Li, J. Ji. I miss every sunny

"Basketball Weekend" with them in Cerdanyola.

Finally, I would like to thank my parents, my sister for their enduring support and encouraging over these years when I was away from home. This thesis is dedicated to them.

The work presented in this thesis are supported by the Chinese Scholarship Council (CSC) grant No.2011611023 and the Spanish projects TRA2011-29454-C03-01 and TIN2011-29494-C03-02.

Abstract

On-board pedestrian detection is crucial for Advanced Driver Assistance Systems (ADAS). An accurate classification is fundamental for vision-based pedestrian detection. The underlying assumption for learning classifiers is that the training set and the deployment environment (testing) follow the same probability distribution regarding the features used by the classifiers. However, in practice, there are different reasons that can break this constancy assumption. Accordingly, reusing existing classifiers by adapting them from the previous training environment (*source domain*) to the new testing one (*target domain*) is an approach with increasing acceptance in the computer vision community. In this thesis we focus on the domain adaptation of deformable part-based models (DPMs) for pedestrian detection. As a prof of concept, we use a computer graphic based synthetic dataset, *i.e.* a virtual world, as the source domain, and adapt the virtual-world trained DPM detector to various real-world dataset.

We start by exploiting the maximum detection accuracy of the virtual-world trained DPM. Even though, when operating in various real-world datasets, the virtual-world trained detector still suffer from accuracy degradation due to the domain gap of virtual and real worlds. We then focus on domain adaptation of DPM. At the first step, we consider single source and single target domain adaptation and propose two batch learning methods, namely A-SSVM and SA-SSVM. Later, we further consider leveraging multiple target (sub-)domains for progressive domain adaptation and propose a hierarchical adaptive structured SVM (HA-SSVM) for optimization. Finally, we extend HA-SSVM for the challenging online domain adaptation problem, aiming at making the detector to automatically adapt to the target domain online, without any human intervention. All of the proposed methods in this thesis do not require revisiting source domain data. The evaluations are done on the Caltech pedestrian detection benchmark. Results show that SA-SSVM slightly outperforms A-SSVM and avoids accuracy drops as high as 15 points when comparing with a non-adapted detector. The hierarchical model learned by HA-SSVM further boosts the domain adaptation performance. Finally, the online domain adaptation method has demonstrated that it can achieve comparable accuracy to the batch learned models while not requiring manually label target domain examples. Domain adaptation for pedestrian detection is of paramount importance and a relatively unexplored area. We humbly hope the work in this thesis could provide foundations for future work in this area.

Resumen

La detección de peatones es crucial para los sistemas de asistencia a la conducción (ADAS). Disponer de un clasificador preciso es fundamental para un detector de peatones basado en visión. Al entrenar un clasificador, se asume que las características de los datos de entrenamiento siguen la misma distribución de probabilidad que las de los datos de prueba. Sin embargo, en la práctica, esta asunción puede no cumplirse debido a diferentes causas. En estos casos, en la comunidad de visión por computador cada vez es más común utilizar técnicas que permiten adaptar los clasificadores existentes de su entorno de entrenamiento (dominio de origen) al nuevo entorno de prueba (dominio de destino). En esta tesis nos centramos en la adaptación de dominio de los detectores de peatones basados en modelos deformables basados en partes (DPMs). Como prueba de concepto, usamos como dominio de origen datos sintéticos (mundo virtual) y adaptamos el detector DPM entrenado en el mundo virtual para funcionar en diferentes escenarios reales. Comenzamos explotando al máximo las capacidades de detección del DPM entrenado en datos del mundo virtual pero, aun así, al aplicarlo a diferentes conjuntos del mundo real, el detector todavía pierde poder de discriminación debido a las diferencias entre el mundo virtual y el real. Es por ello que nos centramos en la adaptación de dominio del DPM.

Para comenzar, consideramos un único dominio de origen para adaptarlo a un único dominio de destino mediante dos métodos de aprendizaje por lotes, el A-SSVM y SA-SSVM. Después, lo ampliamos a trabajar con múltiples (sub-)dominios mediante una adaptación progresiva usando una jerarquía adaptativa basada en SSVM (HA-SSVM) en el proceso de optimización. Finalmente, extendimos HA-SSVM para conseguir un detector que se adapte de forma progresiva y sin intervención humana al dominio de destino. Cabe destacar que ninguno de los métodos propuestos en esta tesis requieren visitar los datos del dominio de origen. La evaluación de los resultados, realizadas con el sistema de evaluación de Caltech, muestran que el SA-SSVM mejora ligeramente respecto al A-SSVM y mejora en 15 puntos respecto al detector no adaptado. El modelo jerárquico entrenado mediante el HA-SSVM todavía mejora más los resultados de la adaptación de dominio. Finalmente, el método secuencial de adaptación de domino ha demostrado que puede obtener resultados comparables a la adaptación por lotes pero sin necesidad de etiquetar manualmente ningún ejemplo del dominio de destino. La adaptación de domino aplicada a la detección de peatones es de gran importancia y es un área que se encuentra relativamente sin explorar. Deseamos que esta tesis pueda sentar las bases del trabajo futuro en esta área.

Resum

La detecció de vianants és crucial per als sistemes d'assistència a la conducció (ADAS). Disposar d'un classificador precís és fonamental per a un detector de vianants basat en visió. Al entrenar un classificador, s'assumeix que les característiques de les dades d'entrenament segueixen la mateixa distribució de probabilitat que la de les dades de prova. Tot i això, a la pràctica, aquesta assumpció pot no complir-se per diferents causes. En aquests casos, en la comunitat de visió per computador és cada cop més comú utilitzar tècniques que permeten adaptar els classificadors existents del seu entorn d'entrenament (domini d'origen) al nou entorn de prova (domini de destí). En aquesta tesi ens centrem en l'adaptació de domini dels detectors de vianants basats en models deformables basats en parts (DPMs). Com a prova de concepte, utilitzem dades sintètiques com a domini d'origen (món virtual) i adaptem el detector DPM entrenat en el món virtual per a funcionar en diferents escenaris reals. Comencem explotant al màxim les capacitats de detecció del DPM entrenant en dades del món virtual, però, tot i això, al aplicar-lo a diferents conjunts del món real, el detector encara perd poder de discriminació degut a les diferències entre el món virtual i el real. És per això, que ens centrem en l'adaptació de domini del DPM.

Per començar, considerem un únic domini d'origen per a adaptar-lo a un únic domini de destí mitjançant dos mètodes d'aprenentatge per lots, l' A-SSVM i el SA-SSVM. Després, l'ampliem a treballar amb múltiples (sub-)dominis mitjançant una adaptació progressiva, utilitzant una jerarquia adaptativa basada en SSVM (HA-SSVM) en el procés d'optimització. Finalment, extenem HA-SSVM per a aconseguir un detector que s'adapti de forma progressiva i sense intervenció humana al domini de destí. Cal destacar que cap dels mètodes proposats en aquesta tesi requereix visitar les dades del domini d'origen. L'evaluació dels resultats, realitzada amb el sistema d'evaluació de Caltech, mostra que el SA-SSVM millora lleugerament respecte el A-SSVM i millora en 15 punts respecte el detector no adaptat. El model jeràrquic entrenat mitjançant el HA-SSVM encara millora més els resultats de la adaptació de domini. Finalment, el mètode sequencial d'adaptació de domini ha demostrat que pot obtenir resultats comparables a la adaptació per lots, però sense necessitat d'etiquetar manualment cap exemple del domini de destí. L'adaptació de domini aplicada a la detecció de vianants és de gran importància i és una àrea que es troba relativament sense explorar. Desitgem que aquesta tesi pugui assentar les bases del treball futur d'aquesta àrea.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

On-board pedestrian detection (Figure 1.1) is crucial to prevent accidents. Vision-based detectors consist of several processing stages [30, 40], namely the generation of image candidate windows, their classification as *pedestrian* or *background*, the refinement into a single detection of multiple ones arising from the same pedestrian, and the tracking of the detections for removing spurious ones or inferring trajectory information.

An accurate classification is fundamental. However, it turns out to be a difficult task due to the large intra-class variability of both pedestrians and background classes, as well as the imaging and environmental conditions. Note that pedestrians are moving objects which vary on morphology, pose and clothes; there is a large diversity of scenarios; and images are acquired from a platform moving outdoors (*i.e.*, the vehicle), thus, pedestrians are seen from different viewpoints at a range of distances and under uncontrolled illumination.

Aiming at overcoming such a complexity, many pedestrian classifiers/detectors have been proposed during the last fifteen years. The reader is referred to [40] for a comprehensive review on pedestrian detection, to [22, 30] for accuracy comparisons of different proposals, as well as to [6, 22] where the focus is on reaching real-time processing. A first outcome of the work done so far in this field is that most accurate pedestrian classifiers are learned from pedestrian and background samples. For instance, this is the case of the well-known pedestrian classifier based on histograms

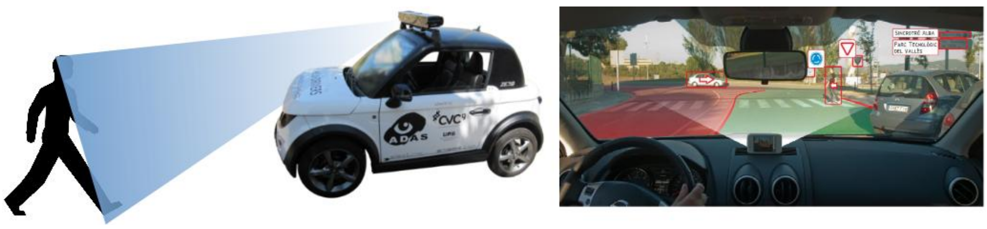


Figure 1.1: On-board pedestrian detection system.

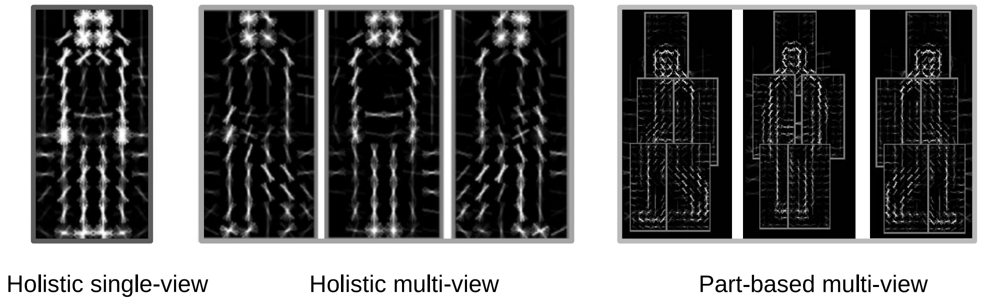


Figure 1.2: Visualization of different pedestrian models trained by using HOG features and linear SVM.

of oriented gradients and linear support vector machines (HOG/Lin-SVM) [15].

Indeed, HOG/Lin-SVM approach was a milestone in the field of pedestrian detection. However, the most relevant contribution of [15] consists in devising HOG features, since the overall pedestrian classifier itself just follows a *holistic* approach and uses a linear frontier to separate pedestrians and background. Holistic approaches regard pedestrians as a whole, *i.e.*, no body-inspired parts are considered separately. Moreover, [15] propose what we term as *single* holistic approach because the intra-class variability of the pedestrians is not explicitly considered. In other words, during the training of the pedestrian classifier all pedestrians are mixed, which tends to generate blurred features. In consequence, the learned classifier does not necessarily improves its accuracy by increasing and/or diversifying the training pedestrian samples [125].

In order to overcome this limitation, prior knowledge about the pedestrian class can be exploited. For instance, we can find multiple holistic ensembles accounting for different pedestrian view and pose combinations (*aspects* hereinafter), or single/multiple body-inspired part-based ensembles. Representative examples can be found in [6, 31, 35, 50, 83, 94, 114]. In fact, the *deformable part-based model* (DPM) presented in [35] is one of the most popular state-of-the-art pedestrian/object detectors. The visualization of the holistic and part-based models trained by using HOG features and linear SVM is shown in Figure 1.2.

An advantage of DPMs is that pedestrian poses unseen during training are implicitly modeled through the allowed deformation, *i.e.*, the generalization capability of the corresponding classifiers increases. This is more effective if view-based DPMs can be used to build a mixture model, which is the case in [35] provided that the aspect ratio of the annotated pedestrian bounding boxes (BBs) correlates with major view differences (*e.g.*, frontal *vs.* side). Accordingly, in this PhD we start by proposing *a new aspect-based mixture of DPMs with part-sharing*. A key point of such a pedestrian model is to have pedestrian samples with reliable and rich annotations. In particular, for each pedestrian, its full-body BB is required along with the BB of its constituent parts, and its aspect label (*e.g.*, either rear-frontal, side-left, side-right). Collecting all this information by human annotation is a tiresome task prone to errors. Thus, other than [4, 11, 13, 29, 32, 85, 94, 95, 117, 125], we propose the use of images from a virtual-world with automatic pixel-wise pedestrian groundtruth. In the first

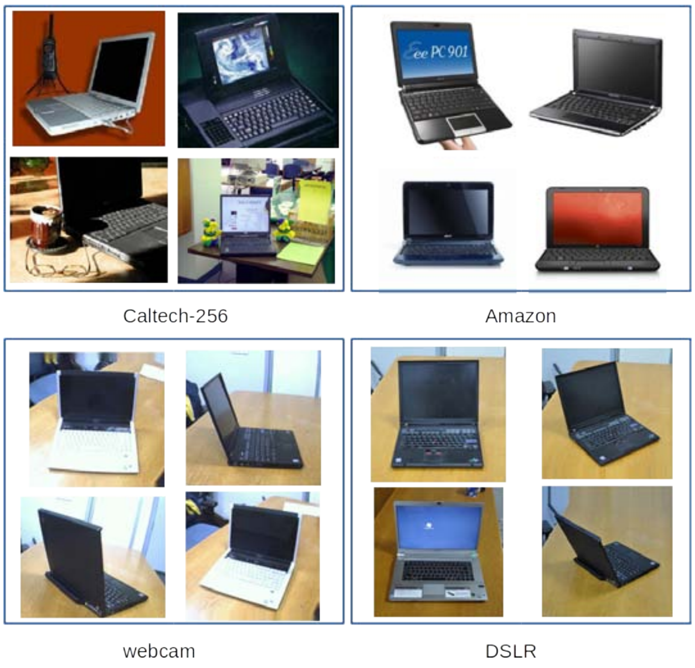


Figure 1.3: Example images of different domains: *Caltech-256* images often have clean backgrounds, *Amazon* is collected from online catalog, *Webcam* images are captured by low-resolution web cameras in an office environment, and *DSLR* consists of high-quality images.

work in this line [68] a single holistic pedestrian classifier trained with virtual-world data performed equally well in automotive real-world images than an equivalent one trained with real-world data. For building our pedestrian model, we go beyond [68] by exploiting part labeling (*i.e.*, part BBs) and aspect clustering, both automatically obtained from the pixel-wise groundtruth.

In the last years the computer vision community has started to consider the decrease in accuracy of a classifier due to differences between training (*source domain*) and testing (*target domain*) data. The discrepancy between different domains is also known as *dataset shift* [89] or *dataset bias* [98], which could be caused by different sensor, (*e.g.*, web camera or digital single-lens reflex camera (DSLR)), different resolution, or human collection bias and so on. Example images from different domains are shown in Figure 1.4.

In [99, 100], it is shown that between virtual- and real-world data this problem exists. However, it is shown also that it is not due to the particular difference between virtual- and real-world imaging but just because this phenomenon can appear between any two camera types, even if both operate in the real world. As DPM is one of the most successful object detection methods, the ability to adapt such a rich model between different domains is essential. The overall aim of the thesis is to provide methods for performing *domain adaptation* (DA) of DPMs.

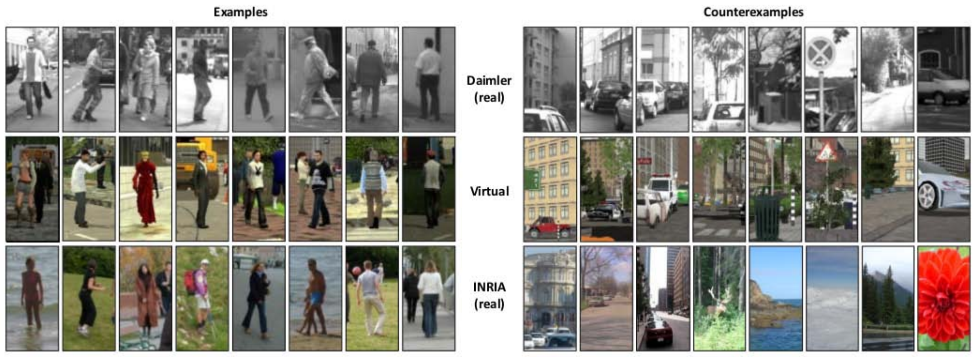


Figure 1.4: Training samples (pedestrians and background) from different datasets. The virtual-world training samples are shown in the second row. The real-world ones: Daimler [30] and INRIA [15] are shown in the first and third row respectively.

We can see a DPM as a particular case of a structural model, where the components and parts define the structure. Accordingly, we formulate the learning of a DPM as a general latent *structural SVM* (SSVM) [102, 118, 124]. Therefore, we cast the DA of a DPM as a particular case of adapting general structural models. In this context, we propose an *adaptive structural SVM* (A-SSVM) method motivated by the adaptive SVM (A-SVM) [116]. Furthermore, since A-SSVM works irrespective of the model structure (*e.g.*, the parts and components in a DPM), we also propose a *structure-aware A-SSVM* (SA-SSVM) method. Remarkably, neither A-SSVM nor SA-SSVM need to revisit the training data from the source domain, instead a relatively low number of training samples (*i.e.*, object instances) from the target domain are needed to adapt the structural model that has been initially learned in the source domain.

Although A-SSVM and SA-SSVM only require a few manually annotated target-domain samples for the adaptation, we also address the more challenging situation of even avoiding such manual annotations. In particular, we have devised an iterative method for automatically discovering and labeling samples in the target domain and re-training an adapted classifier with them using either A-SSVM or SA-SSVM. Our method applies a self-paced learning (SPL) strategy [64] to re-train an initial model with increasingly difficult target-domain examples in an iterative way without requiring source-domain data. The proper definition of what is an *easy/difficult* sample (example or counter-example) is essential for the SPL. However, in general it turns out that discovering easy/difficult samples in a new domain is a non-trivial task. We apply Gaussian process regression (GPR) for performing such a *sample selection*, which simplifies the SPL optimization procedure proposed in [64]. We call our proposal *self-adaptive DPM*.

Many domain adaptation methods assume a single domain shift between the data, *i.e.*, they perform the adaptation from a single source domain to a single target domain [9, 26, 56, 61, 87, 91, 99, 100]. Some others consider multiple source domains

[27, 49, 54, 66, 116] and propose to leverage labeled data from them to perform the domain adaptation, *i.e.*, the underlying idea is to cover as much variability as possible at the source level for making more accurate predictions given a new domain (the target). Instead of performing isolated single-layer adaptations, we propose to make use of the relatedness of multiple target domains while exploiting their differences. Concretely, we organize multiple target domains into a hierarchical structure (tree) and adapt the source model to them jointly. The adaptation to intermediate nodes allows to exploit commonalities between children sub-domains, while the adaptation to the final sub-domains allows to consider their differences. Based on A-SSVM [111], we formulate the hierarchical model as *hierarchical A-SSVM* (HA-SSVM) and apply it to DPM. As an use case in pedestrian detection, we divide target real-world pedestrian dataset into high and low resolution sub-domains and perform domain adaptation with HA-SSVM. It turns out that such progressive adaptation achieves non-trivial improvement in terms of detection accuracy.

In an on-board pedestrian detection system, we can find a large amount of target domains due to many factors, *e.g.*, different geographic scenarios, seasons and even illumination conditions during a day. Training an adapted detector using the aforementioned methods require some static pre-collected data. Such methods can succeed in scenarios where the distribution of the data is similar to the pre-collected training data, but may fail in many other cases. Collecting data to cover all possible scenarios is very expensive and time consuming, because the on-board system faces continuously changing environments.

In order to tackle such kind of variability, we ideally want a detector to self-adapt with sequentially added target domain data. For example, the on-board camera could learn a domain adaptive detector automatically by collecting sequences while driving. In this PhD we also propose an *online domain adaptation* method based on HA-SSVM to address the problem. The online domain adaptation is built on a hierarchical model which consists of instance detectors in the leaf nodes and a category detector at the top level. Each instance detector is an exemplar classifier which is trained online with only one pedestrian per frame. The pedestrian instances are collected by multiple object tracking and the hierarchical model is constructed dynamically according to the trajectories. The proposed method neither requires source domain data, nor labelled target domain data. By doing online domain adaptation, we do not even need to store any target domain data.

1.1 Objectives

In summary, as objectives of this PhD we address the following questions:

- How can we make use of the virtual-world data to train DPM and maximize its accuracy?
- How to adapt a virtual-world trained DPM to operate in a real-world scenario?
- How can we make use of multiple target domains and their hierarchical structures for domain adaptation?

- How to make a virtual-world trained DPM detector self-adapt to a real-world sequence without human intervention?

1.2 Contributions

Answering the previous questions has lead to the following contributions:

- We have proposed a automatic view clustering method and part annotation method based on virtual-world data for training strongly supervised DPM with parts sharing. The virtual-world trained DPM significantly improves the original DPM in detection accuracy.
- Though the virtual-world trained DPM outperforms original DPM, domain shift still exists when applying the virtual-world trained DPM to some real-world datasets. To address this problem, we have proposed adaptive SSVM (A-SSVM) and structure aware SSVM (SA-SSVM). We further proposed a self-paced learning method which incorporates Gaussian Process Regression to handle the situation where target domain labels are not available.
- We have also proposed a hierarchical adaptive SVM (HA-SSVM) which leverages multiple target domains or sub-domains for domain adaptation. Based on HA-SSVM, our multi-resolution adaptive DPM won the pedestrian detection challenge of the KITTI benchmark at the ICCV 2013 Reconstruction Meets Recognition Challenge (RMRC) workshop.
- We have extended HA-SSVM in order to perform automatic self-adaptation to the target sequence without human intervention.

1.3 Outline

The rest of this document is organized as follows. In Chapter 2, we review the state-of-the art literature related to our proposals. In Chapter 3, we introduce our proposed method to learn DPMs in the virtual world, *i.e.*, using view clustering and part annotation propagation. In Chapter 4, we present A-SSVM and SA-SSVM. In Chapter 5, we propose HA-SSVM. In Chapter 6, we extend HA-SSVM for online domain adaptation. Finally, Chapter 7 summarizes the thesis and draws the main conclusions. Although there are obliged references between chapters, we have written them to be as self-contained as possible.

Chapter 2

State of the art

This PhD is related to part-based object detection (particularly, pedestrian detection), and its domain adaptation. Pedestrian detection has been an active research topic in computer vision and pattern recognition community. Domain adaptation has been explored for different applications by machine learning community and it is becoming more and more attractive to the computer vision community. In this chapter, we review the state-of-the-art work[1] in pedestrian detection and visual domain adaptation.

2.1 Pedestrian detection

Pedestrian detection can be regarded as a canonical instance of object detection. It has been a long term active research area and has well established benchmarks and evaluation criteria. As a result, it serves as a perfect playground to explore ideas of different object detection methods. Remarkable progress has been made in pedestrian detection in the last decade. We refer to [39,40] for comprehensive study of pedestrian detection system and [8,22] for the state-of-the-art evaluation of pedestrian detection algorithms.

Following [8], we classify the state-of-the-art methods into three categories: (1) Decision Forest, (2) DPM variants, and (3) Deep networks. In Table 2.1 and Figure 2.1, we denoted these three groups by DF, DPM and DN respectively. In Table 2.1, we show the miss rate of the algorithms on Caltech testing dataset. The results are evaluated by Caltech pedestrian detection benchmark[2]. In addition to the family and miss rate, we also list the properties of each method, *e.g.*, whether they are or not part-based model (Parts), whether they use deep architectures (Deep), feature type and their training dataset. For the dataset, I $\rightarrow$ INRIA [15], V $\rightarrow$ Virtual-world [113], C $\rightarrow$ Caltech [20]. $I+$ and $C+$ stand for INRIA and Caltech with additional data respectively. $I+C$ denotes INRIA plus Caltech. In Figure 2.1, we depict the comparison

[1]Without otherwise specify, the methods and results compared in this chapter are collected on December 30th of 2014, new results may appear later than this date.

[2]`http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians`

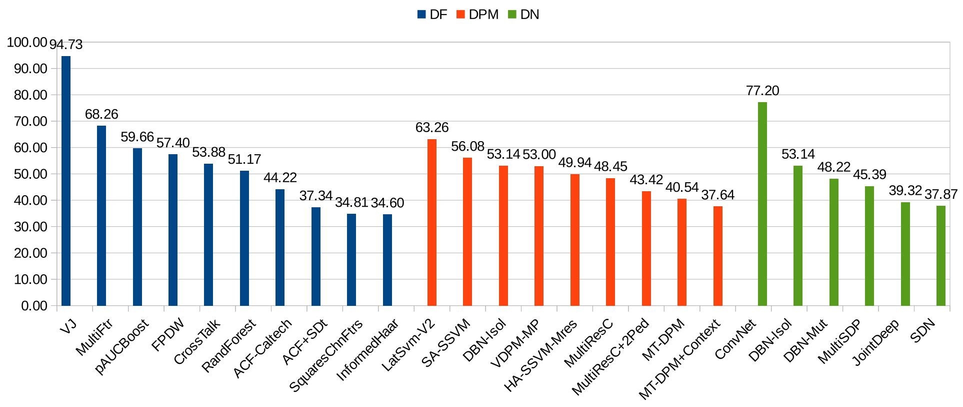


Figure 2.1: Caltech-USA detection results. The values shown are miss rates %, lower is better.

of these three types of detectors. Overall, DF, DPM and DN all reach state-of-the-art performance. In the following, we review the state-of-the-art detectors by category.

2.1.1 Decision forest

The Adaboost cascades proposed by Viola and Jones [103] (*VJ* detector) was one of the most popular detectors at the beginning of last decade. They use Haar-like features and apply Adaboost for automatic feature selection. The cascade consists of several rejection levels of Adaboost classifiers. Additionally, integral images are incorporated for fast computation of the features. In [105], the Haar-like features are further used to model motion information.

Later, Dollár *et al.* [21] extended *VJ* by computing Haar-like feature over multiple channels, including LUV color channels, gray-scale, gradient magnitude and gradient magnitude quantized by orientation (implicitly computing gradient histograms), providing a simple and uniform framework for integrating multiple feature types. This approach was further extended to fast multi-scale detection where feature computed at a single scale is used to approximate feature at nearby scales [20] (*FPDW*). Considerable efforts have also been devoted to improve the computational efficiency. Improved cascade strategy is proposed in [19] (*Crosstalk*). Combining multiple resolution models with a GPU implementation and with additional innovations, the *VeryFast* detector from Benenson *et al.* [6] achieves detection speed over 100 fps. Building on the fast feature pyramid [20], recently, Dollár *et al.* proposed aggregate channel features (*ACF*) which operates at over 30 fps while achieving top results on pedestrian detection [23]. Some other methods improve baseline detectors by considering additional data, *e.g.* *ACF-SDt* [81] exploits optical flow to provide a non-trivial improvement to *ACF-Caltech* (see Table 2.1 for details).

InformedHaar [122] obtains current top results on Caltech dataset by using a set of Haar-like features manually designed for pedestrian detection task. The design of the feature takes into account different body parts, *i.e.*, head, upper body and lower

Table 2.1: Listing of methods tested on Caltech-USA, sorted by log-average miss-rate (lower is better).

Method	MR	Family	Parts	Deep	Feature Type	Training Data
VJ [104]	94.73%	DF			Haar	I
ConvNet [93]	77.20%	DN		Y	Pixels	I
HOG [15]	68.46%				HOG	I
MultiFtr [110]	68.26%	DF			HOG+Haar	I
LatSvm-V2 [35]	63.26%	DPM	Y		HOG	I
pAUCBoost [77]	59.66%	DF			HOG+COV	I
FPDW [20]	57.40%	DF			HOG+LUV	I
SA-SSVM [111]	56.08%	DPM	Y		HOG	V+C
CrossTalk [19]	53.88%	DF			HOG+LUV	I
DBN-Isol [74]	53.14%	DN	Y		HOG	I
VDPM-MP [113]	53.00%	DPM	Y		HOG	V+I
RandForest [67]	51.17%	DF			HOG+LBP	I+C
HA-SSVM-MRes	49.94%	DPM	Y		HOG	V+C
MultiResC [80]	48,45%	DPM	Y		HOG	C
DBN-Mut [76]	48.22%	DN	Y		HOG	C
MultiSDP [120]	45.39%	DN		Y	HOG+CSS	C
ACF-Caltech [23]	44.22%	DF			HOG+LUV	I
MultiResC+2Ped [75]	43.42%	DPM	Y		HOG	C+
MT-DPM [115]	40.54%	DPM	Y		HOG	C
JointDeep [73]	39.32%	DN			Color+Gradient	C
SDN [65]	37.87%	DN	Y	Y	Pixels	C
MT-DPM+Context [115]	37.64%	DPM	Y		HOG	C+
ACF+SDt [81]	37.34%	DF			ACF+Flow	C+
SquaresChnFtrs [7]	34.81%	DF			HOG+LUV	C
InformedHaar [122]	34.60%	DF			HOG+LUV	C

body. Binary and ternary Haar-like features are used. Multiple channel features, which integrates color and gradient information, are also incorporated. In contrast, *SquaresChnFtrs* [7] obtains similar results while using data driven method instead of hand-crafted features.

Marín *et al.* [67] proposed a novel method (*RandForest*) which combines the classical random forest and multiple discriminant local experts. The proposed method is built on a sliding window based pedestrian detection framework and works with rich block-based feature representation such as HOG and LBP. The method provides flexibility in the learned spatial arrangement and certain robustness against partial occlusions. Additionally, the method integrates a cascade architecture which achieves not only high accuracy but also acceptable efficiency.

Receiver Operating Characteristic (ROC) curve is the most commonly adopted evaluation criterion by which to compare the detection accuracy of different algorithms. Particularly, for pedestrian detection, the partial area under the ROC curve (pAUC), typically over the range 0.01 and 1.0 false positives per image, is reported. Paisitkriangkrai *et al.* [77] proposed a novel ensemble learning method (*pAUCBoost*) which directly optimizes the partial area under the ROC curve. It achieves a maximal detection rate at a user-defined range of false positive rates using structured learning.

2.1.2 DPM variants

Another important group of methods are deformable part-based models and its variants. There is a significant body of part-based models. Even from the above boosting based methods, we can see the contribution of using part-based idea, *e.g.*, *RandForest* [67] and *InformedHaar* [122].

Based on the pictorial structure model, the discriminatively trained deformable part-based model (DPM) from Felzenszwalb *et al.* [35] is one the most successful one. In [35], an object is represented by a mixture of multi-scale deformable part-based models. The training only requires bounding boxes of the objects in a set of images, while parts are modelled as latent variables. The model is trained discriminatively in a SVM framework, which is known as latent SVM (*LatSvm-V2*). The DPM is a breakthrough to the classic HOG-SVM detector of Dalal-Triggs detector [15], which is regarded as a milestone in pedestrian detection. The visualization of DPM and the corresponding detection results are shown in Figure 2.2.

As pedestrians always appear at different scales and DPM seems to be most successful at higher resolution ones, Park *et al.* [80] proposed a multi-resolution model (*MultiResC*) that can automatically switch between parts and holistic models. The parts model is used to detect high resolution pedestrians while the holistic model is used to detect low resolution ones.

An alternative method for addressing the problem of detecting multi-resolution pedestrians is proposed by Yan *et al.* [115] (*MT-DPM*). In [115], detecting pedestrians at high and low resolutions are considered as different but related tasks. They propose to map pedestrians in different resolutions in a common space, where a shared detector are learned. The transformation matrix and parameters of the shared detector is learned in a multi-task learning framework using coordinate descent. Additionally, context information is considered in their work to further improve detection accuracy.

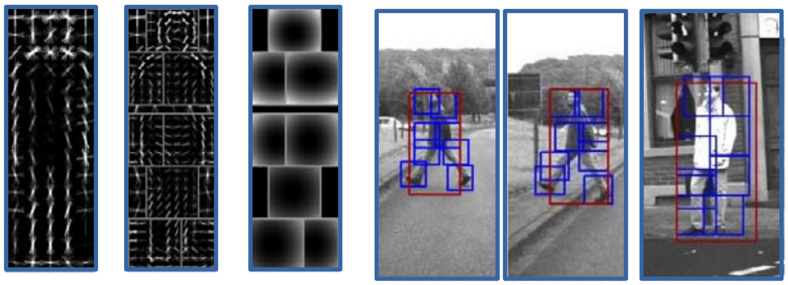


Figure 2.2: DPM model and detections [35]. From left to right: HOG model of the root; HOG model of the parts; spatial layout cost function of the parts (the darker the less deformation cost penalty); detections of root and parts in Daimler dataset.

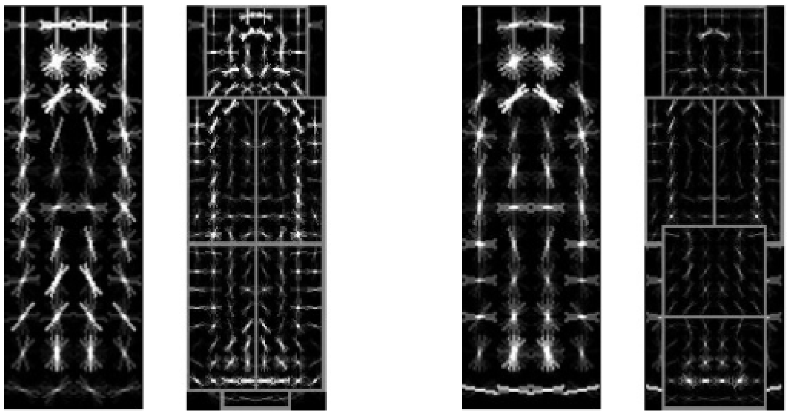


Figure 2.3: Left: original DPM [35]. Right: multi-resolution DPM of [80].

A part from the resolution challenge, occlusion handling is also one of the most difficult task for pedestrian detection. Aiming at improving detector drift and occlusion handling, Ouyang *et al.* proposed to detect single pedestrians by using multi-pedestrian detection (*MultiResC+2Ped*) [75]. A mixture model of multi-pedestrian detectors is designed to capture the visual cues which are formed by nearby multiple pedestrians but cannot be captured by single-pedestrian detectors.

2.1.3 Deep networks

Deep networks (typically convolutional neural networks) are attracting more and more interest in the computer vision community due to the state-of-the-art performance in many vision tasks, *e.g.*, object classification, scene classification [25], fine-grained category detection [121] and object detection [44]. It has also shown fast progress in pedestrian detection recently and their accuracies in Caltech benchmark can be found in Figure 2.1.

ConvNet [93] uses unsupervised convolutional sparse auto-encoders to pre-train features at all levels from the INRIA dataset, and end-to-end supervised training to train the classifier and fine-tune the features in an integrated fashion. This method obtains fair results on INRIA, ETH, and TUD-Brussels, however fails to generalise to the Caltech dataset under *reasonable* setting.

Rather than extracting features from raw pixel values as in *ConvNet*, other works use edge and colour features or initialise network weights to edge-sensitive filters (*e.g.*, *DBN-Isol* [74], *DBN-Mut* [76], *MultiSDP* [120], *JointDeep* [73], *SDN* [65]). These works focus on using deep architectures to model parts and occlusions. For example, the work in [74] aims to improve the occlusion handling of DPM. The score of the part detections is used as input and part visibility is modelled as hidden variables. A deep model (Restrict Boltzmann Machine, RBM) is used to learn the visibility relationship of overlapping parts at multiple layers. Also focusing on occlusion handling, *DBN-Mut* [76] improves *DBN-Isol* by considering mutual visibility relationship between overlapping pedestrians. *MultiSDP* [120] uses a deep model to learn a group of multi-stage classifiers, which is able to simulate cascade classifiers by mining hard samples to train the network stage-by-stage. *JointDeep* [73] formulates feature extraction, deformation handling, occlusion handling and classification into a joint deep architecture, maximizing the performance of each process. The recent work of *SDN* [65] proposed a switchable RBM (SRBM) which jointly learns features, saliency maps, and mixture representations of the whole body and different body parts in a hierarchy. *SDN* achieves superior accuracy among other deep networks models on Caltech benchmark.

2.2 Domain adaptation

Domain adaptation addresses the problem of accuracy drop that a classifier may suffer when the training data (*source domain) and the testing data (*target domain) are drawn from different distributions. The domain adaptation has been explored for different applications by the machine learning community (see [59] for a comprehensive overview). In computer vision, current DA methods can be broadly categorized in two groups, namely *feature-transform-based methods* and *model-transform-based methods.* In the following, we review the related literature on main DA methods, multiple domain adaptation and the application on object detection.

2.2.1 Main approaches for domain adaptation

Feature-transformation-based methods attempt to learn a transformation matrix/kernel over the feature space of different domains, and then apply a classifier [47,49,55,63,91]. For instance, [48,49] use labeled source and unlabeled target data to construct a manifold and learn a classifier from a projected space. The max-margin domain transforms method [56] jointly learns a feature transformation and a discriminative classifier via multi-task learning.

On the other hand, *model-transform-based approaches* concentrate on adapting the parameters of the classifiers, often SVM, including: *weighted combination* of source and target SVMs, *transductive SVM* [9, 100], *feature replication* [87, 99], and *regularization-based methods* as *A-SVM* [116] and its successor the projective model transfer SVM (*PMT-SVM*) [2]. In the following, we elaborate these SVM-based methods.

Weighted combination of SVMs is the most common approach which combines SVMs learned in the source domain and SVMs learned in the target domain [9, 86, 106, 107]. The principal drawback of these methods is that they require both source and target domain training data for the adaptation, which may be computationally expensive, or just not possible if we do not have access to the source data any more. It may even result in negative transfer (*i.e.*, the accuracy decreases for the target domain) as reported in [86]. Alternatively, a *feature replication* approach is proposed in [87], which jointly learns classifiers in both domains with augmented features, *i.e.*, source-domain data is also required. Another approach, the cross-domain SVM (*CD-SVM*) [60], selects the source domain support vectors that are close to the target domain and also adds new support vectors from the target domain to learn a new classifier. Nonetheless, in the case that the target domain data are scarce, the learned classifier may still be source domain oriented.

Among the SVM-based methods, the regularization-based ones (*i.e.*, *A-SVM*, *PMT-SVM*) have a significant advantage as they do not require revisiting source domain data for the adaptation. This would be favourable for many domain adaptation tasks in computer vision, since the source datasets are typically large and computing the features is expensive. Besides, it can even handle the case where the source data is missing at the moment of the adaptation. Basically, these methods learn the target classifier $f^T(\mathbf{x})$ by adding a perturbation function $\Delta f(\mathbf{x})$ to the source classifier $f^S(\mathbf{x})$ so that $f^T(\mathbf{x}) = f^S(\mathbf{x}) + \Delta f(\mathbf{x})$. In this PhD, the proposed *A-SSVM* and *SA-SSVM* in Chapter 4 belong to the regularization-based methods, thus share the advantages of *A-SVM* and *PMT-SVM* of not requiring source-domain training data for the adaptation process. Furthermore, our methods take into account structure knowledge in feature space.

As deep models have emerged as state-of-the-art in large-scale object classification. Recent work of [25] started to investigate DA with deep convolutional networks.

Table 2.2: Comparison of DA methods for pedestrian detection.

	Classifier	Part-based	Prior model	Require source data	Labeled target data	Unlabeled target data
Cao *et al.* [14]	Boosting			Y	Y	
Pang *et al.* [79]	Boosting			Y	Y	
Vázquez *et al.* [99, 101]	SVM			Y	Y	
Vázquez *et al.* [100]	T-SVM		Y	Y	Optional	Y
Wang *et al.* [106, 107]	SVM		Y	Y	Optional	Y
Xu *et al.* [114]	LDA, Boosting			Y	Y	
Donahue *et al.* [24]	PMT-SVM		Y	Y	Y	Y
This work	SSVM	Y	Y		Optional	Y

It is demonstrated that domain shift remain existing even for the deep networks trained with substantial amount of data *e.g.*, images from ImageNet, implying new challenges of DA on deep networks.

2.2.2 Domain adaptation between multiple domains

In the context of domain adaptation between multiple domains, most of the focus is on multiple sources, little attention is paid on the relation of multiple target domains. In the contrary, in this PhD, our proposed HA-SSVM in Chapter 5 method aims to leverage multiple target domains by considering their hierarchical structural relations. Several methods close to our work have been proposed in the natural language processing (NLP) community [36, 88], which are Bayesian-based approaches.

Most of the domain adaptation algorithms are validated assuming that the underlying domains are well-defined. However, multiple unknown domains may exist [54]. In fact, in some cases image data is difficult to manually divide into discrete domains required by adaptation algorithms [46]. In [54], a sub-domain discovery algorithm is proposed, it focuses on discovering multiple hidden source domains. The most recent work of [46] can discover domains among both training and testing data, which benefits existing multi-domain adaptation algorithms. Incorporating [46], our proposed HA-SSVM can also be applied to discover and adapt to multiple hidden target sub-domains.

2.2.3 Domain adaptation for object detection

Most of the related work on DA for computer vision tasks are focused on object recognition [91], while its application to object detection is quite limited. Table 2.2 briefly compares recently proposed DA methods for pedestrian detection. Among these methods, [14, 79, 114] are boosting-based approaches while the others are SVM-based.

Wang *et al.* [106, 107] use a weighted combination to adapt a generic pedestrian detector to a specific scene. Recently, Donahue *et al.* [24] proposed a semi-supervised DA approach which combines an instance-constrained manifold regularization with the PMT-SVM, where a few labeled target domain examples are required. In [86], DA is applied to adapt an object detector from video to images. However, only a weighted combination of source and target classifiers is explored for DPM.

Our previous work of [99,101] investigated the adaptation of a holistic pedestrian model trained with virtual-world samples to operate on real-world images. Using a framework called V-AYLA, virtual-world samples and real-world ones are fused for training and adapting a model within the so-called *cool world*. In these works the focus is on relevant pedestrian descriptors (HOG and LBP [99], Haar and EOH [101]) as well as on the type of complementarity between virtual- and real-world data. In this PhD, we go beyond and focus on a state-of-the-art pedestrian detection method, namely the DPM, providing not only adaptation of pedestrian descriptors but also of the deformable model and the multiple components (A-SSVM, SA-SSVM). Our previous work of [100] also investigated the use of an iterative unsupervised DA technique for the holistic pedestrian detector based on HOG/Lin-SVM. This technique is based on Transductive SVM and, in fact, has turned out to be rather time consuming since both labeled and unlabeled samples are used to learn during each iteration. In this PhD, instead of using a fixed threshold, our self-adaptive DPM uses a combination of self-paced learning (SPL) and Gaussian process regression (GPR) to handle unlabeled target domain samples. Moreover, since we do not need source-domain data for the adaptation, the learning algorithm is faster than the one in [100].

A part from pedestrian detection, DA is also explored for other detection tasks recently, *e.g.*, cars, faces and general objects. In [69], transform component analysis [78] is used to adapt a car detector to the target domain but the overall accuracy may be limited by the holistic detector. Focusing on face detection, the online DA of [58] treats different distributed samples in a single image as different domains and applies GPR for re-scoring. A sufficient number of examples is required per-image in order to perform DA, but the adaptation may be poor if the target image contains very few examples. Recently, a fast adaptation technique base on linear discriminate analysis (LDA) is proposed in [45], which can be used for interactive adaptation of real-time object detectors. As deep models has emerged as state-of-the-art methods for large-scale classification, Hoffman *et al.* treat the transformation of classifiers into detectors as domain adaptation task and proposed a method for large scale detection through adaptation [57].

2.3 Summary

In this chapter, we reviewed the state-of-the-art work related to pedestrian detection and domain adaptation. Compare to the reviewed work, the main contributions of this PhD can be summarized as follows:

- This PhD aims to address the problem of domain adaptation in pedestrian detection. Although domain adaptation has been successfully applied in many problems, it is not well exploited in pedestrian detection, which however is an emergent real-world problem.

- As it is reviewed in this chapter, DPM and its variants have shown superior accuracy in pedestrian detection, we choose DPM as our base detector. As we use virtual-world dataset as our source domain, we improved the conventional DPM by using part-level annotation automatically extracted from virtual world (Chapter 3).

- Based on the virtual-world trained DPM, we proposed two DA methods , namely A-SSVM and SA-SSVM (Chapter 4). These two methods extended adaptive SVM for structured SVM. For SA-SSVM, it can further handle part-level adaptation.
- Conventional multiple domain adaptation focus on multiple source domains, while the proposed HA-SSVM (Chapter 5) leverage multiple target domains for DA. Further more, HA-SSVM considered domain hierarchical relationship, which can perform progressive adaptation.
- Online DA has been seem as a challenging problem and rarely exploited in the literatures. Online DA for pedestrian detection based on DPM is first studied in this PhD (Chapter 6). The proposed online DA method HOLDA is naturally extended from HA-SSVM and it is shown that comparable accuracy to the batch DA methods could be achieved.

Chapter 3

Learning deformable part-based models in virtual world

Detecting pedestrians with on-board vision systems is of paramount interest for assisting drivers to prevent vehicle-to-pedestrian accidents. The core of a pedestrian detector is its classification module, which aims at deciding if a given image window contains a pedestrian. Given the difficulty of this task, many classifiers have been proposed during the last fifteen years. Among them, the so-called (deformable) part-based classifiers including multi-view modeling are usually top ranked in accuracy. Training such classifiers is not trivial since a proper aspect clustering and spatial part alignment of the pedestrian training samples are crucial for obtaining an accurate classifier. In this chapter, first we perform automatic aspect clustering and part alignment by using virtual-world pedestrians, *i.e.*, human annotations are not required. Second, we use a mixture-of-parts approach that allows part sharing among different aspects. Third, these proposals are integrated in a learning framework which also allows to incorporate real-world training data to perform domain adaptation between virtual- and real-world cameras. Overall, the obtained results on four popular on-board datasets show that our proposal clearly outperforms the state-of-the-art deformable part-based detector known as latent SVM.

3.1 Introduction

Learning accurate classification is primarily important for vision-based pedestrian detectors. However, it turns out to be a difficult task due to the large intra-class variability of both pedestrians and background classes, as well as the imaging and environmental conditions. Aiming at overcoming such a complexity, many pedestrian classifiers/detectors have been proposed during the last fifteen years. HOG/Lin-SVM approach has been regarded as a milestone in the field of pedestrian detection which is based on histograms of oriented gradients and linear support vector machines (HOG/Lin-SVM) [15]. However, the HOG/Lin-SVM detector treats the pedestrian model as a holistic model, thus the intra-class variability of the pedestrians is not

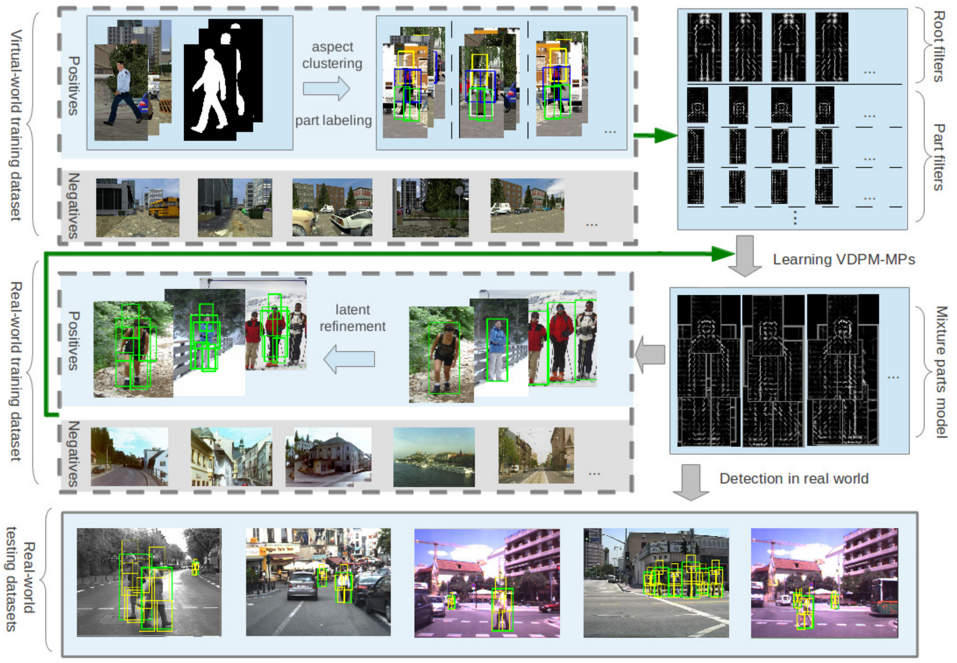


Figure 3.1: Virtual-world trained DPM with our Mixture of Parts (VDPM-MP) framework for training an aspect-based mixture of DPMs with part-sharing.

explicitly considered, neither the body-inspired parts. The accuracy of the holistic model tends to be easily saturated when increasing and/or diversifying the training samples [125].

The *deformable part-based model* (DPM) of [35] overcomes the limitations of the holistic model by taking into account different components and parts. The components are divided by the bounding box aspect ratios. For pedestrian detection, the pedestrian poses are likely to be implicitly modeled in DPM through the allowed deformation. This is more effective if view point information can be explicitly used to build a mixture model. A natural extension of this idea consists in allowing to share parts among different views, which increases the number of implicitly modeled aspects and reduces the number of overall parts to be learned and applied. Up to the best of our knowledge this approach has not been exploited in pedestrian detection for driver assistance. However, part-sharing has recently shown benefits in tasks such as object detection and pose estimation [29, 84, 117, 125].

In this chapter, we proposes a new aspect-based mixture of DPMs with part-sharing. A key point of such a pedestrian model is to have pedestrian samples with reliable and rich annotations. In particular, for each pedestrian, its full-body BB is required along with the BB of its constituent parts, and its aspect label (*e.g.*, either rear-frontal, side-left, side-right). Collecting all this information by human annotation is a tiresome task prone to errors. Thus, other than [4,11,13,29,32,85,94,95,117,125], we propose the use of a virtual-world with automatic pixel-wise pedestrian ground truth. In our first work in this line [68] a single holistic pedestrian classifier trained with virtual-world data performed equally well in automotive real-world images than an equivalent one trained with real-world data. For building our pedestrian model, in this chapter we also exploit part labeling (*i.e.*, part BBs) and aspect clustering, both automatically obtained from the pixel-wise groundtruth.

In [99,100], we show that between virtual- and real-world data domain shift problem exists, *i.e.*, the detector trained in virtual-world dataset suffers from accuracy degradation when applied to real-world datasets. In this chapter, we show how fusing virtual-world training data with a relatively few real-world training data allows to adapt virtual and real domains. While looking for the best *domain adaptation* method for our classifiers is out of the scope of this chapter, we have devised our learning framework to allow such a world's fusion and we demonstrate its effectiveness too. For that we only require the full-body BB of the real-world pedestrians, *i.e.* neither their part BBs nor aspect labels.

Figure 3.1 summarizes our DPM-based proposal. Since we rely on virtual-world data and part-sharing is implemented as a mixture of parts, we term this proposal as VDPM-MP. We test it on four popular on-board datasets focusing on luminance images and HOG features. The results show that VDPM-MP outperforms the state-of-the-art DPM proposed in [35] based on HOG-inspired features and latent SVM (HOG/Lat-SVM).

The rest of the chapter is organized as follows. Section 3.2 introduces the processing of the virtual-world dataset, including aspect clustering and part labelling. Section 3.3 details the training of start model and mixture-of-parts model using virtual-world dataset. Section 3.4 details the testing datasets and evaluation protocol. Section 3.5 gives the experimental results. Finally, Section 3.6 summarizes the conclusions and future work.

3.2 Virtual world training data

3.2.1 Virtual-world images

For this work we have improved the dataset of [68] using the same proprietary game engine (*i.e.*, Half-Life 2). The new images contain higher quality textures and more variability in cars, buildings, trees, pedestrians, etc. Unfortunately, we have no access to the 3D information processed by the game engine. However, a precise 2D segmentation (pixel-wise groundtruth) of the imaged pedestrians is automatically available. Hence, for automatically obtaining BBs, performing aspect clustering and part labeling, we process the 2D pedestrian-segmentation masks as explained in 3.2.2 and 3.2.3. Therefore, our mechanism can also be used when manually drawn object silhouettes are available (*e.g.*, as in [32]). The pedestrian and background samples of virtual-world dataset are shown in Figure 3.2.

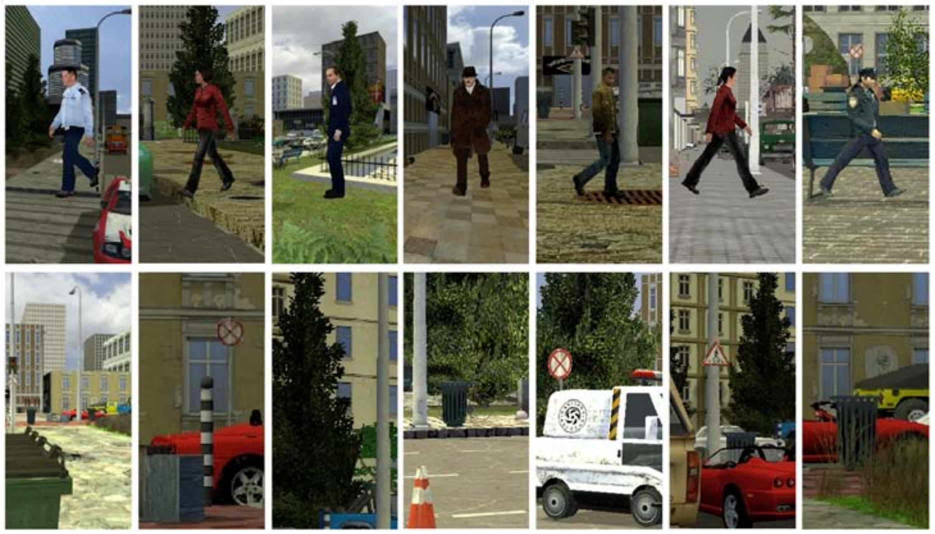


Figure 3.2: Virtual-world pedestrians and background images.

3.2.2 Aspect clustering

The silhouette of the pedestrians can be used to distinguish major aspect tendencies. The available segmentation of the virtual-world pedestrians allows to automatically delineate their precise silhouette. Thus, using a similarity function between silhouettes we can cluster them. A function that does not require point-wise matching between silhouettes is chamfer distance, which has already been successfully used for building shape-based pedestrian hierarchies from manually annotated silhouettes [18]. Given a binary template T and a binary image I, the T to I chamfer distance is defined as $Ch(T, I) = |T|^{-1} \sum_{t \in T} \min_{i \in I} \|t - i\|$, where $|T|$ denotes the area of T. In our case, both T and I are silhouettes. Since $Ch(T, I)$ is not a symmetric function in general, we use the symmetric version $S(X, Y) = Ch(X, Y) + Ch(Y, X)$.

Using $S(X, Y)$ we build a similarity distance matrix, $M(X, Y)$, for the silhouettes. Then, we can organize the pedestrians as a silhouette-based hierarchical cluster by relying on $M(X, Y)$ and K-medoids [82]. K-medoids selects a data point for each cluster center, which is important here since we will further use the *center pedestrians* for part labeling.

First, pedestrian BBs are automatically determined from the segmentation masks. The BBs are set with the same aspect ratio as the *canonical (detection) window* (CW). We crop pedestrians and masks according to the BBs. Then, all cropped windows (appearance and mask) are resized to the CW size.

Second, we exploit vertical symmetry to obtain an initial *alignment* of the pedestrians. In particular, we manually select one left-side viewed pedestrian, which is vertically mirrored to obtain its right-side counterpart (its mask is also mirrored). These two exemplars initialize K-medoids clustering for K=2. This procedure classifies our pedestrians as either left or right aspect. Frontal/rear aspects are assigned to one or another category depending on their aspect tendency. Now, the pedestrians classified as right-aspect are vertically mirrored and joined with the other category. Thus, we obtain a training set of pedestrians that are aspect-aligned in the left-*vs.*-right sense. Regarding the hierarchical clustering, this set of pedestrians constitutes the root level, *i.e.*, no clusters are available yet.

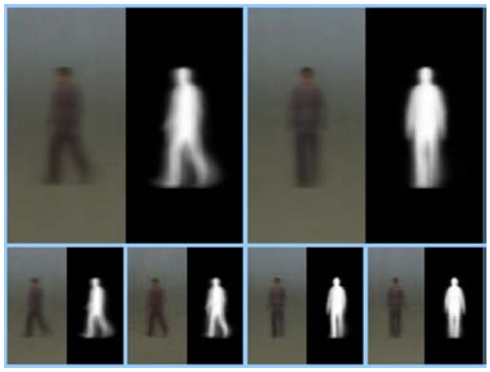
Third, we perform the hierarchical clustering. In particular, we generate a binary tree by iteratively applying K-medoids with K=2 and using $M(X,Y)$. In this case, K-medoids initialization is done just randomly. For instance, the first application of the procedure (2nd level of the hierarchy) divides the pedestrian examples of the root level as frontal/rear-*vs.*-left categories. The second application (3rd level) distinguishes different degrees of left skewness, and so on. Figure 3.3a and 3.3b show the average appearance and mask of pedestrians for the 2nd and 3rd levels of the hierarchy, as well as the *mirrored hierarchy* generated by vertically mirroring the pedestrian examples at each node of the binary tree.

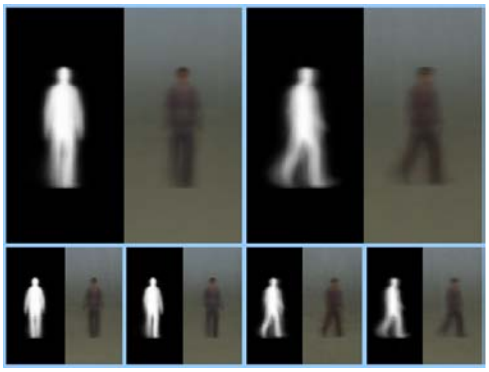
3.2.3 Part labeling

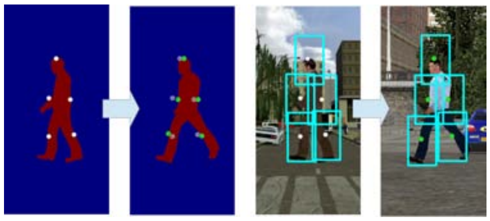
We assume the usual settings of the state-of-the-art part-based models [35, 50, 94], *i.e.*, a fixed number of parts annotated as rectangular sub-windows, where each part rectangle is of fixed size but where such size can vary from part to part. In the deformable case (DPM) the location of the parts changes from one pedestrian example to another. Since we focus on DPMs, we have to provide a procedure to automatically label the parts for each example. Currently we follow the hierarchical cluster described in 3.2.2.

In particular, we select the pedestrian masks representative of the 2nd level clusters, *i.e.*, one exemplar for the frontal/rear aspect and another for the left one. We manually point the parts' centers of these two exemplars. For instance, we can roughly focus on head and extremities, *i.e.*, five parts, and then quickly clicking ten pixels to be these centers. The parts' centers are automatically propagated through the hierarchy, from the 2nd level to the bottom level. From level to level, the centers are propagated between the representatives of cluster nodes. The representatives of the bottom-level clusters propagate the centers to all the pedestrian examples within their respective clusters. Overall, by manually clicking ten points, we can obtain part labeling for thousands of pedestrians.

Propagating the centers from one pedestrian example e_1 to another e_2 is done by a simple but effective procedure. For that we use the distance transform (DT) of the different examples. Since chamfer distance involves DT computation, all pedestrian DTs are already available from the hierarchical clustering. Let c_i^p be the center of the part p of the pedestrian example e_i, and D_j the DT of the example e_j. In order to map c_1^p into c_2^p, the new center c_2^p is defined as the silhouette pixel of e_2 which is at minimum distance $D_2(c_1^p)$ from c_1^p. If the condition holds for more than one pixel, we just choose one at random since they must be quite close and thus the choice will not affect the final pedestrian model. Figure 3.3c and 3.3d illustrate the idea. Note that, like for the hierarchical cluster, here we can define also the *vertically mirrored parts*.


(a)


(b)


(c)

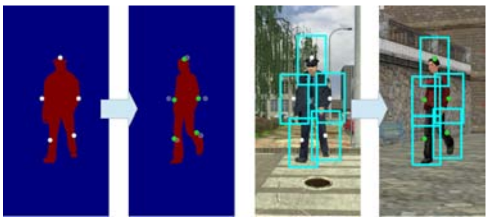

(d)

Figure 3.3: (a) 2nd (top row) and 3rd (bottom row) levels of the silhouette-based hierarchy. The average appearance and segmentation mask of each cluster node are shown. (b) *Mirrored hierarchy.* (c) Part centers marked in the pedestrian mask representative of the left 2nd level cluster, and their automatic propagation to the representative of one of the left subcategories (a 3rd level cluster node). The respective parts are also shown on the corresponding appearance windows. (d) Analogous for the frontal/rear case.

3.3 Pedestrian classifier

In this work, pedestrians are modeled according to their full-body appearance, as well as by the appearance of K body-inspired parts. Such appearances are evaluated by corresponding learned image filters. The size of these filters can be different from part to part. However, each individual filter size is fixed. Contrarily, the location of the parts can vary with respect to the overall full-body location. There are part locations more plausible than others, therefore, there is a deformation penalty given by a deformation cost function. Overall, this is the description of a deformable part-based model (DPM). Moreover, in order to search for pedestrians at multiple resolutions, a pyramidal sliding window is assumed and, following [35], we also assume that parts are detected at twice the resolution of the root.

In addition, we consider different aspect models, thus, our pedestrian model ac-

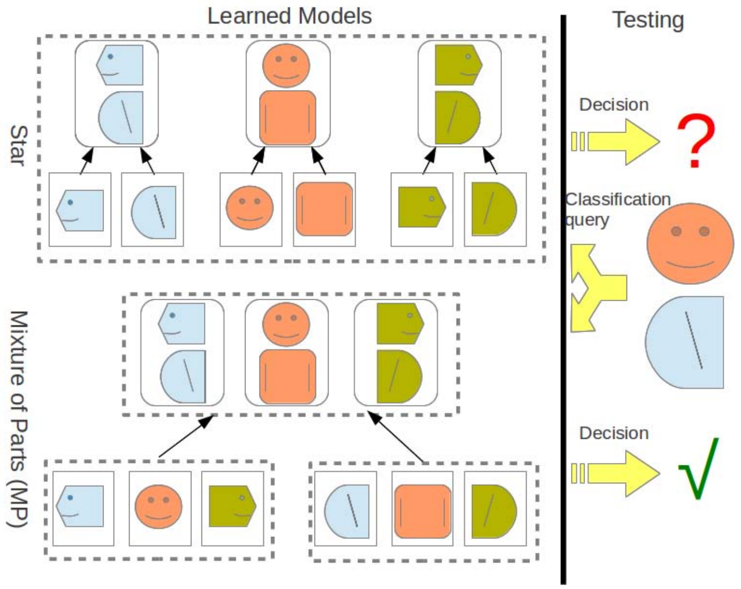


Figure 3.4: Part-sharing allows to model unseen aspects. Imagine the testing sample at the right was not present in the training data. Then, the star model does not include a combination of parts (head/trunk here) corresponding to this testing sample. The MP models part combinations that were not seen during training, thus, it has more chances to rightly classify such a sample.

tually is a *mixture model* of M components (aspect-based mixture of DPMs). When using more than one component we have to decide whether to share parts among components or not (Fig. 3.4). In [35] parts are not shared among components, which corresponds to a star structure. Not sharing parts can lead to a large number of them, while sharing the parts reduces this number and allows to model aspect configurations not explicitly seen during training time. Part-sharing has been successfully used for pose estimation [117] and to share parts among different classes of objects [29]; manual part annotations are required in these works though.

3.3.1 Mixture of DPMs: star model (VDPM-Star)

For describing our pedestrian model we mainly follow the notation of [35] since this is the state-of-the-art multi-component DPM which we take as baseline.

We call $\mathbf{x}$ the pyramid of features built from the image window under consideration. Let $\mathbf{p} = [u, v, s]'$ specify a position $[u, v]$ in the s-th level of $\mathbf{x}$. For instance, if $\mathbf{x}$ is a pyramid with HOG information, then $\mathbf{x}(\mathbf{p})$ contains the features corresponding to a cell of HOG. We term as $\phi_a(\mathbf{x}, \mathbf{p}, \bar{w}, \bar{h})$ the vector obtained by concatenating the feature vectors of a $\bar{w} \times \bar{h}$ sub-window of $\mathbf{x}$ with top-left corner at $\mathbf{p}$ in row-major order. We use $\phi_a(\mathbf{x}, \mathbf{p})$ for short. Let $\mathbf{F}$ be a $\bar{w} \times \bar{h}$ filter, arranged as a vector, *i.e.*, as for the sub-windows of $\mathbf{x}$. We can compute the score of $\mathbf{F}$ at $\mathbf{p}$ as $\mathbf{F}'\phi_a(\mathbf{x}, \mathbf{p})$, where hereinafter we have simplified the notation by assuming equal dimension of filters and sub-windows, and the use of $\mathbf{x}$ underlying appearance features computation (for the deformation features defined later too). Following with the HOG example, note that each entry of $\mathbf{F}$ actually contains a vector of weights for the bins of the four histograms of a HOG cell. In other words, if we think in terms of the traditional HOG/Lin-SVM framework, the filter $\mathbf{F}$ contains the weights learned by the Lin-SVM procedure.

A DPM is then defined by a $(K+2)$-tuple $[\mathbf{F}_0, \mathbf{P}_1, \ldots, \mathbf{P}_K, b]'$, where $\mathbf{F}_0$ is the root filter of size $\bar{w}_0 \times \bar{h}_0$, $\mathbf{P}_i$ describes part i, and b is a real-valued bias term. In particular, $\mathbf{P}_i = [\mathbf{F}_i, \mathbf{d}_i]'$, where $\mathbf{F}_i$ is the filter of part i with size $\bar{w}_i \times \bar{h}_i$, and $\mathbf{d}_i$ is a 4D vector of coefficients of a quadratic function, $\phi_d(du, dv) = [du, dv, du^2, dv^2]'$, that defines the cost of deviating from the anchor position (*i.e.*, the deformation cost).

Now, let $\mathbf{h} = [\mathbf{p}_0, \ldots, \mathbf{p}_K]'$ be a pedestrian hypothesis, *i.e.*, an assumption about where the root and the K parts are located within $\mathbf{x}$, subject to $s_i = s_0 - l$ for $i > 0$, where l defines the number of levels needed to double the resolution. This hypothesis will be validated (*it is pedestrian*) or rejected by thresholding a score, say $f(\mathbf{x}, \mathbf{h})$, which accounts for the scores of the appearance filters in their respective positions as well as the deformation cost of each part, plus the bias term, *i.e.*,

$$f(\mathbf{x}, \mathbf{h}) = \sum_{i=0}^{K} \mathbf{F}_i \cdot \phi_a(\mathbf{x}, \mathbf{p}_i) - \sum_{i=1}^{K} \mathbf{d}_i \cdot \phi_d(du_i, dv_i) + b \ , \tag{3.1}$$

where $[du_i, dv_i]$ is the displacement of the i-th part relative to its anchor point. We can express (3.1) in compact form as

$$f(\mathbf{x}, \mathbf{h}) = \mathbf{w}'\Phi(\mathbf{x}, \mathbf{h}) \ , \tag{3.2}$$

where $\mathbf{w}$ is a vector of model parameters and $\Phi(\mathbf{x}, \mathbf{h})$ is a vector with the appearance and deformation of hypothesis $\mathbf{h}$ as observed in $\mathbf{x}$, *i.e.*,

$$\begin{aligned} \mathbf{w} &= [\mathbf{F}_0, \ldots, \mathbf{F}_K, \mathbf{d}_1, \ldots, \mathbf{d}_K, b]' \ , && (3.3)\\ \Phi(\mathbf{x}, \mathbf{h}) &= [\phi_a(\mathbf{x}, \mathbf{p}_0), \ldots, \phi_a(\mathbf{x}, \mathbf{p}_K), && (3.4)\\ &\quad -\phi_d(du_1, dv_1), \ldots, -\phi_d(du_K, dv_K), 1]' \ . \end{aligned}$$

Based on $f(\mathbf{x}, \mathbf{h})$ we can follow [35] to apply an efficient pedestrian search within an input image.

The parameters in $\mathbf{w}$ must be learned from a set of labeled samples. For Lin-SVM learning with hinge loss, $\mathbf{w}$ can be obtained by solving the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{w}} \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \textstyle\sum_{i=1}^{N} \xi_i \ , \\ &\text{s.t. } \forall i \in [1..N] : \xi_i \geq 0 \wedge (\mathbf{w}'\Phi(\mathbf{x}_i, \mathbf{h}_i)) y_i \geq 1 - \xi_i \ , \end{aligned} \tag{3.5}$$

where, $\{(\mathbf{x}_1, \mathbf{h}_i, y_1), \ldots, (\mathbf{x}_N, \mathbf{h}_N, y_N)\}$ is the set of training samples, with $y_i \in \{+1, -1\}$ labeling sample $\mathbf{x}_i$ as pedestrian $(+1)$ or background (-1). Here we assume $(\mathbf{x}_i, \mathbf{h}_i) = [\mathbf{p}_{0,i}, \ldots, \mathbf{p}_{K,i}]$, *i.e.*, when $y_i = +1$ both the BBs of the root and parts of pedestrian i are provided (see Sect. 3.2.3), while for $y_i = -1$ such BBs can be just sampled from background patches. Moreover, note that $\mathbf{x}_i$ is expressed with respect to the coordinates of the pyramid of features, computed from the original image containing the annotated pedestrian corresponding to $\mathbf{h}_i$. These coordinates already encode resizing pedestrians to CW size.

If only pedestrian root BBs are annotated but not part BBs, this optimization problem is not convex and the Lat-SVM algorithm must be applied by treating root and part BBs as latent information [35]. Lat-SVM is basically a coordinate descent method where holding $\mathbf{w}$ fixed, the root and part BBs are optimized (manually annotated root BBs and ad hoc part BBs relative to such root BBs are used for initialization); then assuming that such BBs are right, $\mathbf{w}$ is optimized.

The basic DPM can be extended to account for M components (*e.g.*, *views*), *i.e.*, the new model can be thought as a mixture of DPMs. Each component has its associated $\mathbf{w}_c$ and $\Phi_c(\mathbf{x}, \mathbf{h})$ vectors, $1 \leq c \leq M$. The score function in this case can be defined as $f(\mathbf{x}, \mathbf{h}) = \max_{c \in [1..M]} \mathbf{w}'_c \Phi_c(\mathbf{x}, \mathbf{h})$. Now, the parameters to be learned take the form of a vector

$$\mathbf{w} = [\mathbf{w}_1, \ldots, \mathbf{w}_M]' \ . \tag{3.6}$$

Again, $\mathbf{w}$ can be obtained by solving the optimization problem in (3.5), where in this case

$$\Phi(\mathbf{x}, \tilde{\mathbf{h}}) = [0, \ldots, 0, \Phi_c(\mathbf{x}, \mathbf{h}), 0, \ldots, 0]' \tag{3.7}$$

for $\tilde{\mathbf{h}} = [c, \mathbf{h}']'$. Note that $\Phi(\mathbf{x}, \tilde{\mathbf{h}})$ is a sparse vector, *i.e.*, all its entries are zero but those of the component c corresponding to $\mathbf{h}$. Accordingly, the training samples are of the form $[\mathbf{x}_i, \tilde{\mathbf{h}}_i, y_i]$. Note that in our case the components are aspects and, thus, during training the aspect information (*i.e.*, c_i) of the pedestrian samples is known (see Sect. 3.2.2), while for the background ones it can be set randomly. This mixture of DPMs just inherits the star structure of the basic DPM for each mixture component. Therefore, and given the fact that we rely on virtual-world data, we term it as VDPM-Star.

As in the single component case, given an image we use the new score $f(\mathbf{x}, \tilde{\mathbf{h}})$ for finding pedestrians following [35].

3.3.2 Mixture of Parts model (VDPM-MP)

The star structure limits the parts to be connected to a single root component. Therefore, sharing parts among different components is not possible. Moreover, when increasing the number of components, the number of part filters grows accordingly. In contrast, models allowing part-sharing can avoid both problems. We follow the mixture of parts (MP) idea presented in [117] for pose estimation, which is based on a tree organization. In particular, a node of the tree conveys different aspects of the same type of part, *e.g.*, one node can include different head aspects, another node can incorporate different trunk aspects, and so on. Moreover, there is a deformation cost between part aspects of child and father nodes.

In this chapter we incorporate several contributions with respect to [117]. First, rather than using just a collection of constituent parts, we also use a root which is treated as a special part. Second, in this case we also detect parts at twice the resolution of the root. Third, as in the DPM seen so far, the deformation cost of all the parts are with respect to the root. Thus, our model is a tree with only two layers. In the first layer (root node of the tree) we have different pedestrian root aspects. In the second layer, we have different nodes, each one being dedicated to a different type of part (*i.e.*, view induced head aspects, left-arm aspects, etc.). Figure 3.4 conceptualizes the idea. Note how any part aspect can be combined with any root aspect. Thus, the variety of modeled pedestrians that were not explicitly seen during training is larger than for star-like models [125], while increasing the number of aspects of a given part (*e.g.*, root aspects) does not require doing the same for the other parts.

Interestingly, by defining the proper $\mathbf{w}$ and Φ vectors, the learning of $\mathbf{w}$ drives us to (3.5) again. First, we define

$$\mathbf{w} = [\Gamma_0, \ldots, \Gamma_K, \Delta_1, \ldots, \Delta_K, b]' \ , \tag{3.8}$$

where

$$\Gamma_i = [\mathbf{F}_i^1, \ldots, \mathbf{F}_i^k]', 0 \le i \le n \ , \tag{3.9}$$

conveys the appearance filters of part i ($i = 0$ refers to the root) for k aspects, while

$$\Delta_i = [\mathbf{d}_i^{1,1}, \ldots, \mathbf{d}_i^{1,k}, \ldots, \mathbf{d}_i^{k,1}, \ldots, \mathbf{d}_i^{k,k}]', 1 \le i \le n \ , \tag{3.10}$$

are the deformation cost parameters of part i with respect to the root, where $\mathbf{d}_i^{a_i,a_0}, 1 \le a_i, a_0 \le k$, stands for the deformation cost parameters of the aspect a_i of part i with respect to the aspect a_0 of the root. We note that, without losing generality, in this work we use the same number of aspects (*i.e.*, k) for each type of part provided it is relatively low (*e.g.*, four in the experiments of Sect. 3.5), otherwise it is straightforward to consider different k_i.

Accordingly, we can define the feature vector $\Phi(\mathbf{x}, \mathbf{h})$ as

$$\begin{aligned} \Phi(\mathbf{x}, \mathbf{h}) \quad = \quad & [\Phi_a(\mathbf{x}, \mathbf{p}_0), \ldots, \Phi_a(\mathbf{x}, \mathbf{p}_K), \\ & -\Phi_d(\delta u_1, \delta v_1), \ldots, -\Phi_d(\delta u_K, \delta v_K), 1]' \ , \end{aligned} \tag{3.11}$$

where

$$\Phi_a(\mathbf{x}, \mathbf{p}_i) = [\phi_a(\mathbf{x}, \mathbf{p}_i^1), \ldots, \phi_a(\mathbf{x}, \mathbf{p}_i^k)]', 0 \le i \le n \ , \tag{3.12}$$

contains the appearance features at $\mathbf{p}_i^{a_i}$, $1 \le a_i \le k$, *i.e.*, the location $\mathbf{p}_i$ for the different aspects a_i. Now, we define the vector $\delta u_i = [du_i^{1,1}, \ldots, du_i^{1,k}, \ldots, du_i^{k,1}, \ldots, du_i^{k,k}]'$ and analogously for δv_i, where $[du_i^{a_i,a_0}, dv_i^{a_i,a_0}], 1 \le a_i, a_0 \le k$, stands for the displacement of aspect a_i of part i with respect to aspect a_0 of the root. Accordingly, we have

$$\begin{aligned} \Phi_d(\delta u_i, \delta v_i) = [&\phi_d(du_i^{1,1}, dv_i^{1,1}), \ldots, \phi_d(du_i^{1,k}, dv_i^{1,k}), \\ &\ldots, \phi_d(du_i^{k,1}, dv_i^{k,1}), \ldots, \phi_d(du_i^{k,k}, dv_i^{k,k})]', \\ & 1 \le i \le n \ . \end{aligned} \tag{3.13}$$

Again, hypothesis of training samples are of the form $\tilde{\mathbf{h}}_i = [c_i, \mathbf{h}_i]'$. The c_i label is used as aspect index. When forming the $\Phi(\mathbf{x}, \tilde{\mathbf{h}}_i)$ vectors for the optimization in (3.5), all appearance related entries are zero but those indexed by aspect index c_i. Regarding the deformation cost entries, the situation is analogous but taking into

Algorithm 1 VDPM Optimization

```
Assume $\mathbf{w}$ and $\Phi$ defined by (3.6) and (3.7) for VDPM-Star, or by (3.8)
and (3.14) for VDPM-MP. Inputs $\mathcal{S}^+$ and $\mathcal{S}^-$ stand for positive and negative
training data, respectively, while $\mathrm{D}_{in}$ is an initial pedestrian detector.
Optimize($\mathcal{S}^+, \mathcal{S}^-, \mathrm{D}_{in}$)
$\mathrm{D}_{out} \leftarrow \mathrm{D}_{in}$
while the optimization does not finish do
   1. Compute the $\Phi$'s as follows:
     1.a. Run Detect($\mathrm{D}_{out}, \mathcal{S}^+$) to obtain the $\phi_a$'s and $\phi_d$'s
     of the pedestrians.
     1.b. Run HardNeg($\mathrm{D}_{out}, \mathcal{S}^-$) to obtain the $\phi_a$'s and
     $\phi_d$'s of the background examples.
   2. Using the $\Phi$'s, solve (3.5) to obtain $\mathbf{w}$.
   3. Update $\mathrm{D}_{out}$ according to the new $\mathbf{w}$.
end while
return $\mathrm{D}_{out}$
```

account that the displacement of each part must be related to all roots, not only to the root whose aspect is indexed by c_i. Note that displacements from any aspect of any part to any root aspect can be computed because during the training all the examples are used according to the CW size. Accordingly, we obtain feature vectors of the form

$$\begin{aligned}\Phi(\mathbf{x}, \tilde{\mathbf{h}}_i) \quad = \quad & [0, \ldots, \phi_a(\mathbf{x}, \mathbf{p}_0^{c_i}), \ldots, 0, \ldots, \phi_a(\mathbf{x}, \mathbf{p}_K^{c_i}), \ldots, 0, \\ & 0, \ldots, \phi_d(du_1^{c_i,1}, dv_1^{c_i,1}), \ldots, \phi_d(du_K^{c_i,1}, dv_K^{c_i,1}), \\ & \ldots, \phi_d(du_1^{c_i,k}, dv_1^{c_i,k}), \ldots, \phi_d(du_K^{c_i,k}, dv_K^{c_i,k}), \\ & \ldots, 0, 1]' \ .\end{aligned} \tag{3.14}$$

We remark that in this case the annotation of the parts and the aspects is strictly necessary. In [117] a manual process is followed to obtain such a rich ground truth, while here we use the virtual world for automatically obtaining it. Accordingly, we term this pedestrian model as VDPM-MP. With the VDPM-MP $f(\mathbf{x}, \mathbf{h})$, we search pedestrians in images following [35].

3.3.3 Training Framework

Algorithms 1 and 2 summarize the training of our VDPMs. We have coded it within the Lat-SVM V5 framework so that comparisons with such a state-of-the-art method are fair.

Algorithm 1 is at the core of Lat-SVM. **HardNeg()** is the data mining procedure used in [35] for collecting hard negatives. **Detect()** has the purpose of self-annotating components (aspects) and parts in training pedestrians, *i.e.*, estimating the Φ's of (3.5) during Lat-SVM learning. Hence, we can adopt **HardNeg()**, while the use

Algorithm 2 VDPM Training

```
*Mandatory*: $\mathcal{V}$, virtual-world data with pixel-wise ground truth for pedestrians as well as pedestrian-free images.
*Optional*: $\mathcal{R}^+$, real-world data with root BB annotations for pedestrians, and $\mathcal{R}^-$ with pedestrian-free images.
**1. Automatic annotation steps.**
**1.a.** Obtain $\mathcal{V}^-$, the pedestrian-free virtual-world images.
**1.b.** Obtain $\mathcal{V}_0^+$ as the complement of $\mathcal{V}^-$ in $\mathcal{V}$.
**1.c.** Obtain $\mathcal{V}^+$ from $\mathcal{V}_0^+$ by performing the automatic annotation of aspects (Sect. 3.2.2) and parts (Sect. 3.2.3).
**2. Build an initial part-based pedestrian detector**
**2.a.** *Appearance classifiers*: using $\{\mathcal{V}^+, \mathcal{V}^-\}$ train each root and parts' initial appearance classifiers.
**2.b.** *Anchor points*: use $\mathcal{V}^+$ to fit a Gaussian mixture model (currently a GMM of five components, *i.e.*, one per part) to the cloud of points generated by considering the centers of the part BBs, independently for each aspect. The mean of each Gaussian is taken as the anchor point of a part.
**2.c.** Build an initial part-based pedestrian detector, $\mathrm{D}_0$, using the appearance classifiers and their anchor locations.
**3. Train the VDPM-MP**
$\mathrm{D}_{out} \leftarrow$ **Optimize**$(\mathcal{V}^+, \mathcal{V}^-, \mathrm{D}_0)$
**4. [optional] Virtual to real world domain adaptation**
$\mathrm{D}_{out} \leftarrow$ **Optimize**$(\mathcal{R}^+, \mathcal{R}^-, \mathrm{D}_{out})$
**return** $\mathrm{D}_{out}$
```

of **Detect()** is different depending on whether we already have BB annotations for aspects and parts (*e.g.*, as for virtual-world data), or we only have root BBs (*e.g.*, as usually for real-world data).

Accordingly, for the step 3 in Alg. 2, the **Detect()** function only needs to return the aspect and part annotations computed in the step 1 of the same algorithm. However, we have found useful to lead the current detector to search for the best detection (highest score) overlapping up to a certain amount with the provided annotations. In particular, we set to 60% such overlapping for roots and parts individually. This flexibility can be understood as a sort of *online jittering* of the training pedestrians. Augmenting the training set with jittered pedestrians is employed in [32] to be more shift invariant because, for the sake of speed, during pedestrian detection the image is explored according to a stride longer than one pixel. We do this process online, thus our pedestrian training set is not augmented. We have seen that this operation leads to gains between two or three percentage points of accuracy.

For real-world data with only root BBs, **Detect()** is exactly the corresponding step of Lat-SVM V5 training. This means that the current detector is used for collecting aspects and part annotations, but without using the prior annotation information available when training with virtual-world data (the 60% overlapping rule). Thus, step 4 of Alg. 2 consists in training with Lat-SVM V5, but initializing the process with a VDPM detector (Star or MP) based only on virtual-world data. Since VDPM detectors are accurate, they provide a good initialization for the optimization process. The rational behind this optional step is to prepare our framework for domain adaptation based on incorporating real-world data [99, 100].

Finally, the initial part-based detector of step 2 in Alg. 2 follows our proposal in [114]. Thus, we obtain an aspect-based mixture of DPMs with a star structure, with the root and parts trained independently from each other.

3.4 Datasets and evaluation protocol

Since our interest is pedestrian detection for cars, we validate our proposals in different datasets acquired on-board, namely Daimler [30], TUD-Brussels [109], Caltech-Testing [22], and CVC (*aka* CVC-02) [41]. Thus, different camera types and cities are covered. Table 3.1 provides relevant statistics of these datasets. Daimler* refers to the *mandatory* set of Daimler we used in [68]. Caltech refers to the *reasonable* testing set of Caltech dataset [22]. Additionally, to asses the accuracy on occluded pedestrians, we use our own occlusion dataset PoleSec (*aka* CVC-05) [67].

As evaluation protocol we run the widely used *Caltech* per-image evaluation [22], *i.e.*, false positives per image (FPPI) *vs.* miss rate. Detected pedestrians must be of height ≥ 50 pixels.

Most pedestrian detectors evaluated in [22] are trained with INRIA training set [15]. Thus, for comparing our proposals with respect to them, we use such INRIA data for adapting virtual world to real one. Regarding domain adaptation, here we only focus on combining all the available virtual and real data, *i.e.*, we leave for the next chapters to incorporate sophisticated DA techniques.

Our virtual-world training set (*aka* CVC-07)[1] contains 2,500 pedestrians and 2,000 pedestrian-free images. Our VDPMs use the root and five parts: shoulder-head, left and right trunk-arms, left and right legs. Lat-SVM V5 uses a 8-part configuration. The root window (*i.e.*, the CW) size is of 48×96 pixels. For detecting pedestrians of height up to 50 pixels, we upscale the images with bilinear interpolation. Part windows are of 24×48 pixels.

Automatic aspect clustering is done once for a desired number of clusters. For the numbers tested in the presented experiments, our clustering procedure (Sect. 3.2.2) roughly takes five minutes for the 2,500 virtual-world pedestrians using MatLab code running on an Intel Xeon CPU E5420 @2.5GHz. The part labeling of the same pedestrians (Sect. 3.2.3) is also done once. By using MatLab code running on the mentioned processor, the part labeling takes around five minutes as well.

The testing of the pedestrian detectors presented in these experiments is always done by running the corresponding part of the Lat-SVM V5 framework. Since the BB prediction post-processing incorporated within this framework requires further training, it is skipped for all tests. In other words, the location of a detected pedestrian

[1]The datasets CVC-02, CVC-05, CVC-07 in this chapter are publicly available within `www.cvc.uab.es/adas/`

Table 3.1: Statistics of the data sets used in this chapter.

Testing sets ⇝	Daimler	Daimler*	TUD	Caltech	CVC
Images	21,790	973	508	4,024	4,363
≥ 50 *pix* ped.	6,090	1,483	1,207	1,014	5,016

Training sets ⇝	INRIA	Virtual
Pedestrian-free images	1,218	2,000
Pedestrian examples	1,208	2,500

Table 3.2: Evaluation of components clustering methods for Lat-SVM V5. Average miss rate % is shown for FPPI in $[10^{-2}, 10^{0}]$.

Clustering Method	Daimler*	TUD	Caltech	CVC
Symmetry, c=2	32.1	72.7	68.1	56.2
HOG K-means, c=4	31.0	73.8	68.6	57.2
Our, c=4	29.3	72.7	**64.9**	**49.6**
Our, c=8	**28.4**	**70.0**	**64.5**	50.9

directly corresponds to the location of the root. Overall, training and testing of all detectors is done under the same conditions for fair comparison.

3.5 Experiments

First we assess the accuracy of the component clustering methods for Lat-SVM V5. We train with our virtual-world data. The results are shown in Table 3.2. Our virtual-world pedestrian examples have a fixed aspect ratio, thus the default Lat-SVM V5 clustering method is equivalent to consider two symmetric components. For completeness, we also include K-means clustering of HOG features [17]. Note how our clustering performs better than the rest. Setting $c = 8$ components tends to perform slightly better than $c = 4$. However, since the difference is small, in the following we assume $c = 4$ (3rd level of the hierarchy in Fig. 3.3a-3.3b) to obtain a faster detector.

Next we compare Star and MP VDPMs using our aspect clustering, with and without domain adaptation. Table 3.3 shows the results. Note how effective is combining the virtual- and real-world data: accuracy improves from 4 to 11 percentage points depending on the dataset, MP clearly outperforming Star. Without the combination, VDPMs perform similarly.

Table 3.4 compares Lat-SVM V5 with VDPM-MP. VDPM-MP uses our aspect clustering. Lat-SVM V5 uses the same clustering input when the training data is the virtual-world one, while it applies its own clustering algorithm when the training uses only INRIA. Note how, using the same aspect clustering and the virtual-world data,

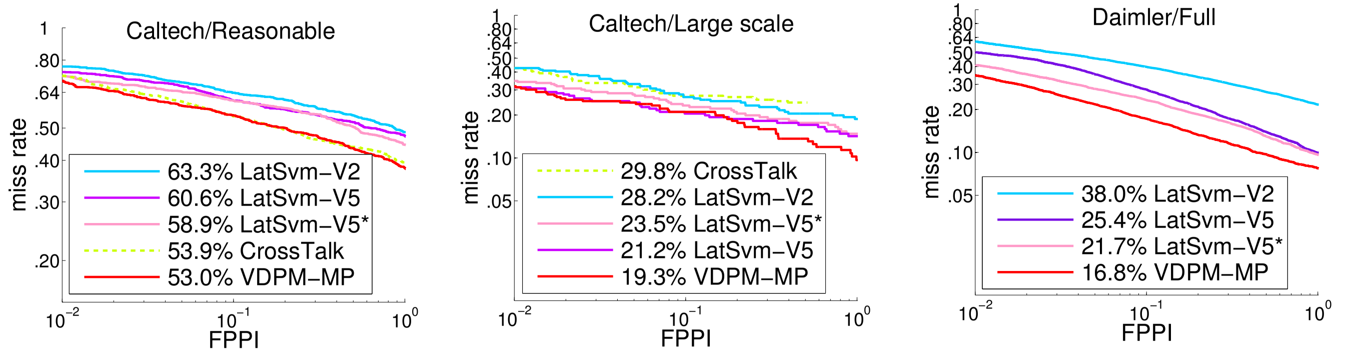


Figure 3.5: LatSvm-V2 and LatSvm-V5 are trained with INRIA training dataset, while LatSvm-V5* and VDPM-MP are trained with Virtual+INRIA training data.

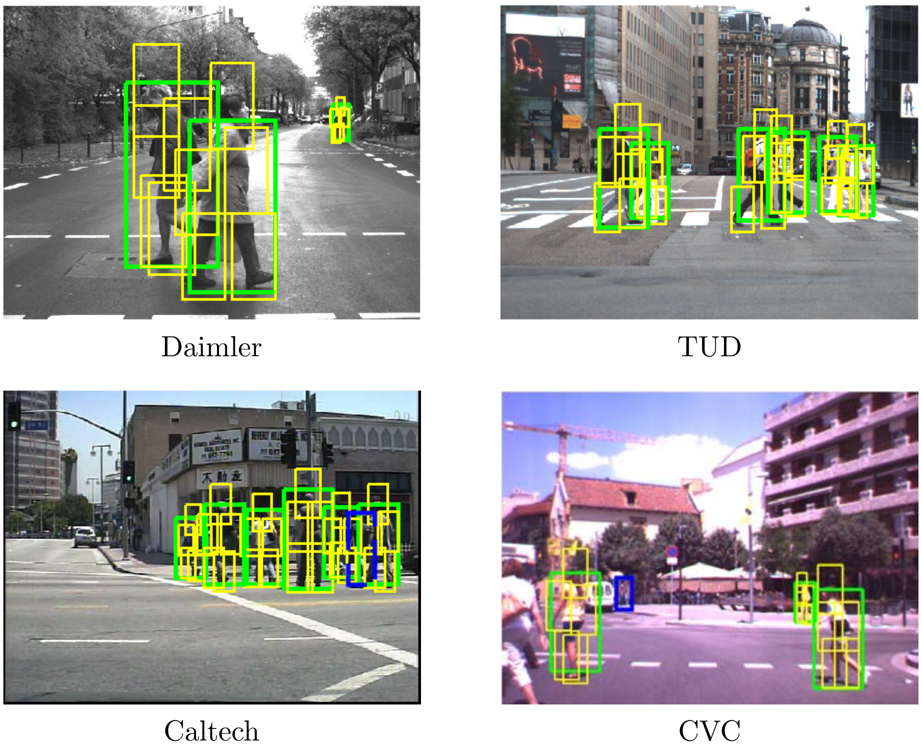


Figure 3.6: Detections at FPPI = 0.1 for our VDPM-MP trained with Virtual+INRIA data. Blue BBs indicate miss detections. Green ones are root right detections, with corresponding detected parts as yellow boxes.

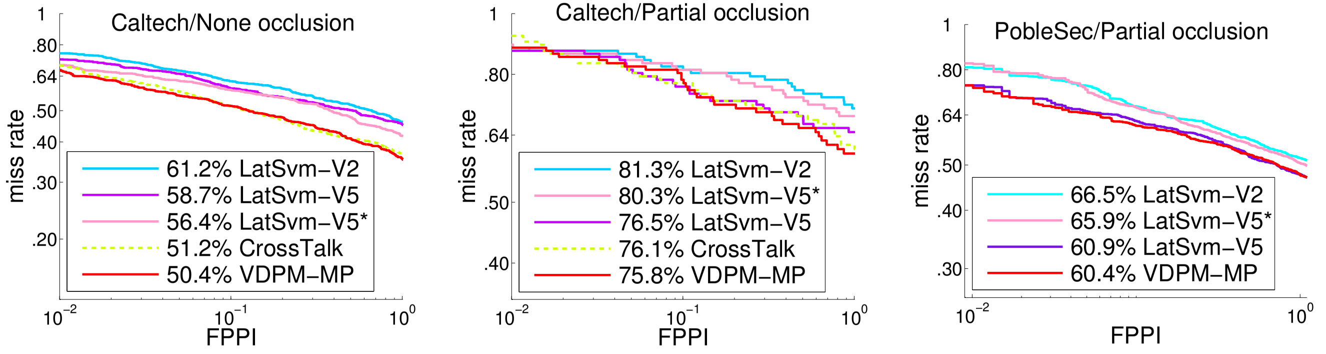


Figure 3.7: Behavior of different detectors with respect to occluded pedestrians.

Table 3.3: VDPM-Star *vs.* VDPM-MP comparison. Average miss rate % is shown for FPPI in $[10^{-2}, 10^{0}]$.

VDPM (training sets)	Daimler*	TUD	Caltech	CVC
Star (V.)	25.1	70.7	63.4	48.8
MP (V.)	**24.3**	**65.9**	**63.3**	**47.5**
Star (V.+INRIA)	21.6	65.7	55.8	42.5
MP (V.+INRIA)	**18.2**	**61.3**	**53.0**	**36.3**

Table 3.4: Average miss rate % is shown for FPPI in $[10^{-2}, 10^{0}]$ for the different DPMs.

DPM (training sets)	Daimler*	TUD	Caltech	CVC
Lat-SVM V5 (V.)	29.3	72.7	64.9	49.6
VDPM-MP (V.)	**24.3**	**65.9**	**63.3**	**47.5**
Lat-SVM V5 (INRIA)	24.7	**60.0**	59.5	42.6
Lat-SVM V5 (V.+INRIA)	23.4	69.6	58.9	42.9
VDPM-MP (V.+INRIA)	**18.2**	61.3	**53.0**	**36.3**

Table 3.5: As Table 3.4 substituting INRIA by CVC and Caltech†. The '—' avoids testing with the training pedestrians.

DPM (training sets)	Daimler*	TUD	Caltech	CVC
Lat-SVM V5 (Caltech†)	57.5	75.4	—	52.5
VDPM-MP (V.+Caltech†)	**18.5**	**60.2**	—	**36.6**
Lat-SVM V5 (CVC)	60.2	81.1	60.0	—
VDPM-MP (V.+CVC)	**20.9**	**56.6**	**50.6**	—

VDPM-MP reports better accuracy than Lat-SVM V5. This is because VDPM-MP is more flexible than the star model of Lat-SVM V5 and relies on a better initialization of the parts. The same happens combining virtual- and real-world data. Overall, if we compare Lat-SVM V5 trained with INRIA to our VDPM-MP trained with INRIA plus our virtual-world data, we see a large decrease in average miss rate, ($\sim$ 6 points for Daimler, Caltech and CVC).

Figure 3.5 draws results for the full Daimler set and different Caltech subsets. We add CrossTalk [19] since it recently reported state-of-the-art results on Caltech. CrossTalk uses a holistic pedestrian model learnt by mining many feature channels using AdaBoost style. Note how in the reasonable setting of Caltech the average accuracy of CrossTalk is comparable to VDPM-MP at the moment, while looking only at close pedestrians (*Large* label corresponds to pedestrians over 100 pixels height, *i.e.*, closer than 18 m [22]) VDPM-MP outperforms CrossTalk in 10.5 points, which is very important in driving scenarios. This is in agreement with the fact that DPMs are expected to work better at higher resolutions than holistic models. Finally, Fig. 3.6 shows qualitative results of VDPM-MP.

For the sake of completeness we have devised a new set of experiments where we have changed the real-world dataset. We have appended the *reasonable* pedestrians of both the training and testing sets of Caltech to obtain a new training set, namely Caltech†, which contains 2,721 pedestrians (roughly twice as much as the INRIA training set). Note that Caltech $\subset$ Caltech†. We have also used the CVC dataset as training set (it contains 5,016 reasonable pedestrians, see Table 3.1). The obtained results, shown in Table 3.5, confirm that our approach clearly outperforms Lat-SVM V5.

For assessing classifiers' accuracy for occluded pedestrians we incorporated the experiments in Fig. 3.7. We tested on the *partial occlusion* set of Caltech and in our own one PobleSec [67]. The former containing 102 partially occluded pedestrians over 50 pix height, the latter containing 577. Note how our VDPM-MP clearly outperforms Latent SVM V5 in the non-occluded pedestrians, while for the occluded ones these methods perform analogously. In fact, the accuracy under partial occlusion tends to decrease compared to the non-occlusion case, showing that DPMs may require mechanisms of occlusion detection and re-scoring as we proposed in [67] for holistic models or, alternatively, explicitly incorporating additional components trained with partially occluded pedestrians as special aspects.

Finally, we assessed the processing time of the training and testing frameworks. The training is conducted in an Intel Xeon(R) CPU E51620 of 8 cores at 3.60GHz. The code has parts in C++ and in MatLab, training in parallel the part filters. DPM and VDPM methods consume a similar time to learn the pedestrian models, *i.e.*, between 11 and 12 hours in average for the presented experiments. For testing we have incorporated the proposal of [28] to speed up our linear part filters. Then, using the same CPU as for training, our current C++ implementation runs in the range of 6 to 10 fps.

3.6 Summary

We have shown how virtual-world data can be used for learning pedestrian DPMs. Using our VDPM-MP proposal and combining virtual- and real-world data, we clearly outperform the state-of-the-art DPM, *i.e.*, Lat-SVM V5. Our automatic aspect clustering and part labeling have two main outcomes. On the one hand, we obtain a more precise initialization for the training optimization procedure. On the other hand, we can train a DPM with part-sharing and aspect clustering. As to the best of our knowledge this is the first work showing how to effectively train such a model by using virtual-world data.

It has been seen in the experiment that adding real-world data in the training can largely improve the detection accuracy. It is in agreement with [99] that domain adaptation is necessary to apply the virtual-world trained detector to the real world. In the next chapter, we will explore domain adaptation techniques for the DPM.

Chapter 4

Domain adaptation of deformable part-based models

The accuracy of object classifiers can significantly drop when the training data (source domain) and the application scenario (target domain) have inherent differences. Therefore, adapting the classifiers to the scenario in which they must operate is of paramount importance. In this chapter, we present novel *domain adaptation* (DA) methods for adapting the deformable part-based model (DPM). We introduce an *adaptive structural SVM* (A-SSVM) that adapts a pre-learned classifier between different domains. By taking into account the inherent structure in feature space (*e.g.*, the parts in a DPM), we propose a *structure-aware A-SSVM* (SA-SSVM). Neither A-SSVM nor SA-SSVM needs to revisit the source-domain training data to perform the adaptation. Rather, a low number of target-domain training examples (*e.g.*, pedestrians) are used. To address the scenario where there are no target-domain annotated samples, we propose a *self-adaptive DPM* based on a self-paced learning (SPL) strategy and a Gaussian Process Regression (GPR). Two types of adaptation tasks are assessed: from both synthetic pedestrians and general persons (PASCAL VOC) to pedestrians imaged from an on-board camera. Results show that our proposals avoid accuracy drops as high as 15 points when comparing adapted and non-adapted detectors.

4.1 Introduction

Training accurate vision-based object classifiers is essential to the development of reliable object detectors. The main focus for training such classifiers has been the search for the most appropriate image representations and learning machines. In this context, most of the methods for learning classifiers assume that the training data (*source domain*) and the data from the application scenario (*target domain*) are sampled from the same probability distribution. However, in many practical situations this is not the case since even changes in the sensor device can break such an assumption [98, 99]. In other words, a *dataset shift* can be present [89] which significantly impacts the accuracy of classifiers and therefore the overall reliability of the overall object detectors. Accordingly, *domain adaptation* (DA) techniques are crucial to maintain detection accuracy across domains.

In Chapter 3, we focused on training accurate deformable part-based models (DPMs) in virtual world. The previous experiments have shown that with even simple DA strategy, *i.e.*, adding a few real-world samples and re-training the virtual-world trained models (VDPM-Star or VDPM-MP), significant accuracy gain has been obtained. In this chapter, we focus on providing methods performing domain adaptation of DPMs. Particularly, we choose VDPM-Star as our source domain detector in this chapter as well as the following chapters. This is because VDPM-Star follows exactly the same model definition as the original DPM [35], thus could be easily generalized to other DPM variants, *e.g.*, the proposed DA methods in this chapter could be used as plug-in tools in the Lat-SVM V5.0 framework. However, VDPM-MP is more specific and more complex.

DPM can be regarded as a particular case of structural model, where the components and parts define the structure. Accordingly, we formulate the learning of a DPM as a general latent *structural SVM* (SSVM) [102, 118, 124]. Therefore, we cast the DA of a DPM as a particular case of adapting general structural models. In this context, we propose an *adaptive structural SVM* (A-SSVM) method motivated by the adaptive SVM (A-SVM) [116]. Furthermore, since A-SSVM works irrespective of the model structure (*e.g.*, the parts and components in a DPM), we also propose a *structure-aware A-SSVM* (SA-SSVM) method. Remarkably, neither A-SSVM nor SA-SSVM need to revisit the training data from the source domain, instead a relatively low number of training examples (*i.e.*, object instances) from the target domain are used to adapt the structural model that has been initially learned in the source domain.

Although A-SSVM and SA-SSVM only require a few manually annotated target-domain examples for the adaptation, we also address the more challenging situation of even avoiding such manual annotations. In particular, we have devised an iterative method for automatically discovering and labelling samples in the target domain and re-training an adapted classifier with them using either A-SSVM or SA-SSVM. Our method applies a self-paced learning (SPL) strategy [64] to re-train an initial model with increasingly difficult target-domain examples in an iterative way without requiring source-domain data. The proper definition of what is an *easy/difficult* sample (example or counter-example) is essential for the SPL. However, in general it turns out that discovering easy/difficult samples in a new domain is a non-trivial task. In this chapter, we apply Gaussian Process Regression (GPR) for performing such a *sample selection*, which can also simplify the SPL optimization procedure proposed in [64]. We call our proposal the *self-adaptive DPM*.

We apply the proposed techniques to *pedestrian detection*. We evaluate two different situations in the context of adapting a pedestrian DPM. We adapt our pedestrian classifiers learned with virtual-world dataset (*i.e.*, VDPM-Star in Chapter 3) to operate on real-world images. Furthermore, we adapt the generic person classifier from the PASCAL VOC to detect people in INRIA data. In the former case the drop in accuracy without adaptation is presumably due to the fact that the synthetic and real-world data differ in appearance. In the latter case the drop in accuracy may be due to the large differences in typical views, poses and resolutions between training

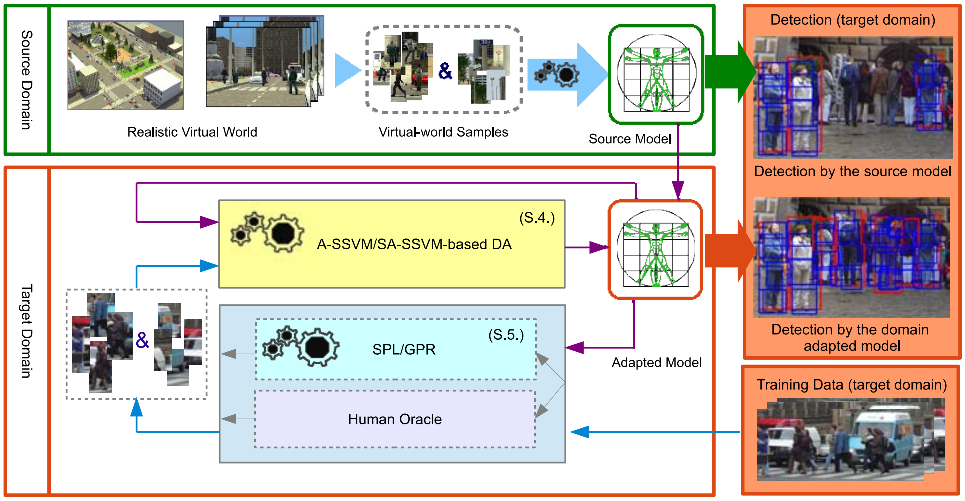


Figure 4.1: Proposed framework for *domain adaptation* (DA) of the SVM-based *deformable part-based model* (DPM). The figure shows the adaptation of a DPM-based pedestrian detector from a virtual-world *source domain* to a real-world *target domain*. As DA module we propose an *adaptive structural* SVM (A-SSVM) and a *structure-aware* A-SSVM (SA-SSVM), see Sect. 4.4. A-SSVM and SA-SSVM require target-domain labeled samples (*e.g.*, a few pedestrians and background) that can be provided by a human oracle. Alternatively, we propose a strategy inspired by *self-paced learning* (SPL) and supported by a *Gaussian Process Regression* (GPR) for the automatic labeling of samples in unlabeled or weakly labeled target domains. The combination of SPL/GPR with either A-SSVM or SA-SSVM gives rise to our *self-adaptive DPM* (see Sect. 4.4).

and testing data, which also represents a very challenging case. The conducted experiments show that our proposals avoid accuracy drops of as high as 15 percentage points when comparing adapted and non-adapted detectors.

The rest of the chapter is organized as follows, in Section 4.2 we summarize the main ideas of the DPM and structural learning. In Section 4.4 we explain our supervised domain adaptation proposals for DPMs, namely A-SSVM and SA-SSVM. In Section 4.4 we present our self-adaptive DPM for working with unlabeled or weakly labeled target domains. In Section 4.5 we assess the results of our proposals in the field of pedestrian detection. Finally, in Section 4.6 we draw the main conclusions and future research lines.

4.2 DPM and structural learning

The DPM [35] is defined by one root filter and a pre-set number of part filters. Part filters operate at twice the resolution of the root filter. The root acts as reference and all other parts are connected to this reference (star model). To better capture intra-class variations, star models can be further combined into a mixture of components (*e.g.*, representing different views).

To detect objects in an image, a sliding window search is applied in the image pyramid. Suppose that the DPM has M components and that each component has K parts. Then, an object hypothesis is defined by $\mathbf{h} = [c, \mathbf{p}'_0, \ldots, \mathbf{p}'_K]', c \in [1, M]$, where $\mathbf{p}_j = [u_j, v_j, s_j]'$ specifies the position (u_j, v_j) and scale level s_j of part $j \in [0, K]$, $j = 0$ identifies the root. The DPM takes into account appearance features as well as part deformations. Given a candidate image window $\mathbf{x}$ and an associated hypothesis $\mathbf{h}$, for a single component c, the decision function can be written in terms of a dot product between the parameter vector $\mathbf{w}_c$ and the feature vector $\Phi_c(\mathbf{x}, \mathbf{h})$ as:

$$\mathbf{w}'_c \Phi_c(\mathbf{x}, \mathbf{h}) = \sum_{j=0}^{K} \mathbf{F}'_{cj} \phi_a(\mathbf{x}, \mathbf{h}) - \sum_{j=1}^{K} \mathbf{d}'_{cj} \phi_d(\mathbf{p}_j, \mathbf{p}_0) + b_c, \tag{4.1}$$

where $\phi_a(\mathbf{x}, \mathbf{h})$ represents the appearance feature vector (*e.g.*, HOG descriptors), and $\phi_d(\mathbf{p}_j, \mathbf{p}_0) = [dx_j, dx_j^2, dy_j, dy_j^2]'$ is the deformation function of part j with respect to part 0 (root). $\mathbf{F}_{cj}$ are the appearance parameters, $\mathbf{d}_{cj}$ is a four-dimensional vector specifying the coefficients of deformation cost, and b_c is the bias term. For the multiple component model, the *one-vs-rest* approach can be employed and the final decision function is written as:

$$f(\mathbf{x}) = \max\nolimits_{\mathbf{h}} \mathbf{w}' \Phi(\mathbf{x}, \mathbf{h}), \tag{4.2}$$

where $\mathbf{w} = [\mathbf{w}'_1, \ldots, \mathbf{w}'_M]'$, $\Phi = [\mathbf{0}'_{n_1}, \ldots, \Phi'_c, \ldots, \mathbf{0}'_{n_M}]'$.
Thus, DPM training aims to learn an optimum $\mathbf{w}$ which encodes the appearance parameters and deformation coefficients. Suppose we are given a set of training samples $(\mathbf{x}_1, y_1, \mathbf{h}_1), \ldots, (\mathbf{x}_N, y_N, \mathbf{h}_N) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{H}$, where $\mathcal{X}$ is the input space, $\mathcal{Y} = \{+1, -1\}$ is the label space, and $\mathcal{H}$ is the hypothesis or output space. We write the features as joint feature vectors $\Phi(\mathbf{x}, \mathbf{h})$. In the DPM case [35], $\mathbf{h}$ is not given and is therefore treated as a latent variable during training.

The discriminative function of (4.2) can be learned by the max-margin method, *e.g.*, using latent SVM as in [35]. The latest version of the DPM (version 5.0) generalizes the SSVM and latent SSVM in a weak-label SSVM, which subsumes latent SVM as a special case [42]. Computing the optimum $\mathbf{w}$ for the score function (4.2) is equivalent to solving the following latent SSVM optimization problem:

$$\begin{aligned} &\underbrace{\min\nolimits_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} \max_{\widehat{y}, \widehat{\mathbf{h}}} [\mathbf{w}' \Phi(\mathbf{x}_i, \widehat{\mathbf{h}}) + L(y_i, \widehat{y}, \widehat{\mathbf{h}})]}_{convex} \\ &\underbrace{-C \sum_{i=1}^{N} \max_{\mathbf{h}} \mathbf{w}' \Phi(\mathbf{x}_i, \mathbf{h}),}_{concave} \end{aligned} \tag{4.3}$$

where parameter C is the relative penalty scalar parameter, $L(y_i, \widehat{y}, \widehat{\mathbf{h}})$ represents the loss function, $\widehat{y}$ the predicted label, and y_i the ground truth label. In particular, we use 0-1 loss for object detection, *i.e.*, $L(y_i, \widehat{y}, \widehat{\mathbf{h}}) = 0$ if $\widehat{y} = y_i$ and 1 otherwise. The latent SSVM optimization objective function (4.3) can be viewed as minimizing the sum of a convex and concave function and it can be solved by the coordinate descend method as in [35] or by the general Convex-Concave Procedure (CCCP) in [42], which is a simple iterative procedure that guarantees the convergence to a local minimum or a stationary point of the objective function. For a comprehensive explanation, we refer the reader to [42, 102, 118, 124].

4.3 Domain adaptive DPM

Based on the DPM framework, we propose our Domain Adaptive DPM (DA-DPM), which is illustrated in Figure 6.1. To adapt a DPM detector from a source domain to a different target domain, we first assume the supervised DA task. This means that both source and target domain labels are given. Let $\mathcal{D}_l^S$ denote the labeled source domain and $\mathcal{D}_l^T$ the labeled target domain. We assume that a DPM has been trained in the source domain, we denote by $\mathbf{w}^S$ the corresponding parameter vector. Thus, our goal is to adapt $\mathbf{w}^S$ to the target domain, using a relatively low number of target-domain labeled examples, so that we obtain a more accurate model $\mathbf{w}$ for the new domain.

4.3.1 Adaptive SSVM (A-SSVM)

Our first proposal is based on the adaptive SVM (A-SVM) [116], an effective DA algorithm that uses a prior model and learns a perturbation function based on a pre-trained source classifier. Given the source domain model $\mathbf{w}^S$, the target domain model $\mathbf{w}^T$ is learned by minimizing the following objective function:

$$\min_{\mathbf{w}^T} \frac{1}{2}\|\mathbf{w}^T - \mathbf{w}^S\|^2 + C\mathcal{L}(\mathbf{w}^T; \mathcal{D}_l^T), \tag{4.4}$$

where the regularization term $\|\Delta\mathbf{w}\|^2 = \|\mathbf{w}^T - \mathbf{w}^S\|^2$ constrains the target model $\mathbf{w}^T$ to be close to the source one $\mathbf{w}^S$. At the testing time, we apply the following decision function to the target domain:

$$f^T(\mathbf{x}) = \mathbf{w}^{T'}\Phi(\mathbf{x}) = f^S(\mathbf{x}) + \Delta\mathbf{w}'\Phi(\mathbf{x}), \tag{4.5}$$

where $\Phi(\mathbf{x})$ is the feature vector for target domain sample $\mathbf{x}$ and $f^S(\mathbf{x})$ is the output score from the source domain classifier. Thus, A-SVM is essentially learning a perturbation function $\Delta f(\mathbf{x}) = \Delta\mathbf{w}'\Phi(\mathbf{x})$ based on the source classifier.

We extend it for structural learning, namely adaptive SSVM (A-SSVM). Given the source model $\mathbf{w}^S$, the final classifier f^T is defined by

$$f^T(\mathbf{x}) = \max_{\mathbf{h}}[\mathbf{w}^{S'}\Phi(\mathbf{x}, \mathbf{h}) + \underbrace{\Delta\mathbf{w}'\Phi(\mathbf{x}, \mathbf{h})}_{\Delta f(\mathbf{x})}] \ , \tag{4.6}$$

where $\Delta f(\mathbf{x})$ is called the perturbation function, $\Delta\mathbf{w} = \mathbf{w} - \mathbf{w}^S$, $\mathbf{w}^S$ is the prior model, and $\mathbf{w}$ the final adapted model. The basic idea is to learn a new decision boundary close to the original source decision one. The new decision function (4.6) can be obtained by solving the following optimization problem:

$$\min_{\Delta\mathbf{w}} \mathcal{R}(\Delta\mathbf{w}) + C\mathcal{L}(\Delta\mathbf{w}, \mathcal{D}_l^T), \tag{4.7}$$

where $\mathcal{R}$ is a regularizer, $\mathcal{L}$ represents the loss term on target data, and C is a penalty scalar parameter as in (4.3). Furthermore, (4.7) can be explicitly written as:

$$\begin{aligned} &\min_{\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}^S\|^2 + C\textstyle\sum_{i=1}^N \xi_i \\ &\text{s.t.} \quad \forall i, y, \mathbf{h}, \quad \xi_i \geq 0, \quad \forall(\mathbf{x}_i, y_i) \in \mathcal{D}_l^T \\ &\mathbf{w}'\Phi(\mathbf{x}_i, \mathbf{h}_i) - \mathbf{w}'\Phi(\mathbf{x}_i, \mathbf{h}) \geq L(y_i, y, \mathbf{h}) - \xi_i \ , \end{aligned} \tag{4.8}$$

where y_i and $\mathbf{h}_i$ are the ground truth label and object hypothesis, y and $\mathbf{h}$ represent all the alternative output label and object hypothesis, and $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_N]'$.

The regularization term shows that A-SSVM adapts the model learned in the source domain towards the target domain by regularizing the distance between $\mathbf{w}$ and $\mathbf{w}^S$. Equivalent to the optimization of SSVM [119], the primal form minimization problem of (4.8) has its closely related maximization dual form problem. By introducing the Lagrange multiplier $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]'$, we can analyse the DA in the dual form:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \textstyle\sum_{i,\overline{y},\overline{\mathbf{h}}} \alpha_i(\overline{y}, \overline{\mathbf{h}})[L(y_i, \overline{y}, \overline{\mathbf{h}}) - \mathbf{w}^{S'}\Delta\Phi_{i,\overline{\mathbf{h}}}] \\ &-\frac{1}{2} \textstyle\sum_{i,\overline{y},\overline{\mathbf{h}}} \sum_{j,\widehat{y},\widehat{\mathbf{h}}} \alpha_i(\overline{y}, \overline{\mathbf{h}})\alpha_j(\widehat{y}, \widehat{\mathbf{h}})\Delta\Phi_{i,\overline{\mathbf{h}}}\Delta\Phi_{j,\widehat{\mathbf{h}}} \end{aligned} \quad , \tag{4.9}$$

where $\overline{y}, \overline{\mathbf{h}}, \widehat{y}, \widehat{\mathbf{h}}$ are alternative labels and object hypotheses other than the ground truth, and $\Delta\Phi_{i,\mathbf{h}} = \mathbf{w}^{S'}[\Phi(\mathbf{x}_i, \mathbf{h}_i) - \Phi(\mathbf{x}_i, \mathbf{h})]$. Comparing (4.9) to the dual form of the standard SSVM [119], the only difference comes from the fact that (4.9) contains the term $\mathbf{w}^{S'}\Delta\Phi_{i,\overline{\mathbf{h}}}$. Let $L_s = \mathbf{w}^{S'}\Delta\Phi_{i,\overline{\mathbf{h}}}$. Then $L_s < 0$ indicates that the output is incorrectly predicted by the source classifier in the target domain. Therefore, a larger α_i is preferred in order to maximize the dual form (4.9) and vice versa. Note that only the target-domain samples $\mathbf{x}_i \in \mathcal{D}_l^T$ are used during the training and α_i is equivalent to the weight of the vector $\mathbf{x}_i$. Thus, the A-SSVM tunes the model parameters towards the target-domain data.

4.3.2 Structure aware adaptive SSVM (SA-SSVM)

The A-SSVM regularization constrains the new classification hyperplane should to not deviate far from the source one, and thus it requires that the source and the target domains have the same feature representation and similar feature distributions. This is very strict for a mixture component part-based model. First of all, it does not take into account the inherent structure knowledge of the model. Secondly, it may not be effective when the source and the target domains have significant differences in the feature space, *e.g.*, substantial differences in view or pose distribution. Since we use the joint feature map, *i.e.*, $\Phi(\mathbf{x}, \mathbf{h})$ for structure learning, the learned hyperplane parameters naturally encode the structural knowledge from the space $\mathcal{X} \times \mathcal{H}$. For example, by taking a deeper look at the learned DPM hyperplane, its corresponding parameter vector can be divided into blocks by the mixture components or parts. This

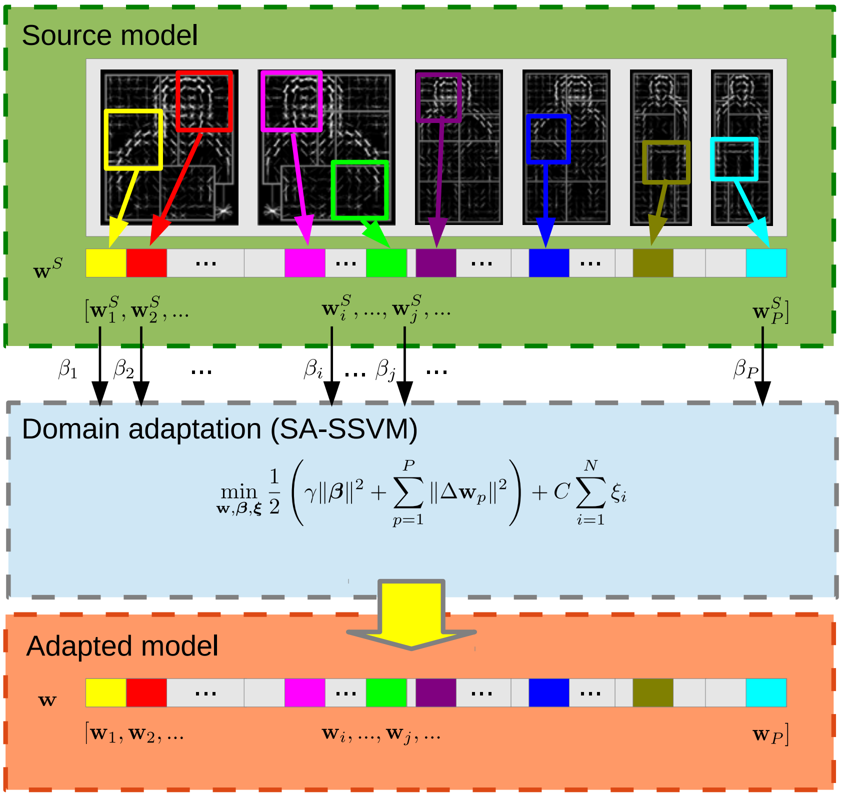

Source model
$\mathbf{w}^S$
$[\mathbf{w}_1^S, \mathbf{w}_2^S, ...$
$\mathbf{w}_i^S, ..., \mathbf{w}_j^S, ...$
$\mathbf{w}_P^S]$
β_1
β_2
β_i
β_j
β_P
Domain adaptation (SA-SSVM)
$\min_{\mathbf{w},\boldsymbol{\beta},\boldsymbol{\xi}} \frac{1}{2}\left(\gamma\|\boldsymbol{\beta}\|^2 + \sum_{p=1}^{P}\|\Delta\mathbf{w}_p\|^2\right) + C\sum_{i=1}^{N}\xi_i$
Adapted model
$\mathbf{w}$
$[\mathbf{w}_1, \mathbf{w}_2, ...$
$\mathbf{w}_i, ..., \mathbf{w}_j, ...$
$\mathbf{w}_P]$

Figure 4.2: Domain adaptation for DPM: Structure-aware Adaptive Structural SVM (SA-SSVM).

motivates us to consider adapting a prior model with structural knowledge, namely our structure-aware A-SSVM (SA-SSVM).

Fig. 4.2 illustrates the SA-SSVM method with a person DPM. First, we learn the DPM in the source domain. This model, $\mathbf{w}^S$, consists of components: half body and full body, as well as persons seen from different viewpoints. Each component consists of parts: head, torso, etc. To adapt this DPM to a different domain, we decompose the structural model as $\mathbf{w}^S = [\mathbf{w}_1^{S'}, \dots, \mathbf{w}_P^{S'}]'$, where P is the number of partitions. Note that each component, $\mathbf{w}_p^S$, may contain both appearance and deformation parameters. The decomposed model parameters are adapted to the target domain by different weights, denoted by $\beta_p, p \in [1, P]$ as in Figure 4.2. In order to learn these adaptation weights, we further introduce a regularization term $\|\boldsymbol{\beta}\|^2$ in the objective function, and we use a scalar parameter γ to control the relative penalty to the hyperplane parameter regularization term.

We define $\Delta\mathbf{w} = [\Delta\mathbf{w}_1', ..., \Delta\mathbf{w}_P']'$, $\boldsymbol{\beta} = [\beta_1, \dots, \beta_P]'$, where $\Delta\mathbf{w}_p = \mathbf{w}_p - \beta_p\mathbf{w}_p^S$, and $p \in [1, P]$. The regularization term of A-SSVM in (4.7) can be modified as:

$$\mathcal{R}\left(\mathbf{w}, \boldsymbol{\beta}, \mathbf{w}^S\right) = \frac{1}{2}\left(\gamma\|\boldsymbol{\beta}\|^2 + \textstyle\sum_{p=1}^{P}\|\Delta\mathbf{w}_p\|^2\right). \tag{4.10}$$

The SA-SSVM optimization is then formulated as:

$$\begin{array}{l} \min_{\mathbf{w},\boldsymbol{\beta},\boldsymbol{\xi}} \mathcal{R}\left(\mathbf{w},\boldsymbol{\beta},\mathbf{w}^S\right) + C\sum_{i=1}^N \xi_i \\ \text{s.t.} \quad \forall i,y,\mathbf{h}, \quad \xi_i \geq 0, \quad \forall (\mathbf{x}_i,y_i) \in \mathcal{D}_l^T \\ \mathbf{w}'\Phi(\mathbf{x}_i,\mathbf{h}_i) - \mathbf{w}'\Phi(\mathbf{x}_i,\mathbf{h}) \geq L(y_i,y,\mathbf{h}) - \xi_i \ . \end{array} \tag{4.11}$$

There are two parameters to be optimized in the SA-SSVM objective function (4.11), *i.e.*, $\boldsymbol{\beta}$ and $\mathbf{w}$.

Directly optimizing (4.11) is difficult using off-the-shelf tools. By re-arranging the feature and parameter representation, we convert (4.11) into a quadratic programming (QP) problem which can be solved by a standard SVM solver. We introduce a concatenated vector $\tilde{\mathbf{w}} = [\Delta\mathbf{w}', \sqrt{\gamma}\boldsymbol{\beta}']'$ and

$$\tilde{\Phi}(\mathbf{x}_i,\mathbf{h}) = [\Phi(\mathbf{x}_i,\mathbf{h})', \frac{1}{\sqrt{\gamma}}\Theta^S(\mathbf{x}_i)']', \tag{4.12}$$

where $\Theta^S(\mathbf{x}_i) = [\mathbf{w}_1^{S'}\Phi_1(\mathbf{x}_i,\mathbf{h}),...,\mathbf{w}_P^{S'}\Phi_P(\mathbf{x}_i,\mathbf{h})]'$, and $\Phi_p(\mathbf{x},\mathbf{h})$ stands for the features of part p given the candidate $\mathbf{x}$ and the hypothesis $\mathbf{h}$. Then, the optimization problem in (4.11) can be rewritten as follows:

$$\begin{array}{l} \min_{\tilde{\mathbf{w}},\boldsymbol{\beta},\boldsymbol{\xi}} \mathcal{R}(\tilde{\mathbf{w}}) + C\sum_{i=1}^N \xi_i \\ \text{s.t.} \quad \forall i,y,\mathbf{h}, \quad \xi_i \geq 0, \quad \forall (\mathbf{x}_i,y_i) \in \mathcal{D}_l^T \\ \tilde{\mathbf{w}}'\tilde{\Phi}(\mathbf{x}_i,\mathbf{h}_i) - \tilde{\mathbf{w}}'\tilde{\Phi}(\mathbf{x}_i,\mathbf{h}) \geq L(y_i,y,\mathbf{h}) - \xi_i \ , \end{array} \tag{4.13}$$

where $\mathcal{R}(\tilde{\mathbf{w}}) = \sum_{p=1}^P \|\tilde{\mathbf{w}}_p\|^2$ and $\tilde{\mathbf{w}}_p = [\Delta\mathbf{w}_p', \sqrt{\gamma}\beta_p]'$.

Note that the regularization term $\mathcal{R}(\tilde{\mathbf{w}})$ is convex and the loss term in (4.13) is also convex, thus the objective function of SA-SSVM is convex. In the following, we discuss several properties of the proposed SA-SSVM.

Part-level adaptation. In contrast to A-SVM, adaptive regularization is performed on partitions. Analogously to the A-SSVM decision function (4.6), we can write the SA-SSVM decision function as:

$$f^T(\mathbf{x}) = \max_{\mathbf{h}}[\sum_{p=1}^P \beta_p\mathbf{w}_p^{S'}\Phi_p(\mathbf{x},\mathbf{h}) + \underbrace{\Delta\mathbf{w}'\Phi(\mathbf{x},\mathbf{h})}_{\Delta f(\mathbf{x})}]. \tag{4.14}$$

Compared to (4.6), (4.14) decomposes the pre-learned classifier into a set of *part classifiers* and the final score is a weighted combination of the prior part classifiers and the perturbation functions. Thus, it takes into account the structural knowledge of the prior model.

Part-level regularization is also proposed in [3], however the parts are taken from multiple holistic templates for transfer learning and the new model is still a rigid holistic template. In contrast to [3], we consider the structure in the single prior model and perform decomposition to the part-based model. The part appearances as well as the deformation in the prior model are adapted in the new model. Using structural correspondence for DA was also proposed in [10]. Structural correspondence is learned with the extracted pivot features from source and target domains. However, the method is specially designed for cross-language text classification tasks.

Properties of γ. The regularization term $\gamma\|\boldsymbol{\beta}\|^2$ controls the adaptation degree of the model. As can be seen from the primal form of the objective function (4.10) and (4.11), when $\gamma \to \infty$ β_p is forced to be zero, due to the infinite penalty. Thus (4.11) converges to non-adaptive SSVM. As $\gamma \to 0$, the penalty on β_p is small, thus it adapts more to the prior model.

Feature augmentation. Note that the joint feature representation in (4.12) is a concatenation of $\Phi(\mathbf{x}_i, \mathbf{h})$ and the part responses of the source classifiers, as $\Theta^S(\mathbf{x}_i)$. Thus, for the adapted classifier $\tilde{\mathbf{w}}$, $\tilde{\Phi}(\mathbf{x}_i, \mathbf{h})$ is an augmented feature with responses in $\Theta^S(\mathbf{x}_i)$.

We can also analyze the properties of the dual form. Letting $\boldsymbol{\alpha}$ be the Lagrange multiplier, the dual form of the optimization problem (4.13) can be written as:

$$\begin{aligned}&\max_{\boldsymbol{\alpha}} \textstyle\sum_{i,\overline{y},\overline{\mathbf{h}}} \alpha_i(\overline{y}, \overline{\mathbf{h}}) L(y_i, \overline{y}, \overline{\mathbf{h}}) \\ &-\frac{1}{2} \textstyle\sum_{i,\overline{y},\overline{\mathbf{h}}} \sum_{j,\widehat{y},\widehat{\mathbf{h}}} \alpha_i(\overline{y}, \overline{\mathbf{h}}) \alpha_j(\widehat{y}, \widehat{\mathbf{h}}) \Delta\tilde{\Phi}'_{i,\overline{\mathbf{h}}} \Delta\tilde{\Phi}_{j,\widehat{\mathbf{h}}}\ ,\end{aligned} \tag{4.15}$$

where the expression $\Delta\tilde{\Phi}'_{i,\overline{\mathbf{h}}}\Delta\tilde{\Phi}_{j,\widehat{\mathbf{h}}} = \Delta\Phi_{i,\overline{\mathbf{h}}}{}'\Delta\Phi_{j,\widehat{\mathbf{h}}} + \frac{1}{\gamma}(\mathbf{w}^{S'}\Delta\Phi_{i,\overline{\mathbf{h}}})(\mathbf{w}^{S'}\Delta\Phi_{j,\widehat{\mathbf{h}}})$ is defined by the labeled training data from the target domain. Thus, the kernel $\Delta\tilde{\Phi}'_{i,\overline{\mathbf{h}}}\Delta\tilde{\Phi}_{j,\widehat{\mathbf{h}}}$ takes into account both visual information from the new domain data and the partial responses of the pre-learned model, which can lead to better discriminative power. Again we see that γ controls the degree of adaptation, as $\gamma \to \infty$ indicates no adaptation and $\gamma \to 0$ indicates maximum adaptation.

4.3.3 Supervised DA-DPM Algorithm

We apply the proposed A-SSVM and SA-SSVM algorithms to learn a domain adapted DPM. The A-SSVM and SA-SSVM are built on the SSVM, which assumes that the ground truth of all outputs $\mathbf{h}$ is given. To apply these techniques to the DPM, we incorporate the latent variables by decomposing the objective functions as the sum of convex and concave parts as in (4.3), thus we can employ the CCCP to solve the latent A-SSVM and SA-SSVM optimization problems. The procedure is formalized in Alg. 3. This algorithm has two main parts: (1) updating the hidden variables (step 3) by approximating the concave function with a linear upper bound; and (2) fixing the hidden variables and updating the parameters by solving a convex A-SSVM or SA-SSVM learning problem.

4.4 Self-adaptive DPM

4.4.1 Self-paced learning (SPL)

To address a DA scenario without target-domain labeled data, we could directly apply the source detector to discover examples (positive samples) and counter-examples (negative samples) in the target domain, and then use them to run A-SSVM or SA-SSVM. However, these collected samples may contain a large number of false positives, due to the domain shift and the inherent detection error of any classifier. In that case,

Algorithm 3 Supervised DA-DPM

```
Input: $\mathbf{w}^S, \epsilon$, target-domain training samples:
$\mathcal{D}_l^T = \{(\mathbf{x}_i, y_i)\}, i \in (1, N)$.
Output: $\mathbf{w}$
0: $\mathbf{w} \leftarrow \mathbf{w}^S$
1: Repeat
2: Update $\mathbf{h}_i^* = \arg\max_{\mathbf{h}} \mathbf{w}'\Phi(\mathbf{x}_i, \mathbf{h}), \forall i$.
3: Update $\mathbf{w}$ by fixing the hidden variables to $\mathbf{h}_i^*$ and solving the DA optimization problem with A-SSVM (4.8) or SA-SSVM (4.13).
4: Until the objective function ((4.8) or (4.13)) cannot be decreased below tolerance $\epsilon$.
```

the DA method can get stuck in a local optimum with high training error due to the fact that the CCCP (and so Alg. 3) considers all samples simultaneously. A strategy analogous to SPL, which starts with the *easiest* samples and gradually considers more complex ones, can be employed to handle this problem.

In SPL, the *easy* samples are defined as those with the highest level of confidence [96], where such a confidence relies on a SVM-based classification score in our case (*e.g.*, the highest absolute value of the score could indicate higher classification confidence). At this point we face a scenario where we must apply a source-domain classifier in a target domain without labels. Therefore, we must distinguish between positive and negative target-domain samples and determine for which samples the decision was easy, all in presence of a domain shift. Accordingly, a simple threshold on the absolute value of the classification score is not an appropriate measure for determining if a sample is easy or not, because: (1) if the easy samples are selected too conservatively (high threshold), the adaptation would be poor since these samples are far away from the hyperplane margin and more likely source-domain oriented; and (2) if the easy samples are aggressively selected (low threshold), many mislabeled ones may be collected for the adaptation. Therefore, rather than defining *easiness* according to a fixed threshold directly applied to our SVM-based classification scores, we propose a more adaptive *sample selection* process based on a GPR.

4.4.2 Gaussian process regression (GPR)

Sample selection must collect object examples and counter-examples (background) from a training sequence of target-domain unlabeled images. The examples will be selected from the detections returned by the current detector (*i.e.*, the source-domain one or an intermediate target-domain adapted version of it). The counter-examples can be selected as background windows overlapping little with the detections (*e.g.*, we use a 10% overlapping threshold). Alternatively, images labeled as object-free (weak labeling) can be used for sampling counter-examples. Collecting examples from the target-domain detections following the SPL relies on a GPR as follows.

We define the thresholds $\overline{r}$ and $\underline{r}$, $\overline{r} \geq \underline{r}$, that divide detections into *conservative* and *aggressive* sets. The conservative set, $\mathcal{D}^{T_{\overline{r}}}$, contains the *easy* examples, defined

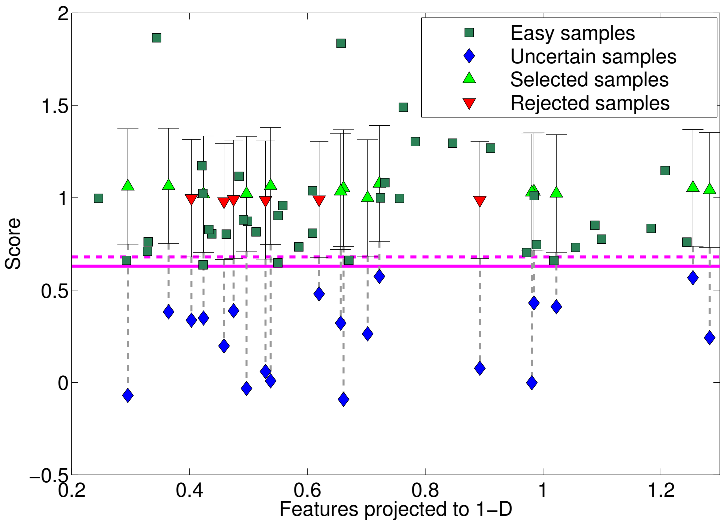


Figure 4.3: Sample selection by GPR (see main text for details). The horizontal axis runs on the sample features projected to 1-D for visualization. The triangles are re-scored values from the diamonds, with vertical segments indicating the $\pm 3\sigma_{*,i}$ variance range. The solid horizontal line draws the threshold $\overline{r}$ and the dashed one $\overline{r} + \theta$. An uncertain sample is selected if its variance range is over $\overline{r} + \theta$.

as those detections with score above $\overline{r}$, *i.e.*, $\mathcal{D}^{\mathcal{T}_{\overline{r}}} = \{(\mathbf{x}_i, z_i) : z_i \geq \overline{r}\}$, where z_i is the classification score of detection $\mathbf{x}_i$. False positives are very unlikely in this set. The aggressive set, $\mathcal{D}^{\mathcal{T}_{\underline{r}}}$, contains the detections with score above $\underline{r}$, *i.e.*, $\mathcal{D}^{\mathcal{T}_{\underline{r}}} = \{(\mathbf{x}_i, z_i) : z_i \geq \underline{r}\}$. The aggressive set minus the conservative one, *i.e.*, $\mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}} = \{(\mathbf{x}_i, z_i) : \underline{r} \leq z_i < \overline{r}\}$, is a set of *uncertain* samples. It contains true positives but containing false positives is more likely than for $\mathcal{D}^{\mathcal{T}_{\overline{r}}}$. In Fig. 4.3 the squares are in $\mathcal{D}^{\mathcal{T}_{\overline{r}}}$ (easy examples) and the diamonds in $\mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}$ (uncertain samples).

A-SSVM and SA-SSVM assume that the target samples have error-free labels. Thus, assigning a proper class to the uncertain samples is important. Accordingly, we propose to use $\mathcal{D}^{\mathcal{T}_{\overline{r}}}$ as confidently classified examples for predicting the scores of the samples in $\mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}$ according to a GPR [90]. In particular, we apply a standard linear regression with Gaussian noise, $z = \mathbf{w}'\Phi(\mathbf{x}) + \eta$, where $\Phi(\mathbf{x})$ is the feature vector, $\mathbf{w}$ is the weight vector and $\eta \sim \mathcal{N}(0, \sigma_z^2)$ is the noise term. In our case, the feature vector consists of the concatenation of the appearance and deformation features of the DPM, *i.e.*, ϕ_a and ϕ_d in Eq. (4.1). We assume a zero mean Gaussian prior on $\mathbf{w}$, *i.e.*, $\mathbf{w} \sim \mathcal{N}(0, \Sigma)$. We use $\mathbf{X}$ to denote the aggregated column vector input from the observed set $\mathcal{D}^{\mathcal{T}_{\overline{r}}}$, and $\mathbf{X}_*$ is the analogous for $\mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}$. The joint density of the observed set and the noise-free function $\mathbf{f}_*$ on the test set $\mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}$ is given by

$$\begin{bmatrix}\mathbf{z}\\ \mathbf{f}_*\end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix}K(\mathbf{X},\mathbf{X}) + \sigma_z^2\mathbf{I} & K(\mathbf{X},\mathbf{X}_*)\\ K(\mathbf{X}_*,\mathbf{X}) & K(\mathbf{X}_*,\mathbf{X}_*)\end{bmatrix}\right), \tag{4.16}$$

where $K()$ is the kernel function for computing the covariance; we use a squared-exponential kernel [90]. The resulting predictive distribution $p(\mathbf{f}_*|\mathbf{X}, \mathbf{y}, \mathbf{X}_*)$ is a Gaussian with mean and covariance defined as:

$$\begin{aligned}\bar{\mathbf{f}}_{*,i} = {} & K(\mathbf{x}_{*,i}, \mathbf{X})[K(\mathbf{X},\mathbf{X}) + \sigma_z^2\mathbf{I}]^{-1}\mathbf{y},\\ \sigma_{*,i} = {} & K(\mathbf{x}_{*,i}, \mathbf{x}_{*,i})\\ & -K(\mathbf{x}_{*,i}, \mathbf{X})[K(\mathbf{X},\mathbf{X}_*) + \sigma_z^2\mathbf{I}]^{-1}K(\mathbf{X}, \mathbf{x}_{*,i}).\end{aligned} \tag{4.17}$$

In analogy with [64], we use variables υ_i indicating if the i^{th} sample is selected ($\upsilon_i = 1$) or not ($\upsilon_i = 0$):

$$\upsilon_i = \begin{cases}1, \left(\bar{\mathbf{f}}_{*,i} - 3\sigma_{*,i} \geq \overline{r} + \theta, \mathbf{x}_{*,i} \in \mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}\right) \vee \mathbf{x}_i \in \mathcal{D}^{\mathcal{T}_{\overline{r}}},\\ 0, \text{otherwise}\end{cases} \tag{4.18}$$

We use $\bar{\mathbf{f}}_{*,i} - 3\sigma_{*,i}$ to ensure that the confidence of the predicted output score is higher than 99.7%. The parameter $\theta > 0$ controls the degree of the acceptance for the samples in $\mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}$ and we use $\theta = 0.05$ in practice. The process is illustrated in Figure 4.3.

4.4.3 Self-adaptive DPM Algorithm

Our self-adaptive DPM is sketched in Alg. 4. At each iteration, we apply GPR to $\mathcal{D}^{\mathcal{T}_{\underline{r}}}, \mathcal{D}^{\mathcal{T}_{\underline{r}\backslash\overline{r}}}$ and compute the υ_i. Supervised DA-DPM (A-SSVM or SA-SSVM) relies on the easy examples and the selected uncertain ones ($\upsilon_i = 1$). Since $\overline{r}$ decreases by a factor of $\Delta > 0$ at each iteration (step 6), $\mathcal{D}^{\mathcal{T}_{\overline{r}}}$ grows and more difficult examples

Algorithm 4 Self-adaptive DPM

```
Input: $\mathbf{w}^S, \underline{r}, \overline{r}, \theta, \Delta, \epsilon$.
Output: $\mathbf{w}$
0: $\mathbf{w} \leftarrow \mathbf{w}^S$
1: Repeat
2: Collect $\mathcal{D}^{T_{\underline{r}}}, \mathcal{D}^{T_{\overline{r}}}$ in the target domain using $\mathbf{w}$.
3: Apply GPR to $\mathcal{D}^{T_{\overline{r}}}, \mathcal{D}^{T_{\underline{r}\setminus\overline{r}}}$ and update $\upsilon_i$ by (4.18).
4: Update $\mathbf{h}_i^* = \arg\max_{\mathbf{h}} \mathbf{w}'\Phi(\mathbf{x}_i, \mathbf{h})$.
5: Update $\mathbf{w}$ by fixing the hidden variables to $\mathbf{h}_i^*$ and solving the corresponding DA optimization problem: A-SSVM (4.8) or SA-SSVM (4.13).
6: $\overline{r} \leftarrow \max(\overline{r} - \Delta,\ \underline{r})$.
7: Until $\overline{r} = \underline{r}$ or the objective function ((4.8) or (4.13)) cannot be decreased below tolerance $\epsilon$.
```

are progressively selected. The training process runs until $\overline{r}$ reaches $\underline{r}$ or the objective function (*i.e.*, (4.8) for A-SSVM or (4.13) for SA-SSVM) cannot be decreased below a tolerance ϵ. We remark that our self-adaptive DPM computes the υ_i at step 3 in an explicit way (Eq. (4.18)), while in the SPL proposal of [64] solving a biconvex optimization problem is required for computing them (see Eq. (4) in [64]). In particular, [64] runs an alternative convex search (ACS).

4.5 Experiments

We built our DA framework based on the latest release of the DPM, *i.e.*, the DPM 5.0 framework [43]. We evaluate first the accuracy of our supervised DA-DPM proposals. We evaluate our self-adaptive DPM, showing its accuracy with and without the GPR.

As we are interested in pedestrian detection, all the experiments rely on public pedestrian datasets. We adapt a generic person detector from the PASCAL VOC 2007 Person dataset to the INRIA pedestrian dataset. In this case, the domain shift is mainly due to the differences in the data distributions in terms of viewpoints and poses. Moreover, following [99], we adapt a pedestrian classifier learned with synthetic data (virtual world) to operate on real-world images.

We use the Caltech evaluation framework [22] following the *reasonable* setting criterion, *i.e.*, detectable pedestrians are those taller than 50 pixels and without heavy occlusion. Thus, we assess the accuracy of a particular pedestrian detector by using per-image evaluation, *i.e.*, computing curves depicting the trade-off between *miss rate* and *number of false positives per image* (FPPI) on a logarithmic scale. For single *detection accuracy* we use one minus the average miss rate in the $[10^{-2}, 10^{0}]$ FPPI range. Moreover, since the target domains are sampled for collecting training data, each DA experiment is repeated five times.

As in [99], to compare our proposals with the state-of-the-art we apply a *paired Wilcoxon test* [108] on the accuracy measures collected from the experiments.

4.5.1 Implementation details

Our DA proposals can be seen as plug-ins for the DPM framework. The solver for optimizing A-SSVM and SA-SSVM is based on the quasi-Newton LBFGS method [92] as in the DPM 5.0 framework [43]. We also use a data-mining procedure to maintain a feature cache with the support vectors, and the minimization of the objective functions is restricted to the cache. Note that in the CCCP the data mining of the examples (*i.e.*, the pedestrians in these experiments) is performed on a constrained set. In particular, the valid examples detected by the model in each iteration are required to have at least 70% overlap with the ground truth bounding box. In the self-adaptive case there are no ground truth bounding boxes, so we designate the detected bounding box with the highest score as *ground truth*. The SPL is implemented to replace the original CCCP. In contrast to the CCCP which uses the entire dataset at each iteration, the SPL first takes the discovered easy examples and gradually collects the difficult ones. Moreover, we use the implementation of [90] to compute the GPR. For the parameters in Alg. 4, we fix $\underline{r}$ by -0.5. The initial value of $\bar{r}$ is estimated in the source domain, which ensures high detection accuracy ($> 90\%$), and we set $\Delta = 0.05$.

In practice, the optimization of the DA converges very fast. We use only two iterations for CCCP, and at each iteration we do data mining twice. For the SPL, we iterate three times and apply data mining twice in each loop. Note that our DA methods only require very few training examples, thus the training is very fast. For instance, training a DA-DPM with 100 pedestrians and $1,000$ negative images takes less than 20 minutes in a $3.60GHz \times 4$ modern desktop PC.

4.5.2 Experiment setting

Datasets

Virtual-world pedestrians. We use the virtual-world dataset in Chapter 3 as source domain and train our source domain DPM detector, *i.e.*, VDPM-Star in Chapter 3.

PASCAL VOC person. We use the PASCAL VOC 2007 Person dataset which contains a large number of general person images, including outdoor vertical full body persons and indoor half body ones, all of them with different poses and some of them highly occluded. The DPM person detector trained on VOC 2007 dataset has six components and eight parts (see the prior model in Figure 4.2) and is publicly available from [43]. The components are trained with person samples of different aspect ratios and views.

Real-world pedestrians. We use popular pedestrian detection datasets, namely INRIA [15], ETH [109], KIT [1], Caltech [22] and CVC (N.02) [41]. Except INRIA, the other datasets are image sequences taken from on-board cameras. In particular, the ETH dataset contains three sub-sequences from on-board cameras, namely 'BAHNHOF', 'JELMOLI' and 'SUNNY DAY'. These sequences are taken in different scenarios and they are named here as ETH0, ETH1 and ETH2 respectively. In all cases pedestrians are standing, either walking or stopped. Caltech and INRIA have separate training and testing sets, for CVC we use the first four sequences for training and the other ten for testing, while for KIT and ETH training images are obtained

Table 4.1: Different types of learned classifiers.

SRC	Trained with labeled source data.
TAR	Trained with labeled target data. For a fair comparison, we initialize the structure of TAR with the source DPM.
MIX	Trained with source and target labeled data.
(S)A-SSVM	Adapted by following Alg. 3, *i.e.* with a source model and labeled target data.
PMT-SSVM	A-SSVM variant that we have developed by extending the PMT-SVM [2] for DA of the DPM.
SA-SSVM-C	SA-SSVM variant where the DPM parameters are partitioned at component level.
U-SA-SSVM	Adapted with target images where the pedestrians are unlabeled, Alg. 4 is followed setting the SA-SSVM case and relying on the threshold $\bar{r}$ to select easy examples (GPR not applied).
U-SA-SSVM-GPR	As U-SA-SSVM but using the GPR.

by sampling the respective sequences keeping the remaining of the sequences for testing. It is worth mentioning that INRIA and Caltech can be considered as weakly labeled since their training sets are split into pedestrian-free images and images with annotated pedestrians. This is not the case for ETH, KIT and CVC.

Learned classifiers

We train the types of classifiers shown in Table 4.1. For TAR, MIX, A-SSVM, SA-SSVM, SA-SSVM-C, and PMT-SSVM we use 100 randomly selected target-domain training pedestrians. For MIX the full source dataset is used too. For U-SA-SSVM and U-SA-SSVM-GPR, we use 150 randomly selected target-domain training images which contain at least 100 pedestrians, but without considering manually annotated bounding boxes. The number of target-domain training images from which to collect background windows is fixed to 1000. For INRIA and Caltech these are pedestrian-free images. For the rest of the datasets these images contain pedestrians, thus background windows are obtained as those overlapping less than a 10% with the annotated pedestrian bounding boxes. The parameter γ in SA-SSVM is fixed by cross validation for all the experiments ($\gamma = 0.08$).

4.5.3 Experiments on supervised DA-DPM

PASCAL to INRIA

We adapt the general person DPM (six components, eight parts) based on PASCAL dataset to detect pedestrians in INRIA testing images. Figure 4.4 shows the accuracy of the different detectors. We evaluated pure mixture of roots (no parts)and part-based models.

Table 4.2: DA from virtual to real world using different models (see main text). Average miss rate (%) is shown.

Method/Dataset	ETH0	ETH1	ETH2	KIT	Caltech	CVC
Lin-SVM	59.6	64.8	74.3	76.5	68.5	63.9
Lat-SVM	49.6	48.9	55.2	58.0	63.3	41.8

Method/Dataset	ETH0	ETH1	ETH2	KIT	Caltech	CVC
SRC	63.1	58.8	48.5	53.3	63.4	45.9
A-SSVM(*)	59.6±1.3	56.4±0.9	45.0±1.5	51.5±1.4	59.5±1.8	43.2±0.8
SA-SSVM(*)	57.0±1.2	54.9±0.6	43.2±1.7	53.7±0.4	57.3±0.6	41.0±0.9
A-SSVM	56.0±0.5	53.2±0.5	37.6±0.8	45.0±0.5	58.6±0.6	32.5±0.7
SA-SSVM	**54.7**±0.3	**51.5**±0.4	**35.5**±0.3	**44.2**±0.7	**56.1**±0.6	**30.8**±0.3

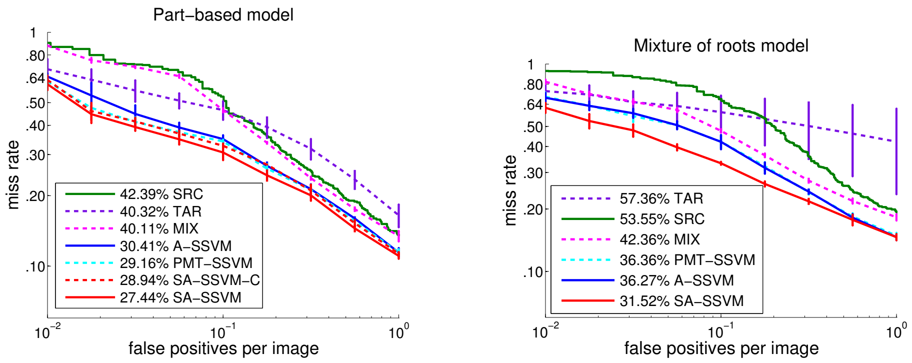


Figure 4.4: Results of adapting PASCAL VOC 2007 DPM person detector to work on the INRIA pedestrian dataset. Percentages correspond to the average miss rate within the plotted FPPI range. Vertical segments illustrate the variance over five runs per experiment.

Virtual to ETH, KIT, Caltech, and CVC

We adapt a pedestrian DPM (three components, five parts) trained with virtual-world data to operate on real-world datasets. For completeness, we include the original HOG/Lin-SVM holistic detector [15] and the DPM state-of-the-art one (Lat-SVM) [35] (three components, eight parts). Lin-SVM and Lat-SVM training uses the full INRIA training set. In fact, a widespread approach consists in training the classifiers using the INRIA training set and then testing on other datasets [22]. Accordingly, we have included analogous experiments. In particular, A-SSVM(*) and SA-SSVM(*) stand for adaptation to INRIA as a sort of *intermediate* domain. However, our interest is the *direct* adaptation to the *final* real-world domain, *i.e.*, to ETH0, ETH1, ETH2, KIT, Caltech, or CVC. The accuracy results of our proposals using intermediate and direct adaptation are listed in Table 4.2. In Figure 4.5, we complete the accuracy results based on direct adaptation. Finally, Table 4.3 shows the adaptation accuracy for the pure mixture of roots (three roots, no parts) and the part-based models.

Discussion

According to Figs. 4.4 and 4.5, TAR shows poor accuracy and high variability, which is due to the low number of target pedestrians used for training. Even SRC performs clearly better than TAR in all cases except the PASCAL-to-INRIA part-based one, which is because the person poses on the PASCAL dataset are too different from those in the INRIA one, while the virtual-world (source) data covers poses similar to the real-world (target) data. MIX clearly outperforms SRC and TAR (INRIA Mixture of Roots, ETH, KIT, CVC) or at least does no harm (INRIA part-based, Caltech). These observations agree with the results of [99]. SSVM adaptations clearly outperform SRC and TAR. The same happens for MIX, except for the ETH1 case where PMT-SSVM and MIX perform similarly. However, we remark that, contrarily to SSVM adaptations, MIX requires re-training with the source data. Thus, we focus on analyzing SSVM DA.

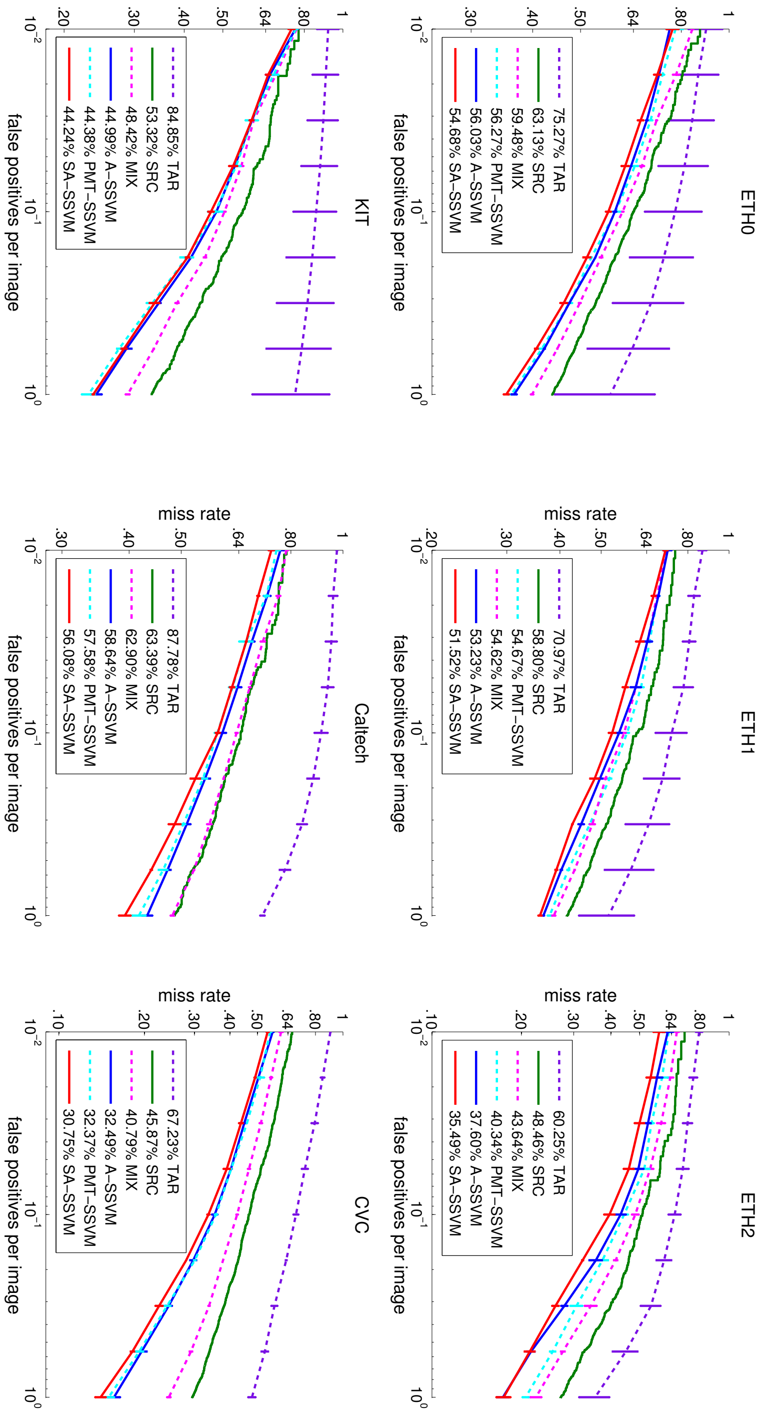


Figure 4.5: Supervised adaptation of DPM from virtual world to specific real-world scenes.

Regarding SA-SSVM, we assessed whether to make it aware of the DPM structure of parts or of components. We used the PASCAL-to-INRIA adaptation problem since the domain shift is not only due to the use of different sensors, as in the virtual-to-real case, but also to large pose differences as mentioned before. In Figure 4.4 top, we see that SA-SSVM accuracy (part aware) is 1.5 points better than SA-SSVM-C accuracy (component aware). Thus, in the rest of SA-SSVM experiments we used the part-aware setting.

Table 4.2 shows how the intermediate adaptations, A-SSVM(*) and SA-SSVM(*), outperform the SRC model in most of the cases. In fact, for CVC the accuracy of SA-SSVM matches Lat-SVM, while for ETH2, KIT and Caltech SA-SSVM clearly outperforms Lat-SVM ($\sim 12, 4,$ and 6 points, respectively), and for ETH1 and ETH2 Lat-SVM is still better ($\sim$ 7 and 6 points, respectively). We remind that Lat-SVM is trained with the full INRIA training set, *i.e.*, using $1,208$ pedestrians, while SA-SSVM only uses 100 ($\sim 8\%$) for the adaptation from the virtual-world (source) model. In any case, we see that the direct adaptations, A-SSVM and SA-SSVM, clearly outperform their intermediate counterparts, especially SA-SSVM. Thus, for the rest of experiments we assumed the direct adaptation setting.

The accuracy of A-SSVM and SA-SSVM have also been assessed for pure mixture-of-roots models. Since parts are not available, for SA-SSVM the component-aware strategy is used. In Figure 4.4 bottom, we see the PASCAL-to-INRIA case, and in Table 4.3 the virtual-to-real one. Observe that A-SSVM and SA-SSVM clearly outperform SRC (Mix. of Roots Δ_a and Δ_{sa} in Table 4.3 show the respective accuracy gains for the virtual-to-real case), thus there is domain adaptation. However, as can be seen in Table 4.3, deformable part-based models achieve a higher relative gain than the mixture-of-roots models for the virtual-to-real case (Part-based Δ_a and Δ_{sa} in Table 4.3 show the corresponding accuracy gains, computed from SRC, A-SSVM and SA-SSVM of Table 4.2). The PASCAL-to-INRIA case is an exception, which is due to the fact that person views at PASCAL dataset are quite different than the ones in INRIA, and therefore strong adaptation can be expected already at component level. Note that in the virtual-to-real case the domain shift is mainly due to the *sensor type* but views and poses of the source and target domains are very similar. In fact, the same reason explains why A-SSVM and SA-SSVM report similar accuracy for the mixture-of-roots adaptation of the virtual-to-real case (Table 4.3), while SA-SSVM outperforms A-SSVM by almost 5 points in the PASCAL-to-INRIA case (Figure 4.4 bottom). In any case, in absolute terms part-based adaptation (either with A-SSVM or SA-SSVM) clearly outperforms pure mixture-of-roots adaptation. Just comparing SA-SSVM from Tables 4.2 and 4.3, we see gains ranging from $\sim$ 11 points for ETH0 to $\sim$ 27 points for CVC.

At this point, we see that A-SSVM, SA-SSVM (part-aware), and PMT-SSVM direct adaptations operating on full DPM models clearly are the best performing methods. Accordingly, we focus on statistical comparisons on them. We use a paired Wilcoxon test by taking into account their respective part-based results for both PASCAL-to-INRIA and virtual-to-real adaptation problems. In this test, when comparing two adaptation methods, the null hypothesis is that they are equal. The hypothesis can be rejected if the p-value of the test is below 0.05 (the p-value running from 0 to 1). Since we test adaptations for seven datasets and each experiment is repeated five times, for each test run we have 35 pairs, which allows us to draw confident conclusions from the paired Wilcoxon test.

The conclusions are: (1) A-SSVM and PMT-SSVM perform equally (p-value = 0.63); (2) SA-SSVM outperforms A-SSVM (p-value = $2.5e^{-07}$) by 1.8 points; and (3) SA-SSVM outperforms PMT-SSVM (p-value = $6.6e^{-07}$) by 1.6 points. Thus, part-aware DA outperforms strategies that ignore model structure.

4.5.4 Experiments on self-adaptive DPM

We evaluate the self-adaptive DPM in the virtual-to-real case. We assume that real world predestrains come without bounding boxes. For Caltech there is a set of pedestrian-free images, while for ETH, KIT, and CVC this is not the case. We restrict our experiments to SA-SSVM since it has shown the best accuracy in the supervised case. Moreover, we evaluate the self-adaptive method with and without GPR.

Virtual to ETH, KIT, Caltech, and CVC

Fig. 4.6 shows the results on all the testing datasets. The paired Wilcoxon test shows that SA-SSVM improves on U-SA-SSVM-GPR by 2.1 points (p-value = $3e^{-06}$). This is mainly due to the difficulty that the self-adaptive DPM (U-SA-SSVM-GPR) faces for discovering target-domain pedestrians without introducing label noise such as false positive detections. In some datasets, the accuracy of the self-adaptive DPM is very close to the supervised DA-DPM (SA-SSVM), *e.g.*, in KIT and ETH0 less than two points. Analogously, the paired Wilcoxon test shows that U-SA-SSVM-GPR improves on U-SA-SSVM by 3.0 points (p-value = $6e^{-06}$). This demonstrates the effectiveness of using GPR rather than a fixed threshold.

Self-paced Learning with GPR

Fig. 4.7 illustrates the GPR-based pedestrian selection for three iterations. At each step the classifier is updated. Thus, all samples are re-scored at the next iteration (Alg. 4, step 2). The bounding boxes (BBs) drawn with continuous lines (yellow) denote easy examples (in $\mathcal{D}^{T_{\overline{r}}}$), *i.e.*, the observations for the GPR. The BBs drawn with discontinuous lines are the uncertain detections (in $\mathcal{D}^{T_{\underline{r}\setminus\overline{r}}}$). The light discontinuous lines (green) show the selected detections after the GPR ($\upsilon_i = 1$), while the dark discontinuous lines (red) denote the rejected ones ($\upsilon_i = 0$). All pedestrians detected in the first iteration, including the one initially rejected, are either classified as easy or selected (both types are the input for steps $4-5$ of Alg. 4) in the last iteration. In fact, two new detections are collected.

4.6 Summary

DA of DPM-based object detectors is of paramount interest for preserving their accuracy across different domains. Accordingly, we have presented two supervised DA-DPM methods (A-SSVM and SA-SSVM), which can be integrated into a self-adaptive

DPM for new unlabeled or weakly labeled domains. Our DA methods do not require revisiting the source-domain data for adaptation, and only relatively little annotated data from the target domain is required to boost detection accuracy. In the case of the self-adaptive technique, samples from the target domain are automatically collected to adapt the model without any supervision, *i.e.* avoiding the need of human intervention. We have tested our proposals in the context of pedestrian detection performing a total of 384 train-test runs. Overall, two types of adaptation are evaluated: both from synthetic and general person domains, to real-world pedestrian images.

So far, we assume the DA task is only performed from one source domain to one target domain and ignored the relatedness of multiple target domains. In the next chapter, we will consider multiple target domain adaptation. To simplify the model representation, we will build higher level hierarchical models based on A-SSVM.

Table 4.3: DA from virtual to real world using different models (see main text). Average miss rate (%) is shown.

Mix. of Roots	ETH0	ETH1	ETH2	KIT	Caltech	CVC
SRC	69.1	70.3	65.8	70.8	72.6	64.9
A-SSVM	67.1±0.2	67.2±0.5	54.9±0.8	64.2±0.3	71.3±1.0	57.6±0.9
SA-SSVM-C	66.5±0.8	66.5±1.0	54.4±0.3	63.5±0.9	71.2±0.6	57.9±0.5
Δ_a	2.0±0.2	3.1±0.5	10.9±0.8	6.6±0.3	1.3±1.0	7.3±0.9
Δ_{sa}	2.6±0.8	3.8±1.0	11.4±0.3	7.2±0.9	1.4±0.6	7.0±0.5

Part-based	ETH0	ETH1	ETH2	KIT	Caltech	CVC
Δ_a	7.1±0.5	5.6±0.5	10.9±0.8	8.3±0.5	6.3±0.6	13.4±0.7
Δ_{sa}	8.4±0.3	7.3±0.4	13.0±0.3	9.1±0.7	8.8±0.6	15.1±0.3

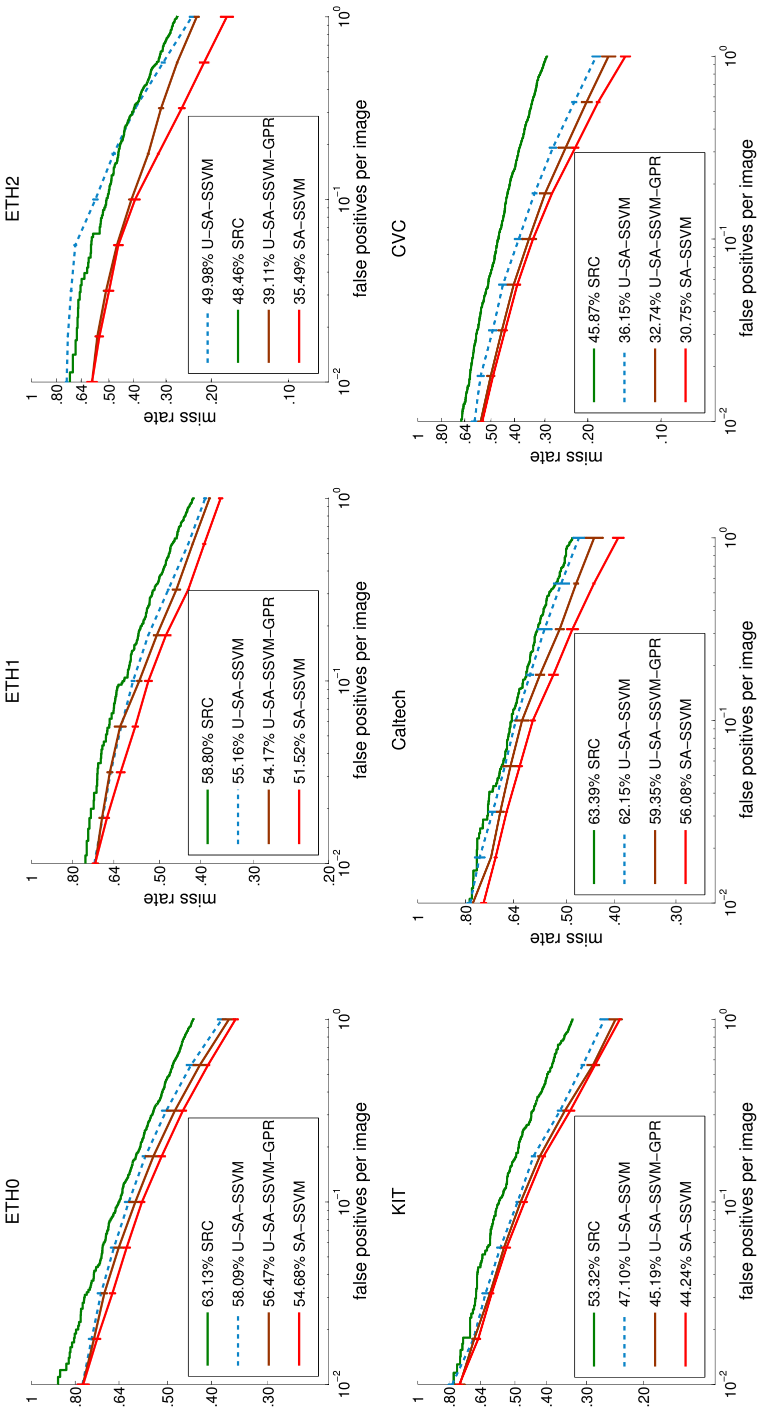


Figure 4.6: Self-adaptive DPM from virtual world to specific real-world scenes.

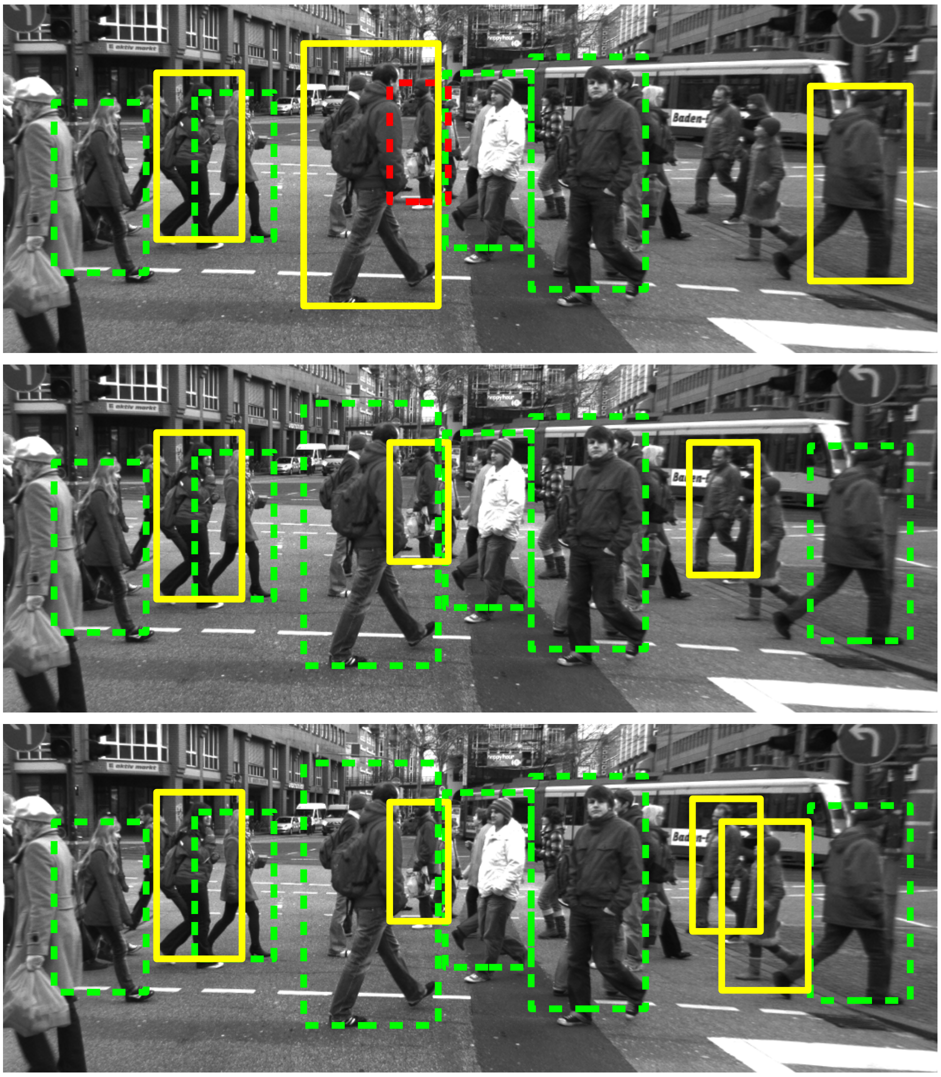


Figure 4.7: Sample selection in U-SA-SSVM-GPR. See 4.4.2 for a complete explanation.

Chapter 5

Hierarchical adaptive structural SVM for domain adaptation

In Chapter 4, we have addressed the problem of adapting deformable part-based models from virtual-world (source) to real-world (target) domains. However, we assume domain adaptation is performed from a single source to a single target domain. In this chapter, we present a novel domain adaptation method that leverages multiple target domains (or sub-domains) in a hierarchical adaptation tree. The core idea is to exploit the commonalities and differences of the jointly considered target domains.

We apply our idea to the proposed adaptive SSVM (A-SSVM), which only requires the target domain samples together with the existing source-domain classifier for performing the desired adaptation. Altogether, we term our proposal as hierarchical A-SSVM (HA-SSVM).

We use HA-SSVM for pedestrian detection and additionally verify its effectiveness on object category recognition. In the former we apply HA-SSVM to the deformable part-based model (DPM) while in the latter HA-SSVM is applied to multi-category classifiers. In both cases, we show how HA-SSVM is effective in increasing the detection/recognition accuracy with respect to adaptation strategies that ignore the structure of the target data. Since, the sub-domains of the target data are not always known a priori, we also shown how HA-SSVM can incorporate sub-domain discovery for object category recognition.

5.1 Introduction

Many domain adaptation methods assume a single domain shift between the data, *i.e.*, they perform the adaptation from a single source domain to a single target domain [9, 26, 56, 61, 87, 91, 99, 100]. Some others consider multiple source domains [27, 49, 54, 66, 116] and propose to leverage labeled data from them to perform the domain adaptation, *i.e.*, the underlying idea is to cover as much variability as possible at the source level for making more accurate predictions given a partially new domain (the target). In this chapter we focus on the complementary case to these works. In

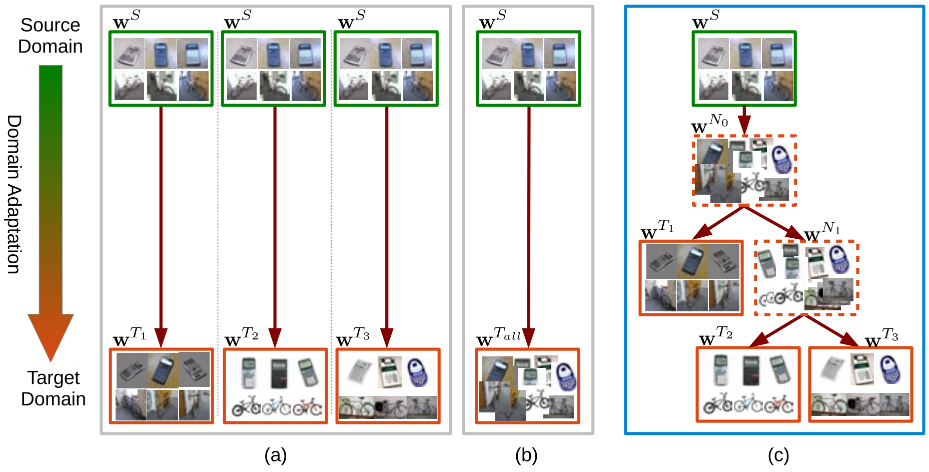


Figure 5.1: Domain adaptation methods: without losing generality we assume a single source domain and three correlated target domains (three different datasets depicting the same object categories). (a) Single layer domain adaptation: adapting to each target domain $\mathbf{w}^{T_j}$ independently. (b) Single layer domain adaptation: pooling multiple target domains. (c) Proposed hierarchical multi-layer domain adaptation. The target domains are organized in an adaptation tree. Adaptation to intermediate nodes allows to exploit commonalities between children sub-domains, while adaptation to final sub-domains allows to consider their differences. Each path from the root to a leaf of the hierarchy can be thought as a progressive adaptation, but all models (intermediate and final) are learned jointly.

other words, the main novelty is the study of the effectiveness of domain adaptation when we can structure the target domain as a hierarchy (*e.g.*, leveraging multiple correlated target domains or using some criteria to build sub-domain partitions).

The main idea of our approach is illustrated in Figure 5.1. Without losing generality assume that we have a prior source model $\mathbf{w}^S$ (*e.g.* a SVM hyperplane) and we would like to adapt it to multiple target domains (T_1, T_2, T_3) from which we have labeled data. Traditionally, $\mathbf{w}^S$ is adapted to each target domain separately, as illustrated in (a). Other option is to pool multiple target domains into a single one and adapt $\mathbf{w}^S$ to a mixed target domain as in (b). We refer to these strategies as single-layer domain adaptation.

Instead of performing isolated single-layer adaptations, we propose to make use of the relatedness of the target domains while exploiting their differences. Concretely, as it is presented in (c), we organize multiple target domains into a hierarchical structure (tree) and adapt the source model to them jointly. The adaptation to intermediate nodes allows to exploit commonalities between children sub-domains (*e.g.*, approach (b) is considered thanks to the root node of the hierarchy), while the adaptation to the final sub-domains allows to consider their differences.

Each path from the root to a leaf of the hierarchy can be thought as a progressive adaptation. However, as we will see, the adaptation of the whole hierarchy is done at once under the same objective function. This implies that our adaptation strategy is also useful in cases where the labeled data from the target domain is scarce but at the same time presents certain variability (sub-domains) worth to consider. Note that by using the approach in (a) such a reduced target domain dataset would be divided into even smaller target sub-domain datasets, which in general would end up in a poorer adaptation. On the other hand, following (b) the potential target sub-domains would be just ignored.

To simplify the model representation but without losing generality, we use the A-SSVM introduced in Chapter 4 as the basic DA algorithm at each layer of the hierarchy. A-SSVM does not require source domain samples, only target domain ones, which can significantly reduce the training (adaptation) time. We term our approach as *hierarchical A-SSVM* (HA-SSVM).

We apply our method in pedestrian detection and object category recognition. The former implies to use HA-SSVM with the deformable part-based model (DPM) while the latter implies to use HA-SSVM with multi-category classifiers. In both cases, we will show how HA-SSVM is effective in increasing the detection/recognition accuracy with respect to state-of-the-art strategies that ignore the structure of the target data. Moreover, as a proof of concept, focusing on the object category recognition application, we will also evaluate HA-SSVM in an scenario were the target sub-domains are not available a priori and must be discovered.

The rest of the chapter is organized as follows. In Section 5.2 we detail the general formulation of the proposed approach and its optimization method, as well as how to incorporate domain reshaping for discovering latent domains. Section 5.3 presents the experimental results of HA-SSVM for pedestrian detection and additionally, Section 5.4 shows its application on object recognition. Finally, Section 5.5 draws the conclusions of this chapter.

5.2 Proposed method

5.2.1 General model

Our proposal is illustrated in Figure 5.1. Assume we have a prior model $\mathbf{w}^S$ from the source domain $\mathcal{D}^S$ and multiple target domains $\mathcal{D}^{T_j}, j \in [1, D]$. Traditionally, $\mathbf{w}^S$ is adapted to each target domain independently, as illustrated in (a), or to the pooled target domain as in (b), which we call single-layer domain adaptation in this chapter. In contrast, we propose to make use of the relatedness of multiple target domains by combining them into a hierarchical adaptation tree, and adapt the prior model to them hierarchically, as in (c).

The proposed hierarchical model can be applied to any supervised learning algorithm which can incorporate prior information. In this work, we focus on the widely used SVM. This learning method considers a loss term $\mathcal{L}(\mathbf{w}; \mathcal{D})$ that captures the error with respect to the training data $\mathcal{D}$ and a regularization term $\mathcal{R}(\mathbf{w})$ that penalizes model complexity. In fact, we will focus on domain adaptation with structural SVM (SSVM), giving rise to our hierarchical A-SSVM (HA-SSVM) in Sect 5.2.2.

5.2.2 Hierarchical adaptive structural SVMs

For the sake of a better understanding, in this subsection, we introduce the involved concepts by progressive order of complexity. We first focus on single-layer domain adaptation based on adaptive SVM (A-SVM). Then, we develop our hierarchical A-SVM (HA-SVM) model. We show how to learn its parameters by using a multiple task learning (MTL) paradigm. Finally, in section 5.2.2 we consider SSVM and, therefore, introduce HA-SSVM.

We have introduced A-SVM in section 4.3.1 of Chapter 4. Details of A-SVM is given in Eq. (4.4) and Eq. (4.5). Eq. (4.4) is also called one-to-one domain adaptation. For the sake of a more understandable explanation but without losing generality, we give the formulation of HA-SVM for a hierarchy of three layers as the one illustrated in Figure 5.1 (c). Assume we have a source model $\mathbf{w}^S$ and three target domains $\mathbf{w}^{T_j}$, $j \in [1,3]$. Let $\mathbf{w} = [\mathbf{w}^{N_0\prime}, \mathbf{w}^{N_1\prime}, \mathbf{w}^{T_1\prime}, \mathbf{w}^{T_2\prime}, \mathbf{w}^{T_3\prime}]'$, then the objective function of the three-layers HA-SVM is written as follows:

$$\begin{aligned} J(\mathbf{w}) = \; & \frac{1}{2}\|\mathbf{w}^{N_0} - \mathbf{w}^S\|^2 + C\textstyle\sum_{j=1}^3 \mathcal{L}(\mathbf{w}^{N_0}; \mathcal{D}_l^{T_j}) \\ & + \frac{1}{2}\|\mathbf{w}^{N_1} - \mathbf{w}^{N_0}\|^2 + C\textstyle\sum_{j=2}^3 \mathcal{L}(\mathbf{w}^{N_1}; \mathcal{D}_l^{T_j}) \\ & + \frac{1}{2}\|\mathbf{w}^{T_1} - \mathbf{w}^{N_0}\|^2 + C\mathcal{L}(\mathbf{w}^{T_1}; \mathcal{D}_l^{T_1}) \\ & + \frac{1}{2}\|\mathbf{w}^{T_2} - \mathbf{w}^{N_1}\|^2 + C\mathcal{L}(\mathbf{w}^{T_2}; \mathcal{D}_l^{T_2}) \\ & + \frac{1}{2}\|\mathbf{w}^{T_3} - \mathbf{w}^{N_1}\|^2 + C\mathcal{L}(\mathbf{w}^{T_3}; \mathcal{D}_l^{T_3}) \end{aligned} \tag{5.1}$$

Eq. (5.1) is in a multi-task learning paradigm form, where the optimization of each $\mathbf{w}^{T_j}$ can be understood as an individual task. The key issue of the multi-task learning lies in how the relationships between tasks are incorporated. As we can see from Eq. (5.1), each task is related by the regularization term, *e.g.*, T_2 and T_3 are connected by $\|\mathbf{w}^{T_j} - \mathbf{w}^{N_1}\|^2$, while T_1 is directly connected to N_0, which is adapted from $\mathbf{w}^S$.

At testing time, for a testing sample from target domain j, we can directly extract the learned parameters $\mathbf{w}_j^T$ and apply the linear decision function:

$$f^{T_j}(\mathbf{x}) = \mathbf{w}^{T_j\prime}\Phi(\mathbf{x}) \; . \tag{5.2}$$

Comparing to the single-layer adaptation $\|\mathbf{w}^{T_j} - \mathbf{w}^S\|^2$ as in Figure 5.1 (a), HA-SVM has several advantages. First, HA-SVM can make use of training samples from multiple related target domains instead of just one. For example, a single-layer domain adaptation only uses the training samples from $T_j, j \in \{1,2,3\}$ in three different optimization runs, while HA-SVM can integrate the samples from the three target domains accounting for their hierarchical structure. Second, the target model $\mathbf{w}^{T_j}$ is not directly regularized by $\mathbf{w}^S$ but some shared intermediate models $\mathbf{w}^{N_i}$, which allows $\mathbf{w}^{T_j}$ to be regularized in a more flexible space. As $\mathbf{w}^{T_j}$ goes down apart from $\mathbf{w}^S$ further in the adaptation tree, less constrain from $\mathbf{w}^S$ is imposed. This can be interpreted as a progressive adaptation.

For single-layer domain adaptation, another straightforward strategy is to pool all target domains and train a single adaptive SVM with all available target samples, as illustrated in Figure 5.1 (b). Comparing to this method, HA-SVM can take the same advantage of using all available labeled data while allows each target domain model to be more discriminative in its own domain. The pooling-based method requires the final model to compromise to each domain in order to minimize the training error, and thus such model may lose the discriminative power in the individual target domains. Our experimental results in Sect. 5.3 and Sect. 5.4 confirm this observation.

To minimize Eq. (5.1), we employ Quasi-Newton LBFGS method, which requires the objective function and the partial derivatives of its parameters. These partial derivatives are:

$$\begin{aligned}
\tfrac{\partial J(\mathbf{w})}{\partial \mathbf{w}^{N_0}} &= 3\mathbf{w}^{N_0} - \mathbf{w}^{S} - \mathbf{w}^{N_1} - \mathbf{w}^{T_1} + C\textstyle\sum_{j=1}^{3} \tfrac{\partial \mathcal{L}(\mathbf{w}^{N_0};\mathcal{D}_l^{T_j})}{\partial \mathbf{w}^{N_0}} \ , \\
\tfrac{\partial J(\mathbf{w})}{\partial \mathbf{w}^{N_1}} &= 3\mathbf{w}^{N_1} - \mathbf{w}^{N_0} - \mathbf{w}^{T_1} - \mathbf{w}^{T_2} + C\textstyle\sum_{j=2}^{3} \tfrac{\partial \mathcal{L}(\mathbf{w}^{N_1};\mathcal{D}_l^{T_j})}{\partial \mathbf{w}^{N_1}} \ , \\
\tfrac{\partial J(\mathbf{w})}{\partial \mathbf{w}^{T_1}} &= \mathbf{w}^{T_1} - \mathbf{w}^{N_0} + C\tfrac{\partial \mathcal{L}(\mathbf{w}^{T_1};\mathcal{D}_l^{T_1})}{\partial \mathbf{w}^{T_1}} \ , \\
\tfrac{\partial J(\mathbf{w})}{\partial \mathbf{w}^{T_2}} &= \mathbf{w}^{T_2} - \mathbf{w}^{N_1} + C\tfrac{\partial \mathcal{L}(\mathbf{w}^{T_2};\mathcal{D}_l^{T_2})}{\partial \mathbf{w}^{T_2}} \ , \\
\tfrac{\partial J(\mathbf{w})}{\partial \mathbf{w}^{T_3}} &= \mathbf{w}^{T_3} - \mathbf{w}^{N_1} + C\tfrac{\partial \mathcal{L}(\mathbf{w}^{T_3};\mathcal{D}_l^{T_3})}{\partial \mathbf{w}^{T_3}} \ .
\end{aligned} \tag{5.3}$$

In our implementation, the LBFGS based optimization converges to the optimum efficiently for both single-layer A-SVM and HA-SVM (as well as for the HA-SSVM defined in next subsection).

The proposed HA-SVM can be extended for SSVM, giving rise to our HA-SSVM. SSVM allows the training of a classifier for general structured output labels. SSVM minimizes the following regularized risk function:

$$\begin{aligned}
&\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 \\
&+C\textstyle\sum_{i=1}^{N}[\max_y \mathbf{w}'\Phi(\mathbf{x}_i, y) + \Delta(y_i, y) - \Phi(\mathbf{x}_i, y_i)] \ ,
\end{aligned} \tag{5.4}$$

where y_i is the ground truth output (label) of sample $\mathbf{x}_i$, and y runs on the alternative outputs. $\Delta(y_i, y)$ is a distance in output space. $\Phi(\mathbf{x}, y)$ is the feature vector from a given sample $\mathbf{x}$ of label y. Accordingly, Eq. (4.4) can be extended to A-SSVM as in Eq. (4.8). Here, we give the explicit form:

$$\begin{aligned}
&\min_{\mathbf{w}^T} \frac{1}{2}\|\mathbf{w}^T - \mathbf{w}^S\|^2 \\
&+C\textstyle\sum_{i=1}^{N}[\max_y \mathbf{w}^{T\prime}\Phi(\mathbf{x}_i, y) + \Delta(y_i, y) - \Phi(\mathbf{x}_i, y_i)] \ .
\end{aligned} \tag{5.5}$$

Correspondingly, the final adapted classifier f^T can be written as:

$$f^T(\mathbf{x}) = \max_y[\mathbf{w}^{S\prime}\Phi(\mathbf{x}, y) + \underbrace{\Delta\mathbf{w}'\Phi(\mathbf{x}, y)}_{\Delta f(\mathbf{x})}] \ , \tag{5.6}$$

where $\Delta\mathbf{w} = \mathbf{w}^T - \mathbf{w}^S$. Therefore, Eq. (5.5) can be integrated into the proposed hierarchical adaptation framework. In particular, we must proceed in the same way as going from (4.4) to (5.1), but now starting with (5.5), thus, giving rise to HA-SSVM.

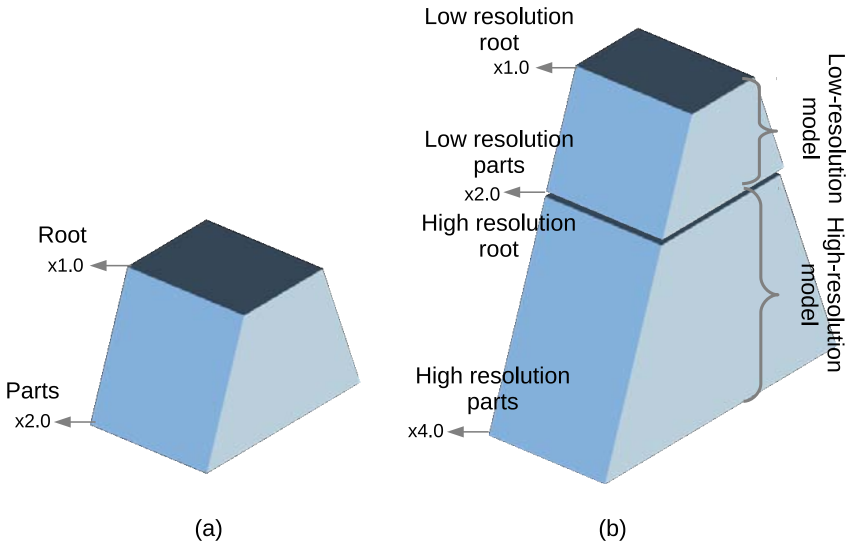


Figure 5.2: (a) Original feature pyramid in DPM. (b) The extended feature pyramid for multi-resolution adaptive DPM.

5.3 Experiments: domain adaptation of DPMs

In this section we apply HA-SSVM in the popular deformable part-based model (DPM) framework [35], focusing on pedestrian detection. The latent structured SVM form of a DPM objective function has been given in Eq. (4.3). Applying A-SSVM for DPM can is given in Eq. (4.8). Thus, we can build HA-SSVM for DPM. We implemented it in the DPM 5.0 framework [43], which is the latest at the moment of doing this research. When applying an adapted DPM in a particular target domain (*i.e.*, in testing time), we do not use the full vector of parameters jointly learned for all the hierarchy of target domains, instead we only use the sub-vector of parameters corresponding to such particular target domain. In other words, we follow Eq. (5.2).

5.3.1 Experiments on Pedestrian Detection

Figure 5.3 illustrates two different cases of DPM domain adaptation using HA-SSVM that we evaluate here. In (a) the source classifier is adapted to three different target domains (different datasets in this case). In (b) we adapt the source classifier to detect pedestrians from image windows of two different resolution categories. The main idea is to divide the target domain into sub-domains according to the resolution of the pedestrian samples, *i.e.*, different resolutions are regarded as different domains. Here we consider only two resolutions, low and high. Note that low resolution pedestrians tend to be blur and their poses are less discriminative than for high resolution ones.

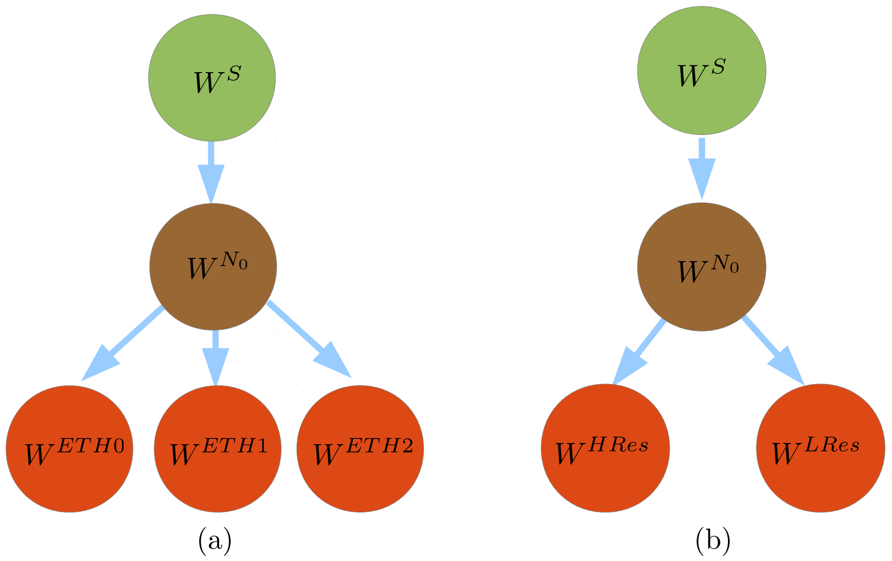


Figure 5.3: HA-SSVM applied to DPM: (a) adaptation to three related datasets (domains), namely ETH0, ETH1 and ETH2, which were acquired with the same camera but different environments; (b) adaptation of a single resolution detector for applying different detectors when processing small and large image windows, which would correspond to pedestrians imaged with low (LRes) and high (HRes) resolution respectively.

ASVM	Adaptive SVM [116]. It does not require the source domain data, only the learned source classifier. In contrast to A-SSVM, ASVM does not consider structural information.
PMT-SVM	Projective model transfer SVM [2], which is a variant of ASVM.
GFK	The geodesic flow kernel method [48], which requires both source and target domain data (including testing data).
MMDT	Max-margin domain transfer method of [56], which learns a mapping from target domain to source domain as well as a discriminative classifier using the mapped target and source domain features.
A-SSVM	Analogous to ASVM.
A-SSVM-ALL	Analogous to ASVM, using all sub-target domains as a single one.

Table 5.1: Different types of learned DPM classifiers.

Datasets

For the source-domain, we use the same virtual-world dataset as in the previous chapters. The target-domain real-world datasets that we use are ETH [33], Caltech [22] and KITTI [38]. The ETH dataset consists of three sub-datasets, namely ETH0, ETH1 and ETH2, and is used to evaluate the setting of Figure 5.3(a). These sub-datasets are collected from different environments but with the same camera sensor and pose, thus, we consider them as related sub-domains. Caltech and KITTI are used in two different experiments, both for evaluating the setting of Figure 5.3(b).

Setup

In supervised domain adaptation it is assumed that there are available just a few labeled data from the different target domains. In order to emulate this setting, we selected only 100 pedestrians for the experiments with ETH0, ETH1, ETH2 and Caltech, which roughly correspond to the 6%, $1.5\%, 3\%, 5\%$, respectively, of the available pedestrians for training. We use all the training pedestrian-free images of these datasets, *i.e.*, $999, 451, 354, 1,824$ images, respectively. We follow the Caltech evaluation criterion [22] and plot the average miss rate *vs* false positive per image (FPPI) curve. We use the suggested reasonable setting and therefore test on the pedestrians taller than 50 pixels. Each train-test experiment is repeated five times and we report the mean and standard deviation of the repetitions. To ensure fair comparisons, we use the same random samples for different training methods. To evaluate the performance of HA-SSVM, we compare it to the baselines described in Table 5.1.

In fact, as touchstone of HA-SSVM for pedestrian detection, our first test was the participation in the *Pedestrian Detection Challenge* of the KITTI benchmark[1] as part of the *Reconstruction Meets Recognition Challenge (RMRC)* held in conjunction with the ICCV'2013 celebrated in Sydney. At that time we did not have written neither a report nor this chapter, so we participated with the multi-resolution HA-SSVM DPM described here, but with the generic name of *DA-DPM* (domain adaptive DPM). In

[1]`http://www.cvlibs.net/datasets/kitti/`

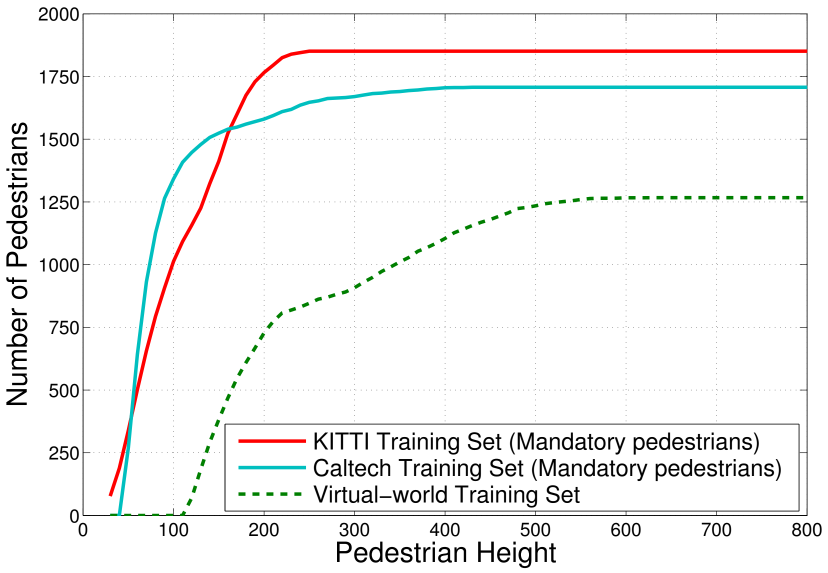
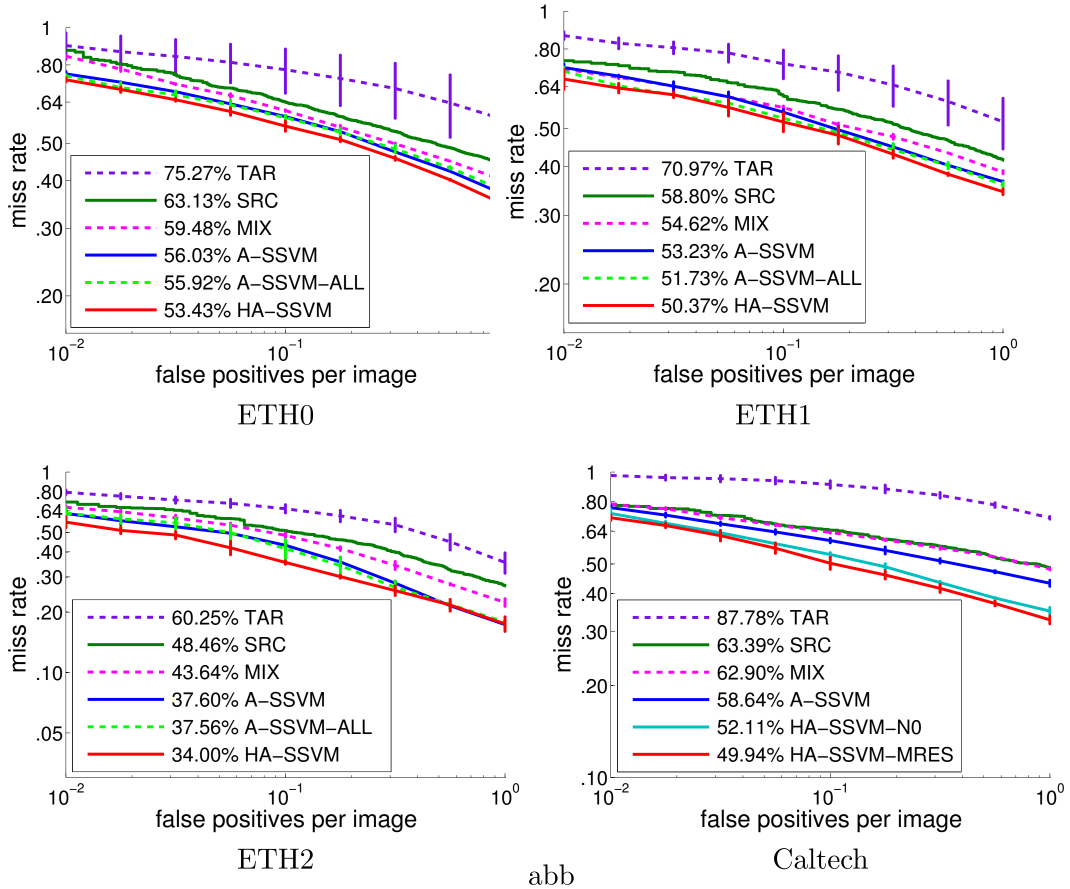


Figure 5.4: Cumulative histogram of the pedestrians' height in Caltech, KITTI and virtual-world training dataset. The virtual-world dataset contains less low-resolution pedestrians than the real-world ones.

this case, we used 200 pedestrians of the KITTI training set, roughly the 11% of the available ones, as well as $2,000$ pedestrian-free images of the $7,518$ available for training. We note that under the KITTI benchmark, the object detection evaluation criterion is different from the Caltech one. Accuracy is measured as precision *vs* recall instead of miss rate *vs* FPPI. Note also that, in order to avoid parameter tuning, the ground truth of the KITTI testing data is not available.

For all the experiments, the SRC classifier (see Table 5.1) is the same DPM, trained with the virtual-world dataset. It is worth to mention that we use a DPM root filter of 12×6 HOG cells (each cell is of 8×8 pixels), *i.e.*, the minimum size of the detectable pedestrians is 96×48 pixels. Then, for the multi-resolution adaptation (to Caltech and KITTI), we build the two-layer hierarchical model of Figure 5.3(b). When computing features, we add an extra octave at the bottom of the feature pyramid and then divide the pyramid into two pyramids: high resolution pyramid which contains pedestrians taller than 96 pixels, and low resolution pyramid which contains pedestrians lower than 96 pixels. The extended feature pyramid is illustrated in Figure 5.2(b). During training time, we assign the training pedestrians to the high and low resolution domains according to the height of their bounding boxes, while the background samples are shared by both domains. In figure Figure 5.4 we show the pedestrian height distribution of the virtual- and real-world training datasets. Note that the virtual-world dataset has few low resolution pedestrians compared with the real-world ones, thus making the pursued adaptation challenging. In testing time, we



abb

Figure 5.5: ETH0, ETH1 and ETH2 show the adaptation results from virtual-world to ETH three sub-datasets. Caltech shows results of adapting virtual-world DPM detector to a multi-resolution detector in Caltech pedestrian dataset. A-SSVM is trained with mixed high and low resolution samples. HA-SSVM-N0 corresponds to W^{N_0} in the multi-resolution adaptation tree and HA-SSVM-MRES is the adapted multi-resolution detector.

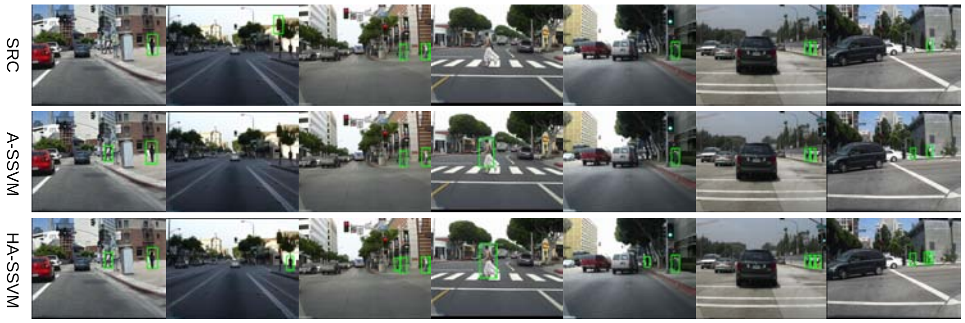


Figure 5.6: Pedestrian detection on Caltech dataset for SRC, A-SSVM and HA-SSVM based models. The results are drawn at FPPI = 0.1.

Rank	Method	Moderate	Easy	Hard
1	DA-DPM	**45.51 %**	**56.36 %**	**41.08 %**
2	LSVM-MDPM-sv	39.36 %	47.74 %	35.95 %
3	LSVM-MDPM-us	38.35 %	45.50 %	34.78 %
4	mBoW (LP)	31.37 %	44.28 %	30.62 %

Table 5.2: Evaluation on KITTI pedestrian detection benchmark during the RMRC'2013. Results are given as average precision (AP). Our method DA-DPM (which is actually the application of HA-SSVM to a DPM trained with virtual-world data) outperforms the previous best method LSVM-MDPM-sv by 5 $\sim$ 8 points in average precision. "LP" means that the method uses point clouds from a Velodyne laser scanner.

apply the two adapted models to the corresponding resolution pyramid and finally we combine their detections and apply non-maximum suppression to obtain the final detections.

Results

In Figure 5.5 we can see the results for the setting of Figure 5.3(a). It can be appreciated that pooling-all-target-domains strategy (A-SSVM-ALL) can have better adaptation accuracy than using single target domain data (A-SSVM). However, HA-SSVM achieves even better accuracy when trained with the same samples as A-SSVM-ALL, which demonstrates the importance of leveraging multiple target domains in a hierarchy.

In Figure 5.5 we can also see the results of applying setting Figure 5.3(a) to Caltech. We additionally assessed the accuracy provided by the intermediate model $\mathbf{w}^{N0}$, which is denoted by HA-SSVM-N0 in contrast to HA-SSVM-MRes which corresponds to the full multi-resolution adaptation. Note how even HA-SSVM-N0 shows better

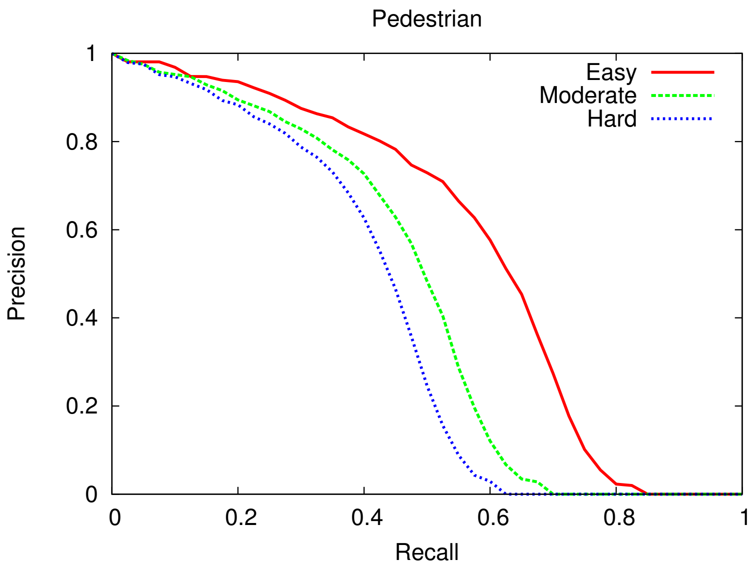


Figure 5.7: Pedestrian detection results on KITTI benchmark.

classification accuracy than A-SSVM. It demonstrates the effectiveness of the progressive adaptation in HA-SSVM, which can be explained by the fact that the multi-task training learns general shared parameters for multiple target domains (*i.e.* high- and low-resolution domains), while single-task A-SSVM does not take into account the differences of multi-resolution samples. Of course, HA-SSVM-MRes is providing the best accuracy. Quantitative results are shown in Figure 5.6, where it can be seen that HA-SSVM-MRes is capable of detecting lower resolution pedestrians.

Finally, as can be see in Table 5.2, we won the pedestrian detection challenge of the RMRC'2013[2], *i.e.*, we outperformed LSVM-MDPM-sv [1], LSVM-MDPM-us [35] and mBoW [5]. Our precision-recall curves can be seen in Figure 5.7.

We think that this was a quite remarkable result because, as we mentioned before, in order to adapt our virtual-world based pedestrian DPM we only used the $\sim$ 11% of the KITTI training pedestrians and the $\sim$ 27% of the available pedestrian-free images. This implies that the adaptation took 20 minutes approximately in our 1 core @ 3.5 Ghz desktop computer, while training the original DPM [35] with all the full KITTI training set may need around 10 hours in the same conditions. It is also worth to point out that, as can be deduced from Figure 5.4, the number of (virtual-world) pedestrians used for building our source model plus the 200 pedestrians selected from the KITTI training dataset is still lower than the total number of pedestrians in the KITTI dataset. Similarly for the pedestrian-free images, since we use $2,000$ from the

[2]In the RMRC'2013 program it can be checked that we did a talk as winners of the pedestrian detection challenge, see `http://ttic.uchicago.edu/~rurtasun/rmrc/program.php`.

virtual world to build the source model and 2,000 from the KITTI training set to do the adaptation, while there are around 7,500 available for training. Moreover, as to the best of our knowledge, this was the first time that a domain adapted object detector wins such a challenge.

5.4 Additional experiments: domain adaptation for object recognition

In the following, we evaluate HA-SSVM on multi-category classifiers. We begin with the scenario in which the target sub-domains are given a priori. After we assess the scenario in which such sub-domains must be discovered. For illustrating how HA-SSVM operates with multi-category classifiers, we focus on object category recognition.

Assume we are given an set of N examples, $\mathcal{D}$, each one labeled as belonging to a category among K possible ones, *i.e.* $\mathcal{D} = \{(\mathbf{x}_i, y_i)|\mathbf{x}_i \in \mathbb{R}^n, y_i \in \{1,\ldots,K\}\}_{i=1}^N$. Let $\mathbf{w}_1,\ldots,\mathbf{w}_K$ be the parameters of K linear category classifiers, so that a new example $\mathbf{x}$ is assigned to a category according to the rule $f(\mathbf{x}) = \mathrm{argmax}_{k\in\{1,\ldots,K\}}\, \mathbf{w}_k'\mathbf{x}$. Let $\mathbf{w} = [\mathbf{w}_1',\ldots,\mathbf{w}_K']'$ in $\mathbb{R}^{Kn}$, and a feature map $\Phi(\mathbf{x},y) = [\mathbf{0}',\ldots,\mathbf{x}',\ldots,\mathbf{0}']'$, where $\mathbf{0}\in\mathbb{R}^n$ is the zero vector and $\mathbf{x}$ is located at the y-th slot in $\Phi(\mathbf{x},y)$. Now the multi-category classification problem can be treated as a special case of structure output prediction:

$$f(\mathbf{x}) = \underset{y\in\{1,\ldots,K\}}{\mathrm{argmax}}\ \mathbf{w}'\Phi(\mathbf{x},y). \tag{5.7}$$

In order to apply HA-SSVM, Eq. (5.5) can be directly used as a basic adaptation unit by writing the loss term as:

$$\textstyle\sum_{i=1}^N[\max_{\hat{y}\in\{1,\ldots,K\}}(\mathbf{w}'\Phi(\mathbf{x},\hat{y}) \quad +\Delta(\hat{y},y_i)) - \mathbf{w}'\Phi(\mathbf{x}_i,y_i)], \tag{5.8}$$

where $\Delta(\hat{y},y_i)$ is the 0-1 loss function.

5.4.1 Known target sub-domains

Datasets

We evaluate HA-SSVM for object category recognition using the benchmark domain adaptation dataset known as *Office-Caltech* [48,56]. This dataset combines the *Office* [91] and *Caltech256* [51] datasets. In particular, *Office-Caltech* consists of the 10 overlapping object categories between *Office* and *Caltech256*, which are `backpack`, `calculator`, `coffee-mug`, `computer-keyboard`, `computer-monitor`, `computer-mouse`, `head-phones`, `laptop-101`, `touring-bike` and `video-projector` in the terminology of *Caltech256*.

ASVM	Adaptive SVM [116]. It does not require the source domain data, only the learned source classifier. In contrast to A-SSVM, ASVM does not consider structural information.
PMT-SVM	Projective model transfer SVM [2], which is a variant of ASVM.
GFK	The geodesic flow kernel method [48], which requires both source and target domain data (including testing data).
MMDT	Max-margin domain transfer method of [56], which learns a mapping from target domain to source domain as well as a discriminative classifier using the mapped target and source domain features.
A-SSVM	Analogous to Table 5.1.
A-SSVM-ALL	Analogous to Table 5.1.

Table 5.3: Different types of learned multi-category classifiers.

From the viewpoint of domain adaptation, *Office-Caltech* consists of four domains. One domain, called *caltech* (C), corresponds to the images of *Caltech256*, which were collected from the internet using Google. The other three domains come from *Office*, namely the *amazon* (A), *webcam* (W) and *dslr* (D) domains. The *amazon* domain is a collection of product images from `amazon.com`. The *webcam* and *dslr* domains contain images taken by a (low resolution) webcam and a (high resolution) digital single-lens reflex camera, respectively. Cross-domain variations are not the only ones, but for a particular domain and category, the objects are imaged under different poses and illumination conditions.

Setup

We follow the experimental setup of [48,56,91], which we summarize in the following. We have four domains (A, W, D, C) and the same 10 object categories per domain. For each experiment, one domain is selected as source domain and the other three as target domains. The number of examples per category varies from domain to domain and from category to category. When A is the source, 20 examples are randomly selected per category for training, while when the sources are either W, D or C, only 8 examples are selected per category. When a domain plays the role of target, only 3 examples are selected per category for performing the domain adaptation (training). All the examples of the target domains not used for training are used for testing. The accuracy of the classification is measured as the number of correctly classified test examples divided by the total number of them (*i.e.*, without distinguishing object categories). In fact, since the splitting of the available examples into training (source and target) and testing (target) is based on random selection, each experiment is repeated 20 times. Therefore, the average of the 20 obtained accuracy values is actually used as final accuracy measure together with its associated standard deviation.

In order to make easier across-paper comparisons, we use the same 20 random train/test splits available from [56]. Moreover, rather than using our own feature computation software, we use the pre-computed SURF-based bag of (visual) words (BoW) available for the images of *Office-Caltech*. Then, following [48], we apply PCA to such original SURF-BoW to obtain histograms of 20 visual words (bins).

Baselines

We compare our algorithm to the baselines summarized in Table 5.3. A-SVM, PMT-SVM, A-SSVM and A-SSVM-ALL are adapted with the target domain examples and the source classifiers, the rest of methods require the target domain examples and the original source domain ones for retraining. For the A-SVM and PMT-SVM methods we use the implementation provided by [2], including MOSEK optimization [70]. We run GFK and MMDT using the code of [56]. Note that in [56], GFK and MMDT are the best performing methods among others, including ARCT [63] and HFA [26] methods. All these methods, except A-SSVM-ALL, follow the one-to-one domain adaptation style (Fig. 5.1(a)), *i.e.*, an independent domain adapation is performed for each target domain.

Results

We first evaluate the accuracy of HA-SSVM with a two-layer adaptation tree, *i.e.*, all the target domain datasets are at the same layer and connected to the source domain dataset by an intermediate node, similar to Figure 5.3(a). The accuracy for each source/targets split is shown in Table 5.4. Table 5.5 shows the accuracy of each algorithm averaged over all domain splits. It is worth to note that our results for GFK and MMDT are totally in agreement with the ones presented in [56] for the same experiments and settings.

From Table 5.4 and Table 5.5, it is clear the importance of using all the available target-domain examples. Note that the best performing methods, A-SSVM-ALL (Fig. 5.1(b) style) and HA-SSVM (Fig. 5.1(c) style), do so in contrast to the rest of methods, which follow the one-to-one domain adaptation style (Fig. 5.1(a)). For instance, if we focus in the A$\rightarrow$[W,D,C] case, both A-SSVM-ALL and HA-SSVM use 90 target domain examples simultaneously, *i.e.* 3 target domains $\times$ 10 object categories per target domain $\times$ 3 examples per category. 1-to-1 domain adaptation style methods use 30 W examples for performing the A$\rightarrow$W domain adaptation, and analogously for A$\rightarrow$D and A$\rightarrow$C. Therefore, potential commonalities between W, D, and C domains are not used. Moreover, HA-SSVM outperforms A-SSVM-ALL, in agreement with our hypothesis that using the underlying hierarchical structure of the target domains is better than just mixing them blindly.

Focusing then on HA-SSVM, it is also interesting to see if other target domain structure (*e.g.*, a three-layer hierarchy) can improve the domain adaptation accuracy obtained so far. We test HA-SSVM with various three-layer adaptation trees. Table 5.6 shows the results. The three-layer adaptation tree achieves results as good as the ones of the two-layer tree and some of them are even better. By further analyzing the domain relationships of the *Office* and *Caltech256* datasets, we found that there are strong connections to previous studies on domain similarity. In particular, to the rank of domain (ROD) [48] and the quantification of domain shift (QDS) [72]. We show the domain similarities in Table 5.7 using QDS measurement. We note that the three-layer hierarchies which yield to best accuracies are those that best capture the underlying domain relationship. For instance, in the first group of Table 5.6, A$\rightarrow$[C, [D, W]] achieves better accuracy than other adaptation trees, which is in agreement with the fact that [D, W] show higher similarity than [D, C] and [C, W] (see Table 5.7).

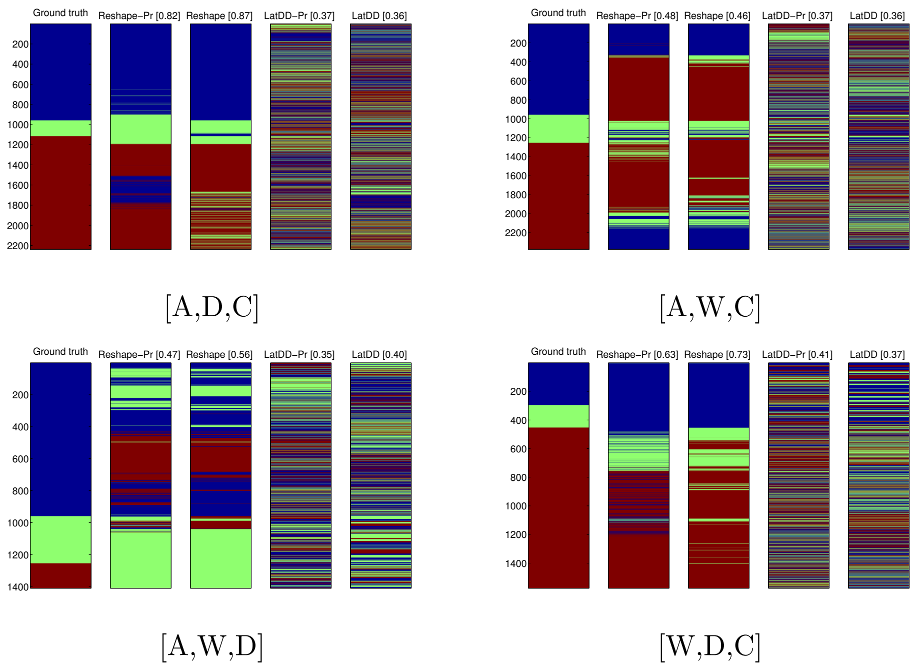
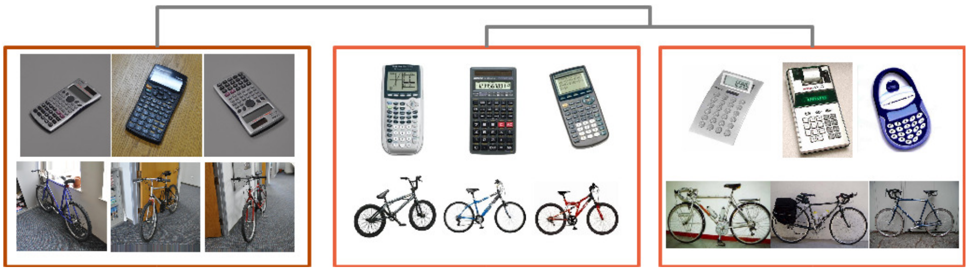


Figure 5.8: Visualization of domain discovery results. The vertical axis indicates the indexes of the examples. Each color represents a different domain. For each sub-figure, the first column shows the original domain (groundtruth). The following columns are domain reshaping with predicted category labels ('Reshape-Pr'), domain reshaping with groundtruth category labels ('Reshape'), latent domain discovery with predicted category labels ('LatDD-Pr') and latent domain discovery with groundtruth category labels ('LatDD'). Within the brackets we show the estimated domain discovery accuracy running in [0,1].

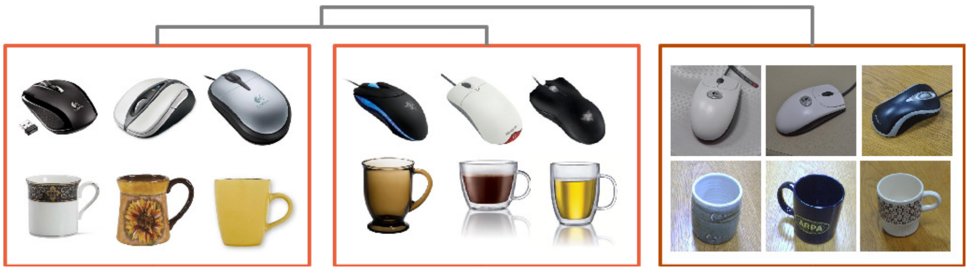
5.4.2 Latent target sub-domains

Now we consider the scenario where the domain labels are not given a priori for the target data. In particular, we use again the *Office-Caltech* dataset with the same settings than in Sect. 5.4.1. However, we mix the target datasets by removing the domain labels. In these experiments, we first compare two recent domain discovery algorithms, in particular, latent domain discovery [54] (we call it *LatDD*), and domain reshaping [46] (we call it *Reshape*). Finally, we evaluate the adaptation accuracies with the discovered domains, using HA-SSVM.

LatDD and *Reshape* require category labels to operate. However, in our domain adaptation setting we assume that only a few target domain examples have category label, which may be a handicap for such domain discovery methods. In this point, as proof-of-concept, we assumed that the target domain data does not have category labels. Therefore, we first applied the source domain model to predict the category



Original target domains: [A,D,C]



Original target domains: [A,W,D]

Figure 5.9: *Reshape-Pr* + HA-SSVM qualitative results. Three exemplars for two categories are shown for each domain discovered by *Reshape-Pr*. The three-layer hierarchy used by HA-SSVM is also indicated for the underlying domains [A,D,C] (top) and [A,W,D] (bottom). In both cases, it correspond to the most accurate HA-SSVM-based multi-category classifier among the different ones that can be obtained for different three-layer hierarchy configurations.

labels in the unlabeled target domain (*i.e.*, the domain obtained by mixing the three domains not used as source). We denote by *LatDD-Pr* and *Reshape-Pr* the cases where we use predicted category labels instead of the groundtruth category labels.

LatDD requires as input the number of sub-domains to be discovered (originally this method has been developed to discover source domains), while *Reshape* involves an iterative process to search for the optimum number of sub-domains. We want to compare the HA-SSVM results in terms of discovered sub-domains *vs* a priori given ones, but only from the point of view of how the target data is distributed among a predefined number of target sub-domains. In other words, in these experiments we do not want the number of domains to be discovered. Therefore, we set this value to 3 for fair comparison with the experiments in Sect. 5.4. It is worth to note that for *Reshape/Reshape-Pr* we only use the so-called *distinctiveness* maximization step [46].

Figure 5.8 depicts the domain discovery results. It can be seen that *Reshape* and *Reshape-Pr* are clearly more accurate than *LatDD* and *LatDD-Pr* predicting the domains. Comparing *Reshape* and *Reshape-Pr*, we see that the former is more accurate as should be expected since it relies on groundtruth data. Comparing *LatDD* with *LatDD-Pr*, the accuracy differences are smaller than for *Reshape* and *Reshape-Pr*.

Now, for applying HA-SSVM, *LatDD-Pr* and *Reshape-Pr* are treated equally and as follows. For each discovered sub-domain, we assume that 3 examples are category-labeled for each category. Since our experiments are with 10 categories, as in Sect. 5.4, 90 target-domain examples must be available for performing domain adaptation (training) and the rest are used for testing. Since this train/test split is based on random selection, we repeat each experiment 20 times in order to emulate the setting of Sect. 5.4. Note that for *LatDD-Pr* and *Reshape-Pr* this means that we discard the predicted category labels, but we require only 90 examples to be labeled. In fact, *LatDD* and *Reshape* are not considered for HA-SSVM since these methods would require the category labels of all the target data (we included them in Figure 5.8 just as reference to compare with their *predicted* counterparts).

Table 5.8 shows the final domain adaptation accuracies. As in Sect. 5.4 we evaluate two- and three-layer hierarchies, for the latter we only show the best obtained result among all possible configurations. We see that these results are comparable to the best obtained in Sect. 5.4 (also included in Table 5.8 as 'Given' for the reader convenience). Although we work with discovered sub-domains, HA-SSVM still outperforms the single-layer adaptation pooling-all strategy. *Reshape-Pr* outperforms *LatDD-Pr* as expected given the domain discovery accuracies seen in Figure 5.8. However, the differences in accuracy are much larger for domain discovery than for the final object category classification, which may be due to the fact that HA-SSVM trains all the object category classifiers simultaneously for all domains in the hierarchy; thus, partially compensating domain assignment errors. Finally, for illustration purposes, Figure 5.9 shows object examples within the domains discovered by *Reshape-Pr* and some of the three-layer adaptation trees used by HA-SSVM.

5.5 Summary

In this chapter, we present a novel domain adaptation method which leverages multiple target domains (or sub-domains) in a hierarchical adaptation tree. The key idea of the method is to exploit the commonalities and differences of the jointly considered target domains. Given the increasing interest on structural SVM (SSVM) classifiers, we have applied this idea to the domain adaptation method known as adaptive SSVM (A-SSVM), which only requires the target domain samples together with the existing source-domain classifier for performing the desired adaptation. Thus, in contrast with many other methods, the source domain samples are not required. Altogether, we term the presented domain adaptation technique as hierarchical A-SSVM (HA-SSVM).

As proof of concept we have applied HA-SSVM to pedestrian detection and object category recognition applications. The former involved to apply HA-SSVM to the widespread deformable part-based model (DPM) while the latter implied their application to multi-category classifiers. In both cases, we showed how HA-SSVM is effective in improving the detection/recognition accuracy with respect to state-of-the-art strategies that ignore the structure of the target data. Moreover, focusing on the object category recognition application, we have evaluated HA-SSVM assuming that the target domains are discovered, obtaining comparable results to the case in which such domains are known a priori.

The hierarchical structure has a lot of nice properties. In addition to the progressive adaptation ability in this chapter, it could be used to model category-to-instance adaptation, *e.g.*, the leaf nodes can be trained as exemplar classifiers while the root node is the category classifier. In the next chapter, we further extend HA-SSVM for hierarchical online domain adaptation where the adaptation tree is learned frame by frame.

	A → W	A → D	A → C	W → A	W → D	W → C
ASVM	65.0 ± 1.0	51.6 ± 1.1	30.9 ± 0.6	48.6 ± 1.1	54.4 ± 1.5	29.8 ± 1.0
PMT-SVM	$\underline{65.9}$ ± 1.0	52.6 ± 1.1	32.3 ± 0.6	49.0 ± 1.1	57.9 ± 1.6	30.4 ± 0.9
GFK	56.5 ± 0.8	45.3 ± 0.9	38.6 ± 0.4	45.8 ± 0.6	**73.8** ± 0.7	32.6 ± 0.6
MMDT	65.1 ± 1.2	54.5 ± 1.0	39.7 ± 0.5	$\underline{50.6}$ ± 0.8	62.5 ± 1.0	34.8 ± 0.8
A-SSVM	60.0 ± 0.9	49.7 ± 0.8	**42.6** ± 0.5	49.5 ± 0.5	67.4 ± 0.7	37.3 ± 0.5
A-SSVM-ALL	64.5 ± 0.7	$\underline{55.1}$ ± 0.8	**42.6** ± 0.3	49.8 ± 0.5	$\underline{67.8}$ ± 0.8	$\underline{39.0}$ ± 0.3
HA-SSVM	**69.8** ± 0.7	**59.7** ± 0.9	$\underline{42.1}$ ± 0.4	**54.4** ± 0.6	66.1 ± 1.1	**39.4** ± 0.3

	D → A	D → W	D → C	C → A	C → W	C → D
ASVM	48.0 ± 1.1	63.5 ± 1.1	29.9 ± 0.8	49.5 ± 1.0	63.2 ± 1.2	52.7 ± 1.3
PMT-SVM	48.6 ± 1.1	66.5 ± 1.2	30.9 ± 0.8	50.0 ± 1.0	64.3 ± 1.2	52.2 ± 1.3
GFK	45.8 ± 0.4	**80.3** ± 0.7	33.3 ± 0.5	46.4 ± 0.7	61.0 ± 1.4	52.7 ± 1.2
MMDT	$\underline{50.4}$ ± 0.7	74.2 ± 0.7	35.7 ± 0.7	$\underline{51.1}$ ± 0.7	62.9 ± 1.1	53.0 ± 1.0
A-SSVM	48.6 ± 0.5	$\underline{74.6}$ ± 0.6	35.5 ± 0.5	**53.4** ± 0.7	63.6 ± 1.2	52.7 ± 1.0
A-SSVM-ALL	49.0 ± 0.5	73.2 ± 0.7	$\underline{37.8}$ ± 0.4	50.6 ± 0.6	$\underline{66.2}$ ± 0.6	$\underline{57.5}$ ± 0.9
HA-SSVM	**52.6** ± 0.5	73.0 ± 0.5	**39.2** ± 0.6	**53.4** ± 0.8	**69.6** ± 0.7	**61.2** ± 0.9

Table 5.4: Multi-category recognition accuracy on target domains. Bold indicates the best result for each domain split. Underline indicates the second best result. The domains are: A: *amazon*, W: *webcam*, D: *dslr*, C: *Caltech256*. We use the nomenclature *Source→Target*.

ASVM	PMT-SVM	GFK	MMDT	A-SSVM	A-SSVM-ALL	HA-SSVM
48.9 ± 1.1	48.9 ± 1.1	51.0 ± 0.7	52.9 ± 0.9	52.9 ± 0.7	<u>54.4</u> ± 0.6	**56.7** ± 0.7

Table 5.5: Multi-category recognition accuracy averaged across all domain splits, with the corresponding standard deviation.

Adaptation Tree	A→W	A→D	A→C	Avg.
A→[W, D, C]	69.8 ± 0.7	59.7 ± 0.9	42.1 ± 0.4	48.4
A→[W, [D, C]]	69.8 ± 0.7	59.5 ± 1.0	40.1 ± 0.4	47.7
A→[D, [W, C]]	69.8 ± 0.6	59.1 ± 0.8	40.9 ± 0.4	47.8
A→[C, [D, W]]	72.7 ± 0.7	63.4 ± 1.2	42.1 ± 0.4	**49.5**
Adaptation Tree	W→A	W→D	W→C	Avg.
W→[A, D, C]	54.4 ± 0.6	66.1 ± 1.1	39.4 ± 0.3	47.3
W→[A, [D, C]]	54.3 ± 0.5	63.3 ± 1.3	38.7 ± 0.5	46.9
W→[D, [A, C]]	55.7 ± 0.6	65.8 ± 1.0	39.5 ± 0.4	**48.1**
W→[C, [A, D]]	54.2 ± 0.6	63.5 ± 1.0	39.5 ± 0.4	47.3
Adaptation Tree	D→A	D→W	D→C	Avg.
D→[A, W, C]	52.6 ± 0.5	73.0 ± 0.5	39.2 ± 0.6	48.7
D→[A, [W, C]]	52.6 ± 0.6	71.8 ± 0.7	39.4 ± 0.6	48.5
D→[W, [A, C]]	54.0 ± 0.6	73.0 ± 0.8	39.9 ± 0.6	**49.6**
D→[C, [A, W]]	53.0 ± 0.6	71.0 ± 0.7	38.9 ± 0.6	48.3
Adaptation Tree	C→A	C→W	C→D	Avg.
C→[A, W, D]	53.4 ± 0.8	69.6 ± 0.7	61.2 ± 0.9	57.3
C→[A, [W, D]]	53.2 ± 0.7	71.2 ± 0.7	63.0 ± 1.1	**57.8**
C→[W, [A, D]]	53.2 ± 0.6	69.5 ± 0.6	59.7 ± 1.3	57.1
C→[D, [A, W]]	52.4 ± 0.6	68.9 ± 1.0	61.0 ± 1.1	56.5

Table 5.6: HA-SSVM trained with various adaptation trees. The first column illustrates the tree structure. A two-layer adaptation tree is represented by X→[Y,Z,T], where X is the source domain and Y, Z and T are sibling target domains. These results are just a copy of the HA-SSVM ones shown in Table 5.4. A three-layer adaptation tree is represented by X→[Y,[Z,T]], where Z and T are siblings on the third layer and Y is located on the second layer.

	Amazon	DSLR	Webcam	Caltech256
Amazon	—	8.13	9.03	**9.78**
DSLR	8.13	—	**9.60**	8.25
WebCam	9.03	**9.60**	—	8.96
Caltech256	**9.78**	8.25	8.96	—

Table 5.7: Domain similarities in terms of QDS values [72]. Lager values indicate higher similarity.

Source Domain			A	W	D	C
Original Target Domains			W, D, C	A, D, C	A, W, C	A, W, D
Method	Hierarchy	Domain Discovery				
A-SSVM-ALL	—	—	47.6 ± 0.4	45.4 ± 0.4	46.5 ± 0.5	54.4 ± 0.6
HA-SSVM	2 Layers	Given	48.4 ± 0.5	47.3 ± 0.5	48.7 ± 0.6	57.3 ± 0.8
HA-SSVM	3 Layers	Given	**49.5** ± 0.4	**48.1** ± 0.6	**49.6** ± 0.5	57.8 ± 0.7
HA-SSVM	2 Layers	LatDD-Pr	46.2 ± 0.3	45.2 ± 0.4	45.9 ± 0.4	53.1 ± 0.6
HA-SSVM	3 Layers	LatDD-Pr	46.3 ± 0.4	45.1 ± 1.4	45.8 ± 0.4	53.0 ± 0.7
HA-SSVM	2 Layers	Reshape-Pr	49.0 ± 0.6	47.0 ± 0.5	48.0 ± 0.5	**59.4** ± 0.5
HA-SSVM	3 Layers	Reshape-Pr	49.1 ± 0.6	47.9 ± 0.5	48.2 ± 0.5	**59.1** ± 0.5

Table 5.8: The average multi-category recognition accuracy for a single target domain, for a priori 'Given' domains (Sect. 5.4), and for discovered latent domains (*LatDD-Pr* and *Reshape-Pr*). For the three-layer hierarchies we only show the best result among all possible three-layer adaptation trees.

Chapter 6

Hierarchical online domain adaptation of deformable part-based models

In this chapter, we extend HA-SSVM for online domain adaptation of the DPMs. The online domain adaptation is based on the hierarchical adaptation tree, which consists of instance detectors in the leaf nodes and a category detector at the top level. Each instance detector is an exemplar classifier which is trained online with only one pedestrian per frame. The pedestrian instances are collected by multiple object tracking (MOT) and the hierarchical model is constructed dynamically according to the trajectories. The proposed method neither requires revisiting source domain data, nor labelled target domain data. The adapted detector achieves comparable accuracy to the state-of-the-art supervised domain adaptation methods, improving the source detector more than 10 percentage points on the benchmark datasets.

6.1 Introduction

Classifiers play a core role in many computer vision tasks as well as in pedestrian detection systems. Collecting a training set is not a cost-free process since the required images must be acquired and the positive/negative samples labelled. In most of the cases, the labelling is a tiresome manual operation prone to errors. Moreover, in many real applications image acquisition involves the deployment of equipment and personnel for days or months, *i.e.*, the images are not *just there*. As we have pointed in the previous chapters, the underlying assumption that the training set and the deployment environment (testing) follow the same probability distribution can always be broken which will cause a significant drop in the accuracy of the learned classifiers.

If a sufficient amount of training data for the new testing *domain* is collected, we may consider retraining the classifiers. However, as we have pointed out, data collection can be costly and, therefore, in the general case doing so is not the optimal use of resources. Accordingly, reusing the existing classifiers by *adapting* them from the previous training environment (*source domain*) to the new testing one (*target domain*) is an approach worth to pursue and with increasing acceptance in the computer vision community [53, 54, 58, 91, 97, 99].

In Chapter 4 and Chapter 5, we focus on domain adaptation of DPMs and proposed methods to adapt a virtual-world trained detector to various real-world datasets with a few labelled target domain data. However, these methods have several drawbacks. First, they require labelling work in the target domain; Second, they need to wait until all the newly labelled training data is available, and then run the adaptation by considering all the training samples at a time. In this chapter, we focus on performing an *online domain adaptation* of DPM-based object detectors. We use the term *online* at video level, not at frame level. In particular, we receive an unlabelled video and we automatically adapt the current DPM to it without human intervention and without using the data that originated the current DPM. We can keep doing that as new videos arrive.

The main benefit is to have an algorithm ready to improve existing source-oriented detectors as soon as a new *unlabelled* target-domain training data is available, and keep improving as more of such data arrives in a continuous fashion.

Due to its challenging setting, online domain adaptation is a relatively unexplored scenario. In [112], we proposed an incremental DA framework which is inspired in online transfer learning (OTL) [97, 123]. Though the weak labels can be handled by following a multiple instance learning (MIL) paradigm for DPM training, it requires labelled target domain data and even at the semi-supervised setting, a human oracle is needed to click out false positives. Similar to our hierarchical model, a recent work of [52] proposed a category-to-instance detector for improving tracking. Target objects are identified with a pre-trained category detector and object identity across frames is established by individual-specific detectors. The individual detectors are re-trained online from a single positive example whenever there is a coincident category detection. The method is built on a boosting classifier framework while our method directly models the category-to-instance adaptation in a unified HA-SSVM framework. The final goal of [52] is online tracking, while ours is to obtain a generic target domain adapted detector.

The rest of the chapter is organized as follows. In Section 6.2, we illustrate the online hierarchical model. We first introduce the overview of the online domain adaptation framework. Then we go to the details of each step, including the MOT process and learning algorithm. In Section 6.3, we show experimental results of the proposed method on ETH datasets. Finally, Section 6.4 summarizes the chapter.

6.2 Hierarchical online domain adaptation (HOLDA)

We first introduce the overall framework of the hierarchical model. Then we introduce how to incorporate multiple object tracking into the pipeline for generating trajectories. Later, we formulate the learning as hierarchical adaptive SSVM and finally, we detail the overall algorithm.

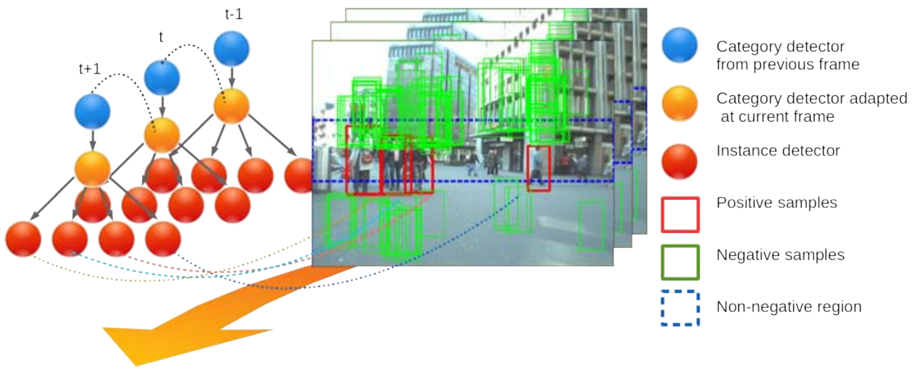

t-1
t
t+1
Category detector from previous frame
Category detector adapted at current frame
Instance detector
Positive samples
Negative samples
Non-negative region

Figure 6.1: The hierarchical online adaptation framework.

6.2.1 Model

The hierarchical online adaptation framework is shown in Figure 6.1. Given a target domain sequence, we first apply the source domain detector to collect the detected bounding boxes (the red bounding boxes). Then, MOT is used to generate trajectories and also remove some false positives. After generating the trajectories, our hierarchical model can be learned frame by frame using online adaptive SSVM. At each frame, the hierarchical model consists of instance detectors at the leaf nodes (the red balls in Figure 6.1) and a category detector at the second layer (the orange ball). The adaptation is executed in a progressive manner, *i.e.*, category detector (orange ball at time t), is adapted from the previously adapted category detector(blue ball at time t, which is the orange ball at time $t-1$). At $t = 0$, the category detector is initialized by the source domain detector. The instance detectors are adapted concurrently with the category detector of the current frame. The hierarchical model is constructed dynamically according to the trajectories at current frame, *i.e.*, each instance detector corresponds to one trajectory. Each instance detector is essentially an exemplar classifier which is trained using only one positive example (red bounding box) and many negative examples (green bounding boxes). The negative examples are collected from the same frame, but outside the *non-negative* area. The non-negative region (blue dash rectangle) is defined using prior geometry knowledge, which can avoid accidentally introducing false negatives as background examples.

6.2.2 Generating trajectories by MOT

We use MOT to provide trajectory information for the hierarchical online learning. The MOT requires bounding boxes from the source domain detector as input and outputs the optimized trajectories which may recover some missing detections and eliminate false detections from the source detections. Besides the selected detections, the trajectories are directly used to build the hierarchical online model, *i.e.*, each trajectory corresponds to a leaf node in the adaptation tree.

Algorithm 5 Motion-based Multiple Object Tracking

```
Input: $N$: length of the sequence
$D_s = \{d_s(t) | t \in [1, N]\}$: set of bounding boxes detected by the source domain, where $d_s(t)$ are such bounding boxes at frame $t$.
Output: $T^*(t)$:set of reliable tracks at frame $t$
0: Initialize the tracks $T^*(1)$ by $d_s(1)$.
1: for t=2, ..., N, do
2: Get the detections $d_s(t)$.
3: Predict new locations of active tracks in $T^*(t-1)$ by Kalman filter.
4: Data association: assign detections $d_s(t)$ to the active tracks in $T^*(t-1)$ using [71].
5: Update tracks:
    5.1: Correct the location estimate of Kalman filter
         for each continued track.
    5.2: Delete lost tracks.
    5.3: Create new tracks from unassigned detections.
6: Evaluate the reliability of the tracks and remove unreliable tracks in
   $T^*(t)$.
7: end for
```

In this work, we implemented a simple motion-based multiple object tracking algorithm based on Kalman filter. Though more sophisticated state-of-the-art MOT algorithms can be readily incorporated in our system, we found our simple MOT has already given promising results. Our MOT method can be divided into three parts: (1) detecting pedestrians at each frame (this is provided by the source detector), (2) associating the detections corresponding to the same pedestrian over time, (3) evaluating each trajectory to collect the reliable ones. The method is summarized in Alg. 5. The reliability of a trajectory is evaluated according to the length and the *confidence*. The length of a trajectory is is defined as the number of frames being active. The confidence of a trajectory is measured by averaging the detection scores of the associated bounding boxes. If the length or confidence is lower than a predefined threshold, the trajectory is defined as not reliable, otherwise as reliable.

6.2.3 Learning with HA-SSVM

We extend the HA-SSVM of Chapter 5 for online domain adaptation and we denote it by *HOLDA*.

Assume we have a source model $\mathbf{w}^S$ and target domain images I_t, $i \in [1, N]$. Without losing generality, assume at frame t, we have 3 pedestrians. The category model we have learned at frame $t-1$ is denoted by $\mathbf{w}_c^{t-1}$. The category model and instance models at frame t are denoted by $\mathbf{w}_c^t$, $\mathbf{w}_{i_j}^t, j \in [1, 3]$ respectively. $\mathbf{w}_{i_j}^t$ is the parameter of the instance classifier i and it is learned with pedestrian example j and all the negative examples in frame t. We denote the training examples for $\mathbf{w}_{i_j}^t$ by

$\mathcal{D}^{T_j}$. Then the objective function of HOLDA is written as follows:

$$\begin{aligned} J(\mathbf{w}) = \ & \frac{1}{2}\|\mathbf{w}_c^t - \mathbf{w}_c^{t-1}\|^2 + C\textstyle\sum_{j=1}^{3}\mathcal{L}(\mathbf{w}_c^t; \mathcal{D}^{T_j}) \\ & +\frac{1}{2}\|\mathbf{w}_{i_1}^t - \mathbf{w}_c^t\|^2 + C\mathcal{L}(\mathbf{w}_{i_1}^t; \mathcal{D}^{T_1}) \\ & +\frac{1}{2}\|\mathbf{w}_{i_2}^t - \mathbf{w}_c^t\|^2 + C\mathcal{L}(\mathbf{w}_{i_2}^t; \mathcal{D}^{T_2}) \\ & +\frac{1}{2}\|\mathbf{w}_{i_3}^t - \mathbf{w}_c^t\|^2 + C\mathcal{L}(\mathbf{w}_{i_3}^t; \mathcal{D}^{T_3}) \end{aligned} \tag{6.1}$$

where $t \in [0, N]$, $\mathbf{w}_c^0 = \mathbf{w}^S$, $C > 0$ is the trade-off parameter. For DPM, $\mathcal{L}(\mathbf{w}; \mathcal{D})$ is the training loss which is defined as:

$$\mathcal{L}(\mathbf{w}; \mathcal{D}) = \textstyle\sum_{i=1}^{N} \max_{\widehat{y},\widehat{\mathbf{h}}}[\mathbf{w}'\Phi(\mathbf{x}_i, \widehat{\mathbf{h}}) + L(y_i, \widehat{y}, \widehat{\mathbf{h}})] - \sum_{i=1}^{N} \max_{\mathbf{h}} \mathbf{w}'\Phi(\mathbf{x}_i, \mathbf{h}). \tag{6.2}$$

where $L(y_i, \widehat{y}, \widehat{\mathbf{h}})$ is the 0-1 loss, *i.e.*, $L(y_i, \widehat{y}, \widehat{\mathbf{h}}) = 0$ if $\widehat{y} = y_i$ and 1 otherwise. For the details of the objective function, we refer the reader to Eq. (4.3) in Chapter 4.

Eq. (6.1) follows a multi-task learning paradigm form, where the optimization of each $\mathbf{w}_{i_j}^t$ can be understood as an individual task. After training with N frames, we obtained an adapted classifier with parameter $\mathbf{w}_c^N$. At testing time, we can directly apply the linear decision function:

$$f(\mathbf{x}) = \max_{\mathbf{h}\in\mathcal{H}} \mathbf{w}_c^{N\prime}\Phi(\mathbf{x}, \mathbf{h}) \ . \tag{6.3}$$

Connecting with our MOT, the overall algorithm of hierarchical online domain adaptation is described in Alg. 6. We denote by HOLDA-MOT the proposed method. Given a source domain trained detector $\mathbf{w}^S$ and target domain sequence of N frames, the first step is to apply MOT proposed in Alg. 5 to obtain refined trajectories $T^*(t), t \in [1, N]$. With these trajectories, we also obtain refined detections $d_s^*(t)$ on each frame. $d_s^*(t)$ consists of the associated bounding boxes of the trajectories, which are supposed to be more confident detections than the original ones, *i.e.*, $d_s(t)$. At frame t, we can build the hierarchical model according to the current trajectory $T^*(t)$, *i.e.* the leaf nodes are corresponding to the individual pedestrians $d_s^*(t)$ and they will be used to train instance detectors $\mathbf{w}_i^t$. Then we extract negative samples from the background of frame t. A *non-negative* region is used to avoid selecting false negative examples. With the positive and negatives examples, we can optimize the hierarchical model at current frame and obtain an adapted category detector $\mathbf{w}_c^t$. In the next frame $t+1$, $\mathbf{w}_c^t$ will be used as *source* detector and adapted to $\mathbf{w}_c^{t+1}$. The intuition behind this is that at frame t the knowledge from the previous frames are encoded in $\mathbf{w}_c^t$; then, by adapting $\mathbf{w}_c^t$ to $\mathbf{w}_c^{t+1}$, $\mathbf{w}_c^{t+1}$ keeps the knowledge of the previous frames at the same time that learns from the examples at frame $t+1$. In this way, the final adapted target domain detector is $\mathbf{w}_c^N$.

Algorithm 6 HOLDA-MOT

```
Input:
Target domain sequence $I_t, t \in [1, N]$
Source domain detector $\mathbf{w}^S$
Output: adapted target domain detector $\mathbf{w}^T$
0: Apply $\mathbf{w}^S$ to $I_t, t \in [1, N]$ and obtain detections $D_s = \{d_s(t)|t \in [1, N]\}$.
1: $\mathbf{w}_c^0 = \mathbf{w}^S$
2: Apply MOT (Alg. 5) on $D_s$ to obtain trajectories $T^*(t), t \in [1, N]$.
3: for t=1,2, ..., N, do
4: Build the hierarchical model according to $T^*(t)$.
5: Get the positive examples (i.e. pedestrians) $d_s^*(t)$ from $T^*(t)$.
6: Extract background examples outside the non-negative region.
7: Optimize the objective function 6.1 and obtain $\mathbf{w}_c^t$.
8: end for
9: $\mathbf{w}^T = \mathbf{w}_c^N$
```

6.3 Experiments

6.3.1 Datasets

As source-domain, we use the same virtual-world dataset as in the previous chapters. For the target domain, we use the three sequences from ETH dataset [109], namely 'BAHNHOF', 'JELMOLI', 'SUNNY DAY', and denoted by ETH0, ETH1, ETH2 respectively. The experiments are evaluated on Caltech pedestrian detection benchmark as in the previous chapters.

6.3.2 Training error

In this section, we investigate the properties of the proposed method by assessing training error. In particular, we compare the proposed online hierarchical model (HOLDA-GT) with batch learned models (A-SSVM) as well as non-hierarchical models (OLDA-GT). As we assess the training error, we train with ground truth and test on the same sequence.

Table 6.1: Different training methods

Method	Description
SRC	Source domain classifier (no adaptation).
A-SSVM [111]	A batch learning baseline, adaptive SSVM, trained with ground truth.
OLDA-GT	Non-hierarchical online adaptation, trained with ground truth.
HOLDA-GT	Hierarchical online adaptation, trained with ground truth.

In Table 6.1, we list the methods (except SRC, which is the source detector) used to train domain adapted detectors. The training error is shown in Figure 6.2. We can see that SRC acts only as a good initialization for DPM, and the online methods HOLDA-GT/OLDA-GT generally do not reach the accuracy of the batch ones (A-SSVM) if we have already labelled all the data. One exception is ETH0, HOLDA-GT/OLDA-GT even outperforms A-SSVM. Comparing HOLDA-GT and OLDA-GT, we see that it is better to have a hierarchy.

From these experiments, we see that HOLDA gives useful results when ground truth is available. In the next experiment, we assess the challenging case when training without ground truth information.

6.3.3 Testing error

In this section, we evaluate the performance of the proposed HOLDA-MOT under the fully unsupervised setting, *i.e.*, no target annotations are provided. We first train and test on each sequence using all available images. Later, we split each sequence into train/test sets, and assess the generalization of the algorithm to unseen images.

In the first part, we do unsupervised training on each sequence and then test on the same sequence. We compare to several baselines in [112]. The compared algorithms are discribed in Table 6.2. Note that, INC-MIL and A-SSVM require labelled training data, and around 100 annotated training images are used. INT-MIL does not use ground truth but requires a human oracle to click out false positives during the training process. Our algorithm does not require any ground truth information, neither human oracle. The results are shown in Figure 6.3. The proposed method HOLDA-MOT improves the source detector more than 10 percentage points on each sequence and approaches the batch learning method A-SSVM, even outperforms A-SSVM on ETH1.

Table 6.2: Different training methods

Method	Description	Require labelled data?
SRC	Source domain classifier (no adaptation).	NO
A-SSVM [111]	A batch learning baseline, adaptive SSVM.	YES
INC-MIL [112]	The incremental adaptive DPM based on multiple instance learning.	YES
INT-MIL [112]	The incremental adaptive DPM based on multiple instance learning, with human in the loop.	Limited (Human in the loop)
HOLDA-MOT	The proposed hierarchical online domain adaptation, incorporating multiple object tracking.	NO

In the second part of this section, we split the sequence into training and testing sets. For ETH0, because it has more images, we train with 200, 400 and 600 consecutive images and test on the rest. For ETH1 and ETH2, we train with first 50, 100 and 200 consecutive images and test on the rest. The main goal of this experiment is to investigate the generalization of the adapted detector and to evaluate its accuracy on unseen images. Figure 6.4 shows the accuracy for each sequence. As we can see from the results, HOLDA-MOT does the adaptation to unseen images. The portion of unlabelled images does have an impact to the final adapted detector. Around one third of the unlabelled sequence is usually adequate to train the adapted detector.

6.4 Summary

In this chapter, we present an online domain adaptation framework based on the hierarchical adaptation model proposed in Chapter 5. The hierarchical model is built on each frame, where leaf nodes are corresponding to pedestrian instance detectors and the root node is corresponding to the pedestrian category detector. The optimization of the hierarchical model is done in the same way as HA-SSVM. The online domain adaptation achieves comparable accuracy to the batch learned models while does not require re-visiting source domain data neither labelled target domain training data. It improves considerably the source classifier too.

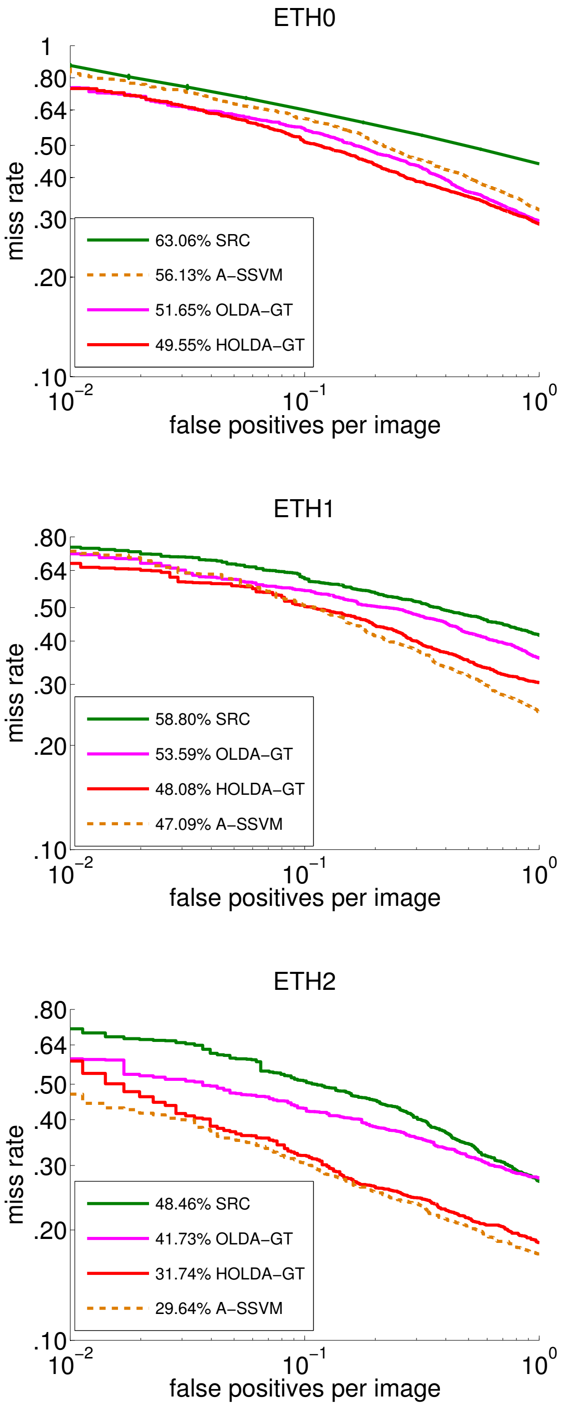


Figure 6.2: Training error of difference DA methods.

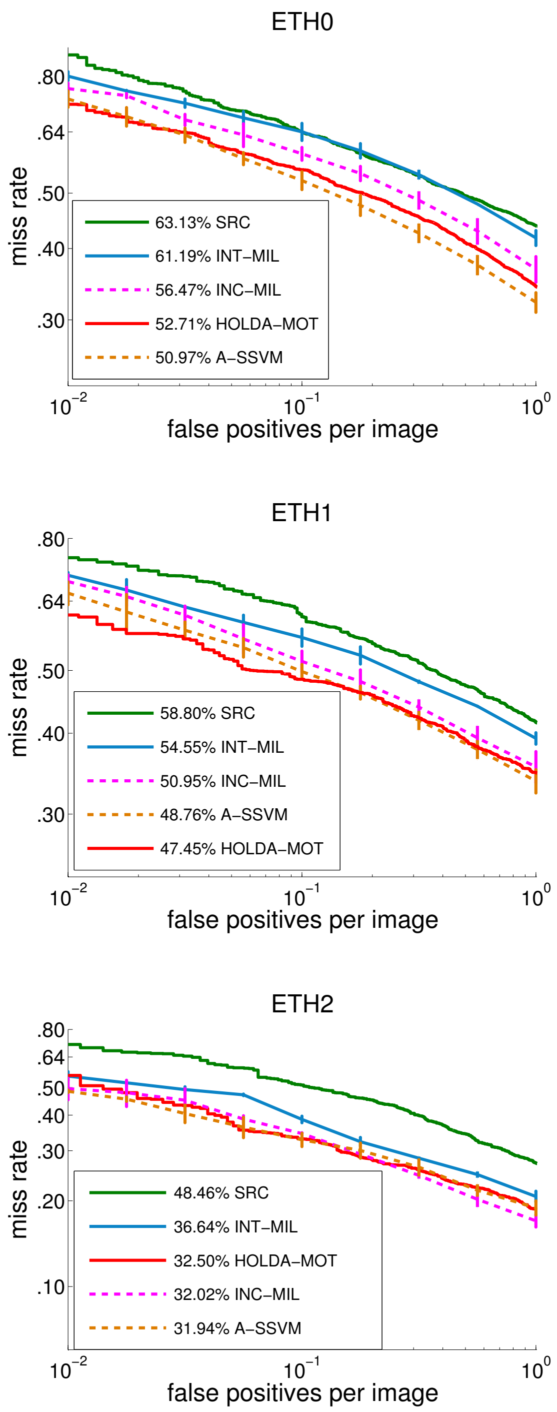
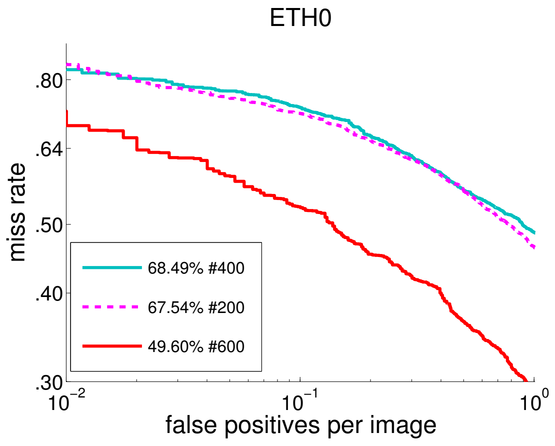
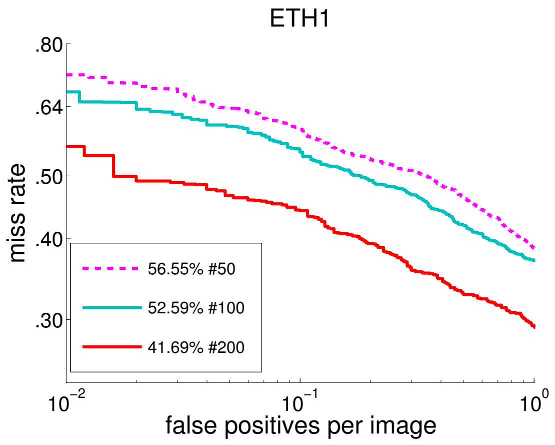


Figure 6.3: Testing error of different DA methods.





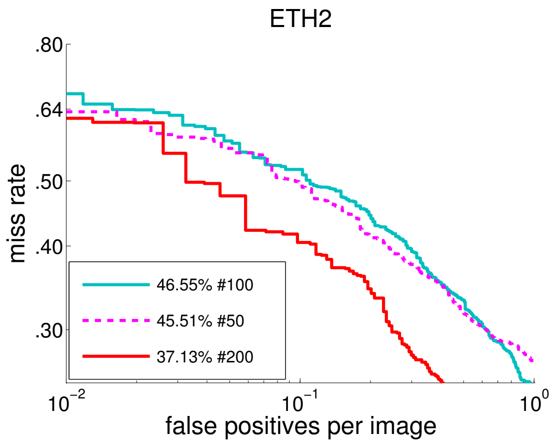


Figure 6.4: Testing error of HOLDA-MOT with various train/test splits. ETH0 has 999 images and 1998 pedestrians. ETH1 has 451 images and 902 pedestrians. ETH2 has 354 images and 708 pedestrians.

Chapter 7

Conclusion

In this chapter, we summarize what we present in this thesis, and also sketch a few future research directions in the area of domain adaptation for object detection, especially related to pedestrian detection.

7.1 Conclusions

In this thesis, we focus on the problem of domain adaptation of DPMs. In particular, we address the domain adaptation problem of adapting a virtual-world trained pedestrian DPM detector to operate in real-world datasets. We start by exploiting various DPM-based methods in the virtual world to maximize the accuracy of a virtual-world trained pedestrian detector. As the accuracy of the virtual-world trained detector drops significantly when operating in the real world, we then focus on domain adaptation methods for DPMs. Various DA methods have been proposed in the thesis, including single domain adaptation, multiple domain hierarchical adaptation and online domain adaptation. Below is the chapter-by-chapter summary of what we have presented.

Chapter 1 introduces the background and the goal of the thesis. We elaborated the concept of domain adaptation and the motivation of applying domain adaptation for pedestrian detection.

In Chapter 2, we reviewed related work in the literatures, including state-of-the-art work in pedestrian detection and domain adaptation.

In Chapter 3, we mainly focus on the source domain, *i.e.*, the virtual world dataset. We have proposed various DPM-based methods (VDPM-MP and VDPM-Star) to make better use of this dataset, aiming at maximizing the accuracy of the virtual-world trained detector. Although significant improvement has been obtained compared to the conventional DPM, the general accuracy on various real-world datasets is still low due to domain shift.

In Chapter 4, we start to explore methods for domain adaptation of DPM. As DPM is essentially a SVM-based model, which in particular, can also be formulated by structural SVM, we extend the classic adaptive SVM (A-SVM) [116] to SSVM and address the DA problem of DPMs. We proposed two methods, namely A-SSVM and SA-SSVM, for domain adaptation of DPMs. The former is a direct extension of A-SVM while the later takes into account the structure information in the part-based model and can perform part-level adaptation. The advantage of A-SSVM and SA-SSVM is that they do not require revisiting source domain data and a few annotated target domain examples are used to adapt the model. We further consider the challenging case of avoiding annotating target domain examples. For that purpose, we explored the use of self-paced learning (SPL) + Gaussian Processing Regression (GPR) method.

The work in Chapter 4 only considers single source to single target domain adaptation, while ignoring the relatedness of multiple target domains. Accordingly, in Chapter 5 we present a hierarchical model based on A-SSVM (HA-SSVM), which leverages multiple target (sub-)domains in an adaptation tree to perform a progressive adaptation. Its effectiveness has been verified on both DPM-based pedestrian detection, as well as multiple category object recognition. By dividing multiple resolution examples into sub-domains, we apply HA-SSVM on KITTI dataset and won the *Reconstruction Meets Recognition Challenge (RMRC)* held in conjunction with the ICCV'2013.

The proposed HA-SSVM in Chapter 5 can be readily extended to online models. In Chapter 6, we proposed a hierarchical online domain adaptation model to handle the challenging problem of online DA. We incorporate multiple object tracking (MOT) to provide optimized trajectories and build hierarchical models using these trajectories. On each frame, we extract pedestrians from the trajectory and train exemplar classifiers for each instance. These instance classifiers are leaf nodes in the adaptation tree while the top node is the category classifier. The target domain detector is adapted continuously when passing the target domain sequence.

7.2 Future perspective

7.2.1 Deep learned pedestrian detector and its domain adaptation

The last decade of progress on visual recognition has been based on the use of SIFT [37] and HOG [15]. As pointed in [42], the convergence in performance of the DPM person detector and the poselets person detector [12] might be an indication that we have squeezed all of the available performance out of HOG features. In 2012, Krizhevsky *et al.* [62] rekindled interest in convolutional neural networks (CNNs) by showing substantially higher image classification accuracy on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [16]. In 2014, Ross *et al.* [44] first show that a CNN can lead to dramatically higher object detection performance on PASCAL VOC [34] as compared to systems based on simpler HOG-like features. As we have reviewed in Chapter 2, using features extracted by deep learning instead of hand-crafted feature as HOG, has achieved top accuracy in pedestrian detection in recent years. Thus, using deep learned features for pedestrian detection has been seen as a promising trend for the immediate future.

One may ask whether deep learned features could eliminate domain shift, as such features are usually learned with substantial among of training examples, *e.g.*, the success of CNNs [44] are resulted from training a large CNN on 1.2 million labelled images. A recent study [25] has shown that even training deep features with millions of images, domain shift remains. Thus, exploring methods for domain adaptation of deep features is still an emergent topic. It would be interesting to extend the proposed methods in this thesis for domain adaptation of the deep learned pedestrian detector.

7.2.2 Self-adaptive detector for real-world autonomous driving

In Chapter 4, we employed SPL+GPR for self-adapting DPMs in batch mode. In Chapter 6, we further explored the challenging problem of online unsupervised domain adaptation. These could be regarded as a proof-of-concept of a self-adaptive detector. A self-adaptive detector, which can improve its detection accuracy automatically, is undoubtedly interesting and challenging topic for academy and industry. One promising application is autonomous driving, where vision plays a core role. The car would perceive millions of images per day during the driving. How to automatically improve its detection accuracy from millions of images remain an unsolved problem. As the car might be driven at different seasons, different cities, it is essential to be able to adapt to all different kinds of scenarios, which must rely on the machine learning algorithm to perform online domain adaptation.

As we said before, the proposed methods in this paper might be used as a simplified prototype of a self-adaptive detector, as we only consider 2-D RGB images and HOG features. For real-world pedestrian detection systems, we can further incorporate other sophisticated techniques, *e.g.*, using depth information from stereo cameras, combining other sensors *etc.*

7.2.3 Virtual world and domain adaptation for other vision applications

In this thesis, we use a virtual world as our source domain and proposed various DA methods for pedestrian detection. However, the idea of using virtual-world datasets and the proposed DA methods are not limited for pedestrian detection. For the virtual-world dataset, the main advantage is that it is annotation free and contains rich information, *e.g.*, view points, part annotations *etc.* These properties are attractive for many other applications which require expensive manual annotation work. For instance, scene segmentation usually requires pixel level annotations. Such annotations are expensive and prone to errors. Using a computer graphic engine to create a virtual world would generate pixel level annotations automatically and precisely. Similar to the situation in this thesis, the testing domain may have different probability distribution. Thus, domain adaptation will be required too.

Appendix A

Evaluation

In the following table, we list all of the proposed methods and their detection accuracy on the benchmark datasets. Values are percentage of average miss rates.

Table A.1: Overview of the proposed methods. I→INRIA, V→Virtual, T→Target dataset. Daimler* is the mandatory subset of Daimler full dataset.

Methods	Supervised	Train	Caltech	Daimler(full)	Daimler*	TUD	CVC	ETH0	ETH1	ETH2
DPM	YES	I	59.5	25.4	24.7	60.0	42.6	49.6	48.9	52.2
VDPM-Star	YES	V	63.4		25.1	70.7	48.8	63.1	58.8	48.5
VDPM-Star	YES	V+I			70.7		45.9	63.1	58.8	48.5
VDPM-MP	YES	V	63.3		24.3	65.9	47.5			
VDPM-MP	YES	V+I	53.0	16.8	18.2	61.3	36.3			
A-SSVM	YES	V+I	59.5				43.2	59.6	56.4	45.0
A-SSVM	YES	V+T	58.6				32.5	56.0	53.2	37.6
PMT-SSVM	YES	V+I	57.6				32.3	56.3	54.7	40.3
SA-SSVM	YES	V+I	57.3				41.0	57.0	54.9	43.2
SA-SSVM	YES	V+T	56.1				30.8	54.7	51.5	35.5
U-SA-SSVM	NO	V+T	62.2				36.2	58.1	55.2	50.0
U-SA-SSVM-GPR	NO	V+T	59.4				32.7	56.5	54.2	39.1
HA-SSVM-MRES	YES	V+T	49.9							
HA-SSVM	YES	V+T						53.4	50.4	34.0
INT-MIL	SEMI	V+T						61.2	64.6	36.6
INC-MIL	YES	V+T						56.5	51.0	32.0
HOLDA-MOT	NO	V+T						52.7	47.5	32.5

Appendix B

Notation

Abbreviation	**Meaning**
DA	Domain adaptation
DPM	Deformable part-based model
VDPM-Star	Virtual world trained DPM, star structure.
VDPM-MP	Virtual world trained DPM, mixture of parts.
A-SVM	Adaptive SVM
A-SSVM	Adaptive structured SVM
SA-SSVM	Structure aware adaptive SSVM
SPL	Self-paced learning
GPR	Gaussian Processing Regression
HA-SSVM	Hierarchical adaptive structured SVM
HOLDA	Hierarchical online domain adaptation

Symbol	**Meaning**
$\mathcal{D}$	Domain
$\mathcal{D}^S$	Source domain
$\mathcal{D}^T$	Target domain
$\mathcal{D}_l$	Labelled domain. Labelled source domain: $\mathcal{D}_l^S$
$\mathcal{D}_u$	Unlabelled domain. Unlabelled target domain: $\mathcal{D}_l^T$
$\mathbf{w}$	Model parameter vector
$\mathbf{w}^S$	Source domain model parameter vector
$\mathbf{w}^T$	Target domain model parameter vector
$\mathbf{F}$	HOG filter
$\mathbf{h}$	Object hypothesis

List of Publications

This dissertation has led to the following publications:

Journal Papers

- Jiaolong Xu, David Vázquez, Antonio M. López, Javier Marín, & Daniel Ponsa. Learning a Part-Based Pedestrian Detector in a Virtual World. *IEEE Trans. on Intelligent Transportation Systems*, 15(5), 2014.

- Jiaolong Xu, Sebastian Ramos, David Vázquez, Antonio M. López. Domain Adaptation of Deformable Part-Based Models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36(12), 2014.

Conference Contributions

- Jiaolong Xu, Sebastian Ramos, David Vázquez, Antonio M. López. Incremental Domain Adaptation of Deformable Part-based Models. In *British Machine Vision Conference*. Nottingham, UK, 2014.

- Jiaolong Xu, Sebastian Ramos, David Vázquez, Antonio M. López. Cost-sensitive structured SVM for multi-category domain adaptation. In *Int. Conf. in Pattern Recognition*. Stockholm, Sweden, 2014.

- Jiaolong Xu and Sebastian Ramos and Xu Hu and David Vázquez and Antonio M. López. Multi-task Bilinear Classifiers for Visual Domain Adaptation. In *Advances in Neural Information Processing Systems Workshop*. Harrah's Lake Tahoe, NV, USA, 2013.

- Jiaolong Xu, David Vázquez, Sebastian Ramos, Antonio M. López, & Daniel Ponsa. Adapting a Pedestrian Detector by Boosting LDA Exemplar Classifiers. In *IEEE Conf. on Computer Vision and Pattern Recognition. Ground Truth Workshop: What is a good dataset?*. Portland, Oregon, 2013.

- David Vázquez, Jiaolong Xu, Sebastian Ramos, Antonio M. López, & Daniel Ponsa. Weakly Supervised Automatic Annotation of Pedestrian Bounding Boxes. In *IEEE Conf. on Computer Vision and Pattern Recognition. Ground Truth Workshop: What is a good dataset?*. Portland, Oregon, 2013.

- Jiaolong Xu, David Vázquez, Antonio M. López, Javier Marín, & Daniel Ponsa. Learning a Multiview Part-based Model in Virtual World for Pedestrian Detection. In *IEEE Intelligent Vehicles Symposium*. Gold Coast, Australia, 2013.

Exhibitions

- Antonio M. López, Jiaolong Xu, David Vázquez, Sebastian Ramos, Germán Ros. Ciutadella, 2013, *Barcelona Party [Science + Technology]*

Bibliography

[1] A.Geiger, C.Wojek, and R.Urtasun. Joint 3D estimation of objects and scene layout. In *Advances in Neural Information Processing Systems*, Granada, Spain, 2011.

[2] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *Int. Conf. on Computer Vision*, 2011.

[3] Y. Aytar and A. Zisserman. Enhancing exemplar SVMs using part level transfer regularization. In *British Machine Vision Conference*, Surrey, UK, 2012.

[4] H. Azizpour and I. Laptev. Object detection using strongly-supervised deformable part models. In *European Conf. on Computer Vision*, Firenze, Italy, 2012.

[5] Jens Behley, Volker Steinhage, and Armin B. Cremers. Laser-based segment classification using a mixture of bag-of-words. In *IEEE Int. Conf. on Intelligent Robots and Systems*, 2013.

[6] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Pedestrian detection at 100 frames per second. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012.

[7] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool. Seeking the strongest rigid detector. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Portland, OR, USA, 2013.

[8] Rodrigo Benenson, Mohamed Omran, Jan Hosang, and Bernt Schiele. Ten years of pedestrian detection, what have we learned? In *Computer Vision for Road Scene Understanding and Autonomous Driving (CVRSUAD, ECCV workshop)*, 2014.

[9] A. Bergamo and L. Torresani. Exploring weakly-labeled web images to improve object classification: a domain adaptation approach. In *Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, 2010.

[10] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Int. Conf. on Empirical Methods in Natural Language Processing*, Sydney, Australia, 2006.

[11] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3D human pose annotations. In *Int. Conf. on Computer Vision*, Kyoto, Japan, 2009.

[12] Lubomir Bourdev, Subhransu Maji, Thomas Brox, and Jitendra Malik. Detecting people using mutually consistent poselet activations. In *European Conf. on Computer Vision*, Hersonissos, Heraklion, Crete, Greece, 2010.

[13] S. Branson, P. Perona, and S. Belongie. Strong supervision from weak annotation: Interactive training of deformable part models. In *Int. Conf. on Computer Vision*, Barcelona, Spain, 2011.

[14] X. Cao, Z. Wang, P. Yan, and X. Li. Transfer learning for pedestrian detection. *Neurocomputing*, 100(0):51–57, 2013.

[15] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, San Diego, CA, USA, 2005.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Miami, Florida, US, 2009.

[17] S.K. Divvala, A.A. Efros, and M. Hebert. How important are "deformable parts" in the deformable parts model? In *European Conf. on Computer Vision–Workshop on Parts and Attributes*, Florence, Italy, 2012.

[18] D.M. Gavrila. A bayesian exemplar-based approach to hierarchical shape matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(8):1408–1421, 2007.

[19] P. Dollár, R. Appel, and W. Kienzle. Crosstalk cascades for frame-rate pedestrian detection. In *European Conf. on Computer Vision*, Firenze, Italy, 2012.

[20] P. Dollár, S. Belongie, and P. Perona. The fastest pedestrian detector in the west. In *British Machine Vision Conference*, Aberystwyth, UK, 2010.

[21] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In *British Machine Vision Conference*, London, UK, 2009.

[22] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: an evaluation of the state of the art. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.

[23] Piotr Dollár, Ron Appel, Serge Belongie, and Pietro Perona. Fast feature pyramids for object detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36(8), 2014.

[24] J. Donahue, J. Hoffman, E. Rodner, K. Saenko, and T. Darrell. Semi-supervised domain adaptation with instance constraints. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Portland, Oregon, USA, 2013.

[25] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Int. Conf. on Machine Learning*, 2014.

[26] L. Duan, D. Xu, and IW. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *Int. Conf. on Machine Learning*, 2012.

[27] Lixin Duan, Ivor W. Tsang, Dong Xu, and Tat-Seng Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *Int. Conf. on Machine Learning*, Montreal, Quebec, Canada, 2009.

[28] C. Dubout and F. Fleuret. Exact acceleration of linear object detectors. In *European Conf. on Computer Vision*, Firenze, Italy, 2012.

[29] I. Endres, V. Srikumar, M.-W. Chang, and D. Hoiem. Learning shared body plans. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012.

[30] M. Enzweiler and D.M. Gavrila. Monocular pedestrian detection: survey and experiments. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2009.

[31] M. Enzweiler and D.M. Gavrila. A multi-level mixture-of-experts framework for pedestrian classification. *IEEE Trans. on Image Processing*, 20(10):2967–2979, 2011.

[32] M. Enzweiler and D.M. Gavrila. A mixed generative-discriminative framework for pedestrian classification. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Anchorage, AK, USA, 2008.

[33] A. Ess, B. Leibe, and L. Van Gool. Depth and appearance for mobile scene analysis. In *Int. Conf. on Computer Vision*, 2007.

[34] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *Int. Journal on Computer Vision*, 88(2):303–338, 2010.

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

[36] J.R. Finkel and D. Christopher. Hierarchical bayesian domain adaptation. In *NAACL*, Colorado, USA, 2009.

[37] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal on Computer Vision*, 60(2):91–110, 2004.

[38] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2012.

[39] D. Gerónimo and A.M. López. *Vision-based pedestrian protection systems for intelligent vehicles.* Springer, 2013.

[40] D. Gerónimo, A.M. López, A.D. Sappa, and T. Graf. Survey of pedestrian detection for advanced driver assistance systems. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(7):1239–1258, 2010.

[41] D. Gerónimo, A.D. Sappa, D. Ponsa, and A.M. López. 2D-3D based onboard pedestrian detection system. *Computer Vision and Image Understanding*, 114(5):583–595, 2010.

[42] R.B. Girshick. *From Rigid Templates to Grammars: Object Detection with Structured Models.* PhD thesis, The University of Chicago, Chicago, IL, USA, 2012.

[43] R.B. Girshick, P.F. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/.

[44] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Columbus, Ohio, USA, 2014.

[45] Daniel Goehring, Judy Hoffman, Erik Rodner, Kate Saenko, and Trevor Darrell. Interactive adaptation of real-time object detectors. In *IEEE Int. Conf. on Robotics and Automation*, 2014.

[46] B. Gong, K. Grauman, and F. Sha. Reshaping visual datasets for domain adaptation. In *Advances in Neural Information Processing Systems*, Lake Tahoe, NV, USA, 2013.

[47] B. Gong, K. Grauman, and F. Sha. Learning kernels for unsupervised domain adaptation with applications to visual object recognition. *Int. Journal on Computer Vision*, 109(1-2):3–27, 2014.

[48] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012.

[49] R. Gopalan, R. Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Int. Conf. on Computer Vision*, Barcelona, Spain, 2011.

[50] K. Goto, K. Kidono, Y. Kimura, and T. Naito. Pedestrian detection and direction estimation by cascade detector with multi-classifiers utilizing feature interaction descriptor. In *IEEE Intelligent Vehicles Symposium*, Baden-Baden, Germany, 2011.

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

[52] D. Hall and P. Perona. Online, real-time tracking using a category-to-individual detector. In *European Conf. on Computer Vision*, 2014.

[53] J. Hoffman, T. Darrell, and K. Saenko. Continuous manifold based adaptation for evolving visual domains. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2014.

[54] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *European Conf. on Computer Vision*, 2012.

[55] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, and K. Saenko. Asymmetric and category invariant feature transformations for domain adaptation. *Int. Journal on Computer Vision*, 109(1-2):28–41, 2014.

[56] J. Hoffman, E. Rodner, J. Donahue, K. Saenko, and T. Darrell. Efficient learning of domain invariant image representations. In *Int. Conf. on Learning Representations*, Arizona, USA, 2013.

[57] Judy Hoffman, Sergio Guadarrama, Eric Tzeng, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. LSDA: Large scale detection through adaptation. In *Advances in Neural Information Processing Systems*, Quebec, Canada, 2014.

[58] V. Jain and E. Learned-Miller. Online domain adaptation of a pre-trained cascade of classifiers. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2011.

[59] J. Jiang. A literature survey on domain adaptation of statistical classifiers. Technical report, School of Information Systems, Singapore Management University, 2008.

[60] W. Jiang, E. Zavesky, C. Shih-Fu, and A. Loui. Cross-domain learning methods for high-level visual concept classification. In *IEEE Int. Conf. on Image Processing*, San Diego, CA, USA, 2008.

[61] M. Kan, J. Wu, S. Shan, and X. Chen. Domain adaptation for face recognition: Targetize source domain bridged by common subspace. *Int. Journal on Computer Vision*, 109(1-2):94–109, 2014.

[62] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, USA, 2012.

[63] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2011.

[64] M.P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, 2010.

[65] Ping Luo, Yonglong Tian, Xiaogang Wang, and Xiaoou Tang. Switchable deep network for pedestrian detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Columbus, Ohio, USA, 2014.

[66] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2008.

[67] J. Marín, D. Vázquez, A.M. López, J. Amores, and B. Leibe. Random forests of local experts for pedestrian detection. In *Int. Conf. on Computer Vision*, Sydney, Australia, 2013.

[68] J. Marín, D. Vázquez, D. Gerónimo, and A.M. López. Learning appearance in virtual scenarios for pedestrian detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, 2010.

[69] F. Mirrashed, V.I. Morariu, S. Behjat, R.S. Feris, and L.S Davis. Domain adaptive object detection. In *WACV*, Washington, DC, USA, 2013.

[70] Mosek. Optimization toolkit. http://www.mosek.com.

[71] James Munkres. Algorithms for assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1), 1957.

[72] Jie Ni, Qiang Qiu, and Rama Chellappa. Subspace interpolation via dictionary learning for unsupervised domain adaptation. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Oregon, USA, 2013.

[73] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *Int. Conf. on Computer Vision*, Sydney Australia, 2013.

[74] Wanli Ouyang and Xiaogang Wang. A discriminative deep model for pedestrian detection with occlusion handling. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2012.

[75] Wanli Ouyang and Xiaogang Wang. Single-pedestrian detection aided by multi-pedestrian detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Oregon, USA, 2013.

[76] Wanli Ouyang, Xingyu Zeng, and Xiaogang Wang. Modeling mutual visibility relationship in pedestrian detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Portland, OR, USA, 2013.

[77] Sakrapee Paisitkriangkrai, Chunhua Shen, and Anton Van Den Hengel. Efficient pedestrian detection by directly optimizing the partial area under the roc curve. In *Int. Conf. on Computer Vision*, Sydney Australia, 2013.

[78] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. In *International Joint Conference on Artificial Intelligence*, Pasadena, California, USA, 2009.

[79] J. Pang, Q. Huang, S. Yan, S. Jiang, and L. Qin. Transferring boosted detectors towards viewpoint and scene adaptiveness. *IEEE Trans. on Image Processing*, 20(5):1388–400, 2011.

[80] D. Park, D. Ramanan, and C. Fowlkes. Multiresolution models for object detection. In *European Conf. on Computer Vision*, Crete, Greece, 2010.

[81] Dennis Park, C Lawrence Zitnick, Deva Ramanan, and Piotr Dollár. Exploring weak stabilization for motion feature extraction. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Portland, Oregon, USA, 2013.

[82] H. S. Park and C. H. Jun. A simple and fast algorithm for K-medoids clustering. *Expert systems with applications*, 36(2):3336—-3341, 2009.

[83] I. Parra, D. Fernández, M.A. Sotelo, L.M. Bergasa, P. Revenga, J. Nuevo, M. Ocaña, and M.A. García. Combination of feature extraction method for SVM pedestrian detection. *IEEE Trans. on Intelligent Transportation Systems*, 8(2):292–307, 2007.

[84] P. Patrick and M. Everingham. Shared parts for deformable part-based models. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011.

[85] L. Pishchulin, A. Jain, M. Andriluka, T. Thormahlen, and B. Schiele. Articulated people detection and pose estimation: reshaping the future. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012.

[86] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012.

[87] H. Daumé III. Frustratingly easy domain adaptation. In *Meeting of the Association for Computational Linguistics*, Prague, Czech Republic, 2007.

[88] H. Daumé III. Bayesian multitask learning with latent hierarchies. In *UAI*, Montreal, QC, Canada, 2009.

[89] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N.D. Lawrence, editors. *Dataset shift in machine learning.* Neural Information Processing. The MIT Press, 2008.

[90] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, 2006.

[91] K. Saenko, B. Hulis, M. Fritz, and T. Darrel. Adapting visual category models to new domains. In *European Conf. on Computer Vision*, 2010.

[92] M. Schmidt. Minconf - projection methods for optimization with simple constraints in matlab. http://www.di.ens.fr/ mschmidt/Software/minConf.html.

[93] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. Pedestrian detection with unsupervised multi-stage feature learning. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Portland, OR, USA, 2013.

[94] A. Shashua, Y. Gdalyahu, and G. Hayun. Pedestrian detection for driving assistance systems: Single–frame classification and system level performance. In *IEEE Intelligent Vehicles Symposium*, Parma, Italy, 2004.

[95] M. Sun and S. Savarese. Articulated part-based model for joint object detection and pose estimation. In *Int. Conf. on Computer Vision*, Barcelona, Spain, 2011.

[96] Kevin Tang, Vignesh Ramanathan, Fei-Fei Li, and Daphne Koller. Shifting weights: Adapting object detectors from image to video. In *Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, USA, 2012.

[97] T. Tommasi, F. Orabona, M. Kaboli, B. Caputo, and C. Martigny. Leveraging over prior knowledge for online learning of visual categories. In *British Machine Vision Conference*, 2012.

[98] A. Torralba and A.A. Efros. Unbiased look at dataset bias. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011.

[99] D. Vázquez, A.M. López, J. Marín, D. Ponsa, and D. Gerónimo. Virtual and real world adaptation for pedestrian detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36(4):797–809, 2014.

[100] D. Vázquez, A.M. López, and D. Ponsa. Unsupervised domain adaptation of virtual and real worlds for pedestrian detection. In *Int. Conf. in Pattern Recognition*, Tsukuba, Japan, 2012.

[101] D. Vázquez, A.M. López, D. Ponsa, and D. Gerónimo. Interactive training of human detectors. *Multimodal Interaction in Image and Video Applications*, 48:169–182, 2013.

[102] A. Vedaldi and A. Zisserman. Structured output regression for detection with partial truncation. In *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2009.

[103] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Kauai, HI, USA, 2001.

[104] P. Viola and M. Jones. Robust real-time face detection. *Int. Journal on Computer Vision*, 57(2):137–154, 2004.

[105] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *Int. Journal on Computer Vision*, 63(2):153–161, 2005.

[106] M. Wang and X. Wang. Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011.

[107] M. Wang and X. Wang. Transferring a generic pedestrian detector towards specific scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012.

[108] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

[109] C. Wojek, S. Walk, and B. Schiele. Multi-cue onboard pedestrian detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Miami Beach, FL, USA, 2009.

[110] Christian Wojek and Bernt Schiele. A performance evaluation of single and multi-feature people detection. In *DAGM symposium on Pattern Recognition*, 2008.

[111] J. Xu, S. Ramos, D. Vázquez, and A.M. López. Domain adaptation of deformable part-based models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36(12):2367–2380, 2014.

[112] J. Xu, S. Ramos, D. Vázquez, A.M. López, and D. Ponsa. Incremental domain adaptation of deformable part-based models. In *British Machine Vision Conference*, Nottingham, UK, 2014.

[113] J. Xu, D. Vazquez, A.M. Lopez, J. Marin, and D. Ponsa. Learning a part-based pedestrian detector in a virtual world. *IEEE Trans. on Intelligent Transportation Systems*, 15(5):2121–2131, 2014.

[114] J. Xu, D. Vázquez, S. Ramos, A.M. López, and D. Ponsa. Adapting a pedestrian detector by boosting LDA exemplar classifiers. In *IEEE Conf. on Computer Vision and Pattern Recognition – Workshop on Ground Truth*, Portland, OR, USA, 2013.

[115] J. Yan, X. Zhang, Z. Lei, S. Liao, and S.Z. Li. Robust multi-resolution pedestrian detection in traffic scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Oregon, USA, 2013.

[116] J. Yang, R. Yan, and A.G. Hauptmann. Cross-domain video concept detection using adaptive SVMs. In *ACM Multimedia*, Augsburg, Germany, 2007.

[117] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011.

[118] C.-N. J. Yu and T. Joachims. Learning structural SVMs with latent variables. In *Int. Conf. on Machine Learning*, Montreal, Quebec, 2009.

[119] Chun Nam Yu. *Improved Learning of Strucutral Support Vector Machines: Training with Latent Variables and Non-linear Kernels.* PhD thesis, Cornell University, Ithaca, NY, USA, 2011.

[120] X. Zeng, W. Ouyang, and X. Wang. Multi-stage contextual deep learning for pedestrian detection. In *Int. Conf. on Computer Vision*, Sydney Australia, 2013.

[121] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *European Conf. on Computer Vision*, 2014.

[122] Shanshan Zhang, Christian Bauckhage, and Armin B Cremers. Informed haar-like features improve pedestrian detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Columbus, Ohio, USA, 2014.

[123] P. Zhao and S. Hoi. OTL: A framework of online transfer learning. In *Int. Conf. on Machine Learning*, 2010.

[124] L. Zhu, Y. Chen, A. Yuille, and W. Freeman. Latent hierarchical structural learning for object detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, 2010.

[125] X. Zhu, C. Vondrick, D. Ramanan, and C. C. Fowlkes. Do we need more training data or better models for object detection? In *British Machine Vision Conference*, Surrey, UK, 2012.