Domain Adaptation for Image Recognition and Viewpoint Estimation

Dissertation

zur

Erlangung des Doktorgrades (*Dr. rer. nat.*)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Pau Panareda Busto

aus

Barcelona, Spanien

Bonn, 2020

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter:	Prof. Dr. Jürgen Gall
2. Gutachter:	Prof. Dr. Zeynep Akata

Tag der Promotion:	02.09.2020
Erscheinungsjahr:	2020

Abstract

by Pau Panareda Busto

for the degree of
Doctor rerum naturalium

Image-based recognition tasks require in their training phase large amounts of data to capture as much visual traits as possible. In many situations, however, the collection of image data implies a tedious effort or, even worse, the test scenarios remain unknown. On top of that, the labelling process is very time consuming, expensive and prone to error. This means that the access to fast, cheap and accurate labelled data arises as ones of the main challenges in classification problems. In this work, we present three major contributions that pursue the attenuation of these issues in image recognition and viewpoint estimation problems. Overall, the main goal is reducing the amount of data collection and labelling effort.

In order to achieve that, we firstly introduce a novel domain adaptation method that allows datasets from different domains to take part in the training process and contribute to improved classification accuracies. We also revise the unrealistic setting of domain adaptation evaluation datasets and introduce open set domain adaptation for target domains that also contain irrelevant samples that belong to unknown classes.

Then, we also propose an optimisation process for fine viewpoint labelling and use synthetic data to refine viewpoints that are coarsely annotated by humans in real images. To this end, due to the differences between the real and the synthetic data, we apply domain adaptation to align both domains and improve the viewpoint refinement. The results have shown that 3D generated models can be successfully used to refine labels in real images.

We finally present an end-to-end multi-task neural network that jointly trains viewpoints and keypoints of rigid objects. We also reinforce the real training data with a novel synthetic dataset that contains annotations for both problems. The experiments show that the proposed approach successfully exploits this implicit correlation between the tasks and outperforms previous techniques that are trained independently.

To the strongest person I have ever met,
my mum.

Acknowledgements

This dissertation was unimaginable when I stepped into Germany for the first time 10 years ago. I only had to stay for one year to finish my diploma thesis at the FAU Erlangen and somehow I ended up staying longer than expected. Interestingly, I have transitioned from specialising in computer graphics and finding English-speaking countries quite attractive to focusing on computer vision and soon obtaining the German citizenship. This evolution can only be explained by the enormous influence and support received from marvellous people, primarily these last years during my industrial doctorate studies at the Airbus Group and supervised by Jürgen Gall at the University of Bonn. Back in 2013, Bastian Leibe from the RWTH Aachen University recommended me Jürgen Gall as advisor to pursue my doctorate studies. I cannot be more thankful for his advice. Jürgen is the reason why I am able to finish my PhD, I have produced high valuable research and I continued moving forward in the most difficult moments. Long-lasting phone calls and complex E-mails between Munich and Bonn were not a barrier to teach me, help me and suggest new ideas and approaches. These lines are not enough to express my gratitude for the insane amount of hours he has spend on me.

During my first year in Germany, I was extremely lucky to be part of the graphics department led by Marc Stamminger at the FAU Erlangen. Christian Eisenacher was an outstanding supervisor and showed me the first insights of research. At the same time, Sven Bauer gave me the opportunity to be at Siemens Healthcare for one year as working student. His kindness was priceless for a young guy who was not used to a new country like Germany. Even after 10 years of my stay in Erlangen, my flatmate Eduardo, the spiritual Elisa and the group Fondo Norte (Minerva, Sally, Laura, Dani S, Dani C and Sergi) are still making my life a bit easier.

After completing my computer engineering degree, I pursued a master's degree in Media Informatics at the RWTH Aachen, where I received the unconditional support from Leif Kobbelt's graphics and vision department. Specially, Bastian Leibe and his vision group were like a second family to me. Friendly students and a professor like Bastian who is extremely kind to everyone made my days in Aachen enjoyable. Bastian's suggestions were always on point and contained an incalculable value. In addition, my life outside university was always entertaining with Laura and Benedikt, who were (and are, in Munich) a great support to me.

I would also like to thank Airbus Group Innovations, the research centre of Airbus, to give me the opportunity to pursue an industrial PhD in a beautiful and charming city like Munich. Many things happened in 4 years and I am very thankful to Falk Schubert and the PhD students I met there. The security personnel from the Airbus campus also deserves a special mention here, since they allowed me to enter into the campus to continue my experiments when special permits are necessary on weekends. Life in Munich is also unforgettable with my bavarian girl Tanja, who has supported me from the first moment, the Churritos (Paula and Dominic), my PhD buddy Nils, the gentleman Patricio and *da boss in da Bronx* Dushan.

The second half of my doctorate studies is located in Bonn at the aforementioned vision group of Jürgen Gall. Jürgen has been well surrounded by his outstanding students. Dimitrios and Abhilash during the first years and Umar, Alexander and Martin towards the end, made my stays in Bonn very pleasant. We stay in contact thanks to conferences and I hope it remains like this in the future.

Last but not least, I want to thank my mum, Mari Carmen Busto Barcos, for teaching me values and habits that are left aside in the current days. She always expresses that a PhD is what changed her life and nothing makes her happier than seeing a son with a PhD. A proud mum might be one of the main reasons why I pursued a PhD. Furthermore, my sister Eva and my nieces Diana and Lidia are also an inspiration to me.

My last words go to the people that know me best, who have always been next to me since many years and are there to give me a hand when needed: Juan Antonio, Aitor and Ben.

Gràcies per tot el que heu fet per mi!

Pau Panareda Busto

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

The following abbreviations have been used in this thesis in alphabetical order:

ATI	Assign-and-Transform-Iteratively
BoW	Bag-of-Words
CNN	Convolutional Neural Network
CRF	Conditional Rando6m Field
CS	Closed Set (Domain Adaptation)
DA	Domain Adaptation
FC	Fully Connected
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
LBP	Local Binary Patterns
MMD	Maximum Mean Discrepancy
NBNN	Naive-Bayes Nearest Neighbour
NN	Nearest Neighbour
OS	Open Set (Domain Adaptation)
ReLu	Rectified Linear Unit
SIFT	Scale Invariant Feature Transform
SVM	Support-Vector Machine
w/o	Without

Mathematical Symbols

The mathematical symbols used in this thesis are listed below in approximate order of first appearance and grouped per chapter:

Chapter 2

x	Extracted feature of an object sample
D	Number of dimensions
y	Object class label
w	Weights of a classifier
W	Set of weights of a classifier
b	Bias term
M	Number of samples of class +1
N	Number of samples of class -1
ξ	Slack variable
C	Input parameter for tuning soft-margins in SVMs
$\phi(x)$	Mapping function to a higher dimensional space

$k(x_i, x_j)$	Kernel function
K	Number of object classes
σ	activation function in neural networks
$\mathcal{L}$	Loss function
$(f * g)$	Convolution of funtions f and g
t	Amount of shift on a convolution
I	2D Image
$\mathcal{L}_{cross}$	Cross-entropy loss function
$\mathcal{L}_{reg}$	Euclidean loss function
p	Pixel

Chapter 4

$\mathcal{T}$	Target samples
$\mathcal{C}$	Set of object classes
c	Object class
D	Number of dimensions
d_{ct}	Euclidean distance between a source cluster and a target sample
S_c	Feature representation of source cluster c
T_t	Feature representation of target sample t
λ	Cut-off distance to reject samples in the assignment of source clusters
x_{ct}	Binary variable in the assignment step stating if c and t are connected
o_{ct}	Binary variable in the assignment step stating if target t is rejected
$\mathcal{L}$	Set of labelled target samples (semi-supervised setting)
$\hat{c}_t$	Source cluster automatically assigned to the labelled target sample t
N_t	Number of neighbours of target sample t
L	All assignments computed between source clusters and target samples
W	Linear transformation computed from source-target assignments
P_S	Matrix with features of source clusters sorted by assignment position
P_T	Matrix with features of target samples sorted by assignment position
ρ	Outlier parameter for samples in the assignment of source clusters

Chapter 5

θ	Azimuth angle
ϕ	Elevation angle
r	Object distance from the camera perspective in a render
$\mathcal{S}$	Source data
$\mathcal{T}$	Target data
D	Number of feature dimensions
S	Set of source samples
s	Source sample
T	Set of target samples
t	Target sample

M	Number of source samples
N	Number of target samples
K	Number of target clusters
C	Set of correspondences between source and target data
c_k	correspondence of a target cluster to source cluster k
W	Linear transformation computed using source-target correspondences
P_S	Matrix with features of source clusters sorted by correspondence
P_T	Matrix with features of target clusters sorted by correspondence
i	Coarse viewpoint, namely *front*, *left*, *rear* and *right*
V_i	Number of fine viewpoints of source data per coarse region i
V	Total number of fine viewpoints of source data
N_i	Target samples per coarse viewpoint
K_i	Number of target clusters per coarse viewpoint
$\hat{S}^i$	Centroids of the source data
$\hat{T}^i$	Centroids of the target data
e_{vk}	Binary value stating if a correspondence between (v) and (k) exist
a_v	Number of associations of a source cluster v to different target clusters
w	Affine weights of a linear SVM
b	Bias term of a linear SVM
x	Input feature descriptor of a target test sample
H	Huber loss
$F(\theta)$	Continuous representation azimuth angle

Chapter 6

C	Set of object classes
c	Object class
K_c	Number of keypoints for class c
k	Keypoint
$\mathcal{M}$	Set of training samples with viewpoint and keypoint annotations
$\mathcal{N}$	Set of training samples with viewpoint annotations only
$\mathcal{O}$	Set of training samples with keypoint annotations only
s	Stage for pose estimation on a multi-stage CNN architecture
x_i	Training sample
$y_{i,k}$	Ground-truth 2D heatmap for sample i and keypoint k
f_s	Predicted heatmap at the given stage s
$\mathcal{L}_{kp_s}$	Euclidean loss for keypoint estimation
$\mathcal{L}_{vp_b}$	Cross-entropy loss for viewpoint estimation at a specific bin size b
$\mathcal{L}$	Total loss combining viewpoint and keypoint losses
$\Delta(R_{gt}, R_{pred})$	Geodesic distance between R_{gt} and R_{pred} rotations
R_{gt}	Ground-truth rotation matrix
R_{pred}	Predicted rotation matrix

Publications

The work presented in this thesis has been evaluated and approved by the computer vision community through the peer-reviewed papers listed below. The conference papers had to undergo a double-blind review process, while the journals articles had to undergo a single-blind review process of high standards. The code developed for some of the publications is public and available in the provided web links.

Conference Papers:

- P. Panareda Busto, J. Liebelt and J. Gall.
  Adaptation of Synthetic Data for Coarse-to-Fine Viewpoint Refinement
  In British Machine Vision Conference (BMVC), 14.1-14.12, 2015.
  `pages.iai.uni-bonn.de/gall_juergen/download/jgall_adaptview_bmvc15.pdf`
  [Panareda Busto et al., 2015]
- P. Panareda Busto and J. Gall.
  Open Set Domain Adaptation
  In International Conference on Computer Vision (ICCV), 754-763, 2017.
  `pages.iai.uni-bonn.de/gall_juergen/download/jgall_opensetdomain_iccv17.pdf`
  Code: `github.com/Heliot7/open-set-da`
  [Panareda Busto and Gall, 2017]
- P. Panareda Busto and J.Gall.
  Joint Viewpoint and Keypoint Estimation with Real and Synthetic Data
  In German Conference on Pattern Recognition (GCPR), 107-121, 2019.
  `pages.iai.uni-bonn.de/gall_juergen/download/jgall_viewkeypoint_gcpr19.pdf`
  Code: `github.com/Heliot7/viewpoint-cnn-syn`
  [Panareda Busto and Gall, 2019]

Journal Articles:

- P. Panareda Busto and J. Gall.
  Viewpoint refinement and estimation with adapted synthetic data
  In Computer Vision and Image Understanding (CVIU) 169:75-89, 2018
  `pages.iai.uni-bonn.de/gall_juergen/download/jgall_viewtransfer_cviu18.pdf`
  [Panareda Busto and Gall, 2018]
- P. Panareda Busto, A. Iqbal and J. Gall.
  Open Set Domain Adaptation for Image and Action Recognition
  In Transactions on Pattern Analysis and Machine Intelligence (PAMI), 2018
  `pages.iai.uni-bonn.de/gall_juergen/download/jgall_opensetdomain_pami18.pdf`
  [Panareda Busto et al., 2018]

Chapter 1
Introduction

"The Bundesliga is different from La Liga, it's different from Serie A, it's different from the Premier League, so you have to adapt to the circumstances."

—MICHAEL LAUDRUP (1964–)
ex-football player & coach

Contents

1.1 Motivation

We define computer vision as the understanding of the real world through the analysis of an image or a video sequence with one or more cameras. In an abstract way, it tries to replicate human vision capabilities which, at some point, will enhance our own ones. These comprise from the visual perception of tangible elements to the interpretation of events, just in the way a cognitive process would do. Therefore, a vision system usually provides a response in the form of visual enhancements and/or decisions from prior scene estimations.

Computer vision applications have become an efficient solution to daily tasks that involve a wide range of interactions between humans and their surroundings. Indeed, the recognition of well-defined objects using cameras is one of the most critical aspects of these systems. Some examples include the early detection of errors in the automatised quality control of manufacturing factories that produce large quantities of the same product, the suggestions of similar items to indecisive customers in online shopping sites and the correct detection of speed limit signs in highways for sticking to the maximum speeds allowed by law for road vehicles. For these three scenarios, behaviours like overlooking faulty products, the inclusion of non-related sale items on the customer's screen and the configuration of wrong numbers in the speed regulation function, respectively, derive in the immediate degradation of the offered feature, which therefore becomes unusable. The source of all these problems: a poor image-based object recognition.

More specifically, computer vision applications are often focused on identifying the differences of a set of discrete categories. Therefore, they classify an object to a known class by assigning a label to a given image or a selected region of it. This class identification has been coined in the field of computer vision as *object recognition* or *classification*. This recognition process might be applied to not only identifiable objects, e.g. *car*, *motorbike*, and *truck*, but also to any discretisable information seen in the image, e.g. set of viewpoints of a specific object, global image information that defines weather conditions or geographic traits and unique instances like human faces.

In general, image-based object recognition makes use of supervised machine learning techniques in order to *learn*, in the training phase, the best possible *classifier* that decides, in the test phase, what label is assigned to every new incoming image. As illustrated in Figure 1.1, this learning process requires as input images and manually annotated labels that specify what class is associated to what image region. The training of the classifier usually happens before the test phase starts, i.e. offline, and optimises the best possible separation among all labelled classes in a multi-dimensional space of *feature* descriptors computed for every image. Then, test images are sent to the classifier, which outputs the predicted result in the form of a class label or a list of class probabilities.

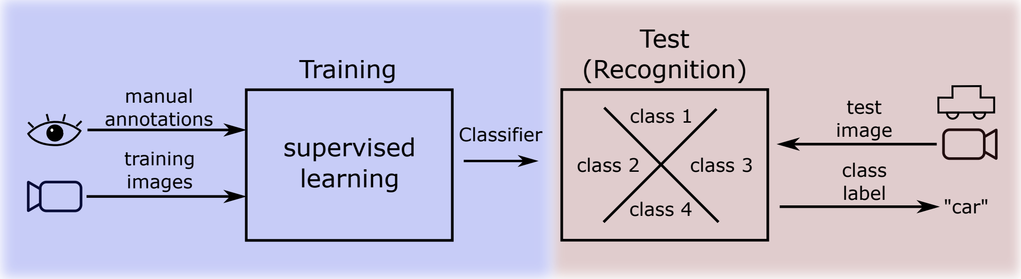


Figure 1.1: Pipeline of a standard object recognition task, including an offline training phase (blue) and its posterior test phase (red), where the trained classifier assigns a label to the incoming test images.

In order to guarantee the best possible recognition results and consequently a classifier that accurately recognises the list of trained objects, the images used in the training phase must derive from the same scenario where the test phase happens. Besides, the amount of training images must suffice and provide enough examples of the object classes involved in this task. Both requirements are, however, very challenging in many situations and might not be fully satisfied.

1.2 Problem Description

The collection of high quality training data becomes a key component in the success of object recognitions tasks. Under these circumstances, we encounter two important challenges when preparing the training data.

Firstly, the lack of training samples of the scenarios we want to deploy the recognition system. Due to the specificity or rareness of some situations, it might be very hard to find enough data of the classes of interest, regardless of the amount of stored images. Similarly, we can gather a considerable amount of images, but unfortunately only find object samples with the same shape and viewpoint, inducing not enough variety to describe the class in the training set. In extreme cases, it is possible to have no training data for a given class, either for not finding any sample in the recorded scenes or because the test scenario is still unknown.

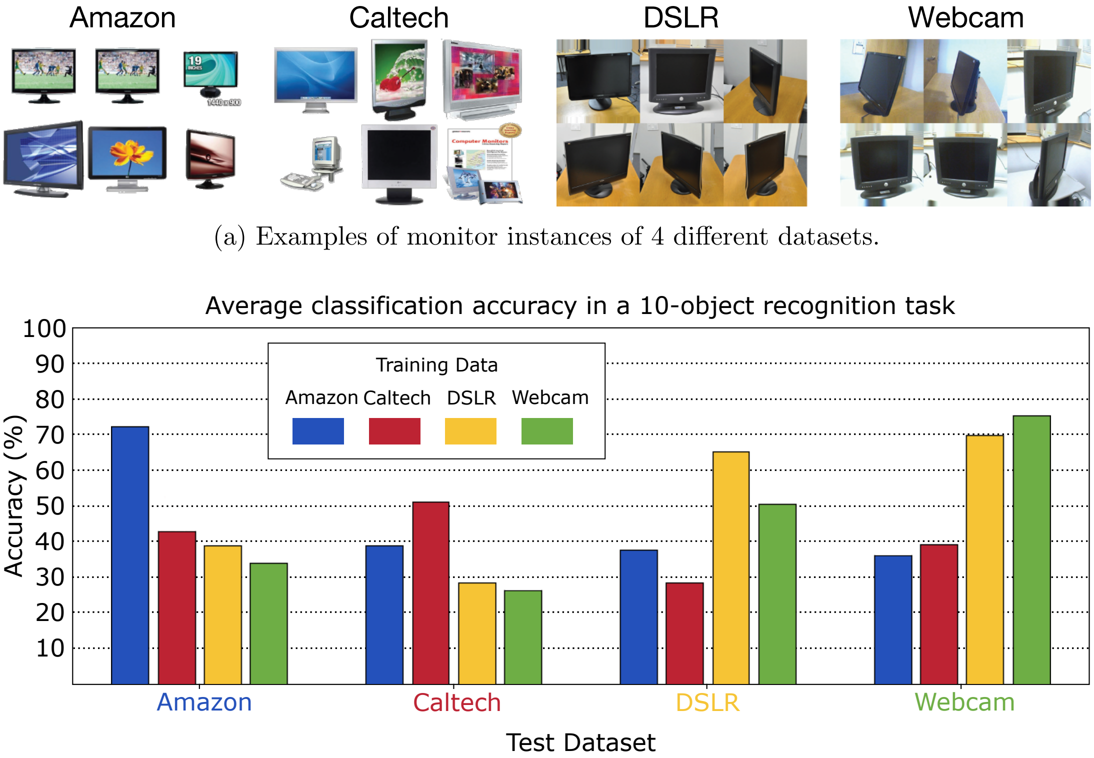
Secondly, the biological constraints of humans when annotating images also have an impact in the trained classifier. Humans are very efficient at high-level labelling, where visual differences among classes are clear and well defined. However, they lack precision at fine tasks that go beyond human levels of perception. On top of that, the labelling effort is very taxing and quality decreases over time, i.e. manual annotations are prone to error, at the same time that the labelling process slows down.

A straightforward solution to the presented issues is including additional training data that has been previously collected for other or similar purposes [Park et al., 2016, Dobrescu et al., 2017]. The drawback of this first attempt is not the availability of usable datasets, but the visual differences between this additional training data and the expected test images. That is, although we can re-use or even re-annotate existing images for our application, the differences between training and test images will likely hinder the quality of our trained system. For instance, the multi-dataset image collection by Saenko et al. [2010] exemplifies the negative impact of learning classifiers with datasets that do not belong to the same domain as the test images. This data collection includes 4 datasets with the same 10 classes. Examples of one of these, *monitor*, for each dataset are shown in Figure 1.2a. If we evaluate the performance of the trained classifiers against the test data of all datasets, as depicted in Figure 1.2b, we observe that the accuracies, defined as the percentage of correct classified categories, of the 10-class classifiers that belong to the same dataset obtain the best results by a significant margin. We use support-vector machines [Cortes and Vapnik, 1995] (see Section 2.1.2 for more details) as decision classifiers. For the *DSLR* and *Webcam* datasets, which were recorded in the same scenarios but with different camera sensors, this behaviour still holds but to a lesser extent. Therefore, less visual differences among datasets, also known as *domain shift*, implies better accuracies.

In general, common dissimilarities among datasets arise due to the following settings and characteristics:

- Camera specification (hardware and software): colour filter array, exposure times, lens distortion, level of sharpness, undesired noise, ...

- Extrinsic factors: day vs. night light conditions, sunny vs. foggy weather condi-



(a) Examples of monitor instances of 4 different datasets.

(b) Classification accuracies of 4 datasets with 10 object classes.

Figure 1.2: Evaluation of classification accuracies on the multi-dataset collection introduced by Saenko et al. [2010]. We present all possible combinations of trained classifiers evaluated on test samples for a total of 4 datasets: *Amazon*, *Caltech*, *DSLR* and *Webcam*. We make use of the supervised learning model called support-vector machines [Cortes and Vapnik, 1995] as decision classifier. The best results are reported by the classifiers learned and tested on the same dataset. The number of training and test images for each dataset is equated and do not affect the final results.

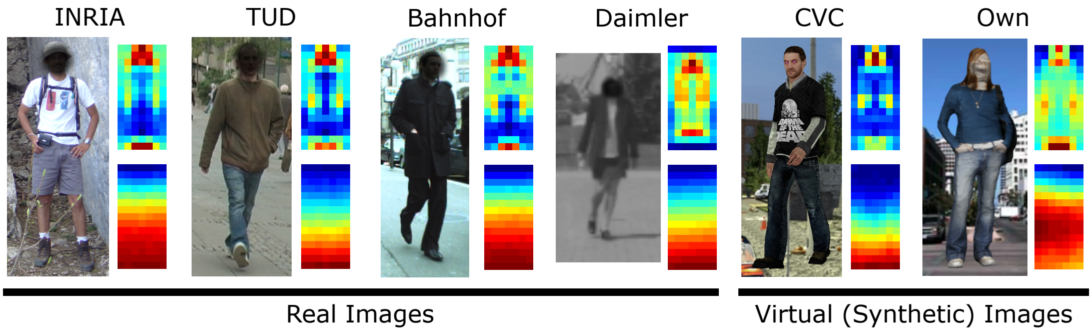


Figure 1.3: Examples of human datasets with their summed intensities of humans samples (top) and non-human related samples (bottom), i.e. background, from histograms of oriented gradients [Dalal and Triggs, 2005]. From left to right, these datasets were presented by Dalal and Triggs [2005], Wojek et al. [2009], Ess et al. [2008], Enzweiler and Gavrila [2009] and Marín et al. [2010].

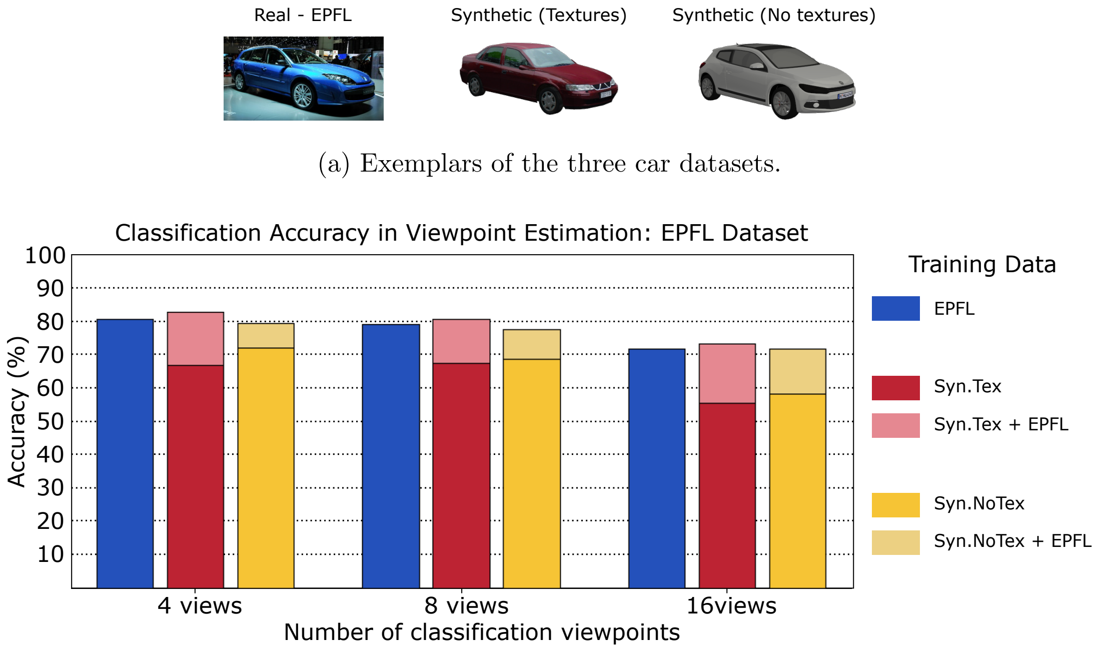
tions, urban vs. countryside background scenes, ...

- Intrinsic configurations: viewpoint, pose of non-rigid objects, subclasses, textures, attached accessories, ...

Figure 1.3 includes real images of humans that despite of differing in all three categories are still grouped in the same object class *pedestrian*.

Another widely used approach is the creation of synthetic data, either by generating new images from rendered 3D graphics models for each object class or by altering the already existing training images, e.g. calculating linear and perspective transformations.

From the former, only a selection of refined 3D models and a graphics engine are required to generate thousands of samples in just a few minutes [Peng et al., 2015, Su et al., 2015]. Annotations are also fully automatised, providing pixel-precision bounding boxes that represent the image region containing the object of interest. Other labels such as viewpoints and depth can also be extracted from the render without human intervention. However, the preparation of a considerable number of fine meshes with textures might become very challenging. Besides, these virtual images barely contribute to the overall performance due to their poor photorealistic results. A visual comparison between real and computer generated images of pedestrians is shown in Figure 1.3. We observe that the differences of the features computed from histograms of oriented gradients [Dalal and Triggs, 2005] (see Section 2.2.1 for more details) are more noticeable on the synthetic datasets. Renders with high levels of realism are unfortunately a very costly and time consuming effort and still differ to a large degree from real datasets when comparing their computed feature descriptors. The impact of including computer generated data with two different levels of realism is reported in Figure 1.4. We observe that textured cars allow for a slight classification improvement compared to the non-textured cars that even decrease the accuracy when estimating the viewpoint of vehicles at different fine levels.



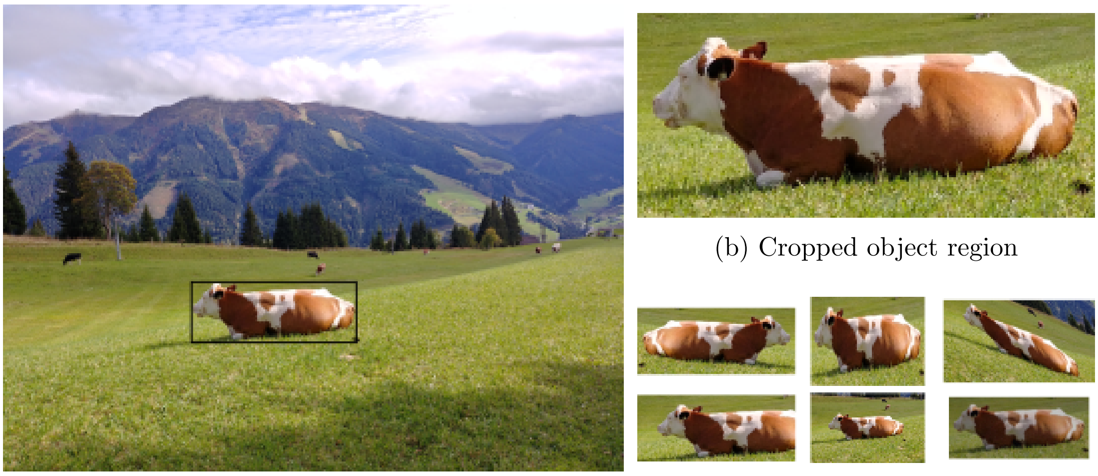
(a) Exemplars of the three car datasets.

(b) Classification accuracies on the EPFL dataset for 4, 8 and 16 viewpoints.

Figure 1.4: Classification accuracies on the real dataset EPFL presented by Ozuysal et al. [2009] for different levels of car viewpoint granularities: 4, 8 and 16 views. A textured and a non-textured synthetic datasets are also employed for training the support-vector machine classifiers evaluated on the test data. The addition of well curated synthetic images (with textures in this example) to the real training data leads to better results.

From the latter approach, we can process the images by computing linear transformations, adding noise and altering the light conditions, among others, in order to modify the appearance of the object sample, as presented in Figure 1.5. Nonetheless, the quantity of additional valuable information is highly constrained by the provided data and thus classification improvements are marginal. Hence, data augmentation is mostly applied to avoid *overfitting*, i.e. prevent a poor generalisation of the object classification based on the limited number, given the case, of training images.

These aforementioned strategies that introduce training data that differ from the test domain derived in a new field of study that intends to alleviate the dissimilarities between training and test datasets in order to produce better classification results [Wu and Dietterich, 2004, Yang et al., 2007, Mansour et al., 2009]. This methodology is called *domain adaptation* and addresses the problem of leveraging training images recorded in other scenarios, denoted as *source* data, to the problem relevant training and test images (if available), denoted as *target* data. Therefore, the source and target datasets belong to at least partially different domains, i.e. there exist a domain shift. While these adaptation techniques have a lot of potential, current domain adaptation algorithms are highly hand-crafted for specific scenarios, e.g. semantic seg-



(b) Cropped object region

(a) Original Image

(c) Data augmented samples

Figure 1.5: Data augmentation of an image portion (a) for a single object sample (b). The image transformations in (c) are the following (from top to bottom and from left to right): mirroring, bounding box offset, Y-axis scaling, bounding box scaling, perspective transformation and contrast alteration.

mentation [Zhang et al., 2017b], or techniques, e.g. tuning the level of adaptation of each layer in deep neural networks [Cariucci et al., 2017].

More recently, multi-task learning appears as another interesting solution to improve the classification performance when the amount of training data is limited [Socher et al., 2012, Doersch and Zisserman, 2017, Kendall et al., 2018]. We define multi-task learning strategies as the joint training process of correlated problems that benefit from each other in order to improve the overall results. More concretely, other computer vision problems such as *optical flow* and *depth estimation* with different labels but with a common interest in the object understanding, will likely transfer relevant information to our recognition task. Another side-effect of this approach is the saved effort of not re-annotating datasets for our specific needs or not having the need to collect new images, but simply adding new labels to our dataset.

1.3 Contributions

Based on the presented data collection issues and solutions, we propose several improvements in order to alleviate the process of curating training data, while at the same time we improve the object recognition accuracies.

Firstly, this dissertation contributes in the field of object recognition by introducing a novel domain adaptation algorithm that is very flexible to a wide range of problems and simultaneously obtains significant improvements in classification accuracies. This technique alleviates the issues presented in the utilisation of training data from other domains and allows them to become an active part of the learning process. In addition, we focus on understanding the insights of viewpoint estimation for rigid objects, e.g.

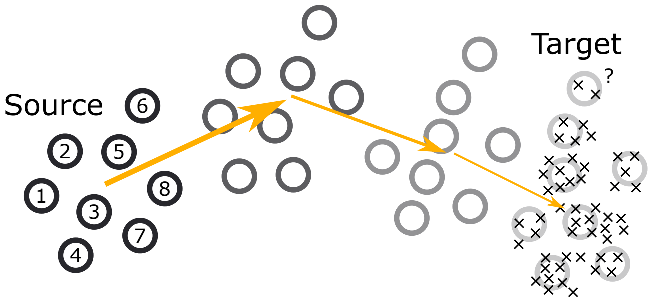


Figure 1.6: Domain adaptation from the source to the target domain in an iterative process with progressive linear transformations of the source towards the target dataset. There are 8 classes represented in the image. The source domain contains 8 clusters (circles) and the target domain contains unlabelled samples (crosses).

vehicles, appliances and daily tools, and present state-of-the-art results on this topic. Particularly, we pay attention to the unseen applicability of domain adaptation in the estimation of viewpoints between real and synthetic, i.e. computer generated, images. We thus bridge the gap between the accurate fine pose annotations of unrealistic synthetic images and the coarse but real images of the test scenario we evaluate our method. Finally, we also introduce an efficient end-to-end pipeline that combines in its learning framework viewpoints and keypoints and validates the usage of multi-task learning approaches. We also obtain state-of-the-art results on both topics.

1.3.1 Domain Adaptation

The main contribution presented in this thesis applies to the field of domain adaptation for object recognition tasks. We present a novel domain adaptation algorithm that transforms (and thus aligns) the source data towards the target data, which is part of a different but similar visual domain, as shown in Figure 1.6. This process is done iteratively and associates in every iteration the labelled training samples, clustered per class label, and the unlabelled test images. We demonstrate that this technique combines several positive aspects that have not been presented in the literature before. In first place, this algorithm accommodates to scenarios where not only all test images belong to the same known classes of the training data, but also to more challenging scenarios where some classes are still unknown (see contribution in Section 1.3.2 for more details). Then, we also demonstrate that the proposed domain adaptation also works for action recognition, where we evaluate video sequences and not single frames, and viewpoint estimation, where we adapt poses as classes at sub-category level. Thirdly, this iterative process is resilient to data compression, reporting the same accuracies with up to 20% of the feature descriptor dimensions. And last but not least, the overall results of our method outperform well-established domain adaptation algorithms in all the test evaluations, including several deep learning approaches. This algorithm belongs to our work presented in Panareda Busto and Gall [2017], which is also adapted in Panareda Busto et al. [2015] for viewpoint estimation problems.

1.3.2 Open Set Recognition

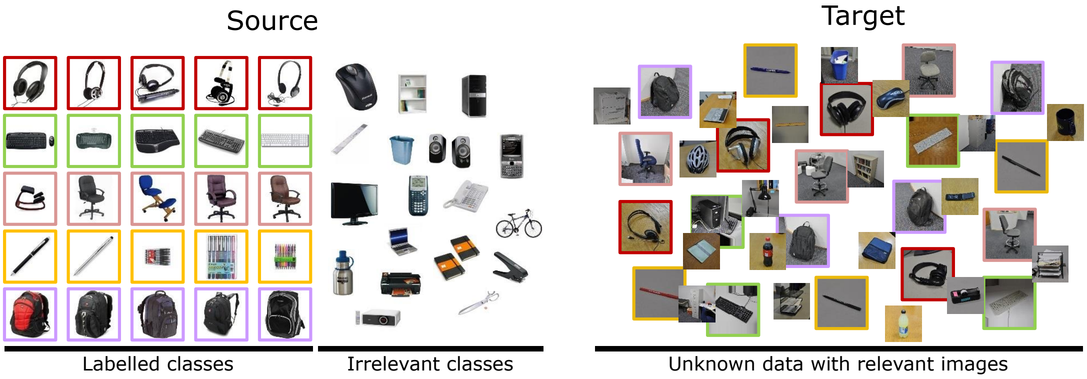


Figure 1.7: Outline of an open set domain adaptation problem. The target data contains not only relevant images for our application, but also samples that belong to unknown and uninteresting classes. Optionally, this might also occur to unsorted source data.

All domain adaptation publications prior to our work in Panareda Busto and Gall [2017] and Panareda Busto et al. [2018] were evaluated on a unique setting, where the same classes are shared between the source and target domains. However, in many real applications, the target images contain irrelevant images that do not belong to any class of interest. The same applies to the training images of other domains, where the re-used datasets contain additional classes that are not part of the recognition system under development. Therefore, we introduce *open set* domain adaptation, including additional training and test samples of unknown classes that are not relevant for the primary recognition task, as shown in Figure 1.7. This new scenario broadens the field of domain adaptation and encourages future works to take more realistic and challenging configurations into account. We also report that our adaptation technique, mentioned as a contribution of this thesis in Section 1.3.1, directly acclimates to this new scenario. Therefore, all these uninteresting and/or unknown images are processed to reduce an undesired impact on the relevant classes of interest in the learning phase of our application.

1.3.3 Viewpoint Estimation

In addition, we also present two contributions in the field of viewpoint estimation for rigid objects.

In the first approach, we take advantage of the full 360° span of viewpoint annotations that are automatically created when rendering synthetic images. Based on the assumption that humans fail at labelling fine-grained poses, we request them to simply annotate coarse viewpoints. These labels are then refined using a classifier trained with domain adapted synthetic images, i.e. aligned to the feature space of real images with our proposed domain adaptation technique. As illustrated in Figure 1.8, this adaptation process associates target samples to clusters of synthetic images with fine

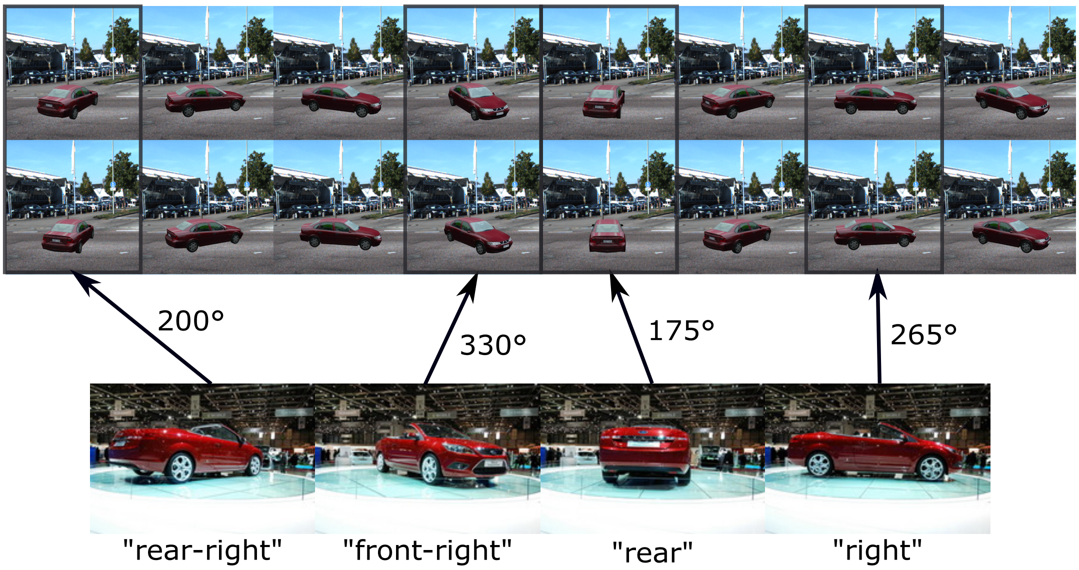


Figure 1.8: Assignment of real images with coarse annotated viewpoints to clusters of fine-grained viewpoints of synthetic data. These are then used on a standard optimisation problem to compute a transformation matrix that aligns the source towards the target domain.

viewpoints and transform them based on these associations. The viewpoint classifiers learned with the resulting refined training datasets can easily be embedded after any traditional object detector, taking the detected bounding boxes as input. This part of the thesis was presented in the following publications: Panareda Busto et al. [2015] and Panareda Busto and Gall [2018].

The second contribution shows how the joint training of viewpoints and keypoints using an end-to-end deep neural network architecture allows for better estimations on both problems. Specifically, embedding the viewpoint classifiers into a backbone for keypoint estimation produces increased accuracies and reduced median errors in the viewpoint angle. This is especially noticeable when no synthetic data supports the neural network learning process. This second contribution in viewpoint estimation was recently published in Panareda Busto and Gall [2019], obtaining state-of-the-art results on popular evaluation datasets.

All articles mentioned in this Section also demonstrate through a rigorous evaluation that dealing with the estimation of viewpoints as a standard object recognition problem usually obtains better results than regression formulations. In this configuration, fine viewpoints are treated as independent classes.

1.3.4 Keypoint Estimation

The annotation of keypoints requires a tedious effort. Therefore, we propose two solutions in order to improve the keypoint estimation while not requiring any additional labelling.

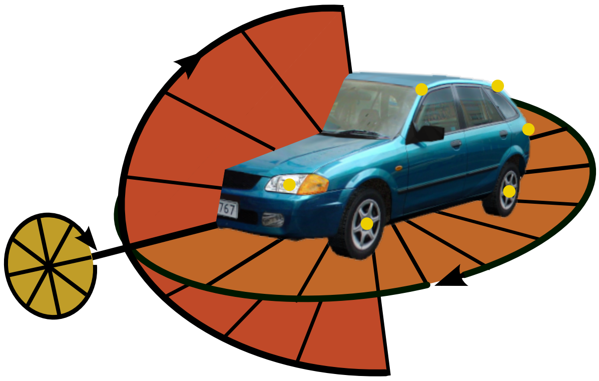


Figure 1.9: Visualisation of the 3 viewpoint angless, i.e. azimuth (orange), elevation (red) and tilt (yellow), and the keypoints of a car. The displacement of the non-occluded keypoints will remain the same for all car instances that are placed with the same pose.

As already discussed in the previous Section 1.3.3, including viewpoint annotations in the learning process suffices to get more precise keypoints. This proofs that highly-correlated tasks can benefit from each other when trained together on the same supervised model. A sketch of both tasks is shown in Figure 1.9, including the 3 viewpoint angles, azimuth, elevation and tilt, and all keypoints of a vehicle.

Furthermore, we present a novel synthetic dataset with automatically generated keypoints for a total of 12 rigid classes. Given a 3D rendering of a rigid object, the only necessary manual intervention is placing spheres at every 3D location of interest. All remaining steps, including the occlusion handling and the projection of the 3D keypoint from its world position to the 2D image coordinate, are fully automatised. The learning process using this dataset outperforms those with only the real images. This second contribution in viewpoint estimation was recently published in Panareda Busto and Gall [2019].

1.4 Dissertation Outline

The dissertation is structured in seven chapters, being chapters 4, 5 and 6 the core of this work. Before this main block, the two chapters that follow this introduction are focused on the fundamentals needed to better understand the major contributions of this thesis.

The code of this thesis is publicly available at: `https://www.github.com/Heliot7`. We refer to the summary of Publications on page xix for the code repositories of each specific publication.

- In **Chapter 2** we present the fundamentals needed to better understand the algorithms and strategies introduced in this thesis.
- In **Chapter 3** we summarise the literature that is highly related to the topics presented in this thesis.

- In **Chapter 4** we present open set domain adaptation and a domain adaptation technique that deals with irrelevant data on the target domain. This method works not only for open sets, but also for standard domain adaptation protocols.
- In **Chapter 5** we present the applicability of domain adaptation to the task of viewpoint estimation and propose a pipeline to avoid the labelling errors made by humans on fine annotations.
- In **Chapter 6** we present a joint training of viewpoints and keypoints with real and synthetic data by designing an end-to-end deep neural network.
- In **Chapter 7** we expose the conclusions of this thesis. Extensions of the presented methodologies are also extensively discussed.

Chapter 2

Technical Background

"I find it terrible when talents are rejected based on computer stats. Based on the criteria at Ajax now I would have been rejected. When I was 15, I couldn't kick a ball 15 meters with my left and maybe 20 with my right. My qualities technique and vision, are not detectable by a computer."

—JOHAN CRUYFF (1947–2016)
ex-football player & coach

Contents

2.1 Visual Object Classification

As we already discussed in Chapter 1, object classification tasks generally require a supervised classifier that decides whether an incoming test image belongs to one of several known classes, learned from a set of training images that are accordingly labelled. These classifiers are divided into two distinctive approaches: *generative* and *discriminative*. Formally, the two categories can be described as follows: given an input example in feature space $x \in \mathbb{R}^D$, with D dimensions, and an object class y, a generative classifier learns a statistical model of the joint probability $p(x, y)$ and classifies test images based on posterior probabilities $p(y|x)$. These are obtained by using Bayes' rule, i.e. with prior probabilities $p(y)$ and class-conditional densities $p(x|y)$. In contrast, a discriminative classifier directly models the posterior probability $p(y|x)$ or

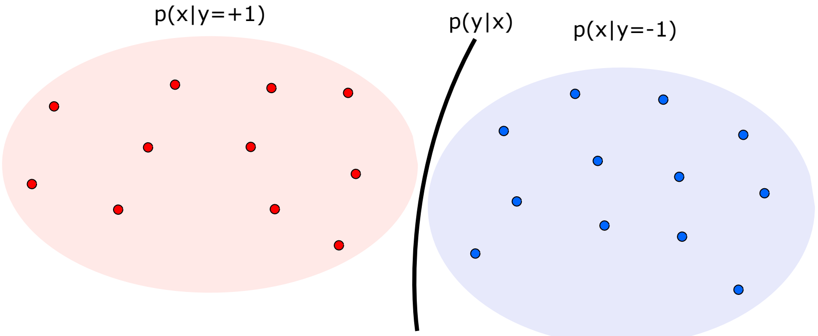


Figure 2.1: 2D sketch of two distributions for classes $y = \{-1, +1\}$ and their discriminative decision boundaries based on the conditional probability of being class y given the observation x.

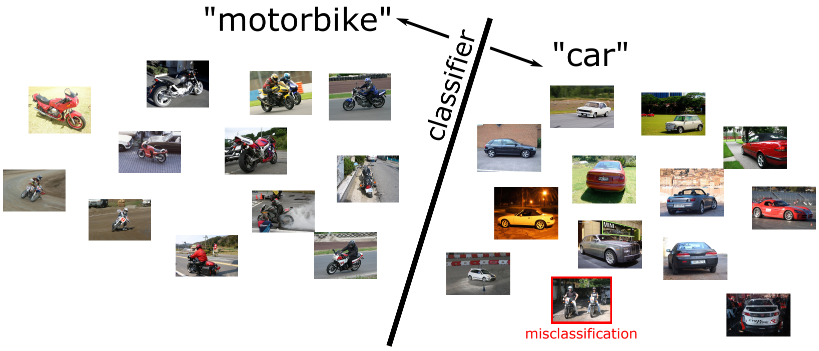
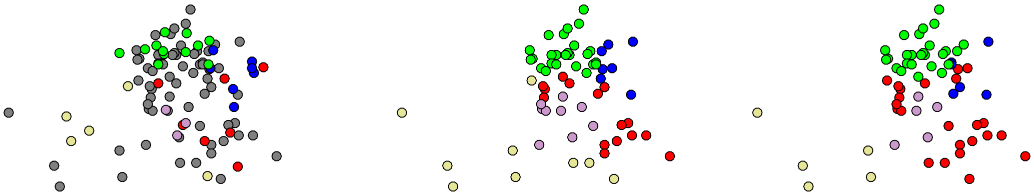


Figure 2.2: Test images containing either cars or motorbikes are correctly classified, excepting an image with motorbikes that are wrongly classified as *car*.

learns a map from the input sample to the class label $y = f(x)$, based on the training data. For instance, if we want to assign a label y from two classes $\{-1, +1\}$, a discriminative learning approach reduces to the binary question if a given image contains one object or the other. Figure 2.1 depicts the differences between both models in such a binary problem. A more specific example of a discriminative model between the classes *motorbike* and *car* is shown in Figure 2.2 with a linear separation between test images of both classes and a misclassified sample of a motorbike that lies on the wrong side of the decision boundary.

Empirically, not only discriminative approaches tend to result in better recognition accuracies, but are also usually easier to formulate, since generative models hardly build robust models with only a few parameters [Ng and Jordan, 2002]. For this reason, the great majority of current methodologies have opted for using discriminative classifiers for object recognition tasks. This thesis also emphasises on discriminative solutions, including the 3 classifiers described in the following sections.



(a) Initialisation of NN (b) 1-NN Classifications (c) 3-NN Classifications

Figure 2.3: Class assignment using a Nearest Neighbour (NN) approach for 5 different classes. (a) In the initial setting, we encounter labelled training samples (coloured) and unlabelled test samples (grey). (b) $k = 1$: a given test sample gets the label of its closest training sample. (c) $k = 3$: a given test sample obtains the most represented label among its 3 closest training samples.

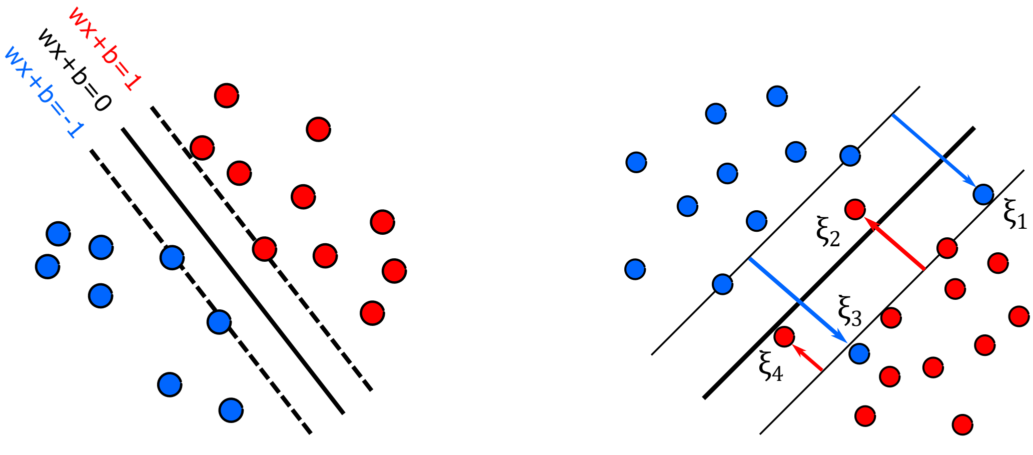
2.1.1 *k*-Nearest Neighbour

One of the simplest methods for object classification is the so-called nearest neighbour (NN), which requires no training process. For every new test sample that we want to classify, the algorithm assigns the object class label of the closest training example. In order to increase the robustness against outliers, the algorithm generalises to the k-nearest neighbours, i.e. checks the annotated label of the k closest training examples. The assigned class label is therefore the one with the highest number of occurrences. Examples of NN and k-NN are shown in Figure 2.3.

This method, however, has two major drawbacks: (1) an exhaustive search of all training examples is necessary for each test example and (2) outcomes tend to be biased when one class dominates over the others with many more examples. Both issues are typically attenuated by using fast search structures like kd-trees [Mount, 2010] and weighting the distances between training and test examples, respectively. A more robust version is presented in the Naive-Bayes nearest neighbour (NBNN). Assuming uniform class priors, the posterior probabilities are reduced to maximum likelihoods, which are efficiently approximated using the closest example neighbours for each class [Boiman et al., 2008]. In some applications, this method reports competitive results regardless of its extremely modest formulation.

2.1.2 Support-Vector Machine

Given a set of training images from two different classes that are linearly separable, a support-vector machine (SVM) constructs a hyperplane in a D-dimensional space, which separates the training examples of both classes by maximising the margin between them [Cortes and Vapnik, 1995]. We define the following quadratic formulation, represented in Figure 2.4a:



(a) Hard-margin SVM

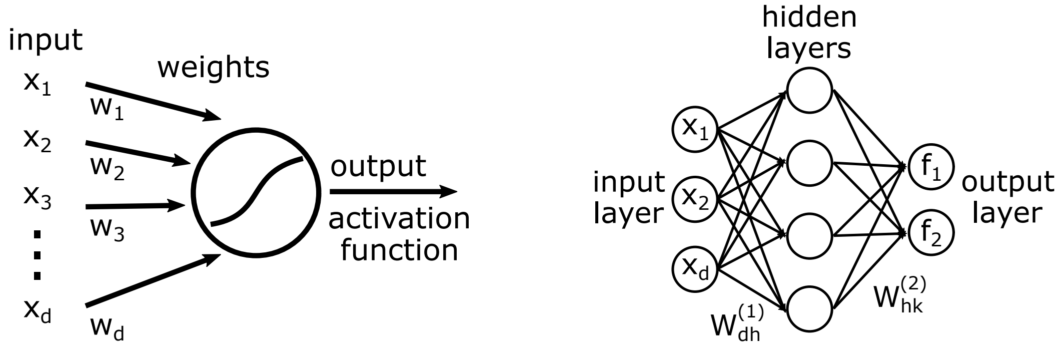
(b) Soft-margin SVM

Figure 2.4: Visual representation in 2D of a 2-class linear separation using a support-vector machine. (a) Maximum separability with a linear SVM. (b) Soft-margin separability with slack variables $\xi_.$.

$$\begin{aligned} &\underset{w,b}{\operatorname{argmin}} \frac{1}{2}||w||^2 \\ \text{subject to}& \\ &y_n(w^T x_n + b) \geq 1 \quad n = 1..M+N, \end{aligned} \tag{2.1}$$

where M and N are the amount of training samples for class labels $y = +1$ and $y = -1$, respectively, and x_n is a training sample associated to y_n. The decision values of the classifier $y(x) = w^T x + b$ for the test sample x are formed by affine weights w and a bias term b, which can be computed using standard quadratic solvers, e.g. Lagrange multipliers. However, features from different classes are commonly non-separable and additional unknowns, namely slack variables ξ_n, extend the formulation to allow for misclassifications, i.e. moving from a hard to a soft margin scheme, as shown in Figure 2.4b. Therefore, the final minimisation function of (2.1) adds $C\sum_{n=1}^{M+N}\xi_n$ with $1-\xi_n$ on the right hand side of its constraint with the input parameter C, which modules the tolerance of wrong classifications. The lower the value C, the softer the margin.

SVMs can also define non-linear separations by mapping the data into a higher dimensional space, $\phi(x)$. Since this mapping appears in dot products throughout the extended formulations, they can be replaced by kernel functions, $k(x_i, x_j) = \phi(x_i)^T\phi(x_j)$, known as kernel trick, that implicitly map the data to the higher-dimensional space without having to compute ϕ explicitly. Valid kernel functions must be positive definite symmetric and thus satisfy the Mercer's theorem [Smola and Schölkopf, 1998]. Examples of commonly used kernels are the polynomial kernel, $k(x_i, x_j) = (x_i^T x_j + 1)^p$, and the radial basis function kernel, $k(x_i, x_j) = exp{-\frac{(x_i-x_j)^2}{2\theta^2}}$.



(a) Perceptron

(b) Multi-Layer Perceptron

Figure 2.5: Representation of a biologically inspired perceptron (a) and its multi-layer version with hidden layers (b).

There are two main strategies that extend the binary decision nature of SVMs into a discriminative method for K object classes, being $K > 2$. The *one-vs-all* approach learns K classifiers, training each object category against all training samples from the remaining classes. We assign the label $y = \max(y_{1..K}(x))$ to the test example x. On the contrary, the *one-vs-one* approach learns $K(K-1)/2$ classifiers by training all possible combinations of the K classes. The class with the most number of votes is assigned.

2.1.3 Neural Network

Neural network models derive from the family of the biologically inspired perceptrons. A (single-layer) perceptron corresponds to a generalised linear discriminant that through an activation function σ, e.g. Sigmoid,

$$\sigma(x) = \frac{1}{1+e^{-x}} \ , \tag{2.2}$$

or hyperbolic tangent (tanh), returns a discriminative prediction given an input vector x, as depicted in Figure 2.5a. Activation functions are usually non-linear, i.e. continuous differentiable, in order to better model complex data, e.g. images, and decide whether an input vector (neuron) must be triggered or not, approximating a binary decision. Adding an additional *hidden* layer with an arbitrary number of nodes, continuous functions are then approximated in the form of a fully connected network that at the end of its model contains k different outputs. The network weights W are then trained by minimising the error between true training labels y_n for the examples x_n and the estimated output labels $f_k(x_n)$:

$$E_k(W) = \sum_n \mathcal{L}(y_n, f_k(x_n; W)) \ , \tag{2.3}$$

where $\mathcal{L}$ represents the loss function, e.g. least-squares (L_2 distance) or cross-entropy, for the final learnt decision function of output k with one hidden layer, as illustrated in Figure 2.5b:

$$f_k(x_n) = \sigma(\sum_{i=0}^{h} W_{ki}^{(2)}(\sum_{j=0}^{d} W_{ij}^{(1)} x_{nj})) \ . \tag{2.4}$$

By stacking more hidden layers, neural networks, i.e. multi-layer perceptrons, become *deep* enough to better describe and discriminate object categories at the cost of having a training phase that is computationally expensive, since the weights in every hidden layer need to be jointly updated. This process can be done by using the gradient descent method provided f is differentiable, whose gradients of the loss function are computed through backpropagation [LeCun et al., 1998].

Neural networks for computer vision applications receive images as input data and therefore a few modifications in the model are necessary to understand the 2D spatial layout of images. Hence, this considerably more complex network architecture requires a few adaptations to efficiently cope with an additional dimension and the implicit spatial relations among neighbouring pixels. The first and most relevant building block are convolutional layers, which substitute fully connected layers from standard neural networks to save computational time with the assumption that the network is locally connected and that the information is shared by using the same parameters and convolutions, i.e. learned weights. A convolution of functions f and g is expressed as

$$(f * g)(t) = \int_{\infty}^{-\infty} f(x)g(t-x)dx \tag{2.5}$$

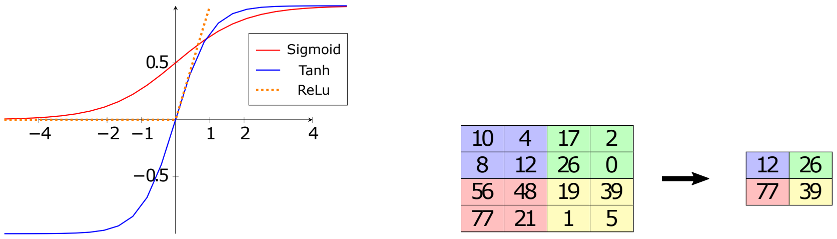
in its continuous formulation shifted by t. Its discretisation for a given pixel (i, j) in image I is written as

$$(f * I)[i,j] = \sum_{p_i=-\lfloor \frac{P}{2} \rfloor}^{\lfloor \frac{P}{2} \rfloor} \sum_{p_j=-\lfloor \frac{P}{2} \rfloor}^{\lfloor \frac{P}{2} \rfloor} f[p_i,p_j]I[i-p_i, j-p_j] \tag{2.6}$$

for a squared kernel g of path size $P \times P$. In this type of layer, function f operates as a 2D weighting kernel over I, which is an input image or a feature map for initial or intermediate layers, respectively. Besides, there is at each convolutional layer not just one but a list of filters that are appended in the depth channel, resulting in structures of size $width \times height \times depth$. Networks that use convolutional layers receive the name of *convolutional neural networks* (CNNs).

After the convolutional layer, a non-linear activation function is applied. Currently, rectified linear units (ReLu),

$$\sigma(x_i) = max(0, x_i) \ , \tag{2.7}$$



(a) Non-linear activations

(b) Max-pooling

Figure 2.6: Two basic layers in convolutional neural networks: (a) plot with popular non-linear activation functions and (b) example of max-pooling subsampling from a 4x4 to a 2x2 patch.

are preferred than Sigmoid and hyperbolic tangent functions, since its simplicity speed-up the optimisation process. In addition, ReLus favour sparsity, which tend to be more beneficial than dense representations, and for values of $x_i > 0$ the gradient does not vanish and remains constant. Figure 2.6a shows these three different types of activation functions.

The next major building layer produces spatial pooling to reduce the image dimensionality, i.e. subsampling. Standard strategies either average among all pixels or propagate the highest value of a given patch. An example of a reduction by half is shown in Figure 2.6b. Downsampling is also achieved with a stride in the convolutions, which skips after each convolution a given number of pixels. The main goal of pooling is capturing higher-levels of image information and gaining spatial robustness in the computed features. More specifically, the initial levels of hidden layers become self-crafted feature descriptors moving from a lower to a higher-level of abstraction by appending consecutive convolutions with interleaved pooling layers.

At the end, the concatenation of these layers result in a feature map, previously normalised, that becomes the input of deeper layers based on the same principle. Slight changes, including different types of convolutions and avoiding pooling steps are used along the CNN architecture. The last part of the system contains fully connected layers that act as object classifiers. A softmax activation function is commonly used for transforming the last layer into class probabilities that sum to 1 and is expressed as

$$\sigma(x)_i = \frac{exp(x_i)}{\sum_{j=1}^{K} exp(x_j)} \tag{2.8}$$

for the feature x of class i normalised by the sum of all K output dimensions, i.e. classes, which is usually optimised defining a cross-entropy loss function, $\mathcal{L}_{cross}$, that measures the error of the softmax probabilities given by

$$\mathcal{L}_{cross} = -\sum_{i=1}^{K} y_i log(\sigma(x_i)) \ , \tag{2.9}$$

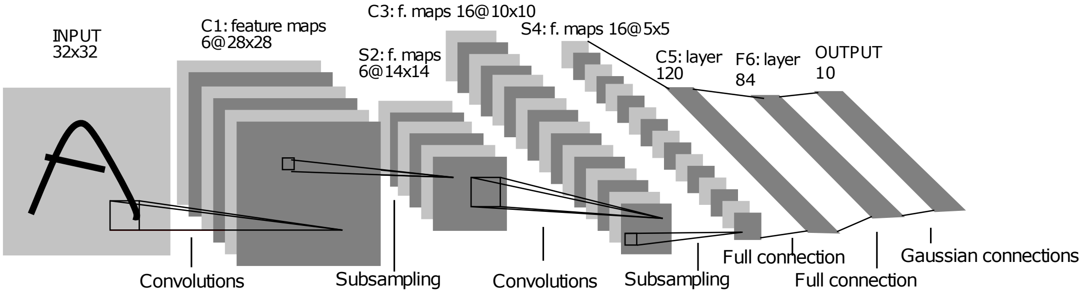


Figure 2.7: Scheme of a convolutional neural network for digit recognition with concatenated convolutional, pooling and fully connected layers. A 32x32 input image feeds the network with 10 outputs from digit '0' to '9'. From [LeCun et al., 1998].

where y_i is a binary indicator that states whether class i is assigned to the input sample of the network. An example of an end-to-end architecture for digit recognition is shown in Figure 2.7.

Currently, deep CNNs have significantly outperformed previous supervised learning methods and produce state-of-the-art results in object classification tasks. Since CNNs are also widely used for learning complex feature encoders, the output of CNN layers as feature descriptors are further described in Section 2.2.2.3. In spite of its high accuracy in object recognition, neural networks require large amounts of data and a deep structure, i.e. a large number of concatenated layers, to cope and process as much scene and class information as possible. Although these requirements produce impractical computational times for training models with millions of unknowns, the recent advances in hardware and design optimisations have highly reduced the training times of deep neural networks. The first seminal work with top object classification accuracies was presented in the AlexNet model by Krizhevsky et al. [2012], which outperformed by a large margin not only previous CNN methods, but also other techniques, e.g. SVM-based methods, in popular object recognition challenges. This network is modelled as a 1000-object classification system.

Some computer vision problems are solved by estimating a 2-dimensional feature map as output of the CNN. In this case, image labelling appears at pixel level when annotating semantic labels, e.g. segmentation tasks, or heat maps, e.g. keypoint estimation. The networks are in most of the cases fully convolutional and are optimised employing a modified cross-entropy loss of (2.9) in 2D space or a regression formulation that predicts continuous values, respectively. From the latter, the Euclidean loss function between the predicted value y in the last layer of the CNN and a given training sample x is formulated as

$$\mathcal{L}_{reg} = \sum_{i=1} \sum_{j=1} \| f(x_{ij}) - y_{ij} \|_2^2 \tag{2.10}$$

along all indexes of a 2D map of a given height and width.

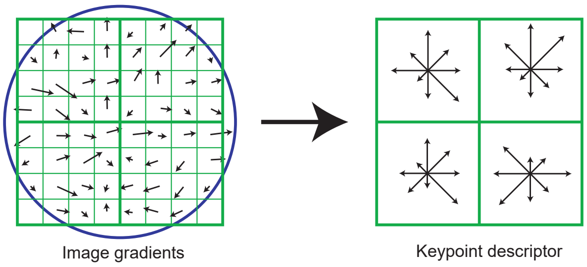


Figure 2.8: Extracted SIFT features from a region of interest with its 4x4 sub-patch representation. From [Lowe, 2004].

2.2 Feature Descriptors

During the last decades, there has been many ways of representing images, interest points or annotated objects, in order to better describe their visual traits while keeping robustness against image transformations and variations within the same class. These encoded image representations are called *features* or *feature descriptors*. Currently, appearance-based approaches, which extract such features from the pixel values of the images, e.g. from coloured and greyscale channels, are used in all relevant object recognition tasks.

2.2.1 Local Features

In some scenarios, certain regions contain more discriminative information than others. For instance, corners, unique shapes or textured regions are likely to provide better feature descriptors that straight lines or flat textures. Local features appear from this assumption, extracted from small interest points in the image, previously selected through low-level vision techniques, such as corner detectors [Harris et al., 1988]. Using a selection of local features to describe an object provides a lot of robustness against occlusions and small transformations that can affect the correspondences between features in order to identify the object. Although these features are mainly applied to image matching algorithms and not strictly for recognition purposes, it creates the basis for further widely used feature descriptors. The most popular local descriptor is the Scale Invariant Feature Transform (SIFT) Lowe [2004]. From an initially smoothed image, its extraction starts dividing the region of an interest point into 4x4 subpatches for a total of 16 cells. For each pixel, it calculates the gradient magnitude and orientation later accumulated in a histogram of oriented gradients (usually 8 reference angles) at each sub-patch. As shown in Figure 2.8, the descriptor results in a feature vector of 128 dimensions that shows robustness against small scale invariances. Other relevant captured invariances are illumination changes due to the use of gradients, small rotations due to the discretised orientations and small translation shifts by using sub-patches.

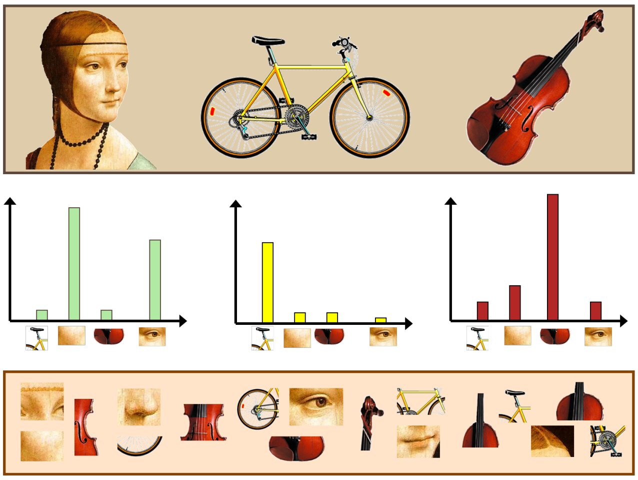


Figure 2.9: Example of a Bag-of-words feature descriptor using a bust, a bicycle and a violin as the object classes (top). Their computed global histograms/features (middle) are based on a set of local regions of interest (bottom). From Fei-Fei et al. [2005].

2.2.2 Global Features

In contrast to features extracted from interest points, global features make use of the whole annotated object class to describe it as a whole. Before the abrupt arrival of CNNs that produce state-of-the-art feature descriptors, object recognition methods were mainly designed with hand-crafted global features, which in practical applications outperformed locally-based approaches. Hand-crafted features represent those that are manually fixed to problem needs and are no longer modified. In contrast, neural networks belong to the family of learnt features, since their filters and convolutions are learned from training images without user intervention.

2.2.2.1 Bag-of-Words

The main idea behind Bag-of-words (BoW) [Csurka et al., 2004] is computing local features from interest regions and then generating a histogram from these descriptors. This histogram is used as a global feature descriptor, which represents unordered points with a single vector. However, the object's spatial layout is removed, as illustrated in Figure 2.9. This theoretical drawback is indeed robust against deformations and occlusions, but lacks the inclusion of spatial cues which might end up playing a big role in the later recognition. Therefore, a spatial pyramid representation has also been proposed for an in-between solution to preserve spatial information [Lazebnik et al., 2006].

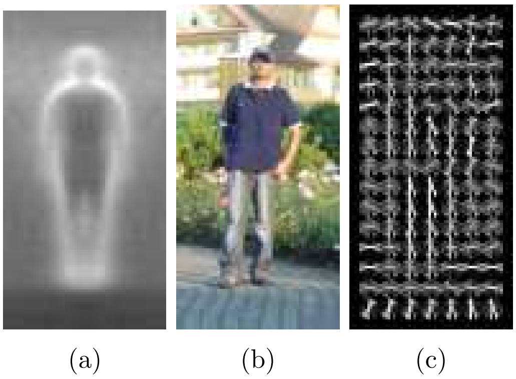


Figure 2.10: HOG descriptors mainly provide cues of contours. (a) Average gradient image over all training examples. (b) Test image. (c) Visual representation of the extracted HOG features. Pedestrian dataset and figures from Dalal and Triggs [2005].

2.2.2.2 Histogram of Oriented Gradients

The most popular global descriptor and with the most prominent classification results in the family of hand-crafted features is the Histogram of Oriented Gradients (HOG) [Dalal and Triggs, 2005]. Based on the same idea as SIFT descriptors, using gradients and bins for maximising invariance against transformations, HOGs move to a global dense extraction of features, which has been proven to be of great success in well-defined object classes, i.e. relatively rigid objects, with not so high level of intra-class variation and few occlusions. The steps are summarised as follows, given an input image region:

1. Gamma correction to reduce the impact of strong gradients.

2. Compute gradients in all colour channels and take the strongest one.

3. Weighted vote in spatial and orientation cell bins (8x8 pixels for 8 orientations), in the same fashion as SIFT.

4. Contrast normalisation over overlapping spatial cells. Trilinear interpolation is crucial for robustness: bilinear in image space where each pixel contributes to 3 neighbouring cell histograms and then another linear interpolation over the measured gradient angles.

5. Create the descriptor as an array of all these cell channels. The resulting feature descriptor for an object class is shown in Figure 2.10c for pedestrian detection. The preserved spatial layout, if the classifier allows it, perfectly fits in standard sliding window approaches, including their speed-ups, convolving such feature descriptor, seen as a template, all over the extracted dense features of the detection image. Therefore, the detection score map obtained indicates regions that are likely to contain pedestrians.

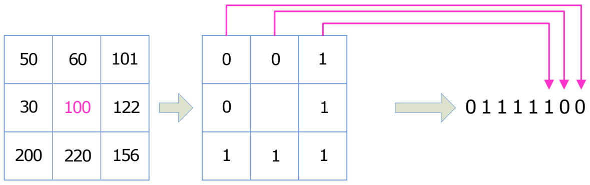


Figure 2.11: LBP example from an extracted 3x3 patch. All decimal outputs for each cell are then represented in a 256-bit histogram. From Wang et al. [2009].

In addition, HOG features are easily extendable by appending new feature data. From all introduced extensions, Local Binary Patterns (LBP) [Ojala et al., 1994] arise as the HOG enhancement with the best recognition results [Wang et al., 2009]. In every cell of the computed feature and for each pixel p, one computes from a 3x3 patch formed by the 8-connected neighbours of p:

$$LBP(x_i, y_j) = \begin{cases} 1, & \text{if } patch(x_i, y_j) \geq p \\ 0, & \text{if } patch(x_i, y_j) < p \end{cases} \quad \text{for } i, j = [1, 2, 3] \ . \tag{2.11}$$

The computed binary values for all neighbours are then concatenated, converted to decimal (0...255) and appended to the HOG cell using a 256-bit histogram. A visual description of LBPs is shown in Figure 2.11.

2.2.2.3 Neural Networks

As explained in Section 2.1.3, architectures based on deep neural networks currently obtain state-of-the-art results in a wide range of image classification problems. Although this end-to-end system embeds a classifier in its last layer, another major trait of this methodology is the learning of features, in the early layers, instead of using hand-crafted structures designed by the user. Therefore, intermediate neural network layers, namely feature maps, are extracted and used as powerful feature descriptors that are then applied to other supervised learning models.

The first widely used feature encoder from a CNN was the last fully connected layer of the AlexNet model [Krizhevsky et al., 2012]. Then, more advanced CNNs were introduced, contributing in some critical aspects: (1) more stacked convolutional layers divided in more efficient 3-by-3 convolution kernels from the VGG model [Simonyan and Zisserman, 2014], (2) parallelisation of multiple layers with their own specific filters that are concatenated in a later stage from the GoogLeNet/Inception model [Szegedy et al., 2015] and (3) the inclusion of residual connections that append early layers in later stages, skipping one or more intermediate layers, that avoid vanishing gradients in early training stages and therefore allow for a much deeper design, e.g. up to 152 layers, from the ResNet model [He et al., 2016]. These models are usually fine-tuned by retraining them with a new output layer, i.e. class-probabilities, based on the specific needs of the problem that needs to be solved. The advantage of fine-tuning is that only a small portion of new training exemplars are necessary to obtain substantial improvements, since all network weights from the previous layers are initialised with pre-trained values that already provide decent accuracies. These CNN weights are typically trained with an extremely large database, e.g. ImageNet [Deng et al., 2009], with millions of exemplars and data augmentation strategies.

A noticeable constraint in these CNNs is the fixed image size, e.g. 224x224, required as input due to the rigid nature of containing fully connected layers in the later stages of the network. Extracting feature encoders from the fully convolutional part of the network, however, do not require a fixed input size and thus becomes more manageable for problems that include object categories with variable resolutions.

Chapter 3

Previous Work

”I’ve never scored a goal in my life without getting a pass from someone else.”

—ABBY WAMBACH (1980–)
ex-football player

Contents

3.1 Domain Adaptation

The interest in studying domain adaptation techniques for computer vision problems increased with the release of a benchmark by Saenko et al. [2010] for domain adaptation in the context of object classification. The first relevant works on unsupervised domain adaptation for object categorisation were presented by Gopalan et al. [2011] and Gong et al. [2012], who proposed an alignment in a common subspace of source and target samples using the properties of Grassmanian manifolds. Jointly transforming source and target domains into a common low dimensional space was also done together with a conjugate gradient minimisation of a transformation matrix with orthogonality constraints [Baktashmotlagh et al., 2013] and with dictionary learning to find subspace interpolations [Ni et al., 2013, Shekhar et al., 2013, Xu et al., 2015]. Sun and Saenko [2014] and Sun et al. [2015a] presented a very efficient solution based on second-order statistics to align a source domain with a target domain. Herath et al. [2017] also match second-order statistics with a joint estimation of latent spaces. To obtain an estimate of the target distribution in the latent space, Gholami et al. [2017] introduce a Bayesian approximation to jointly learn a softmax classifier across-domains. Similarly, Csurka et al. [2016] jointly denoise source and target samples to reconstruct data without partial random corruption. Zhang et al. [2017a] also align distributions, but they include geometrical differences in a joint optimisation. Sharing certain similarities with associations between domains, Gong et al. [2013a] minimise the Maximum Mean Discrepancy (MMD) [Gretton et al., 2006] of two datasets. They assign instances to latent domains and solve it by a relaxed binary optimisation. Ming Harry Hsu et al. [2015] use a similar idea and allow instances to be linked to all other samples.

Semi-supervised domain adaptation approaches take advantage of knowing the class labels of a few target samples. Aytar and Zisserman [2011] proposed a transfer learning formulation to regularise the training of target classifiers. Exploiting pairwise constraints across domains, Saenko et al. [2010] and Kulis et al. [2011] learn a transformation to minimise the effect of the domain shift while also training target classifiers. Following the same idea, Hoffman et al. [2013] considered an iterative process to alternatively minimise the classification weights and the transformation matrix.

The idea of selecting the most relevant information of each domain has been studied in early domain adaptation methods in the context of natural language processing [Blitzer et al., 2006]. Pivot features that behave the same way for discriminative learning in both domains were selected to model their correlations. Gong et al. [2013b] presented an algorithm that selects a subset of source samples that are distributed most similarly to the target domain. Another technique that deals with instance selection has been proposed by Sangineto [2014]. They train weak classifiers on random partitions of the target domain and evaluate them in the source domain. The best performing classifiers are then selected. Other works have also exploited greedy algorithms that iteratively add target samples to the training process, while the least relevant source samples are removed [Bruzzone and Marconcini, 2010, Tommasi and Caputo, 2013].

During the last years, a large number of domain adaptation methods have been based on deep CNNs [Krizhevsky et al., 2012], which learn more discriminative feature representations than hand-crafted features and substantially reduce the domain bias between datasets in object recognition tasks [Donahue et al., 2014]. Non-adapted classifiers trained with features extracted from CNN layers outperform domain adaptation methods with shallow feature descriptors [Donahue et al., 2014, Sun et al., 2015a]. Many of these deep domain adaptation architectures are inspired by the traditional methods and seek to minimise the MMD distance as a regulariser to learn features for source and target samples jointly [Ghifary et al., 2014, Tzeng et al., 2014, Long et al., 2015, 2016, Yan et al., 2017]. Going one step further, Saito et al. [2018a] utilise a minimax problem that finds two classifiers that maximise the discrepancy on the target sample, but at the same time generate features that minimise it. The impact of intra-class discrepancies are addressed by Kang et al. [2019], who jointly train an intra- and inter-class discrepancy loss in alternating updates. Extending this type of networks, Carlucci et al. [2017] use intermediate layers for the alignment of distributions before batch normalisation. They learn a parameter that steers the contribution of each domain at a given layer. Similarly with one backbone network for each domain, Rozantsev et al. [2018] introduced loss functions shared by both source and target networks at each intermediate layer that prevent weights from being too dissimilar. Ganin and Lempitsky [2015] added a domain classifier network after the CNN to maximize the discriminatory loss of both domains while jointly minimising the classification loss using source data. More recently, Tzeng et al. [2017] propose a generalised framework for adversarial adaptation, extended by Long et al. [2018] with a randomised multilinear map between feature representations and class predictions that improves the discriminability among classes.

In the semi-supervised setting, Motiian et al. [2017] present a deep domain adaptation method that exploits the domain loss minimisation while maximizing the distances between labelled samples from different domains and classes.

From a different visual perspective, Bousmalis et al. [2017] tackles the problem of domain adaptation by introducing an adversarial network that transforms source images at pixel level with a joint minimisation of the classification and similarity loss. Other forms of data representation, such as hash codes [Venkateswara et al., 2017] and scatter tensors [Koniusz et al., 2017, Lu et al., 2017], have also been combined with deep domain adaptation architectures to further reduce the domain bias.

Lately, some works expanded the application of domain adaptation techniques to not just object classification, but also localisation. Inoue et al. [2018] fine-tune a model trained with only source data by using generated target images from a generative adversarial network that transforms source images to visually similar target samples. Another approach is presented by [Chen et al., 2018], who model a consistency loss at sample and image level after the Faster R-CNN object detector [Ren et al., 2015].

3.2 Open Set Recognition

The inclusion of *open sets* in recognition tasks appeared in the field of face recognition, where evaluation datasets contain unseen face instances as impostors that have to be rejected [Phillips et al., 2000, Li and Wechsler, 2005]. Such open set protocols are nowadays widely used for evaluating face recognition approaches [Sun et al., 2015b].

The generalisation towards an open set scenario for multi-object classification was introduced by Scheirer et al. [2013], who addressed the more realistic case of a finite set of known objects mixed with many unknown ones. Based on this principle, Jain et al. [2014] and Scheirer et al. [2014] propose multi-class classifiers that detect unknown instances by learning SVMs that assign probabilistic decision scores instead of class labels. More recently, Bendale and Boult [2016] adapt traditional neural networks for open set recognition tasks by introducing a new layer that estimates the probability of an object to be labelled as unseen class. Closely related are also the works by Zhang and Metaxas [2006] and Bartlett and Wegkamp [2008] that add a regulariser to detect uninformative data and penalise a misclassification during training. Lately, Gavves et al. [2015] present an active learning technique, whose initially trained SVMs on a subset of known classes are used as priors to further train novel object classes from other target datasets.

Its application to domain adaptation, presented in this thesis, was further researched by Saito et al. [2018b], which modifies the closed set specific network by Ganin and Lempitsky [2015] and propose an adversarial network for open set domain adaptation with an additional unknown class.

3.3 Viewpoint Estimation

Methods for viewpoint estimation are often based on popular object class detectors [Leibe et al., 2004, Dalal and Triggs, 2005, Felzenszwalb et al., 2010, Girshick et al., 2014] and learn a discrete set of pose classifiers. In [Liebelt and Schmid, 2010, Fidler et al., 2012, Pepik et al., 2012, Hejrati and Ramanan, 2014], annotations from 2D images are enhanced with 3D metadata to formulate 3D geometric models. On the contrary, Gu and Ren [2010] learn a mixture-of-templates that inherently captures the characteristics of projected views and Ozuysal et al. [2009] refine the hypothesis of 16 viewpoint detectors from 2D images with additional view specific Naïve Bayes classifiers. More recently, CNNs for object classification [Krizhevsky et al., 2012] have been retrained using 2D pose annotations in order to provide viewpoint probabilities as output channels coupled with the object class probability [Tulsiani and Malik, 2015, Pepik et al., 2015]. In the study pursued by Ghodrati et al. [2014], simple frameworks that extract features from 2D bounding boxes with powerful encoders provided the same or even better viewpoint accuracies than state-of-the-art methods based on complex 3D models. Su et al. [2015] propose a classification-based CNN model with one bin per degree, i.e. 360 bins for the azimuth angle, and a Gaussian function that spreads the optimisation to neighbouring bins. The training phase uses millions of synthetic samples to compensate the fine viewpoint representation. A coarser discretisation was introduced by Tulsiani and Malik [2015], which showed better accuracies when trained on real data. More recently, Divon and Tal [2018] introduced a triplet loss to increase the dissimilarity of viewpoints that are far apart. Viewpoint estimation can also benefit from 3D object detections, as shown by Kehl et al. [2017], who extended a popular real-time object detector with 3D viewpoint predictions.

In contrast to classification approaches, regression approaches [Torki and Elgammal, 2011, Fenzi et al., 2013] do not require a discretisation of the viewpoints. In the work by He et al. [2014], the viewpoint regression is integrated into a joint discriminative continuous parametrised model. The localisation and the continuous pose of objects are jointly estimated by Redondo-Cabrera et al. [2014] using a Hough forest regression voting scheme. Accumulated votes in the Hough space are later refined with a kernel density estimator to consolidate votes in a local region close to the current maxima. Similarly, Hough forests have been used for head pose estimation [Fanelli et al., 2013] where patches from depth images are used. Glasner et al. [2012] also utilise a voting process to refine the prediction of discretised pose classifiers. Recent studies [Massa et al., 2016, Pepik et al., 2015] concluded, nonetheless, that CNNs for viewpoint classification outperform CNNs for viewpoint regression by a considerable margin when the number of discrete viewpoints is formed by at least 16 bins. For further details on joint object detection and pose estimation, we refer to the studies by Massa et al. [2014] and Elhoseiny et al. [2016].

The 3D spatial information of graphics models was already addressed in several works to estimate the viewpoint of object instances, as well as its localisation [Mottaghi et al., 2015, Liebelt and Schmid, 2010, Pepik et al., 2012, Schels et al., 2012, Stark et al., 2010, Zia et al., 2013]. These algorithms are computationally expensive, since the object geometry is used to learn the spatial 3D relations of parts or features.

Some approaches already used the spatial information of keypoints to estimate accurate viewpoints. Torki and Elgammal [2011] learn a regression function to compute the azimuth angle of vehicles based on pre-computed local features and their spatial arrangements. Pepik et al. [2012] extend the deformable part model [Felzenszwalb et al., 2010] to 3D objects, optimising at the same time the location and the viewpoint of the object for a fixed number of bins. Concretely for hand pose estimation, Zimmermann and Brox [2017] compute the camera parameters by using keypoint confidence maps as input of the network. A deep regression technique is presented by Wu et al. [2016], where 2D keypoints are used to estimate the camera parameters after concatenating several fully connected layers. Lately, Grabner et al. [2018] use the Perspective-n-Point algorithm to extract the viewpoint from a detected 3D bounding box and Zhou et al. [2018a] takes as input RGB-D images for 3D keypoint localisation combined with an unsupervised domain adaptation technique among views to obtain better accuraciess.

3.4 Keypoint Estimation

Research in keypoint estimation with CNNs has mostly been centred on human articulated poses. Toshev and Szegedy [2014] initially proposed a 2-stage architecture, which firstly estimates the 2D coordinates of each keypoint using a fully connected layer as input of its regression loss, and then refines the prediction feeding the region of interest on a second CNN model. Later, Tompson et al. [2015] improved the keypoint localisation with a cascade architecture that overlaps croppings. The human pose estimation proposed by Wei et al. [2016] optimises confidence maps for each keypoint. This model appends the later portion of the network several times, i.e. the input of the new stage comes from the output of the previous one, creating larger receptive fields. The deeper the stacked network the more it suppresses ambiguities and better captures the spatial layout of the keypoints. Similar in spirit, Belagiannis and Zisserman [2017] designed a recurrent model with multiple stages to reduce ambiguities and thus the amount of false positives combined with the prediction of keypoint visibility. Newell et al. [2016] refined this type of multi-stage architecture by adding transposed convolutions at the end of each stage for finer confidence maps. This model was extended with a Conditional Random Field (CRF) model that adds spatial cues [Chu et al., 2017].

Previous to the work presented in this dissertation, keypoint estimation in rigid objects has already been in focus. Long et al. [2014] initially addressed the capabilities of CNNs for keypoint estimation by dividing the last convolutional layer in smaller cells and training each keypoint as an independent class in a multi-class SVM. Moving towards a purely neural network approach, Tulsiani and Malik [2015] concatenate the spatial information in a fully connected layer and only activate through the network those receptive fields that include the corresponding keypoints. The prediction is further refined with independently computed viewpoints. The human pose estimation by Newell et al. [2016] has been modified by Pavlakos et al. [2017] and Zhou et al. [2018b] to detect 3D keypoints of multiple rigid classes to consequently estimate the translation and rotation of the object by fitting the keypoints into a shape model.

The impact of multi-task learning in deep neural networks to improve keypoint estimation has been researched by Liu et al. [2016], who designs a hierarchical CNN that recognises human grouping and their individual actions. On a more generalised approach, Zamir et al. [2018] also trains 2D and 3D keypoints combined with a total of 26 2D, 2.5D and 3D semantic supervised tasks.

3.5 Synthetic Data

The use of synthetic images from rendered models and scenes as training data started to gain attention in the context of pedestrian detection. While Marín et al. [2010] only use synthetic data generated from a popular game engine, Pishchulin et al. [2011] combine real with synthetic data from highly accurate 3D reconstructed humans. Both methods, however, do not consider the 3D information and collect only 2D images with automatically annotated bounding boxes.

In recent years, new published datasets based on computer generated models with accurate 3D pose information have been proposed. For instance, ShapeNet [Chang et al., 2015] provides a large dataset of 3D graphics models for hundreds of object classes. Its drawback comes from the low quality of most of their 3D models. Tremblay et al. [2018] introduce synthetic data of 21 household objects with fine 3D annotations that handle occlusions. However, they only use one 3D model for each object class. More in the direction of human pose estimation, Varol et al. [2017] generate synthetic humans to include 3D joint information, i.e. depth, in the learning process. Instead of rendering 3D data, synthetic data can also be generated by defining a parametric model for synthesising geometric shapes from a particular object class, used in both recognition and reconstruction, as proposed by Hejrati and Ramanan [2014]. Recently, Su et al. [2015] and Peng et al. [2015] tested the impact of synthetic data in CNNs by training millions of synthesised images from 3D models. Thus, the main challenge becomes the generation of extremely large amounts of data with as much intra-class variation as possible, e.g. viewpoint and shape, to avoid overfitting.

3D models have also been used to annotate datasets [Matzen and Snavely, 2013, Xiang et al., 2014, Wang et al., 2018] by manually superposing them on top of 2D object instances. While the 3D models are supported by humans and improve the accuracy of the annotation, the annotation process with 3D models is very slow and still prone to annotation errors. Instead of using synthetic data, Sedaghat and Brox [2015] proposed a supervised approach that automatically annotates cars, bounding boxes and azimuth angles, in videos using structure from motion.

In the context of domain adaptation, Peng et al. [2018] recently published a novel dataset for adapting synthetic data to real data. Influenced by our work presented in this thesis [Panareda Busto and Gall, 2017], this dataset also specifies challenges for both closed and open sets.

Specifically, the field of autonomous driving, which already counts with well established datasets, is starting to incorporate synthetic datasets with precise 3D information. While the KITTI [Geiger et al., 2012] and Cityscapes [Menze and Geiger, 2015] datasets provide fine 3D annotations based on expensive lidar and radar systems, Cordts et al. [2016] published a fully self-supervised synthetic dataset with optical flow and depth information.

Chapter 4

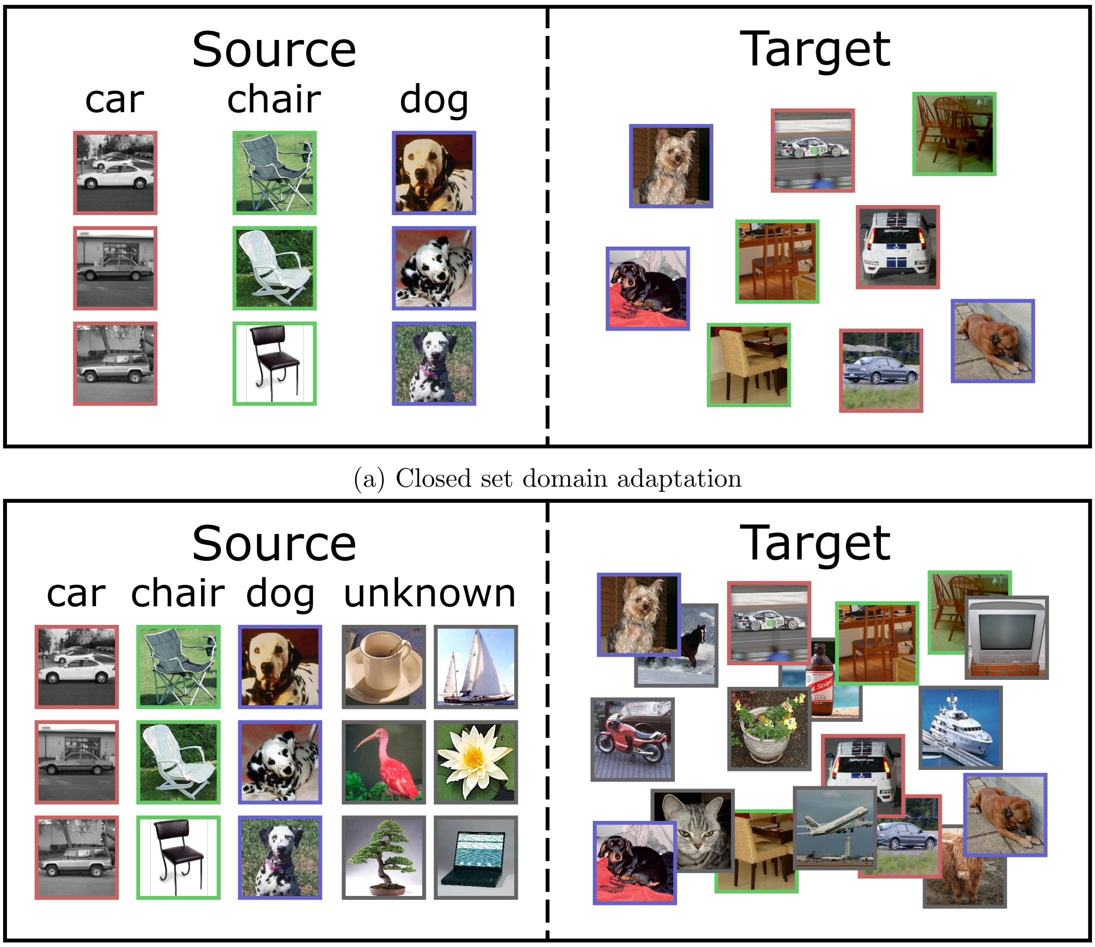
Open Set Domain Adaptation for Image and Action Recognition

Contents

4.1 Introduction

In the last years, impressive results have been achieved on large-scale datasets for image classification or action recognition. Acquiring such large annotated datasets, however, is very expensive and there is a need to transfer the knowledge from existing annotated datasets to unlabelled data that is relevant for a specific application. If the labelled and unlabelled data have different characteristics, they have been sampled from two different domains. In particular, datasets that have been collected from the Internet, e.g. from platforms for sharing videos or images, differ greatly from data that needs to be processed for an application. To address the domain shift between the labelled dataset, which is the source domain, and the unlabelled data from the target domain, various unsupervised domain adaptation approaches have been proposed. If the data from the target source is partially labelled, the problem is termed semi-supervised domain adaptation. In this work, we address unsupervised and semi-supervised domain adaptation in the context of image and action recognition.

Although the methods for domain adaptation have been advanced tremendously in the last years [Saenko et al., 2010, Gopalan et al., 2011, Gong et al., 2012, Chopra et al., 2013, Hoffman et al., 2014, Ganin and Lempitsky, 2015, Ming Harry Hsu et al., 2015,



(a) Closed set domain adaptation

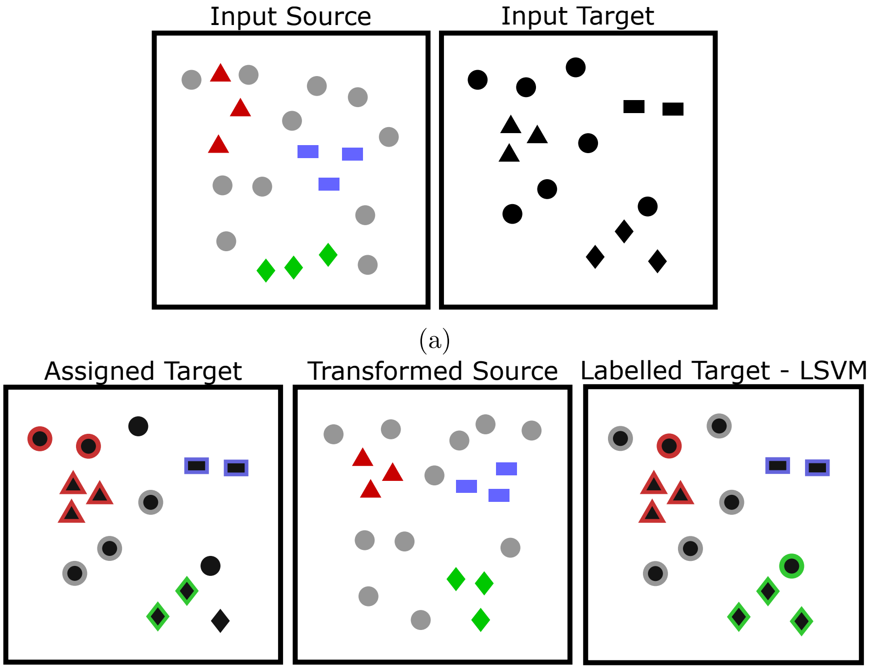
(b) Open set domain adaptation

Figure 4.1: (a) Standard domain adaptation benchmarks assume that source and target domains contain images or videos only of the same set of categories. This is denoted as *closed set domain adaptation* since it does not include samples of unknown categories or categories which are not present in the other domain. (b) We propose *open set domain adaptation*. In this setting, both source and target domain contain images or videos that do not belong to the categories of interest. Furthermore, the target domain contains images or videos that are not related to any image or video in the source domain and vice versa.

Ghifary et al., 2016, Tzeng et al., 2017, Motiian et al., 2017], the evaluation protocols were restricted to a scenario where all categories in the target domain are known and match the categories in the source domain. Figure 4.1a illustrates such a *closed set domain adaptation* setting. The assumption that all images or videos that are in the target domain belong to categories in the source domain, however, is violated in most cases. In particular if the number of potential categories is very large as it is the case for object or action categories, the target domain contains images or videos of categories that are not present in the source domain since they are not of interest for a specific application. We therefore propose a more realistic evaluation setting for unsupervised or semi-supervised domain adaptation, namely *open set domain adaptation*, which builds on the concept of open sets [Scheirer et al., 2013, 2014, Bendale and Boult, 2016]. As illustrated in Figure 4.1, the source and target domains are not any more restricted in the open set case to share the same categories as in the closed set case, but both domains contain images or videos from categories that are not present in the other domain.

To address the problem of open set domain adaptation, we propose a generic approach that learns a linear mapping that maps the feature space of the source domain to the feature space of the target domain. It assigns a subset of images or videos of the target domain to the categories of the source domain and transforms the feature space of the source domain gradually towards the feature space of the target domain. By using a subset instead of the entire set, the approach handles images or videos in the target domain that are not related to any sample in the source domain. The approach can be applied to any feature space, which includes features extracted from images as well as features extracted from videos. The approach works in particular very well for features spaces that are extracted by convolutional networks and outperforms most end-to-end learning approaches for domain adaptation. The good performance of the approach coincides with the observation that deep convolutional networks tend to linearise manifolds of image domains [Bengio et al., 2013, Upchurch et al., 2017]. In this case, a linear mapping is sufficient to map the feature space of the source domain to the feature space of the target domain. In particular, the flexibility of the approach, which can be used for images and videos, for open set and closed set domain adaptation, as well as unsupervised and semi-supervised domain adaptation, makes the approach a versatile tool for applications. An overview of the approach for unsupervised open set domain adaptation is given in Figure 4.2.

In this chapter, we introduce open set domain adaptation for object and action recognition tasks, describing our novel adaptation algorithm, and provide a thorough experimental evaluation. We revisit popular domain adaptation data collections with our new open set protocol, both unsupervised and semi-supervised. We also present an open set evaluation for a new action recognition adaptation, from synthetic data to real data and an evaluation of the proposed approach for standard closed set protocols. In total, we evaluate the approach on 26 *open set* and 34 *closed set* combinations of source and target domains including the *Office* dataset Saenko et al. [2010], its extension with the *Caltech* dataset Gong et al. [2012], the *Cross-Dataset Analysis* Tommasi and Tuytelaars [2014], the *Sentiment dataset* Blitzer et al. [2007], synthetic data Peng



(a)

(b) (c) (d)

Figure 4.2: Overview of the proposed approach for unsupervised *open set domain adaptation.* (a) The source domain contains some labelled images, indicated by the colours red, blue and green, and some images belonging to unknown classes (grey). For the target domain, we do not have any labels but the shapes indicate if they belong to one of the three categories or an unknown category (circle). (b) In the first step, we assign class labels to some target samples, leaving outliers unlabelled. (c) By minimising the distance between the samples of the source and the target domain that are labelled by the same category, we learn a mapping from the source to the target domain. The image shows the samples in the source domain after the transformation. This process iterates between (b) and (c) until it converges to a local minimum. (d) In order to label all samples in the target domain either by one of the three classes (red, green, blue) or as unknown (grey), we learn a classifier on the source samples that have been mapped to the target domain (c) and apply it to the samples of the target domain (a). In this image, two samples with unknown classes are wrongly classified as red or green.

et al. [2017], and two action recognition datasets, namely the *Kinetics Human Action Video Dataset* Kay et al. [2017] and the *UCF101 Action Recognition Dataset* Soomro et al. [2012]. Our approach achieves state-of-the-art results in all settings both for unsupervised and semi-supervised open set domain adaptation and obtains competitive results compared state-of-the-art deep leaning approaches for closed set domain adaptation.

4.2 Open Set Domain Adaptation

We present an approach that iterates between solving the labelling problem of target samples, i.e. associating a subset of the target samples to the known categories of the source domain, and computing a mapping from the source to the target domain by minimising the distances of the assignments. The transformed source samples are then used in the next iteration to re-estimate the assignments and update the transformation. This iterative process is repeated until convergence and is illustrated in Figure 4.2.

In Section 4.2.1, we describe the unsupervised assignment of target samples to categories of the source domain. The semi-supervised case is described in Section 4.2.2. Section 4.2.3 finally describes how the mapping from the source domain to the target domain is estimated from the previous assignments. This part is the same for the unsupervised and semi-supervised setting.

4.2.1 Unsupervised Domain Adaptation

We first address the problem of unsupervised domain adaptation, i.e. none of the target samples are annotated, in an open set protocol. Given a set of classes $\mathcal{C}$ in the source domain, including $|\mathcal{C} - 1|$ known classes and an additional unknown class that gathers all instances from other irrelevant categories, we aim to label the target samples $\mathcal{T} = \{T_1, \ldots, T_{|\mathcal{T}|}\}$ by a class $c \in \mathcal{C}$. We define the cost of assigning a target sample T_t to a class c by $d_{ct} = \|S_c - T_t\|_2^2$ where $T_t \in \mathbb{R}^D$ is the feature representation of the target sample t and $S_c \in \mathbb{R}^D$ is the mean of all samples in the source domain labelled by class c. To increase the robustness of the assignment, we do not enforce that all target samples are assigned to a class as shown in Figure 4.2b. The cost of declaring a target sample as outlier is defined by a parameter λ, which is discussed in Section 4.3.1.

Having defined the individual assignment costs, we can formulate the entire assignment problem by:

$$
\begin{aligned}
\underset{x_{ct}, o_t}{\text{minimise}} \quad & \sum_t \left(\sum_c d_{ct} x_{ct} + \lambda o_t \right) & & \\
\text{subject to} \quad & \sum_c x_{ct} + o_t = 1 & \forall t \ , & \\
& \sum_t x_{ct} \geq 1 & \forall c \ , & \\
& x_{ct}, o_t \in \{0, 1\} & \forall c, t \ . &
\end{aligned}
\tag{4.1}
$$

By minimising the constrained objective function, we obtain the binary variables x_{ct} and o_t as solution of the assignment problem. The first type of constraints ensures that a target sample is either assigned to one class, i.e. $x_{ct} = 1$, or declared as outlier, i.e. $o_t = 1$. The second type of constraints ensures that at least one target sample is assigned to each class $c \in \mathcal{C}$. We use the constraint integer program package SCIP [Achterberg, 2009] to solve all proposed formulations.

As it is shown in Figure 4.2b, we label the targets also by the unknown class. Note that the unknown class combines all objects that are not of interest. Even if the unknowns in the source and target domain belong to different semantic classes, a target sample might be closer to the mean of all negatives than to any other positive class. In this case, we can confidentially label a target sample as unknown. In our experiments, we show that it makes not much difference if the unknown class is included in the unsupervised setting since the outlier handling discards target samples that are not close to the mean of negatives.

4.2.2 Semi-supervised Domain Adaptation

The unsupervised assignment problem naturally extends to a semi-supervised setting when a few target samples are annotated. In this case, we only have to extend the formulation (4.1) by additional constraints that enforce that the annotated target samples do not change the label, i.e.

$$x_{\hat{c}_t t} = 1 \qquad\qquad \forall (t, \hat{c}_t) \in \mathcal{L}, \tag{4.2}$$

where $\mathcal{L}$ denotes the set of labelled target samples and $\hat{c}_t$ the class label provided for target sample t. In order to exploit the labelled target samples better, one can use the neighbourhood structure in the source and target domain. While the constraints remain the same, the objective function (4.1) can be changed to

$$\sum_t \left(\sum_c x_{ct} \left(d_{ct} + \sum_{t' \in N_t} \sum_{c'} d_{cc'} x_{c't'} \right) + \lambda o_t \right), \tag{4.3}$$

where $d_{cc'} = \|S_c - S_{c'}\|_2^2$. While in (4.1) the cost of labelling a target sample t by the class c is given only by d_{ct}, a second term is added in (4.3). It is computed over all neighbours N_t of t and adds the distance between the classes in the source domain as additional cost if a neighbour is assigned to another class than the target sample t.

The objective function (4.3), however, becomes quadratic and therefore NP-hard to solve. Thus, we transform the *quadratic assignment problem* into a mixed 0-1 linear program using the Kaufman and Broeckx linearisation [Kaufman and Broeckx, 1978]. By substituting

$$w_{ct} = x_{ct} \left(\sum_{t' \in N_t} \sum_{c'} d_{cc'} x_{c't'} \right), \tag{4.4}$$

we derive to the linearised problem

$$\begin{aligned}
\underset{x_{ct},w_{ct},o_t}{\text{minimise}} \quad & \sum_t \left(\sum_c d_{ct}x_{ct} + \sum_c w_{ct} + \lambda o_t \right) & & \\
\text{subject to} \quad & \sum_c x_{ct} + o_t = 1 & & \forall t \ , \\
& \sum_t x_{ct} \geq 1 & & \forall c \ , \\
& a_{ct}x_{ct} + \sum_{t' \in N_t} \sum_{c'} d_{cc'}x_{c't'} - w_{ct} \leq a_{ct} & & \forall s,t \ , \\
& x_{ct}, o_t \in \{0,1\} & & \forall c,t \ , \\
& w_{ct} \geq 0 & & \forall c,t \ ,
\end{aligned} \tag{4.5}$$

where $a_{ct} = \sum_{t' \in N_t} \sum_{c'} d_{cc'}$.

4.2.3 Mapping

As illustrated in Figure 4.2, we iterate between solving the assignment problem, as described in Section 4.2.1 or 4.2.2, and estimating the mapping from the source domain to the target domain. We consider a linear transformation, which is represented by a matrix $W \in \mathbb{R}^{D \times D}$. We estimate W by minimising the following loss function:

$$f(W) = \frac{1}{2} \sum_t \sum_c x_{ct} \|WS_c - T_t\|_2^2 \ , \tag{4.6}$$

which can be written in matrix form:

$$f(W) = \frac{1}{2} ||WP_S - P_T||_F^2 \ . \tag{4.7}$$

The matrices P_S and $P_T \in \mathbb{R}^{D \times L}$ with $L = \sum_t \sum_c x_{ct}$ represent all assignments, where the columns denote the actual associations. The quadratic nature of the convex objective function may be seen as a linear least squares problem, which can be easily solved by any available QP solver. State-of-the-art features based on convolutional neural networks, however, are high dimensional and the number of target instances is usually very large. We use therefore non-linear optimisation [Svanberg, 2002, Johnson, 2007–2010] to optimise $f(W)$. The derivatives of (5.2) are given by

$$\frac{\partial f(W)}{\partial W} = W(P_S P_S^T) - P_T P_S^T \ . \tag{4.8}$$

If $L < D$, i.e. the number of samples, which have been assigned to a known class, is smaller than the dimensionality of the features, the optimisation also deals with an underdetermined linear least squares formulation. In this case, the solver converges to the matrix W with the smallest norm, which is still a valid solution.

After the transformation W is estimated, we map the source samples to the target domain. We therefore iterate the process of solving the assignment problem and estimating the mapping from the source domain to the target domain until it converges. After the approach has converged, we train linear SVMs in a one-vs-one setting on the transformed source samples. For the semi-supervised setting, we also include the annotated target samples $\mathcal{L}$ (4.2) to the training set. The linear SVMs are then used to obtain the final labelling of the target samples as illustrated in Figure 4.2d.

4.3 Experiments

We evaluate our method in the context of domain adaptation for image classification and action recognition. In this setting, the images or videos of the source domain are annotated by class labels and the goal is to classify the images or videos in the target domain. We report the accuracies for both unsupervised and semi-supervised scenarios, where target samples are unlabelled or partially labelled, respectively. For consistency, we use *libsvm* [Chang and Lin, 2011] since it has also been used in other works, e.g. [Fernando et al., 2013] and [Sun et al., 2015a]. We set the misclassification parameter $C = 0.001$ in all experiments, which allows for a soft margin optimisation that works best in such classification tasks [Fernando et al., 2013, Sun et al., 2015a].

4.3.1 Parameter Configuration

Our algorithm contains a few parameters that need to be defined. For the outlier rejection, we use

$$\lambda = \rho\big(\max_{t,c} d_{ct} + \min_{t,c} d_{ct}\big), \tag{4.9}$$

i.e. λ is adapted automatically based on the distances d_{ct} and ρ, which is set to 0.5 unless otherwise specified. While higher values of λ closer to the largest distance barely discard any outlier, lower values almost reject all assignments. We iterate the approach until the maximum number of 10 iterations is reached or if the distance

$$\sqrt{\sum_t \sum_c x_{ct} \, \|W_k S_c - T_t\|_2^2} \tag{4.10}$$

is below $\varepsilon = 0.01$, where W_k denotes the estimated transformation at iteration k. In practice, the process converges after 3-5 iterations.

4.3.2 Open Set Domain Adaptation

4.3.2.1 Office Dataset

We evaluate and compare our approach on the *Office* dataset [Saenko et al., 2010], which is the standard benchmark for domain adaptation with CNN features. It provides three different domains, namely *Amazon (A)*, *DSLR (D)* and *Webcam (W)*. While the *Amazon* dataset contains centred objects on white background, the other two comprise pictures taken in an office environment but with different quality levels. In total, there are 31 common classes for 6 source-target combinations. This means that there are 4 combinations with a considerable domain shift (A → D, A → W, D → A, W → A) and 2 with a minor domain shift (D → W, W → D). Following the standard protocol and for a fair comparison with the other methods, we extract feature vectors from the fully connected layer-7 (fc7) of the AlexNet model [Krizhevsky et al., 2012].

We introduce an open set protocol for this dataset by taking the 10 classes that are also common in the *Caltech* dataset [Gong et al., 2012] as shared classes. In alphabetical order, the classes 11-20 are used as unknowns in the source domain and 21-31 as unknowns in the target domain, i.e. the unknown classes in the source and target domain are not shared. For evaluation, each sample in the target domain needs to be correctly classified either by one of the 10 shared classes or as unknown. In order to compare with a closed setting (CS), we report the accuracy when source and target domain contain only samples of the 10 shared classes. Since OS is evaluated on all target samples, we also report the numbers when the accuracy is only measured on the same target samples as CS, i.e. only for the shared 10 classes. The latter protocol is denoted by OS*(10) and provides a direct comparison to CS(10).

Unsupervised domain adaptation. We firstly compare the accuracy of our method in the unsupervised set-up with state-of-the-art domain adaptation techniques embedded in the training of CNN models. DAN [Long et al., 2015] retrains the AlexNet model by freezing the first 3 convolutional layers, finetuning the last 2 and learning the weights from each fully connected layer by also minimising the discrepancy between both domains. RTN [Long et al., 2016] extends DAN by adding a residual transfer module that bridges the source and target classifiers. BP [Ganin and Lempitsky, 2015] trains a CNN for domain adaptation by a gradient reversal layer and minimises the domain loss jointly with the classification loss. For training, we use all samples per class as proposed in Gong et al. [2013a], which is the standard protocol for CNNs on this dataset. As proposed in Ganin and Lempitsky [2015], we use for all methods linear SVMs for classification instead of the soft-max layer for a fair comparison.

To analyse the formulations that are discussed in Section 4.2, we compare several variants: ATI (*Assign-and-Transform-Iteratively*) denotes our formulation in (4.1) assigning a source class to all target samples, i.e. $\lambda = \infty$. Then, ATI-λ includes the outlier rejection and ATI-λ-N_1 is the unsupervised version of the locality constrained formulation corresponding to (4.3) with 1 nearest neighbour. In addition, we denote LSVM as the linear SVMs trained on the source domain without any domain adaptation.

The results of these techniques using the described open set protocol are shown in Table 4.1. Our approach ATI improves over the baseline without domain adaptation (LSVM) by +6.8% for CS and +14.3% for OS. The improvement is larger for the combinations that have larger domain shifts, i.e. the combinations that include the *Amazon* dataset. We also observe that ATI outperforms all CNN-based domain adaptation methods for the closed (+2.2%) and open setting (+5.2%). It can also

	A→D			A→W		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)
LSVM	87.1	70.7	72.6	77.5	53.9	57.5
DAN	88.1	76.5	77.6	**90.5**	70.2	72.5
RTN	**93.0**	74.7	76.6	87.0	70.8	73.0
BP	91.9	77.3	78.3	89.2	73.8	75.9
ATI	92.4	78.2	78.8	85.1	**77.7**	**78.4**
ATI-λ	**93.0**	**79.2**	**79.8**	84.0	76.5	77.6
ATI-λ-N1	91.9	78.3	78.9	84.6	74.2	75.6

	D→A			D→W		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)
LSVM	79.4	40.0	45.1	97.9	87.5	88.5
DAN	83.4	53.5	57.0	96.1	87.5	88.4
RTN	82.8	53.8	57.2	97.9	88.1	89.0
BP	84.3	54.1	57.6	97.5	88.9	89.8
ATI	93.4	**70.0**	71.1	**98.5**	92.2	92.6
ATI-λ	**93.8**	**70.0**	**71.3**	**98.5**	93.2	93.5
ATI-λ-N1	93.3	65.6	67.8	97.9	**94.0**	**94.4**

	W→A			W→D			AVG.		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)	CS	OS*	OS
LSVM	80.0	44.9	49.2	**100**	96.5	96.6	87.0	65.6	68.3
DAN	84.9	58.5	60.8	**100**	97.5	98.3	90.5	74.0	75.8
RTN	85.1	60.2	62.4	**100**	98.3	**98.8**	91.0	74.3	76.2
BP	86.2	61.8	64.0	**100**	98.0	98.7	91.6	75.7	77.4
ATI	93.4	76.4	76.6	**100**	99.1	98.3	**93.8**	82.1	82.6
ATI-λ	**93.7**	**76.5**	**76.7**	**100**	99.2	98.3	93.7	**82.4**	**82.9**
ATI-λ-N1	93.4	71.6	72.4	**100**	**99.6**	**98.8**	93.5	80.6	81.3

Table 4.1: Open set domain adaptation on the unsupervised Office dataset with 10 shared classes (OS) using all samples per class [Gong et al., 2013a]. For comparison, results for closed set domain adaptation (CS) and modified open set (OS*) are reported.

be observed that the accuracy for the open set is lower than for the closed set for all methods, but that our method handles the open set protocol best. While ATI-λ does not obtain any considerable improvement compared to ATI in CS, the outlier rejection allows for an improvement in OS. The locality constrained formulation, ATI-λ-N_1, which we propose only for the semi-supervised setting, decreases the accuracy in the unsupervised setting.

The evolution of the percentage of correct assignments and the intermediate classification accuracies are shown in Table 4.2. The approach converges after two or three iterations. While the accuracy of the LSVMs that are trained on the transformed source samples increases with each iteration, the accuracy of the assignment can even decrease in some cases.

Additionally, we report accuracies of popular domain adaptation methods that are not related to deep learning. We report the results of methods that transform the data to a common low dimensional subspace, including Transfer Component Analysis (TCA) [Pan et al., 2009], Geodesic Flow Kernel (GFK) [Gong et al., 2012] and Subspace

	A→D		A→W		D→A		D→W		W→A		W→D	
	assign-λ	LSVM	assign-λ	LSVM	assign-λ	LSVM	assign-λ	LSVM	assign-λ	LSVM	assign-λ	LSVM
init		72.6		57.5		45.1		88.5		49.2		96.6
it 1	78.4	76.8	74.5	69.8	73.6	68.1	90.4	90.3	71.9	70.0	89.6	97.8
it 2	77.7	79.1	80.1	77.6	80.4	71.3	91.5	93.5	77.2	75.9	84.7	98.3
it 3	75.3	79.8							77.8	76.7		

Table 4.2: Evolution of the percentage of correct assignments (assign-λ) when taking into account the selected target samples and the average class accuracy of all target samples using linear SVMs (LSVM). The approach converges after 2 or 3 iterations.

alignment (SA) [Fernando et al., 2013]. In addition, we also include CORAL [Sun et al., 2015a], which whitens and recolours the source towards the target data. Following the standard protocol of Saenko et al. [2010], we take 20 samples per object class when *Amazon* is used as source domain, and 8 for *DSLR* or *Webcam*. As in the previous comparison with the CNN-based methods, we extract feature vectors from the last convolutional layer (fc7) from the AlexNet model [Krizhevsky et al., 2012]. Each evaluation is executed 5 times with random samples from the source domain. The average accuracy and standard deviation of the five runs are reported in Table 4.3. The results are similar to the protocol reported in Table 4.1. Our approach ATI outperforms the other methods both for CS and OS and the additional outlier handling (ATI-λ) does not improve the accuracy for the closed set but for the open set.

Impact of unknown class. The linear SVM that we employ in the open set protocol uses the unknown classes of the transformed source domain for the training. Since unknown object samples from the source domain are from different classes than the ones from the target domain, using an SVM that does not require any negative samples might be a better choice. Therefore, we compare the performance of a standard SVM classifier with a specific open set SVM (OS-SVM) [Scheirer et al., 2014], where only the 10 known classes are used for training. OS-SVM introduces an inclusion probability and labels target instances as unknown if this inclusion is not satisfied for any class. Table 4.4 compares the classification accuracies of both classifiers in the 6 domain shifts of the Office dataset. While the performance is comparable when no domain adaptation is applied, ATI-λ obtains significantly better accuracies when the learning includes negative instances.

As discussed in Section 4.2.1, the unknown class is also part of the labelling set $\mathcal{C}$ for the target samples. The labelled target samples are then used to estimate the mapping W (5.2). To evaluate the impact of including the unknown class, Table 4.5 compares the accuracy when the unknown class is not included in $\mathcal{C}$. Adding the unknown class improves the accuracy slightly since it enforces that the negative mean of the source is mapped to a negative sample in the target. The impact, however, is very small.

Additionally, we also analyse the impact of increasing the amount of unknown samples in both source and target domain on the configuration *Amazon* → *DSLR+Webcam*. Since the domain shift between *DSLR* and *Webcam* is close to zero (same scenario, but different cameras), they can be merged to get more unknown samples. Following the

	A→D			A→W		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)
LSVM	84.4±5.9	63.7±6.7	66.6±5.9	76.5±2.9	48.2±4.8	52.5±4.2
TCA	85.9±6.3	75.5±6.6	75.7±5.9	80.4±6.9	67.0±5.9	67.9±5.5
gfk	84.8±5.1	68.6±6.7	70.4±6.0	76.7±3.1	54.1±4.8	57.4±4.2
SA	84.0±3.4	71.5±5.9	72.6±5.3	76.6±2.8	57.4±4.2	60.1±3.7
CORAL	85.8±7.2	79.9±5.7	79.6±5.0	81.9±2.8	68.1±3.6	69.3±3.1
ATI	**91.4±1.3**	80.5±2.0	81.1±2.8	**86.1±1.1**	73.4±2.0	**75.3±1.7**
ATI-λ	91.1±2.1	**81.1±0.4**	**82.2±2.0**	85.5±2.1	**73.7±2.6**	**75.3±1.4**

	D→A			D→W		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)
LSVM	75.5±2.1	36.1±3.7	42.2±3.3	96.2±1.0	81.5±1.5	83.1±1.3
TCA	88.2±1.5	71.8±2.5	71.8±2.0	97.8±0.5	92.0±0.9	91.5±1.0
gfk	79.7±1.0	45.3±3.7	49.7±3.4	96.3±0.9	85.1±2.7	86.2±2.4
SA	81.7±0.7	52.5±3.0	55.8±2.7	96.3±0.8	86.8±2.5	87.7±2.3
CORAL	89.6±1.0	66.6±2.8	68.2±2.5	97.2±0.7	91.1±1.7	91.4±1.5
ATI	93.5±0.3	69.8±1.4	70.8±2.1	97.3±0.5	89.6±2.1	90.3±1.8
ATI-λ	**93.9±0.4**	**71.1±0.9**	**72.0±0.5**	**97.5±1.1**	**92.1±1.3**	**92.5±0.7**

	W→A			W→D			AVG.		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)	CS	OS*	OS
LSVM	72.5±2.7	34.3±4.9	39.9±4.4	99.1±0.5	89.8±1.5	90.5±1.3	84.1	58.9	62.5
TCA	85.5±3.3	68.1±5.1	68.6±4.6	98.8±0.9	94.1±2.9	93.6±2.6	89.5	78.1	78.2
gfk	75.0±2.9	43.2±5.1	47.6±4.6	99.0±0.5	92.0±1.5	92.2±1.4	85.2	64.7	67.3
SA	76.5±3.2	49.7±5.1	53.0±4.6	98.8±0.7	92.4±2.9	92.4±2.8	85.7	68.4	70.3
CORAL	86.9±1.9	63.9±4.9	65.6±4.3	**99.2±0.7**	96.0±2.1	95.0±2.0	90.1	77.6	78.2
ATI	92.2±1.1	75.1±1.7	76.0±2.0	98.9±1.3	95.5±2.3	95.4±2.1	**93.2**	80.7	81.5
ATI-λ	**92.4±1.1**	**75.4±1.8**	**76.4±1.8**	98.9±1.3	**96.5±2.1**	**95.8±1.8**	**93.2**	**81.5**	**82.3**

Table 4.3: Open set domain adaptation on the unsupervised Office dataset with 10 shared classes (OS). We report the average and the standard deviation using a subset of samples per class in 5 random splits [Saenko et al., 2010]. For comparison, results for closed set domain adaptation (CS) and modified open set (OS*) are reported.

described protocol, we take 20 samples per known category, also in this case for the target domain, and we randomly increase the number of unknown samples from 20 to 400 in both domains at the same time. As shown in Table 4.6, that reports the mean accuracies of 5 random splits, adding more unknown samples decreases the accuracy if domain adaptation is not used (LSVM), but also for the domain adaptation method CORAL [Sun et al., 2015a]. This is expected since the unknowns are from different classes and the impact of the unknowns compared to the samples from the shared classes increases. Our method handles such an increase and the accuracies remain stable between 80.3% and 82.5%.

Subsampling of target samples. In order to evaluate the robustness of our method when having a reduced amount of target samples for domain adaptation, we subsample the target data. Figure 4.3 shows the results for ATI-λ on the 6 domain shifts of the Office dataset with the standard open set protocol (OS). We vary the number of target

	A→D		A→W		D→A		D→W		W→A		W→D		AVG.	
	OS-SVM	LSVM	OS-SVM	LSVM	OS-SVM	LSVM	OS-SVM	LSVM	OS-SVM	LSVM	OS-SVM	LSVM	OS-SVM	LSVM
No Adap.	67.5	72.6	58.4	57.5	54.8	45.1	80.0	88.5	55.3	49.2	94.0	96.6	68.3	68.3
ATI-λ	72.0	**79.8**	65.3	**77.6**	66.4	**71.3**	82.2	**93.5**	71.6	**76.7**	92.7	**98.3**	75.0	**82.9**

Table 4.4: Comparison of a standard linear SVM (LSVM) with a specific open set SVM (OS-SVM) [Scheirer et al., 2013] on the unsupervised Office dataset with 10 shared classes using all samples per class [Gong et al., 2013a].

	A→D	A→W	D→A	D→W	W→A	W→D	AVG.
	OS(10)						
ATI-λ ($\mathcal{C}$ w/o unknown)	79.0	77.1	70.5	93.4	75.8	98.2	82.3
ATI-λ ($\mathcal{C}$ with unknown)	**79.8**	**77.6**	**71.3**	**93.5**	**76.7**	**98.3**	**82.9**

Table 4.5: Impact of including the unknown class to the set of classes $\mathcal{C}$. The evaluation is performed on the unsupervised Office dataset with 10 shared classes using all samples per class [Gong et al., 2013a].

Amazon → DSLR+Webcam								
number of unknowns	20	40	60	80	100	200	300	400
unknown / known	0.10	0.20	0.30	0.40	0.50	1.00	1.50	2.00
LSVM	74.2	70.0	66.2	63.4	61.4	53.9	50.4	48.2
CORAL	77.2	76.4	76.2	74.8	73.7	71.5	70.8	69.7
ATI-λ	80.3	82.4	81.2	81.7	82.5	80.9	80.7	81.9

Table 4.6: Impact of increasing the amount of unknown samples in the domain shift *Amazon → DSLR+Webcam* on the unsupervised Office dataset with 10 shared classes using 20 random samples per known class in both domains.

samples from 50 to the total number of instances. For a fixed number of target samples, we randomly sample 5 times from the target data and plot the lowest, highest and average accuracy of the 5 runs. The accuracy is always measured on the whole target dataset. The results show that between 300 and 400 target instances are sufficient to achieve similar accuracies than our method with all target samples. When the domain shifts are smaller, e.g. $D \rightarrow W$ and $W \rightarrow D$, even less target samples are required.

Scalability analysis of target samples. The number of sampled target samples has an impact on the execution time of the assignment and the transformation steps of the iterative process. Therefore, we also test the scalability of the two steps of our method with respect to the number of target samples. The average execution times of both techniques in the domain shift *Amazon* → *DSLR+Webcam* for all the random splits and unknown sets of the previous evaluation are shown in Figure 4.4. We observe that the assignment problem takes less than a second to be solved for any size of target data from the evaluated settings. Most of the computation time is required for estimating the transformation W, which requires at least 120 seconds. The computation time of this step, however, increases only moderately with respect to the number of target samples.

Impact of parameter ρ. The cost that determines whether a target sample is considered as outlier during the assignment process is defined by λ (4.9), which is based on the current minimum and maximum distance between the source clusters and target samples. Thus, λ is updated at each iteration. The value of λ, however, also depends on the parameter ρ. For all experiments, we use $\rho = 0.5$ as default value, aiming for a moderate outlier rejection. Figure 4.5 shows the impact of ρ on the accuracy. Using $\rho = 0.5$, which rejects around 10-20% of the target samples, achieves the best results in 5 out of the 6 domain shifts on the Office dataset. When ρ gets closer to 0 the accuracy drops substantially since too many samples are discarded.

Impact of constraint $\sum_t x_{ct} \geq 1$. Our formulation in (4.1) ensures that at least one target sample is assigned to an object category. Therefore, all classes contribute to the estimation of the transformation matrix W. In order to measure its impact on the adaptation problem, we run experiments with $\sum_t x_{ct} \geq 1$ and without the constraint, i.e. when a class might not be assigned to any target sample at all. As illustrated in Figure 4.6, the inclusion of this constraint provides higher accuracies when $\rho < 0.3$. For greater values of ρ, the constraint can be omitted since it does not influence the accuracy.

Impact of wrong assignments. During the iterative process of our method, wrong assignments take part in the optimisation of W, introducing false associations between the source and the target domain that negatively affect the final transformation. A general assumption in our method is that the correct assignments largely compensate the wrong ones and, thus, the transformed source data allows for better classification accuracies in the target domain. Therefore, we artificially generate assignments in the first iteration by assigning a random subset of target samples to the correct class in the source domain and the remaining target samples to random classes. We then run our approach without any additional modifications until it converges. We report

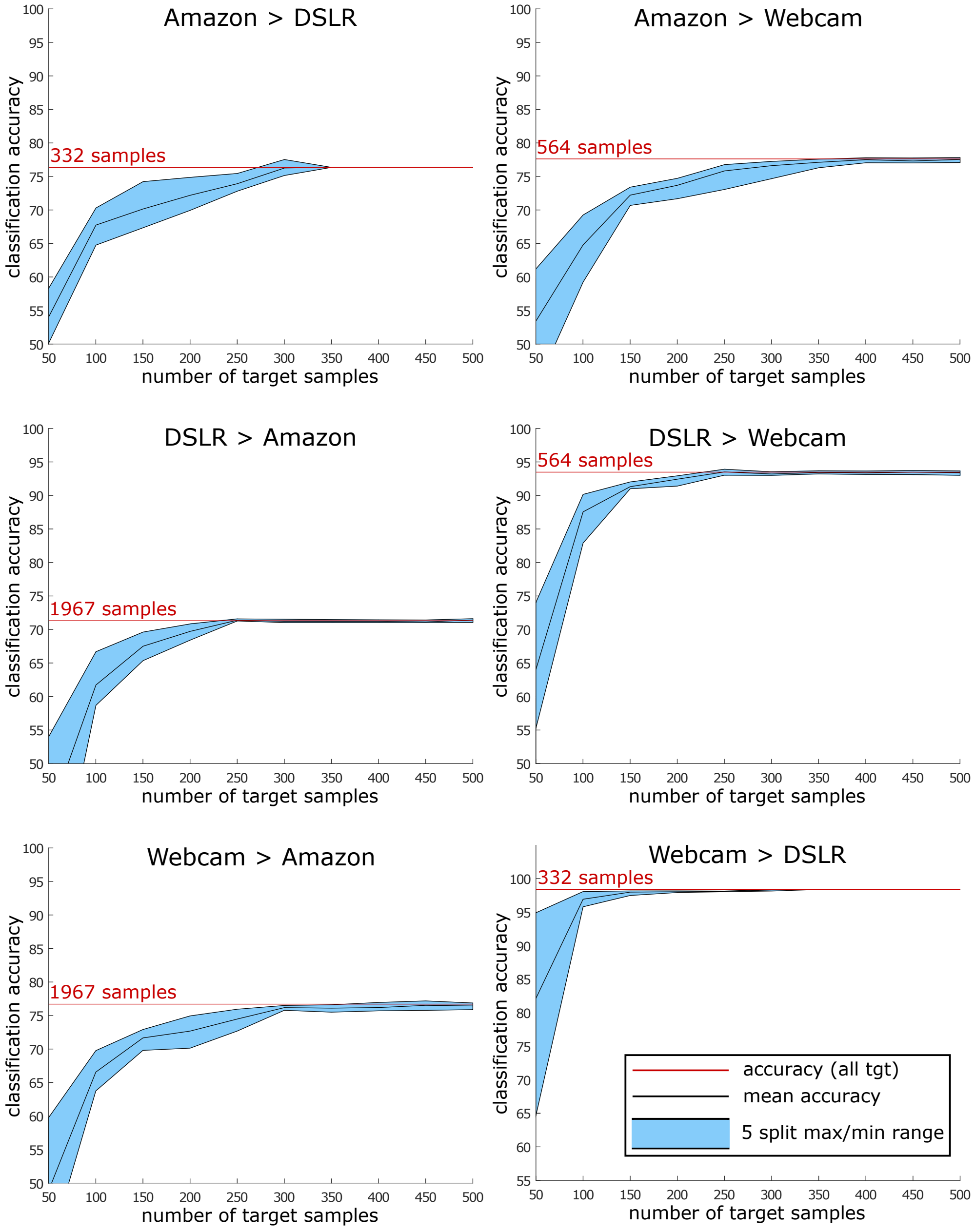


Figure 4.3: Impact of using a random subset of target samples. The blue region shows the difference between the best and worst result of the 5 randomly sampled subsets for a given number of target samples and the black line within the region is the mean accuracy of the 5 subsets. The red line indicates the classification accuracy when using all target samples. The results are reported for ATI-λ using the open set protocol on the unsupervised Office dataset with 10 shared classes using all samples per class.

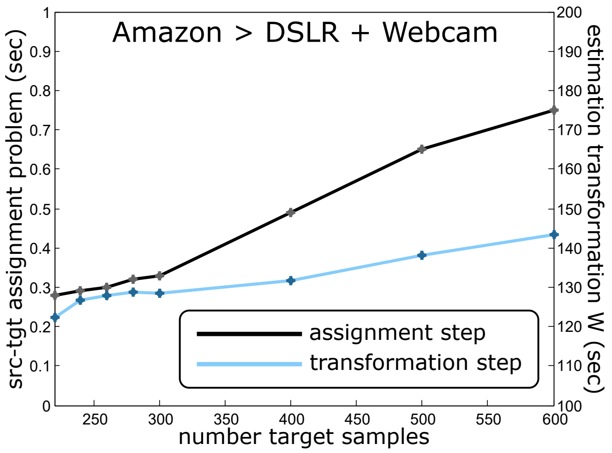


Figure 4.4: Execution time in seconds for the assignment and transformation estimation steps of a single iteration with respect to the number of target samples.

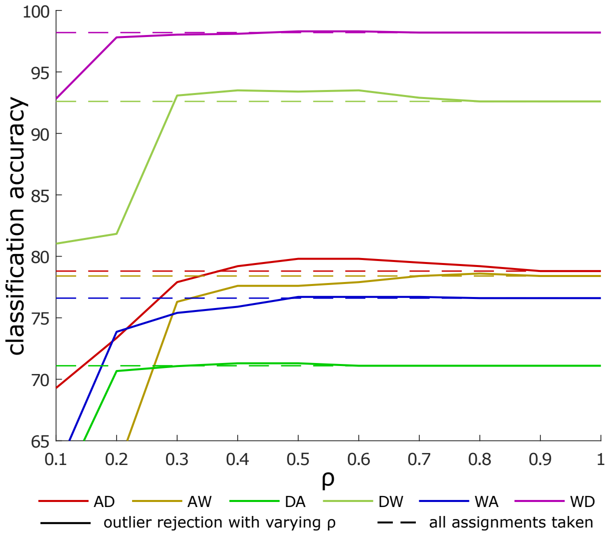


Figure 4.5: Impact of varying ρ in order to restrict the outlier handler λ. An intermediate value, i.e. $\rho = 0.5$, tends to obtain the best accuracies. We follow the presented open set protocol on the unsupervised Office dataset with 10 shared classes using all samples per class [Gong et al., 2013a].

in Table 4.7 the average percentage of correct assignments of 5 random splits for the domain shift *Amazon* → *DSLR+Webcam* with 400 unknown samples. While the first iteration represents the accuracy of correct and random assignments that we generate, the last row shows the accuracies after the approach has converged. As it can be observed, the approach ends in a local optimum, but the accuracies increase for all cases except if we initialise the approach with 100% correct assignments. It is expected that the assignment accuracy does not remain at 100% since the image manifolds are not perfectly linearised and even for the best estimate of W wrong assignments can occur.

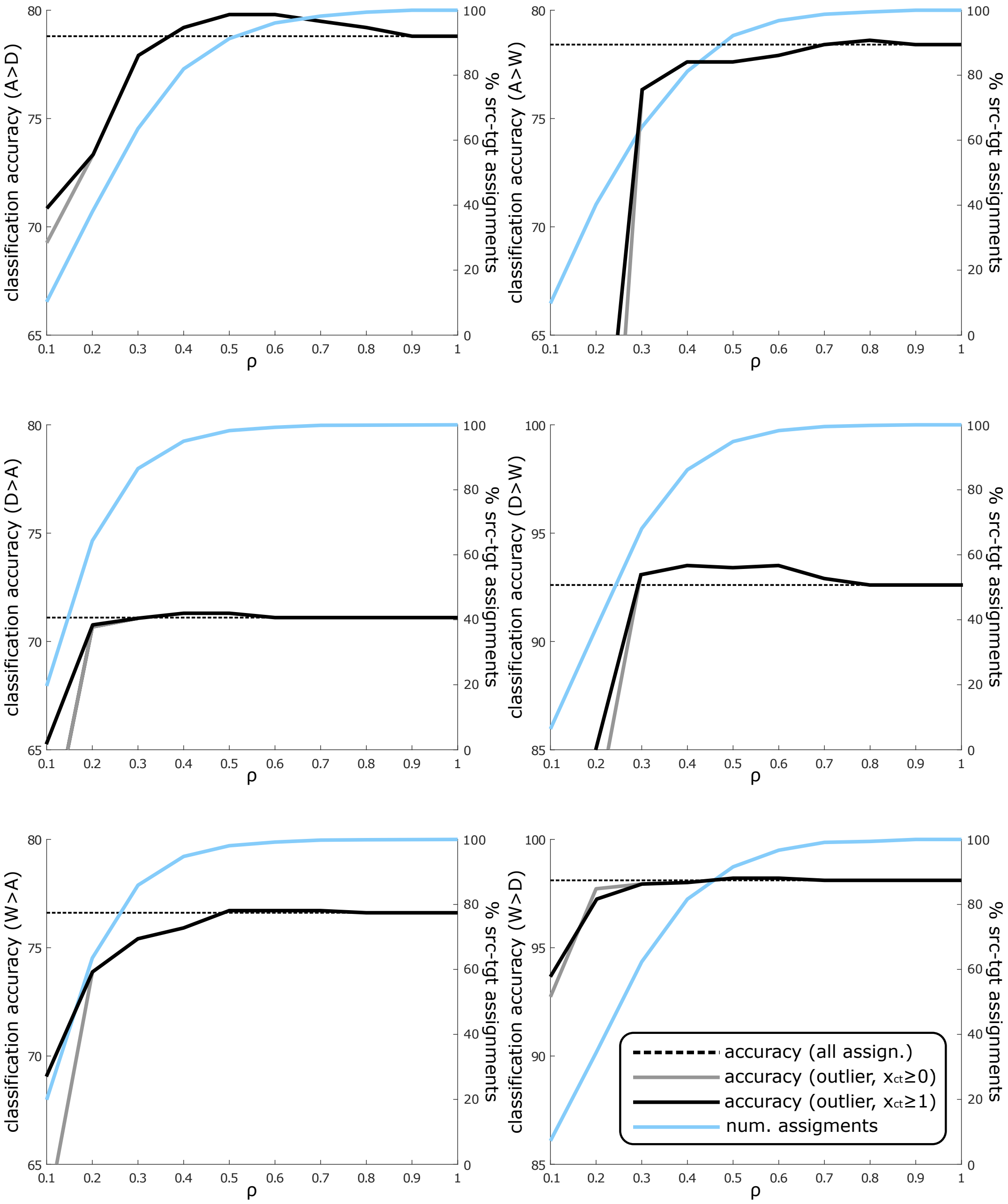


Figure 4.6: The black and grey curves show the classification accuracies for varying values of ρ when including or not the constraint $\sum_t x_{ct} \geq 1$, respectively. $\rho = 0.5$ obtains the best accuracies in 5 out of 6 domain shifts. The blue curve shows the percentage of selected assignments to compute the transformation matrix W in the first iteration. The results are reported for ATI-λ using the open set protocol on the unsupervised Office dataset with 10 shared classes using all samples per class.

Amazon → *DSLR+Webcam* (400 unknown samples)											
%gt (+rnd)	10	20	30	40	50	60	70	80	90	100	*std*
iteration 1	18.2	27.0	36.1	45.2	54.3	63.5	72.7	81.7	90.7	100.0	*85.1*
final	24.4	40.1	54.7	65.4	72.8	79.2	83.6	88.8	93.1	96.7	*88.6*

Table 4.7: Impact of limiting the amount of correct assignments in the first iteration. We report the average percentage of correct assignments over 5 random splits and increase the percentage of correctly selected assignments from 10% to 100%, leaving the rest randomly selected. The last column shows the percentage of correct assignments of the method without modifying the initial assignments.

Semi-supervised domain adaptation. We also evaluate our approach for open set domain adaptation on the *Office* dataset in its semi-supervised setting. Applying again the standard protocol of Saenko et al. [2010] with the subset of source samples, we also take 3 labelled target samples per class and leave the rest unlabelled. We compare our method with the deep learning method MMD [Tzeng et al., 2014]. As baselines, we report the accuracy for the linear SVMs without domain adaptation (LSVM) when they are trained only on the source samples (s), only on the annotated target samples (t) or on both (st). As expected, the baseline trained on both performs best as shown in Table 4.8. Our approach ATI outperforms the baseline and the CNN approach [Tzeng et al., 2014]. As in the unsupervised case, the improvement compared to the CNN approach is larger for the open set (+4.8%) than for the closed set (+2.2%). While the locality constrained formulation, ATI-λ-N, decreased the accuracy for the unsupervised setting, it improves the accuracy for the semi-supervised case since the formulation enforces that neighbours of the target samples are assigned to the same class. The results with one (ATI-λ-$N1$) or two neighbours (ATI-λ-$N2$) are similar.

4.3.2.2 Dense Cross-Dataset Analysis

In order to measure the performance of our method and the open set protocol across popular datasets with more intra-class variation, we also conduct experiments on the *dense* set-up of the *Testbed for Cross-Dataset Analysis* [Tommasi and Tuytelaars, 2014]. This protocol provides 40 classes from 4 well known datasets, *Bing (B)*, *Caltech256 (C)*, *ImageNet (I)* and *Sun (S)*. While the samples from the first 3 datasets are mostly centred and without occlusions, *Sun* becomes more challenging due to its collection of object class instances from cluttered scenes. As for the Office dataset, we take the first 10 classes as shared classes, the classes 11-25 are used as unknowns in the source domain and the classes 26-40 as unknowns in the target domain. We use the provided DeCAF features (DeCAF7). Following the unsupervised protocol described by Tommasi et al. [2015], we take 50 source samples per class for training and we test on 30 target images per class for all datasets, except *Sun*, where we take 20 samples per class.

The results reported in Table 4.9 are consistent with the Office dataset. ATI outperforms the baseline and the other methods by +4.1% for the closed set and by +5.3% for the open set. ATI-λ obtains the best accuracies for the open set.

	A→D			A→W		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)
LSVM (s)	85.8±3.2	62.1±7.9	65.9±6.2	76.4±2.1	45.7±5.0	50.4±4.5
LSVM (t)	92.3±3.9	68.2±5.2	71.1±4.7	91.5±4.9	59.6±3.7	63.2±3.4
LSVM (st)	95.7±1.3	82.5±3.0	84.0±2.6	92.4±1.8	72.5±3.7	74.8±3.4
MMD	94.1±2.3	86.1±2.3	86.8±2.2	92.4±2.8	76.4±1.5	78.3±1.3
ATI	95.4±1.3	89.0±1.4	89.7±1.3	95.9±1.3	84.0±1.7	85.1±1.5
ATI-λ	97.1±1.1	**89.5±1.4**	90.2±1.3	96.1±2.0	84.1±1.8	85.2±1.5
ATI-λ-N1	97.6±1.0	**89.5±1.3**	**90.3±1.2**	**96.4±1.7**	**84.4±3.6**	**85.5±1.5**
ATI-λ-N2	**97.9±1.4**	89.4±1.2	90.1±1.0	92.8±1.6	84.3±2.4	85.4±1.5

	D→A			D→W		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)
LSVM (s)	85.2±1.7	40.3±4.3	45.2±3.8	97.2±0.7	81.4±2.4	83.0±2.2
LSVM (t)	88.7±2.2	52.8±6.0	57.0±5.5	91.5±4.9	59.6±3.7	63.2±3.4
LSVM (st)	91.9±0.7	68.7±2.5	71.2±2.3	98.7±0.9	87.3±2.3	88.5±2.1
MMD	90.2±1.8	69.0±3.4	71.3±3.0	98.5±1.0	85.5±1.6	86.7±1.4
ATI	**93.5±0.2**	74.4±2.7	76.1±2.5	98.7±0.7	91.6±1.7	92.4±1.5
ATI-λ	**93.5±0.2**	74.4±2.5	76.2±2.3	98.7±0.8	91.6±1.7	92.4±1.5
ATI-λ-N1	93.4±0.2	74.6±2.5	76.4±2.3	98.9±0.5	92.0±1.6	92.7±1.5
ATI-λ-N2	**93.5±0.1**	**74.9±2.3**	**76.7±2.1**	**99.3±0.5**	**92.2±1.9**	**92.9±1.7**

	W→A			W→D			AVG.		
	CS (10)	OS* (10)	OS (10)	CS (10)	OS* (10)	OS (10)	CS	OS*	OS
LSVM (s)	78.8±2.9	32.4±3.8	38.2±3.4	99.5±0.3	88.7±2.2	89.6±1.9	87.1	58.4	62.0
LSVM (t)	88.7±2.2	52.8±6.0	57.0±5.5	92.3±3.9	68.2±5.2	71.1±4.7	90.9	60.2	63.8
LSVM (st)	90.8±1.3	66.2±4.4	69.0±4.1	99.4±0.7	93.5±2.7	94.0±2.5	94.8	78.4	80.3
MMD	89.1±3.2	65.1±3.8	67.8±3.4	98.2±1.4	93.9±2.9	94.4±2.7	93.8	79.3	80.9
ATI	**93.0±0.5**	71.3±4.6	74.3±4.3	99.3±0.6	96.3±1.8	96.6±1.7	96.0	84.4	85.7
ATI-λ	**93.0±0.5**	71.5±4.8	73.6±4.4	**99.5±0.6**	96.3±1.8	96.6±1.7	96.3	84.6	85.7
ATI-λ-N1	**93.0±0.6**	72.2±4.5	74.2±4.1	99.3±0.6	**96.7±2.1**	**97.0±1.9**	96.4	**84.9**	**86.0**
ATI-λ-N2	**93.0±0.6**	**72.8±4.2**	**74.8±3.9**	99.3±0.6	95.5±2.2	95.9±2.0	**96.6**	84.8	**86.0**

Table 4.8: Open set domain adaptation on the semi-supervised Office dataset with 10 shared classes (OS). We report the average and the standard deviation using a subset of samples per class in 5 random splits [Saenko et al., 2010].

4.3.2.3 Sparse Cross-Dataset Analysis

We also introduce an open set evaluation using the *sparse* set-up from Tommasi and Tuytelaars [2014] with the datasets *Caltech101 (C)*, *Pascal07 (P)* and *Office (O)*. These datasets are quite unbalanced and offer distinctive characteristics: *Office* contains centred class instances with barely any background (17 classes, 2300 samples in total, 68-283 samples per class), *Caltech101* allows for more class variety (35 classes, 5545 samples in total, 35-870 samples per class) and *Pascal07* gathers more realistic scenes with partially occluded objects in various image locations (16 classes, 12219 samples in total, 193-4015 samples per class). For each domain shift, we take all samples of the shared classes and consider all other samples as unknowns. Table 4.10 summarises the amount of shared classes for each shift and the percentage of unknown target samples, which varies from 30% to 90%.

	B→C		B→I		B→S	
	CS (10)	OS (10)	CS (10)	OS (10)	CS (10)	OS (10)
LSVM	82.4±2.4	66.6±4.0	75.1±0.4	59.0±2.7	43.0±2.0	24.2±3.0
TCA	74.9±3.0	62.8±3.8	68.4±4.0	56.6±4.5	38.3±1.7	29.6±4.2
gfk	82.0±2.2	66.2±4.0	74.3±1.0	58.3±3.1	42.2±1.4	23.8±2.0
SA	81.1±1.8	66.0±3.4	73.9±0.9	57.8±3.2	41.9±2.4	24.3±2.6
CORAL	80.1±3.5	68.8±3.3	73.7±2.0	60.9±2.6	42.2±2.4	27.2±3.9
ATI	86.3±1.6	**71.4±1.8**	80.1±0.7	68.0±1.9	**49.2±3.2**	36.8±1.2
ATI-λ	**86.7±1.3**	**71.4±2.3**	**80.6±2.4**	**69.0±2.8**	48.6±2.5	**37.4±2.6**

	C→B		C→I		C→S	
	CS (10)	OS (10)	CS (10)	OS (10)	CS (10)	OS (10)
LSVM	53.5±2.1	40.1±1.9	76.9±4.3	62.5±1.2	46.3±2.7	28.2±1.4
TCA	49.2±1.1	38.9±1.9	73.1±3.6	60.2±1.4	45.9±3.6	29.7±1.6
gfk	53.2±2.6	40.2±1.8	77.1±3.3	62.2±1.5	46.2±3.0	28.5±1.0
SA	53.4±2.5	40.3±1.7	77.3±4.2	62.5±.8	46.1±3.3	29.0±1.5
CORAL	53.6±2.9	40.7±1.5	78.2±5.1	64.0±2.6	48.2±3.9	31.4±0.8
ATI	53.2±3.4	45.4±3.4	81.7±3.7	66.7±4.2	52.0±3.4	35.8±1.8
ATI-λ	**54.2±1.9**	**45.7±3.0**	**82.2±3.7**	**67.9±4.2**	**53.1±2.8**	**37.5±2.7**

	I→B		I→C		I→S	
	CS (10)	OS (10)	CS (10)	OS (10)	CS (10)	OS (10)
LSVM	**59.1±2.0**	42.7±2.0	86.2±2.6	73.3±3.9	50.1±4.0	32.1±3.2
TCA	56.1±3.8	40.9±2.9	83.4±3.2	68.6±1.8	49.3±2.6	34.5±3.8
gfk	58.7±1.9	42.6±2.4	86.1±2.7	73.3±3.6	49.5±3.6	32.7±3.6
SA	58.7±1.8	43.1±1.6	85.9±2.9	72.8±3.1	50.0±3.6	32.2±3.7
CORAL	58.5±2.7	44.6±2.5	85.8±1.5	74.5±3.4	49.5±4.8	35.4±4.4
ATI	57.9±1.9	**48.8±2.3**	89.3±2.2	77.1±2.6	55.0±5.0	42.2±4.0
ATI-λ	58.6±1.4	48.7±1.8	**89.7±2.3**	**77.5±2.2**	**55.3±4.3**	**43.4±4.8**

	S→B		S→C		S→I		AVG.	
	CS (10)	OS (10)	CS (10)	OS (10)	CS (10)	OS (10)	CS (10)	OS (10)
LSVM	33.1±1.7	16.4±1.1	53.1±2.6	27.9±2.9	52.3±1.8	25.2±0.5	59.3	41.5
TCA	30.6±1.3	19.4±2.1	47.5±3.5	32.0±3.9	45.2±1.9	31.1±4.6	55.2	42.0
gfk	33.3±1.4	16.9±1.5	53.1±3.0	28.6±3.8	52.5±2.0	26.4±1.1	59.0	41.6
SA	34.2±1.1	17.5±1.6	52.5±3.2	29.2±4.2	52.6±2.4	27.1±1.3	59.0	41.1
CORAL	32.9±1.6	18.7±1.2	52.1±2.8	33.6±5.3	52.9±1.8	31.3±1.3	59.0	44.2
ATI	**34.9±2.6**	22.8±3.1	59.8±1.3	46.9±2.5	**60.8±3.4**	32.9±2.2	63.4	49.5
ATI-λ	34.1±2.4	**23.2±3.2**	**60.2±2.7**	**47.3±2.9**	60.3±2.4	**33.0±1.1**	**63.6**	**50.2**

Table 4.9: Unsupervised open set domain adaptation on the Testbed dataset (dense setting) with 10 shared classes (OS). In addition, the results for closed set domain adaptation (CS) are reported for comparison.

Unsupervised domain adaptation. For the unsupervised experiment, we conduct a single run for each domain shift using all source and unlabelled target samples. The results are reported in Table 4.10. ATI outperforms the baseline and the other methods by +5.3% for this highly unbalanced open set protocol. ATI-λ improves the accuracy of ATI slightly.

	C→O	C→P	O→C	O→P	P→C	P→O	AVG.
shared classes	8	7	8	4	7	4	
unknown / all (t)	0.52	0.30	0.90	0.81	0.54	0.78	
LSVM	46.3	36.1	60.8	29.7	78.8	70.1	53.6
TCA	45.2	33.8	58.1	31.1	63.4	61.1	48.8
gfk	46.4	36.2	61.0	29.7	79.1	**72.6**	54.2
SA	46.4	36.8	61.1	30.2	79.8	71.1	54.2
CORAL	48.0	35.9	60.2	29.1	78.9	68.8	53.5
ATI	**51.6**	**52.1**	63.1	38.8	80.6	70.9	59.5
ATI-λ	51.5	52.0	**63.4**	**39.1**	**81.1**	71.1	**59.7**

Table 4.10: Unsupervised open set domain adaptation on the sparse set-up from Tommasi and Tuytelaars [2014].

	C→O	C→P	O→C	O→P	P→C	P→O	AVG.
LSVM (s)	46.5±0.1	36.2±0.1	60.8±0.3	29.7±0.0	79.5±0.3	73.5±0.7	54.4
LSVM (t)	53.1±3.7	44.6±2.1	73.7±1.5	40.5±3.0	81.1±2.5	70.5±4.3	60.6
LSVM (st)	56.0±1.3	44.5±1.2	68.9±1.1	40.9±2.2	80.9±0.6	76.7±0.3	61.3
ATI	59.6±1.2	55.2±1.3	75.8±1.2	45.2±1.4	81.6±0.2	**77.1±0.8**	65.8
ATI-λ	60.3±1.2	56.0±1.2	75.8±1.1	**45.8±1.2**	81.8±0.2	76.9±1.3	66.1
ATI-λ-N1	**60.7±1.2**	**56.3±1.2**	**76.7±1.6**	**45.8±1.4**	**82.0±0.4**	76.7±1.1	**66.4**

Table 4.11: Semi-supervised open set domain adaptation on the sparse set-up from Tommasi and Tuytelaars [2014] with 3 labelled target samples per shared class.

Semi-supervised domain adaptation. In order to evaluate the semi-supervised setting, we take all source samples and 3 annotated target samples per shared class as it is done in the semi-supervised setting for the Office dataset [Saenko et al., 2010]. The average and standard deviation over 5 random splits are reported in Table 4.11. While ATI improves over the baseline trained on the source and target samples together (st) by +4.5%, ATI-λ and the locality constraints with one neighbour boost the performance further. ATI-λ-N_1 improves the accuracy of the baseline by +5.1%.

4.3.2.4 Action Recognition

We extend the applicability of our technique to the field of action recognition in video sequences. We introduce an open set domain adaptation protocol between the *Kinetics Human Action Video Dataset* [Kay et al., 2017] (Kinetics) and the *UCF101 Action Recognition Dataset* [Soomro et al., 2012] (UCF101). The Kinects dataset is used as source domain and contains a total of 400 human action classes. The UCF101 dataset serves as target domain including 101 action categories, mainly of sports events. Since the labels of the same action differ between the datasets, e.g. *massaging persons head* (Kinetics) and *head massage* (UCF101), we manually map the class labels between the datasets. Additionally, we also merge all action classes in one dataset if they correspond to a single class in the other dataset, e.g. *dribbling basketball*, *playing basketball*, *shooting basketball* (Kinetics) are merged and associated to *basketball* (UCF101). We

Kinetics → UCF101						
LSVM	TCA	gkf	SA	CORAL	ATI	ATI-λ
64.9	71.2	64.9	65.1	69.4	76.6	**76.9**

Table 4.12: Unsupervised open set domain adaptation for action recognition.

Kinetics → UCF101			
LSVM (st)	ATI	ATI-λ	ATI-λ-N1
73.5±0.5	84.1±0.7	84.2±0.8	**84.5±0.6**

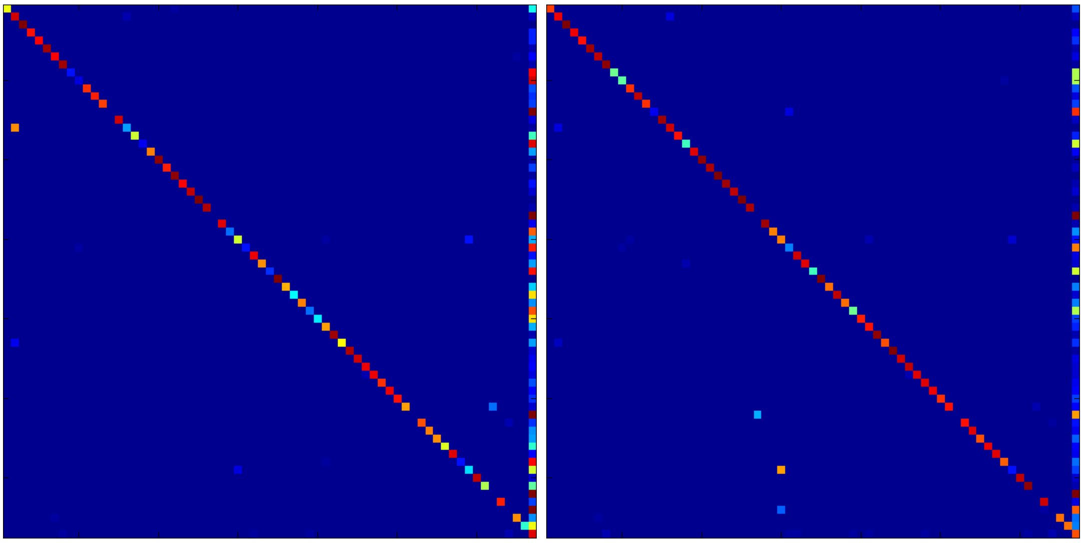
Table 4.13: Semi-supervised open set domain adaptation for action recognition.

finally obtain an open set protocol with 66 shared action classes, with 391 actions from Kinetics and 97 from UCF101. The list of shared classes, as well as all unrelated categories between both datasets, are given in Table A.1 and Table A.2 of Appendix A, respectively. The list also includes what similar actions are clustered for each common class.

For action recognition, we use the features extracted from the 5c layer of the spatial and temporal stream of the I3D model [Carreira and Zisserman, 2017], which is pre-trained on Kinetics [Kay et al., 2017]. We forward the complete video sequences through the spatial and temporal stream of I3D [Carreira and Zisserman, 2017] and the 5c layer of each stream provides an $7 \times 7 \times 1024$ output for a temporal fragment. We then apply spatial average pooling using a 7×7 kernel and average over time to obtain a 1024-dimensional feature vector from both the spatial and temporal stream of the I3D model [Carreira and Zisserman, 2017]. Finally, the feature vectors from the spatial and temporal streams are concatenated to get a single 2048-dimensional feature vector per video sequence.

Unsupervised domain adaptation. In the unsupervised setting, we evaluate our method by taking all source samples in a single run. Table 4.12 shows that the proposed approach outperforms the baseline and other approaches. ATI-λ achieves the highest accuracy and improves the accuracy by +12.0% compared to LSVM. The resulting confusion matrices of LSVM and ATI-λ are shown in Figure 4.7. LSVM misclassifies many instances of shared classes in the target domain as unknown instances (last column of confusion matrix), which is a well-known problem for open set recognition. Although ATI-λ does not resolve this problem completely, it reduces this effect substantially.

Semi-supervised domain adaptation. We extend the unsupervised protocol to evaluate our method on a semi-supervised setting by labelling 3 target samples per shared class. We report the average accuracies of 5 random splits in Table 4.13. Like in the previous semi-supervised experiments, ATI-λ-N1 obtains the best classification accuracies, outperforming the baseline without adaptation, LSVM (st), by +11.0%.



(a) No adaptation (LSVM): 64.9%

(b) ATI-λ: 76.9%

Figure 4.7: Confusion matrices without (a) and with adaptation (b) for the 66 shared classes and unknowns (last row and last column) for the unsupervised open set protocol for *Kinetics* [Kay et al., 2017] and *UCF101* [Soomro et al., 2012]. Many instances of the shared classes in the target domain are wrongly classified as unknown instances (last column) if domain adaptation is not applied. The figure is best viewed by zooming in.

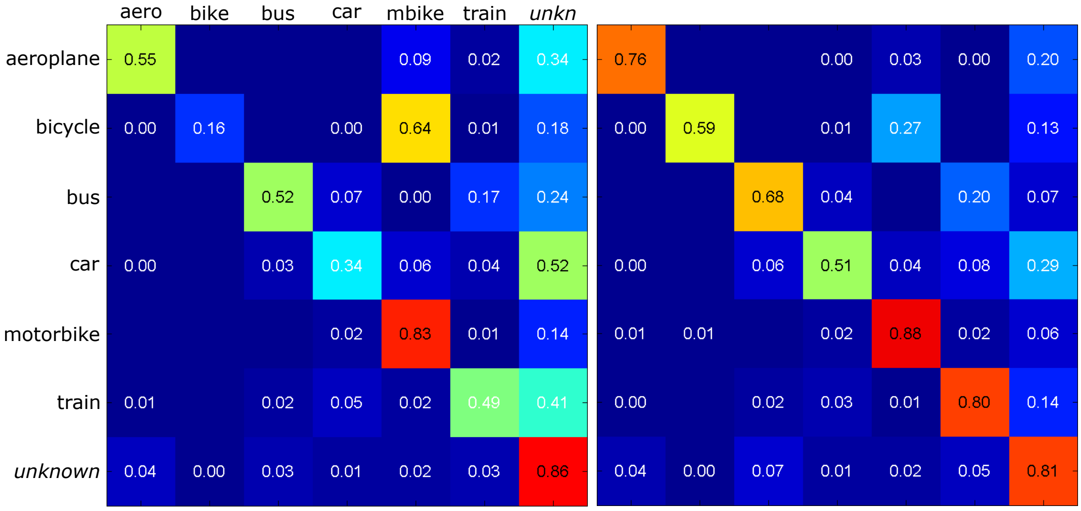
4.3.2.5 Synthetic Data

We also introduce another open set protocol with a domain shift between synthetic and real data. In this case, we take 152,397 synthetic images of the VISDA'17 challenge [Peng et al., 2017] as source domain and 5970 instances of real images from the training data of the Pascal3D dataset [Xiang et al., 2014] as target domain. Since both datasets contain several types of vehicles, we obtain 6 shared classes, namely, *aeroplane*, *bicycle*, *bus*, *car*, *motorbike* and *train*, within the 12 categories of each dataset. Following the protocol used in Section 4.3.2.1, we extract deep features from the fully connected layer-7 (fc7) from the AlexNet model [Krizhevsky et al., 2012] with 4096 dimensions. In addition, we also extract features from the VGG-16 model [Simonyan and Zisserman, 2014] to evaluate the impact of using deeper features.

The results of the classification task are shown in Table 4.14. The proposed domain adaptation method achieves the best results for both types of CNN features. When we compare the performance of the deep features from AlexNet and VGG-16, the accuracy of the baseline (LSVM) increases by +5.6% when using the deeper network VGG-16 instead of AlexNet. ATI and ATI-λ, however, benefit even more from the deeper architecture. For instance, the accuracy of ATI-λ increases by +10.5%. This coincides with the observation that deeper networks have a stronger linearisation effect on manifolds of image domains [Bengio et al., 2013, Upchurch et al., 2017] than shallow

	VISDA → Pascal3D						
	LSVM	TCA	gkf	SA	CORAL	ATI	ATI-λ
AlexNet	48.0	49.7	50.1	51.2	52.0	61.1	**61.4**
VGG-16	53.6	55.0	55.2	56.5	60.0	**72.0**	71.9

Table 4.14: Open set domain adaptation using synthetic images from the VISDA'17 challenge [Peng et al., 2017] as source and real data from the Pascal3D dataset [Xiang et al., 2014] as target dataset. There are 6 shared classes between both datasets.



(a) No adaptation (LSVM): 53.6%

(b) ATI-λ: 71.9%

Figure 4.8: Confusion matrices without (a) and with adaptation (b) for an open set classification task with 6 shared classes and a domain shift between synthetic [Peng et al., 2017] (source) and real [Xiang et al., 2014] (target) data. The features are extracted from the fc7 layer of the VGG-16 model [Simonyan and Zisserman, 2014].

networks. Since the proposed approach learns a linear mapping from the feature space of the source domain to the feature space of the target domain, it benefits from a better linearisation. The confusion matrices of the classification task with features extracted from the VGG-16 model are shown in Figure 4.8. ATI-λ improves the overall accuracy of LSVM by +18.3% since it resolves confusions between similar classes. For instance, LSVM frequently misclassifies *bicycle* as *motorbike* and *car* as instances of trucks, which are part of the unknown class.

4.3.3 Closed Set Domain Adaptation

We also report the accuracies of our method for popular domain adaptation datasets using the standard closed set protocols, where all classes are known in both domains.

	A→D	A→W	D→A	D→W	W→A	W→D	AVG.
NN	51.3±1.4	45.7±2.1	26.0±0.9	65.5±1.4	28.0±0.5	69.8±1.8	47.7
LSVM	62.3±3.8	55.8±3.1	42.8±1.6	90.1±0.6	41.2±0.4	92.6±1.5	64.1
TCA	60.3±4.0	54.7±3.0	49.4±1.6	90.7±0.4	46.9±2.3	92.0±0.9	65.7
gfk	61.3±3.7	55.7±3.0	45.6±1.6	90.6±0.4	43.1±2.3	93.4±0.9	65.0
SA	60.6±3.5	55.0±3.1	47.3±1.6	90.9±0.6	44.4±1.4	93.3±0.8	65.3
CORAL	64.4±3.9	58.9±3.3	52.1±1.2	**92.6±0.3**	50.0±1.0	**94.0±0.6**	68.7
ATI	**67.6±3.0**	62.3±3.1	54.8±1.3	90.3±0.8	52.4±2.1	92.6±1.7	70.0
ATI-λ	67.3±2.3	**62.6±2.5**	**55.2±2.6**	90.1±0.6	**53.4±2.5**	92.7±2.5	**70.2**
ATI-λ-N_1	64.6±2.9	60.9±1.3	51.9±1.9	90.2±0.9	48.1±1.6	93.7±2.1	68.2

Table 4.15: Comparison on the unsupervised Office dataset [Saenko et al., 2010] with 31 shared classes and 6 domain shifts using the protocol from Saenko et al. [2010] and features from the AlexNet model (fc7 layer).

4.3.3.1 Office Dataset

For the *Office* dataset [Saenko et al., 2010], we run experiments for the 6 domain shifts of the three provided datasets and use deep features extracted from the fc7 feature map from the AlexNet [Krizhevsky et al., 2012] and VGG-16 [Simonyan and Zisserman, 2014] models.

Unsupervised domain adaptation. For unsupervised domain adaptation, we first report the results for the protocol from Saenko et al. [2010], where we run 5 experiments for each domain shift using randomised samples of the source dataset. The results are shown in Table 4.15, where we compare our method with generic domain adaptation methods, i.e. TCA [Pan et al., 2009], gfk [Gong et al., 2012], SA [Fernando et al., 2013] and CORAL [Sun et al., 2015a] using AlexNet features. The results are in accordance with the observations from Section 4.3.2.1. While ATI outperforms all generic domain adaptation methods in average and ATI-λ performs slightly better than ATI, ATI-λ-N_1 decreases the accuracy in the unsupervised setting. In addition, we also include the accuracies of using nearest neighbours without domain adaptation, NN, which reports significant lower accuracies than LSVM. LSVM also outperforms NN in other closed set evaluation protocols by a large margin.

We also compare our method with current state-of-the-art CNN-based domain adaptation methods [Long et al., 2015, 2016, Ganin and Lempitsky, 2015, Venkateswara et al., 2017, Tzeng et al., 2017, Carlucci et al., 2017]. In this case, we report the accuracies when all source samples are used in a single run as described by Gong et al. [2013a]. As shown in Table 4.16, our method achieves competitive results even for the standard closed set protocol.

Semi-supervised domain adaptation. We also evaluate our approach for semi-supervised domain adaptation on the *Office* dataset. We follow the protocol from Saenko et al. [2010] and report the accuracies and standard deviations over 5 runs with random samples. In the first experiment with AlexNet features, we also include ATI-λ-N_2 with locality constraints using 2 nearest neighbours and compare our approach with

	A→D	A→W	D→A	D→W	W→A	W→D	AVG.
	AlexNet features (fc7)						
NN	55.9	49.7	27.4	75.3	31.5	86.2	54.3
LSVM	65.7	60.3	43.2	94.7	44.0	98.9	67.8
DAN	66.8	68.5	50.0	96.0	49.8	99.0	71.7
DAH	66.5	68.3	55.5	96.1	53.0	98.8	73.0
RTN	**71.0**	73.3	50.5	96.8	51.0	**99.6**	73.7
BP	-	73.0	-	96.4	-	99.2	-
ADDA	-	**75.1**	-	**97.0**	-	**99.6**	-
ATI	70.3	68.7	55.3	95.0	**56.9**	98.7	**74.2**
ATI-λ	69.0	67.0	**56.2**	95.0	**56.9**	98.7	73.8
	VGG-16 features (fc7)						
NN	61.3	55.4	33.1	78.6	49.4	88.8	61.1
LSVM	76.1	68.6	55.3	95.9	61.5	99.6	76.2
DAN	74.4	76.0	61.5	95.9	60.3	98.6	77.8
AutoDIAL	**82.3**	**84.2**	64.6	**97.9**	64.2	**99.9**	**82.2**
ATI	80.6	81.4	**67.1**	96.1	66.4	99.3	81.8
ATI-λ	80.8	81.3	66.9	96.1	**66.5**	98.9	81.8

Table 4.16: Comparison on the unsupervised Office dataset [Saenko et al., 2010] with 31 shared classes and 6 domain shifts taking all source samples as in Gong et al. [2013a].

state-of-the-art CNN-based methods [Tzeng et al., 2014, Long et al., 2015, Tzeng et al., 2015]. As in Section 4.3.2.1, we train the SVMs on the transformed source samples and labelled target samples (st). The results are reported in Table 4.17.

Our method achieves the same average accuracy as MMC [Tzeng et al., 2015] and performs slightly worse than the method by Motiian et al. [2017] for the VGG-16 features. In addition, we report the accuracy for AlexNet features when the mapping W (5.2) is estimated using only the labelled target samples without solving the individual assignments (4.1). This variant is denoted by ATI (labels t) and performs worse than ATI.

4.3.3.2 Office+Caltech dataset

We also evaluate our approach on the extended version of the Office evaluation set [Gong et al., 2012], which includes the additional *Caltech (C)* dataset. This results in 12 domain shifts, but reduces the amount of shared classes to only 10. As shown in Table 4.18, our method obtains very competitive results with AlexNet features, outperforming in overall the generic domain adaptation method [Sun et al., 2015a] and 3 out of 4 CNN-based methods. If features from a deeper network such as VGG-16 are used, our method obtains the best overall results.

4.3.3.3 Dense Testbed for Cross-Dataset Analysis

We also present an evaluation on the Dense dataset of the Testbed for Cross-Dataset Analysis [Tommasi et al., 2015] using the provided DeCAF features. This protocol comprises 12 domain shifts between the 4 datasets *Bing (B)*, *Caltech (C)*, *ImageNet*

	A→D	A→W	D→A	D→W	W→A	W→D	AVG.
	AlexNet features (fc7)						
LSVM (st)	82.6±5.5	77.0±2.5	63.4±1.6	94.0±0.8	61.8±1.1	96.3±0.8	79.2
DDC	-	84.1±0.6	-	95.4±0.4	-	96.3±0.3	-
DAN	-	**85.7±0.3**	-	**97.2±0.2**	-	96.4±0.2	-
MMC	86.1±1.2	82.7±0.8	**66.2±0.3**	95.7±0.5	65.0±0.5	**97.6±0.2**	**82.2**
ATI (labels t)	85.0±2.1	78.3±2.3	63.6±1.5	94.0±0.8	62.3±0.9	96.4±0.8	79.9
ATI	85.5±2.9	82.4±1.1	65.1±1.3	93.4±0.9	65.6±1.5	95.7±1.1	81.3
ATI-λ	85.6±2.6	82.6±0.5	65.3±1.3	93.3±1.0	65.7±1.7	95.7±1.1	81.4
ATI-λ-N_1	**88.1±1.7**	83.1±2.3	66.0±1.4	93.9±1.2	**65.9±1.5**	96.2±0.8	**82.2**
ATI-λ-N_2	87.0±3.5	84.6±3.5	65.3±1.0	93.6±1.2	**65.9±1.8**	95.8±1.3	82.0
	VGG-16 features (fc7)						
LSVM (st)	86.1±1.5	83.4±1.2	67.9±1.0	96.1±0.7	67.1±0.6	**96.6±1.0**	82.9
SO	84.5±1.7	86.3±0.8	65.7±1.7	97.5±0.7	66.5±1.0	95.5±0.6	82.7
CCSA	88.2±1.0	**89.0±1.2**	**72.1±1.0**	**97.6±0.4**	**71.8±0.5**	96.4±0.8	**85.8**
ATI-λ-N_1	**90.3±1.9**	88.0±1.4	70.8±0.9	95.1±0.7	70.3±2.0	96.3±0.9	85.1

Table 4.17: Comparison on the semi-supervised Office dataset [Saenko et al., 2010] with 31 shared classes and 6 domain shifts, following the protocol from Saenko et al. [2010].

(I) and *Sun (S)*, which share 40 classes. Following the protocol described in Tommasi et al. [2015], we take 50 source samples per class for training and we test on 30 target images per class for all datasets, except *Sun*, where we take 20 samples per class. The results reported in Table 4.19 show that ATI-λ outperforms other generic domain adaptation methods.

4.3.3.4 Sentiment Analysis

To show the behaviour of our method with a different type of feature descriptor, we also present an evaluation on the *Sentiment analysis* dataset [Blitzer et al., 2007]. This dataset gathers reviews from Amazon for four products: *books (B)*, *DVDs (D)*, *electronics (E)* and *kitchen appliances (K)*. Each domain contains 1000 reviews labelled as *positive* and another set of 1000 reviews as *negative*. We use the data provided by Gong et al. [2013b], which extracts bag-of-words features from the 400 words with the largest mutual information across domains. We report the mean accuracy over 20 splits, where for each run 1600 samples are randomly selected for training and the other 400 for testing. The results in Table 4.20 show that our approach not only works very well for image and video data, but it can also be applied to other types of data. This demonstrates the versatility of the proposed approach.

4.4 Summary

We have introduced the concept of open set domain adaptation in the context of image classification and action recognition. In contrast to closed set domain adaptation, we do not assume that all instances in the source and target domain belong to the same set

	A→C	A→D	A→W	C→A	C→D	C→W	D→A	D→C	D→W	W→A	W→C	W→D	AVG
	AlexNet features (fc7)												
NN	78.4	78.1	71.7	90.7	84.4	80.8	64.2	58.6	89.0	63.2	58.8	95.4	76.1
LSVM	83.3	84.1	77.5	91.8	89.1	82.3	79.4	70.2	97.9	80.0	72.7	**100.0**	84.0
CORAL	83.2	86.5	79.6	91.4	86.6	82.1	87.3	77.5	**99.3**	85.2	76.1	**100.0**	86.2
BP	84.6	92.3	90.2	91.9	92.8	93.2	84.0	74.9	97.8	86.9	77.3	**100.0**	88.2
DDC	83.5	88.4	83.1	91.9	88.8	85.4	89.0	79.2	98.1	84.9	73.4	**100.0**	87.1
DAN	84.1	91.1	91.8	92.0	89.3	90.6	90.0	80.3	98.5	92.1	81.2	**100.0**	90.1
RTN	**88.1**	**95.5**	**95.2**	93.7	**94.2**	**96.9**	**93.8**	84.6	99.2	**95.5**	**86.6**	**100.0**	**93.4**
ATI	86.5	92.8	88.7	**93.8**	89.6	93.6	93.4	**85.9**	98.9	93.6	86.3	**100.0**	91.9
ATI-λ	87.1	90.6	90.7	93.4	85.4	93.4	93.6	85.8	**99.3**	93.6	86.1	**100.0**	91.8
	VGG-16 features (fc7)												
NN	86.7	84.4	83.4	91.4	88.2	88.0	78.9	75.0	95.2	80.9	78.5	100.0	85.6
LSVM	87.8	88.7	87.2	93.3	91.8	91.4	82.5	77.9	98.4	87.8	84.9	100.0	89.3
ATI	91.0	92.4	95.9	94.7	93.1	97.4	93.7	89.8	98.1	95.1	90.3	99.5	94.3
ATI-λ	90.4	92.4	91.4	94.5	93.9	96.0	94.6	89.4	98.4	95.3	89.4	99.6	93.8

Table 4.18: Classification accuracies on the unsupervised Office+Caltech dataset [Gong et al., 2012] with 10 shared classes and 12 domain shifts using deep features. We take all source samples on a single run [Gong et al., 2013a].

of classes, but allow that each domain contains instances of classes that are not present in the other domain. We furthermore proposed an approach for unsupervised and semi-supervised domain adaptation that achieves state-of-the-art results for open sets and competitive results for closed sets. In particular, the flexibility of the approach, which can be used for images, videos and other types of data, makes the approach a versatile tool for real-world applications.

	B→C	B→I	B→S	C→B	C→I	C→S	I→B	I→C	I→S	S→B	S→C	S→I	AVG
LSVM	63.8±2.2	57.4±0.7	20.2±1.0	38.3±0.8	62.9±0.9	21.7±1.6	39.3±1.4	70.8±1.5	24.6±1.8	16.6±1.0	26.1±2.0	26.3±0.7	39.0
TCA	53.8±1.3	49.1±1.1	17.1±1.1	35.6±1.8	59.2±0.8	18.9±1.2	36.4±1.2	66.3±2.3	22.2±1.4	13.8±1.4	23.2±1.5	23.2±1.5	34.9
gfk	63.4±1.8	57.2±1.1	20.6±1.3	38.3±0.9	62.9±1.2	21.7±1.4	38.8±1.3	70.9±1.1	24.4±1.4	16.3±0.9	26.7±1.8	26.1±1.0	38.9
SA	63.0±1.9	57.1±1.4	20.2±1.4	38.3±0.9	62.8±1.0	21.5±1.2	39.0±1.3	71.1±1.3	24.2±1.4	16.0±0.9	26.8±1.9	26.4±1.1	38.9
CORAL	63.9±2.1	57.8±0.8	20.4±2.0	38.3±0.8	63.4±0.9	22.5±1.2	39.0±1.2	71.2±1.3	24.9±1.6	16.8±1.0	27.4±2.2	27.7±0.5	39.4
ATI	69.1±1.3	62.4±1.9	23.4±1.1	**39.0±1.4**	**66.9±1.2**	25.2±0.9	39.7±1.8	74.4±1.6	**25.9±2.1**	18.3±1.1	37.1±3.2	**35.0±1.0**	42.8
ATI-λ	**69.4±1.4**	**62.9±1.3**	**23.6±1.0**	**39.0±1.4**	**66.9±1.1**	**25.3±0.9**	**39.8±1.8**	**74.8±1.5**	25.8±2.0	**18.7±0.7**	**37.4±2.9**	34.8±0.8	**43.2**

Table 4.19: *Testbed* dataset [Tommasi and Tuytelaars, 2014] with 40 common classes and 12 domain shifts.

	B→E	D→B	E→K	K→D	AVG.
LSVM	75.5±1.6	78.2±2.5	83.1±1.8	73.3±1.8	77.5
TCA	76.6±2.2	78.5±1.6	**83.8±1.5**	75.0±1.4	78.5
gfk	77.0±2.0	**79.2±1.8**	83.7±1.7	73.7±1.9	78.4
SA	75.9±1.9	78.4±2.1	83.0±1.7	72.1±1.9	77.4
CORAL	76.2±1.7	78.4±2.0	83.1±2.0	74.2±3.0	78.0
ATI	**79.9±2.0**	**79.2±1.9**	83.7±2.1	**75.6±1.9**	**79.6**
ATI-λ	79.6±1.4	79.0±1.8	83.6±2.1	74.4±1.7	79.2

Table 4.20: Accuracies of 4 domain shifts on the Sentiment dataset [Blitzer et al., 2007] using the bag-of-words features and the protocol from Gong et al. [2013b].

Chapter 5

Viewpoint Refinement and Estimation with Adapted Synthetic Data

Contents

5.1 Introduction

In Chapter 4, we presented a domain adaptation approach that improves the classification accuracies in object and action recognition tasks. In this chapter, we extend this idea and propose a specifically tailored domain adaptation algorithm for viewpoint estimation problems, where classification-based approaches have recently shown excellent results [Tulsiani and Malik, 2015, Massa et al., 2016, Divon and Tal, 2018]. Compared to standard classification of object categories, viewpoint estimation presents more challenges than just gathering training data that copes with the intra-class variation of objects. In order to estimate the viewpoint of objects in images precisely, an accurate annotation of the training data is also required. Humans, however, perform poorly for estimating the viewpoint of an object accurately as illustrated in Figure 5.1. Instead of annotating real images, synthetic data can be generated using

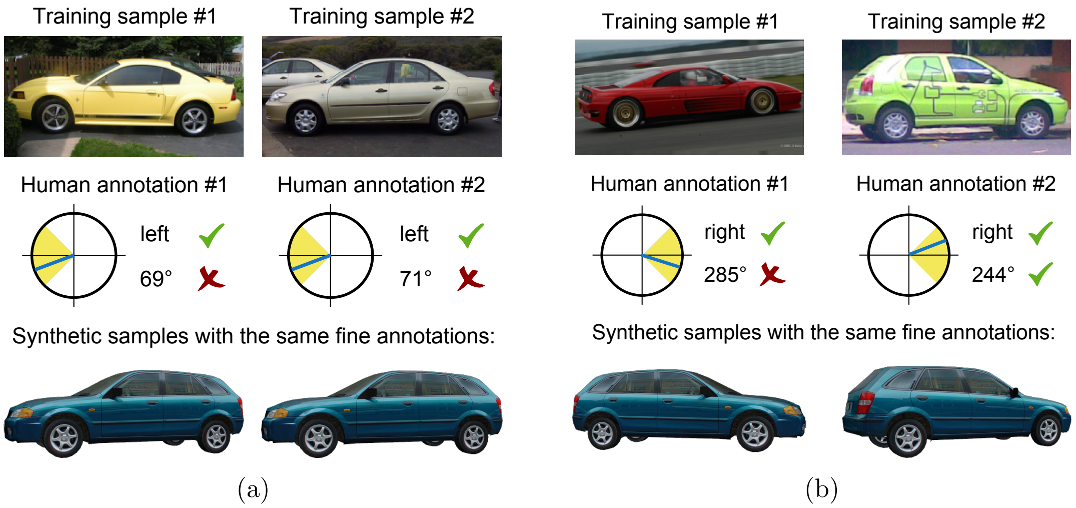


Figure 5.1: Faulty annotations of fine viewpoints are introduced in human-annotated training datasets. While coarse labels like left or right are correct, the viewpoint annotations in degrees are not precise (a) and sometimes inconsistent (b). Samples and fine annotations are taken from the Pascal3D+ dataset [Xiang et al., 2014].

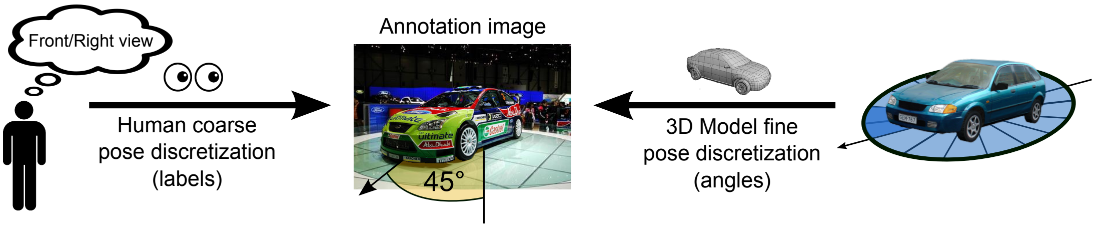
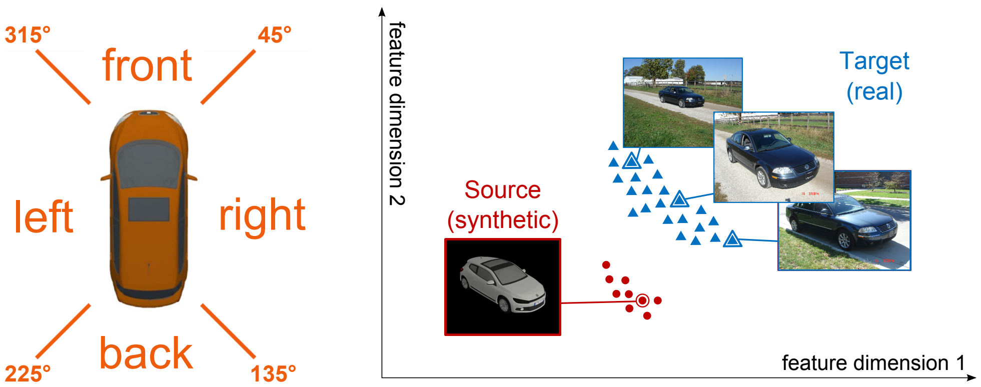


Figure 5.2: Humans are perfect for annotating coarse viewpoints of objects in real images, but fail to estimate pose accurately at a fine level. 3D graphic models can be used to synthesize data at very accurate fine angles, but it is time-consuming to model all appearance variations present in real images. We therefore propose to leverage the abilities of humans of estimating coarse viewpoints and the pose accuracy of synthetic data.

3D models [Mottaghi et al., 2015, Sun and Saenko, 2014, Vázquez et al., 2011, 2014, Pishchulin et al., 2011, Marín et al., 2010]. While synthetic data provides accurate viewpoints, it either lacks the realism of real images or it is very expensive to generate. In particular, collecting a large variation of textured 3D shapes and combining them with coherent background scenes and illumination conditions is time-consuming.

We address this issue by leveraging human annotators and synthetic data, as depicted in Figure 5.2, to avoid manual annotation by humans of fine viewpoints, which is time-consuming and erroneous, and to avoid the synthesis of a realistic dataset that captures the variations of real images, which is time and memory consuming. To this end, we ask humans to annotate only four coarse views, sketched in Figure 5.3a, and introduce an approach that refines the labels using synthetic data. Since syn-



(a) Coarse views

(b) Illustration of features space

Figure 5.3: (a) The four views available for real images. (b) Synthetic and real images with the same annotated viewpoint lie in different domains within the feature space.

thetic data and real images belong to different domains as illustrated in Figure 5.3b, a domain adaptation approach is used for the refinement. General domain adaptation approaches like Gong et al. [2012] and Hoffman et al. [2013], however, are not sufficient for label refinement since they fail to distinguish viewpoint rotations by 180 degrees. We therefore present a task-specific approach that takes advantage of the coarse labels of the real training samples. While in the previous chapter we focused on unsupervised and semi-supervised classification problems, we introduce a 2-step approach with a weekly supervised step using human labels and the resulting coarse views that are 4 unsupervised domain adaptation tasks.

In order to test the performance of our method, we provide a thorough experimental evaluation on several rigid object categories, focusing especially on cars, computing different feature descriptors, including state-of-the-art features extracted from CNNs [Krizhevsky et al., 2012, Simonyan and Zisserman, 2014]. In addition, we study the effect of truncated and occluded object instances and also show how the refined datasets are able to obtain in some cases comparable or even better results than annotated training data with full human supervision. The evaluation, which is performed on six datasets for viewpoint estimation, reveals that our approach outperforms state-of-the-art domain adaptation methods.

5.2 Adapted Synthetic Data for Viewpoint Refinement and Estimation

In this section we describe the automatic process of refining coarse annotations of real data into fine viewpoints using adapted synthetic data. As depicted in Figure 5.4, we initially request humans to coarsely annotate viewpoints of given 2D bounding boxes. Additionally, we also generate synthetic data with fine viewpoint annotations. This process is discussed in Section 5.2.1. Then, we adapt the synthetic data towards the real data, explained in Section 5.2.2, and assign fine viewpoints to the real data,

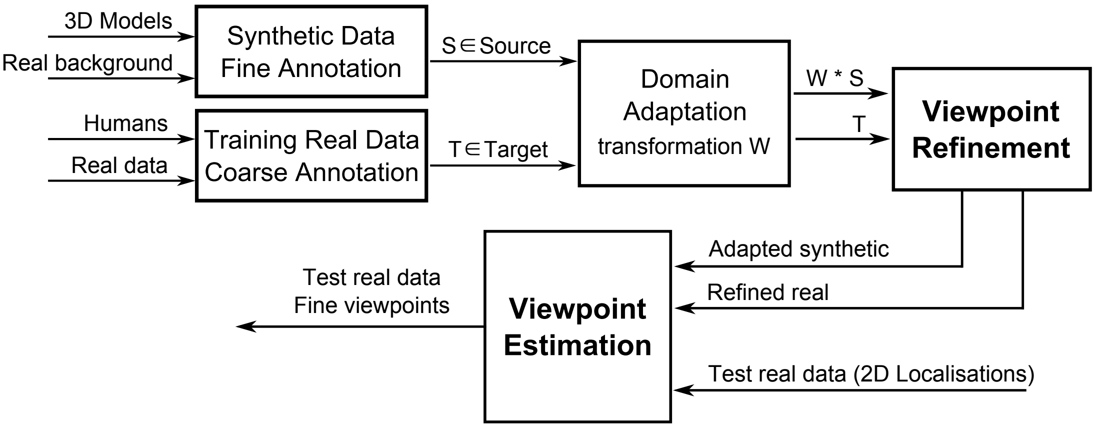

3D Models
Real background
Synthetic Data Fine Annotation
Humans
Real data
Training Real Data Coarse Annotation
S∈Source
T∈Target
Domain Adaptation transformation W
W * S
T
Viewpoint Refinement
Adapted synthetic
Refined real
Test real data (2D Localisations)
Viewpoint Estimation
Test real data Fine viewpoints

Figure 5.4: Proposed pipeline for viewpoint refinement and estimation of real data.

further detailed in Section 5.2.3. We evaluate our approach for viewpoint refinement and viewpoint estimation. For viewpoint refinement, the coarse viewpoint is given and the goal is to estimate the fine viewpoint. For viewpoint estimation, the refined real and adapted synthetic data is used to train an estimator for fine-grained viewpoint estimation. The estimator is then evaluated on unseen test instances.

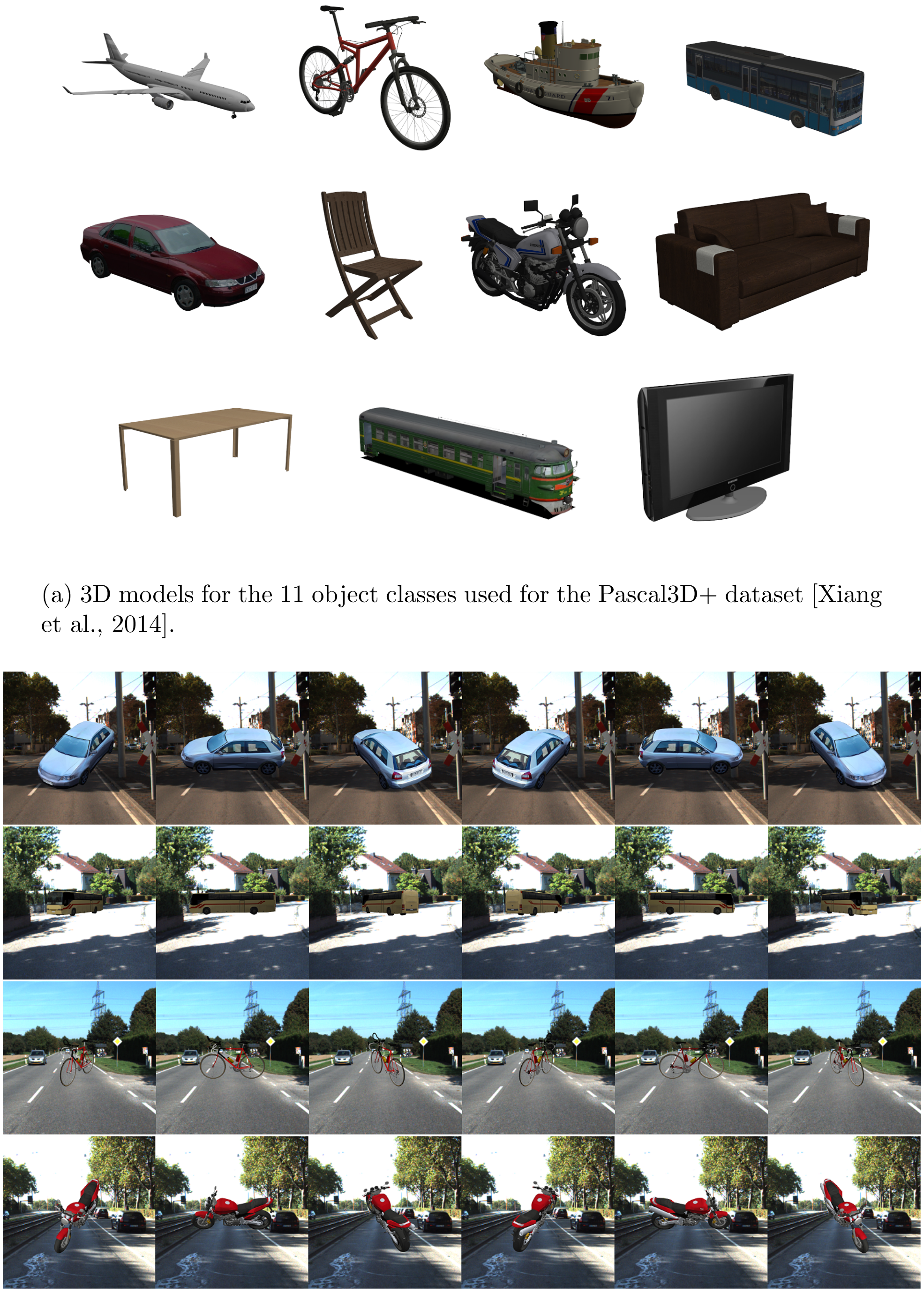
5.2.1 Generation of Synthetic Data from 3D Models

In order to produce thousands of synthetic images, we first download free available 3D graphics models from the Internet. We then render the models, centred in the screen coordinate system, with 8 different light sources evenly spread around the object. Based on a Phong reflection model [Phong, 1975], we emphasise the usage of diffuse lighting to highlight shape variations and deformations, reducing the impact of ambient illuminations and specular reflections. The resulting rendered virtual classes used in the experiments are shown in Figure 5.5a. The scene is completed with a real background image taken from Geiger et al. [2012] placed behind the rendered object.

Finally, the generation process reduces to a parametrised camera displacement with azimuth θ, elevation ϕ and object distance r. Although this configuration allows to move along the whole view-sphere, we simplify the fine viewpoint annotations to the Y-axis rotation, being the azimuth angle the most dominant factor to recognise viewpoint differences in feature space, as well as the most relevant plane in viewpoint estimation tasks [He et al., 2014]. Figure 5.5b shows some examples of synthetic images. While the process of synthesizing images does not require much effort, it does not generate realistic images since the unknown 3D geometry and light conditions of the background are not taken into account.

5.2.2 Domain Adaptation of Synthetic Data

Since synthetic data and real images belong to different domains, as illustrated in Figure 5.3b, we adapt the domain of the synthetic data to the real data. Our approach



(a) 3D models for the 11 object classes used for the Pascal3D+ dataset [Xiang et al., 2014].

(b) Synthesised images with different azimuth, elevation and distance configurations.

Figure 5.5: 3D graphics models for different object classes are rendered in front of real background images from Geiger et al. [2012] in order to automatically generate thousands of synthetic images with different accurate viewpoint annotations.

clusters the source (synthetic) and target (real) domains, and establishes correspondences between the clusters. The correspondences are then used to learn a mapping from the source domain to the target domain. The viewpoint annotations of the real images are then refined with viewpoint classifiers trained on the transformed synthetic data.

The learning of the mapping from the source to the target domain is discussed in Section 5.2.2.1 and the establishment of correspondences between clusters of both domains is discussed in Section 5.2.2.2.

5.2.2.1 Alignment from Synthetic to Real Domain

To map the source data to the target domain, we have to learn a mapping from $\mathcal{S} \in \mathbb{R}^D$ to $\mathcal{T} \in \mathbb{R}^D$, where D denotes the dimensionality of the features. For label refinement, the dimensionality of the source and the target domain is the same. We consider a linear transformation, which is represented by a matrix $W \in \mathbb{R}^{D \times D}$, i.e. $t = Ws$.

Let $S = \{s_1, ..., s_M\}$ and $T = \{t_1, ..., t_N\}$, where $s \in S$ and $t \in T$, denote the training samples of the source and target domains, respectively. M and N are the total amount of samples of each domain and we can assume that $M \geq N$, since we can always generate more synthetic data than annotated real images. We first assume that for a subset of the target elements t_k we have already established a corresponding element in the source domain. The establishment of the correspondences $C = \{c_1, ..., c_K\}$ with (s_{c_k}, t_k) and $K \leq N$ will be explained in Section 5.2.2.2.

Given the correspondences, W can be learned by minimizing the objective

$$f(W) = \frac{1}{2} \sum_{k=1}^{K} ||Ws_{c_k} - t_k||_2^2, \tag{5.1}$$

which can be expressed in matrix form:

$$f(W) = \frac{1}{2} ||WP_S - P_T||_F^2. \tag{5.2}$$

The matrices P_S and $P_T \in \mathbb{R}^{D \times K}$ represent all assignments between source and target elements, where the columns denote the actual correspondences. We optimise the objective by non-linear optimisation. To this end, the derivatives of (5.2) are calculated by

$$\frac{\partial f(W)}{\partial W} = W(P_S P_S^T) - P_T P_S^T. \tag{5.3}$$

In our implementation, we use the local gradient-based optimization method of moving asymptotes [Svanberg, 2002], which is part of the NLOPT package [Johnson, 2007–2010].

The advantage of this method compared to other widely used algorithms, such as stochastic gradient descent or conjugate gradient, is that, in each iteration, the optimisation is reduced into a convex approximating sub-problem, which is easier to solve. This formulation is thus suitable for solving large-scale unconstrained optimisation

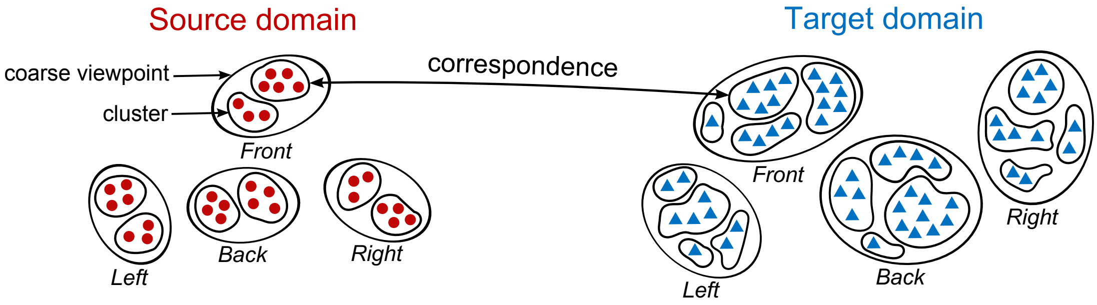


Figure 5.6: Each cluster in the target domain is assigned to a source cluster that belongs to the same coarse viewpoint. In this example, for an 8-view refinement: $V_i = 2$ and $K_i = 4$.

problems like (5.1), a very expensive function evaluation due to the size of transformation matrix W, which is still differentiable. However, dimensionality reduction techniques may also be considered for feature descriptors with extremely large D. e.g. data compression with Principal Component Analysis [Pearson, 1901]. Experiments regarding viewpoint refinement with dimensionality reduced features are presented in Section 5.3.

5.2.2.2 Source-Target Correspondences

In order to minimize (5.1), we first have to establish correspondences between the source and the target data. To this end, we cluster the data in both domains. For the synthetic data, we use the known fine-grained poses where each pose can be associated with one of the four coarse viewpoints $i = \{\text{front}, \text{back}, \text{left}, \text{right}\}$, i.e. $V = \sum_i V_i$, where V_i is the number of fine viewpoints for refinement in each coarse region. Fine viewpoints that lie between two coarse views are always assigned to the front or back views. For the target domain, we only have the coarse viewpoints and therefore cluster the N_i training samples of one coarse viewpoint further by K-Means, where the number of clusters for each coarse viewpoint is given by K_i, i.e. $K = \sum_i K_i$. and $V_i \leq K_i \leq N_i$. If $K_i = N_i$ clustering is not performed since each target instance is considered as one cluster. If $K_i = V_i$, the number of clusters is equal to the number of fine viewpoints. For the clustering, we represent each image by a HOG or CNN feature vector and append the aspect ratio of the bounding box surrounding the object.

As illustrated in Figure 5.6, we establish correspondences between the clusters in the source and target domains, separately for each coarse viewpoint. To this end, we represent each cluster by its centroid. The sets of centroids are denoted by $\hat{S}^i = \{\hat{s}^i_1, ..., \hat{s}^i_{V_i}\}$ and $\hat{T}^i = \{\hat{t}^i_1, ..., \hat{t}^i_{K_i}\}$. The correspondences are then established by solving a bipartite matching problem:

$$\begin{aligned} &\underset{e_{vk}}{\operatorname{argmin}} \sum_{v=1}^{V_i} \sum_{k=1}^{K_i} e_{vk} \left\| \hat{s}^i_v - \hat{t}^i_k \right\|_2^2 \\ &\text{subject to } \sum_v e_{vk} = 1 \;\; \forall k \;, \quad \sum_k e_{vk} = a_v \;\; \forall v \;\; \text{and} \;\; e_{vk} \in \{0,1\} \;\; \forall v,k \;. \end{aligned} \tag{5.4}$$

It assigns to each cluster in the target domain a unique cluster in the source domain. Since there can be more clusters in the target domain than in the source domain, each source is associated to $a_v = K_i/V_i$ target clusters. If K_i is not a multiple of V_i, i.e. $aV_i < K_i < (a+1)V_i$, we set $a_v = a+1$ for the first $K_i - aV_i$ source clusters and $a_v = a$ otherwise. We use the Hungarian algorithm [Kuhn, 1955] to solve the problem and for any cluster pair with $e_{vk} = 1$, we obtain a correspondence c_k. Due to the nature of the Hungarian method, which requires all combinations of centroid distances to be precomputed, we have not observed noticeable differences when solving (5.4) with other norms. The correspondences from all coarse views are then used to estimate the transformation W in (5.1).

5.2.3 Viewpoint Refinement and Estimation

The last step in our pipeline is the viewpoint refinement of the real training images. This is seen as a classification problem where we train on the transformed synthetic samples a linear SVM for each of the fine viewpoints $v = \{1, ..., V\}$, as effectively presented in other works [Liebelt and Schmid, 2010, Pepik et al., 2012, Glasner et al., 2011]. Then, we apply the linear SVMs corresponding to the coarse viewpoint i of the real image and assign the fine pose with the highest scoring function:

$$f(x, i) = \underset{v=\{1,...,V_i\}}{\operatorname{argmax}} \; w_v^T x + b_v, \tag{5.5}$$

where w_v and b_v are the weights and bias of the linear SVM for the fine viewpoint v. Since the transformation of synthetic data is guided by correspondences that deal with a discretised representation of viewpoints, i.e. source samples are clustered in exactly V centroids, we consider that the usage of classifiers for the final viewpoint refinement naturally fits in the overall formulation.

For pose estimation on real test images, we also use linear SVMs in a one-vs-all classification procedure. For each fine viewpoint, we train a linear SVM using the real training images with refined pose labels and the synthetic training images, which have been transformed by domain adaptation, together.

5.3 Experiments

We evaluate our algorithm on three car and three multi-object datasets with fine annotated poses. From the former group, the *Multi-View Car* [Ozuysal et al., 2009] dataset contains sequences of 20 cars as they rotate by 360°, where one image is taken every 3-4°. These fine-grained poses allow us to test the refinement at higher levels of viewpoint discretisation. We take the first 10 car sequences as training (1179 images) and the last 10 as test data (1120 images). Since the cars in this dataset are in a fixed location, we also evaluate our method on the more realistic *KITTI* [Geiger et al., 2012] benchmark, where images are recorded while driving along streets and roads. Due to the lack of bounding box annotations in the test data, we perform a 2-fold cross validation on the fully visible cars of the training set, containing 7481 images with 17463 cars, 7811 of those which are non-occluded. For a cross-dataset experiment in Section 5.3.5, we also use the dataset [Sedaghat and Brox, 2015] where the bounding boxes and viewpoints have been annotated in a fully unsupervised manner. From the latter, the *3D Object Categorization* [Savarese and Fei-Fei, 2007] dataset provides 10 image sets of cars and bikes in 8 different angles (every 45 degrees), permitting a refinement from 4 to 8 fine viewpoints. There are 2 elevations and 3 distances for each view, giving 48 images per object. We take 7 sets for training and 3 for testing. We also evaluate the method on the *Pascal3D+* [Xiang et al., 2014], which contains occlusions and truncated object instances of several classes. The main part of this dataset enriches the PASCAL VOC 2012 [Everingham et al., 2010] categories with 3D annotations for 11 rigid objects (In the original protocol, the class "bottle" is discarded due to its lack of viewpoint reference): *aeroplane*, *bike*, *boat*, *bus*, *car*, *chair*, *dining table*, *motorbike*, *sofa*, *train* and *tv monitor*. The dataset has been further increased by images from the ImageNet dataset [Deng et al., 2009], which are also augmented with 3D annotations for the same rigid objects, and contain a larger amount of samples but with reduced number of occluded instances. Therefore, we opt for evaluating both subsets separately, denoted in our experiments as *Pascal3D* and *ImageNet3D*, respectively, using their validation sets as test data. The setup for the experiments is as follows. At first, we automatically generate synthetic data of textured 3D models for each object class. Following the evaluation protocol of Panareda Busto et al. [2015], we take 10 graphics models for each of the 11 rigid object categories, thus decreasing the number of cars from 15 to 10 in order to keep an even quantity among all classes. The attached background images, randomly taken from the KITTI dataset [Geiger et al., 2012], point towards the car's driving direction, allowing for synthetic vehicle placements, e.g. bike, bus, car and motorbike classes, in the centre of the image. In comparison to Panareda Busto et al. [2015], the synthetic images are obtained with a finer viewpoint granularity, rotating the θ angle of the camera every 1 degree in clockwise order, instead of every 10 degrees, allowing for a total of 360 fine viewpoints. Since elevation ϕ varies among the objects classes, we take the elevation ranges of each object class from the training data of Xiang et al. [2014] and discretise them in 4 different levels, independently. Besides, we make use of one single distance, $r = 2.0$, in virtual world coordinates. The pose labels are then quantised to their closest angle of the V fine poses. The first viewpoint $v = 1$ lies at $\theta = 0$ in all quantisation levels. Overall, we generate 14400 samples per object class. Some examples of the synthesised data are illustrated in Figure 5.5b.

Our first evaluation, in Section 5.3.1, measures the accuracy of our viewpoint refinement, extracting the bounding boxes of the real training images and converting the given viewpoints into the four coarse views, that is: $front = (315°, ..., 45°)$, $right = [45°, ..., 135°]$, $back = (135°, ..., 225°)$ and $left = [225°, ..., 315°]$. Then, in Section 5.3.2, we evaluate the viewpoint estimation of the real test images having as training the adapted synthetic data and the refined real data. We use the given bounding boxes if the images are not already cropped. Neither coarse nor fine viewpoints are used for the test images. Section 5.3.3 discusses the impact of occluded object instances and Section 5.3.4 evaluates the accuracy of CNN-based methods for pose estimation using the refined datasets. We finally perform a cross-dataset experiment in Section 5.3.5.

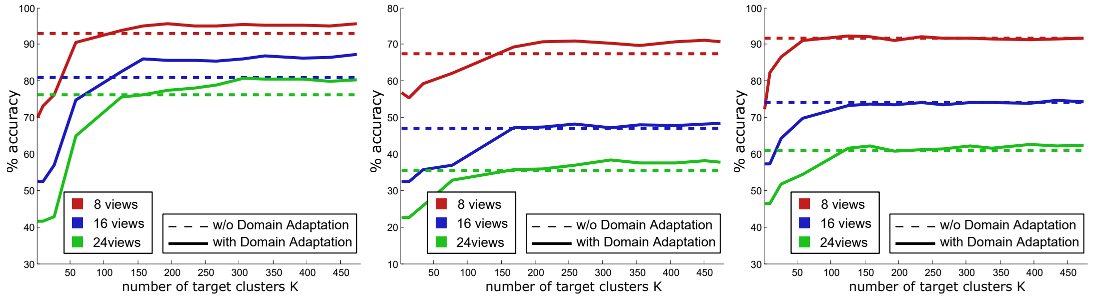
Several widely used feature descriptors are evaluated to measure the performance of the method in different feature spaces. For the hand-crafted features, we rescale the bounding boxes to 128×128 pixels and extract HOG descriptors [Dalal and Triggs, 2005] with 8 bins (31 channels/bin), as in Panareda Busto et al. [2015]. For the deep features, we take the AlexNet [Krizhevsky et al., 2012] and VGG [Simonyan and Zisserman, 2014] models and we extract the feature maps from the last convolutional layer (CNN-pool5), with 9216 and 25088 dimensions from the standard 227×227 and 224×224 input patches, respectively. As we will show in Section 5.3.1, we reduce the dimensionality for AlexNet to 3041 dimensions (33%) and for VGG to 6272 dimensions (25%) without loss of accuracy. Additionally, we also evaluate the features from the last fully connected layer (CNN-fc7) of a re-trained VGG model, using the synthetic dataset and modifying the output layer with 360 classification channels. In the experiments with hand-crafted features, the annotated instances are rescaled preserving the aspect ratio. For the evaluations with deep features, the annotations are warped as in Tulsiani and Malik [2015].

5.3.1 Viewpoint Refinement

We first evaluate the accuracy of our approach for pose refinement on the real training images. To this end, we use the coarse labels of the real training images and refine the viewpoints as described in Section 5.2.3. We then evaluate the accuracy of the refined labels on the real training images in conjunction with the transformed synthetic samples after the domain adaptation process. For the initial parameter evaluation of our method, we stick to extracted AlexNet (CNN-pool5) features of car models. Then, we test the performance of our viewpoint refinement for all descriptors and classes.

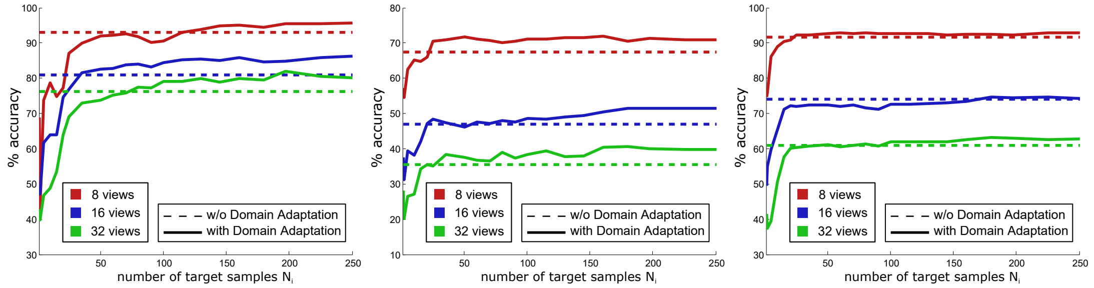
Impact of number of target clusters As described in Section 5.2.2.2, we cluster each coarse view by K-Means. We therefore evaluate the impact of the number of target clusters K on the viewpoint refinement. The results for the different datasets and V refined viewpoints used for evaluation are shown in Figure 5.7. As baseline, we use linear SVMs trained on the synthetic data without domain adaptation. The accuracy tends to stabilize when the number of clusters is sufficiently large. The finer the viewpoints are the more clusters are also needed.

Impact of number of target samples Although annotating real images by coarse viewpoints is easy to do, it also takes time. We therefore evaluate the impact of the number of coarsely labelled target samples N. To avoid any clustering artefacts, we set $K_i = N_i$, i.e. each target sample itself is a cluster. We also keep the numbers of the real images N_i for each of the four viewpoints equal while increasing N. The results in Figure 5.8 show that already 100-150 annotated samples per coarse view give a boost in performance compared to the baseline. This means that very little time is actually required for the annotation task.



(a) Multi-View Car dataset (b) Pascal3D dataset (c) ImageNet3D dataset

Figure 5.7: Impact of the number of target clusters K for viewpoint refinement.



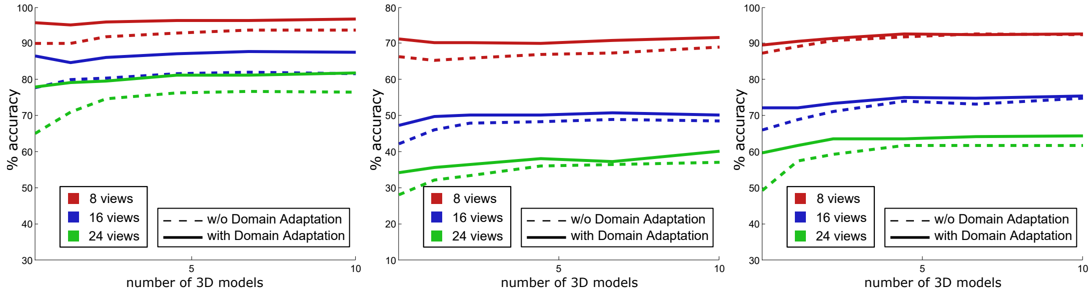
(a) Multi-View Car dataset (b) Pascal3D dataset (c) ImageNet3D dataset

Figure 5.8: Impact of the number of target samples N_i per coarse view for the refinement.

Impact of number of 3D models We also evaluate the impact of the amount of 3D models used to generate synthetic data. Figure 5.9 shows how the accuracy tends to stabilise with already 5 models.

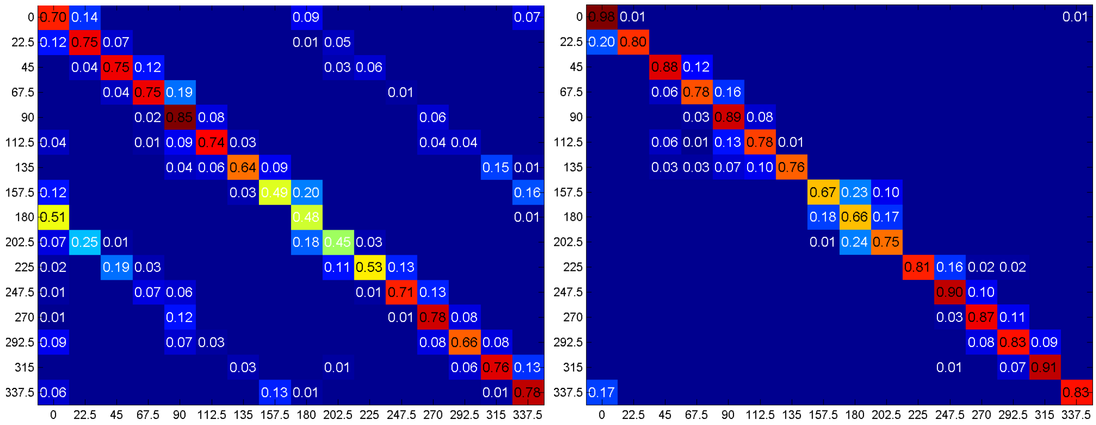
Weak supervision If the target samples are not annotated by the four coarse views, we can still perform unsupervised domain adaptation. In this case, we observe a substantial amount of wrong viewpoint estimates by 180 degrees as shown by the confusion matrix in Figure 5.10a. In contrast, we resolve these errors by using the coarse viewpoints of the real images as weak supervision as shown in Figure 5.10b. This shows that using coarse annotations of real images, which are inexpensive to annotate, significantly increases the viewpoint refinement accuracy.

Accuracy of the viewpoint refinement We finally compare the refinement accuracy of our method with popular domain adaptation techniques [Gong et al., 2012, Fernando et al., 2013, Sun et al., 2015a]. The geodesic flow kernel (GFK) [Gong et al., 2012] is an unsupervised domain adaptation method that maps both domains to a common subspace in a Grassmannian manifold. The same applies to the subspace alignment technique (SA) [Fernando et al., 2013], that maps both domains to a common subspace using the d largest eigenvectors. In both cases, the number of chosen sub-dimensions d is kept as large as possible to avoid a significant loss in accuracy. Lastly, we also test the current state-of-the-art adaptation method named



(a) Multi-View Car dataset (b) Pascal3D dataset (c) ImageNet3D dataset

Figure 5.9: Impact of the number of 3D car models for viewpoint refinement.



(a) Unlabelled target samples (b) 4 viewpoint labels in target samples

Figure 5.10: Confusion matrix for the Multi-View Car dataset in a 16-viewpoint refinement. (a) Without supervision rotations by 180 degrees are sometimes confused. (b) When weak supervision from the four coarse viewpoint labels is used, these confusions are resolved.

CORAL [Sun et al., 2015a]. Without any dimensionality reduction, it decorrelates the source samples by whitening and re-colours them by the covariance matrix of the target data. For all methods, we exploit the weak supervision and apply them for each coarse viewpoint, independently. As already shown in Panareda Busto et al. [2015], supervised methods that internally process the coarse labelling [Hoffman et al., 2013] report worse viewpoint accuracies than the unsupervised methods. For the refinement after domain adaptation, we use linear SVMs as described in Section 5.2.3. As baseline, we use the linear SVMs trained on the synthetic data without domain adaptation (*w/o DA*).

For our method, we report the refinement accuracy for four different clustering settings. For the first three, we set V equal to the number of views for fine-grained viewpoint estimation as in the previous experiments. We report numbers for $K = V$, $K = 100$ and $K = N$. For the first two settings, we report the mean accuracy and its standard deviation over 10 runs since K-Means depends on the random initialization.

In the last setting, each target sample is a cluster.

We first report the results only for the fully visible object exemplars and compare the hand-crafted features (HOG) and the deep features, i.e. the last convolutional features from AlexNet and VGG models (CNN-pool5) after dimensionality reduction and the re-trained fully connected layer of VGG (CNN-fc7), in Table 5.1. The accuracies of CNN-pool5 features from both models outperform the results of the HOG features, obtaining VGG slightly better results than AlexNet, especially for finer viewpoints. While both CNN-pool5 features achieve the best overall results, VGG CNN-fc7 performs slightly better on the Multi-View Car dataset for $V \geq 72$.

While $K = N$ performs best in almost all cases, $K = 100$ and $K = V$ achieves the highest accuracy in only very few cases, with only marginal improvements compared to $K = N$. Overall, $K = N$ with CNN-pool5 features performs best. We also evaluated the accuracy when V is also set to the number of synthetic samples M, i.e. each synthetic image is a cluster. In this case, the accuracy drops significantly for all datasets and feature descriptors. This shows that the synthetic data needs to be quantized according to the fine-grained views.

Table 5.1 also compares our approach to other domain adaptation methods [Gong et al., 2012, Fernando et al., 2013, Sun et al., 2015a]. In nearly all setting and feature combinations, our method outperforms the generic domain adaptation methods. Although CORAL obtains better results with CNN-fc7 features, the reported accuracies are still lower than the results of the CNN-pool5 features with our method.

In contrast to the datasets [Ozuysal et al., 2009, Geiger et al., 2012, Savarese and Fei-Fei, 2007], the datasets Pascal3D and ImageNet3D contain many occluded and truncated objects. The results for these two datasets are reported in Table 5.2. We report the accuracies for both CNN models with the CNN-pool5 features using $K = N$ and compare it to the baseline without domain adaptation. Except for the 8 view refinement on ImageNet3D, our approach outperforms the baseline by around 4-6%. In general, the reported results of the AlexNet and VGG models are comparable.

Viewpoint refinement without coarse annotations We complete the evaluation of the viewpoint refinement by showing how it behaves if the real images are not weakly labelled by humans. Concretely, we test ATI-λ for closed set domain adaptation, presented in Chapter 4, and compare it against the baseline, whose real images also remain unsupervised. In this scenario, the total number of source clusters V and the amount of target samples N are the input parameters of the adaptation algorithm. We report their accuracies in Table 5.3 and 5.4, which show a substantial accuracy reduction, with and without domain adaptation, compared with the methods that use coarse annotations, shown in Table 5.1 and 5.2. This implies that just a small amount human effort produces remarkable improvements in the quality of viewpoint annotation. Besides, we observe that ATI-λ only outperforms the baseline in datasets with more differences between mirrored angles, proving the advantages of weakly supervised domain adaptation in viewpoint estimation tasks. The adaptation also fails when using *fc7* features, since its fully connected layer highly diminishes the subtle differences of similar viewpoints.

	3DObjCat		Multi-View Car							KITTI	
							HOG				
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	98.2	98.4	88.8	78.1	68.5	55.6	30.9	13.3	6.5	82.5	**69.9**
GFK [Gong et al., 2012]	98.2	98.0	89.3	79.7	71.3	55.9	31.7	14.9	6.5	82.2	69.1
SA [Fernando et al., 2013]	96.4	98.4	87.5	77.0	69.4	55.2	31.8	13.2	6.4	**87.4**	69.3
CORAL [Sun et al., 2015a]	95.8	95.8	89.9	79.1	65.7	52.4	24.6	10.2	4.4	78.3	66.5
V=views, K=V	83.4 (0.8)	81.6 (0.7)	78.6 (1.7)	63.7 (2.1)	63.0 (2.2)	51.5 (1.8)	30.6 (1.4)	14.5 (1.0)	7.0 (0.6)	65.7 (1.9)	65.8 (1.4)
V=views, K=100	99.4 (0.2)	98.8 (0.4)	**92.1** **(0.6)**	81.2 (0.8)	71.1 (1.5)	59.3 (1.2)	32.2 (1.1)	14.6 (0.9)	**7.6** **(0.4)**	80.4 (1.4)	67.5 (1.5)
V=views, K=N	**100.0**	**99.8**	91.0	**85.3**	**76.8**	**64.4**	**38.1**	**15.6**	7.4	83.8	69.0
V=M, K=N	98.2	98.4	89.3	78.1	67.6	53.8	28.8	13.4	7.1	82.0	67.5
							AlexNet CNN-pool5				
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	99.7	97.0	93.5	81.5	76.8	61.4	35.1	12.8	6.1	81.9	70.4
GFK [Gong et al., 2012]	99.4	97.8	94.9	83.5	78.5	60.4	35.1	14.4	6.8	83.2	67.4
SA [Fernando et al., 2013]	99.7	96.8	92.5	81.4	76.1	61.1	35.5	13.0	6.8	83.5	**71.3**
CORAL [Sun et al., 2015a]	98.8	94.8	94.4	81.5	71.5	54.8	27.1	7.0	2.0	79.7	64.1
V=views, K=V	83.3 (1.7)	68.5 (2.2)	70.8 (2.7)	52.7 (1.5)	42.2 (1.3)	29.2 (1.7)	30.2 (1.0)	14.4 (0.5)	8.3 (0.8)	67.3 (2.2)	40.1 (2.8)
V=views, K=100	99,7 (0.0)	95.6 (0.9)	94.7 (0.5)	83.2 (1.2)	71.2 (1.1)	56.4 (1.4)	30.9 (1.2)	14.5 (0.8)	**8.7** **(0.8)**	75.9 (1.9)	64.9 (1.7)
V=views, K=N	**100.0**	**99.0**	**96.7**	**87.5**	**81.7**	**67.7**	**40.5**	**16.3**	7.3	**84.7**	68.8
V=M, K=N	99.7	97.0	93.6	81.2	71.4	60.0	34.3	13.3	6.9	82.1	63.3
							VGG CNN-pool5				
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	99.7	96.2	93.5	84.4	76.2	62.5	34.5	13.0	6.7	82.1	68.3
GFK [Gong et al., 2012]	99.4	97.0	95.1	85.0	78.1	61.0	33.9	14.0	7.1	83.1	66.1
SA [Fernando et al., 2013]	98.2	91.3	93.3	83.9	75.5	59.6	33.4	13.2	7.2	82.5	67.6
CORAL [Sun et al., 2015a]	98.2	94.6	95.0	82.8	75.4	60.0	31.3	9.8	4.6	77.2	65.8
V=views, K=V	54.5 (3.4)	60.1 (3.7)	54.8 (4.1)	37.0 (3.2)	22.4 (2.8)	25.4 (2.0)	20.4 (1.7)	11.9 (1.0)	7.9 (1.1)	49.5 (4.0)	30.7 (2.2)
V=views, K=100	97.3 (0.5)	93.5 (0.6)	92.6 (0.7)	73.6 (1.1)	60.2 (1.3)	41.1 (0.9)	21.2 (0.5)	12.4 (0.8)	8.0 (0.6)	82.0 (1.0)	64.5 (1.3)
V=views, K=N	**100.0**	**98.8**	**95.5**	**87.0**	**82.1**	**70.1**	**42.7**	**19.5**	**9.0**	**84.7**	**68.5**
V=M, K=N	99.4	96.2	93.6	84.1	72.2	60.8	34.0	13.3	7.5	82.5	62.2
							VGG CNN-fc7				
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	96.4	96.8	90.6	79.8	74.2	63.6	43.2	20.2	9.0	78.9	65.3
GFK [Gong et al., 2012]	96.5	96.9	91.2	81.1	76.0	63.3	42.8	19.9	10.0	78.1	64.5
SA [Fernando et al., 2013]	95.5	96.2	90.5	80.1	73.4	61.9	**43.6**	18.9	10.7	79.4	64.7
CORAL [Sun et al., 2015a]	91.7	94.1	**93.9**	**83.6**	76.0	**63.9**	42.0	19.1	9.2	74.9	59.6
V=views, K=V	86.9 (1.2)	**97.4** **(0.5)**	89.8 (1.0)	72.9 (1.5)	69.3 (2.0)	58.9 (1.8)	40.9 (2.1)	19.7 (0.9)	**10.4** **(0.7)**	66.4 (2.3)	56.3 (1.7)
V=views, K=100	97.0 (0.5)	96.8 (0.5)	90.9 (0.7)	80.2 (0.9)	**76.2** **(0.9)**	63.8 (1.1)	43.5 (0.8)	21.5 (0.8)	10.2 (0.6)	76.8 (2.1)	63.2 (2.2)
V=views, K=N	**97.9**	96.8	90.8	80.6	74.8	**63.9**	43.3	**22.2**	9.9	78.6	**67.2**
V=M, K=N	96.4	97.0	90.7	81.4	75.9	63.6	39.1	20.2	9.7	**79.8**	64.4

Table 5.1: Accuracy of the coarse-to-fine viewpoint refinement for different domain adaptation techniques. For the methods with K-Means clustering, the mean and standard deviation (brackets) over 10 runs are provided.

		PASCAL3D											
		AlexNet CNN-pool5											
views		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**63.4**	67.5	57.6	69.3	68.2	58.6	64.9	**70.6**	61.4	65.2	64.4	64.6
	V=views, K=N	59.0	**68.6**	**60.7**	**72.0**	**70.9**	**63.2**	**66.5**	70.1	**66.0**	**67.8**	**68.4**	**66.7**
16	w/o DA	**42.0**	42.0	31.5	53.9	47.8	38.4	41.3	44.8	43.2	42.6	**41.2**	42.6
	V=views, K=N	36.0	**49.3**	**35.1**	**57.6**	**49.5**	**42.4**	**45.0**	**54.5**	**44.1**	**47.4**	34.3	**45.0**
24	w/o DA	**28.6**	34.2	19.2	43.8	36.2	29.4	**28.7**	34.1	32.3	23.7	19.7	30.0
	V=views, K=N	28.0	**39.8**	**27.5**	**44.7**	**39.3**	**31.1**	27.7	**39.4**	**35.7**	**30.4**	**35.6**	**34.5**
		VGG CNN-pool5											
		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**62.1**	67.3	55.6	**68.7**	67.8	60.0	**47.9**	68.5	64.1	66.8	55.4	62.2
	V=views, K=N	57.4	**72.5**	**58.2**	67.9	**70.7**	**63.6**	46.7	**69.4**	**76.0**	**70.1**	**59.7**	**64.7**
16	w/o DA	**38.7**	40.9	32.1	**62.7**	46.4	36.1	34.6	47.5	35.3	38.7	43.8	41.5
	V=views, K=N	36.1	**53.2**	**36.0**	56.8	**50.7**	**41.5**	**45.2**	**52.1**	**40.0**	**52.4**	**49.1**	**46.6**
24	w/o DA	24.3	30.7	18.2	43.3	36.1	26.2	22.2	32.7	25.6	30.0	29.6	29.0
	V=views, K=N	**29.0**	**39.4**	**26.5**	**46.1**	**40.4**	**30.8**	**27.7**	**38.9**	**38.2**	**37.8**	**37.4**	**35.7**

		ImageNet3D											
		AlexNet CNN-pool5											
		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**64.8**	**78.8**	**56.5**	**94.9**	91.3	75.5	73.0	73.8	**77.4**	**64.8**	**81.6**	**75.7**
	V=views, K=N	60.1	78.7	55.9	92.8	**91.5**	**75.8**	**76.6**	**77.5**	77.2	63.6	**81.6**	75.6
16	w/o DA	**46.5**	56.0	36.3	70.7	73.6	62.1	34.9	52.1	57.0	34.7	**39.3**	51.2
	V=views, K=N	42.1	**60.0**	**37.8**	**74.6**	**74.2**	**62.5**	**60.0**	**58.8**	**63.8**	**45.5**	37.3	**56.1**
24	w/o DA	**37.5**	41.3	25.7	54.4	60.5	48.9	28.7	36.1	**45.0**	28.1	**40.0**	40.6
	V=views, K=N	36.8	**48.2**	**27.8**	**62.4**	**63.2**	**53.3**	**50.8**	**40.6**	43.4	**34.7**	35.3	**45.1**
		VGG CNN-pool5											
		aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	**64.8**	76.4	**60.7**	**92.2**	**91.5**	**77.3**	71.4	**71.8**	**85.1**	**77.4**	**80.5**	**77.2**
	V=views, K=N	61.1	**76.5**	58.3	87.9	90.1	73.7	**74.8**	77.7	73.8	70.7	74.5	74.5
16	w/o DA	**44.2**	55.0	36.7	69.0	**73.5**	55.8	44.1	52.5	57.6	**44.0**	25.2	50.7
	V=views, K=N	**44.2**	**59.1**	**38.6**	**73.3**	72.6	**59.0**	**57.9**	**57.1**	**60.3**	40.8	**46.5**	**55.4**
24	w/o DA	33.5	40.4	26.0	53.9	**63.5**	44.1	33.2	34.2	42.0	22.1	22.1	37.7
	V=views, K=N	**35.2**	**50.1**	**30.2**	**57.1**	63.2	**47.4**	**44.5**	**43.1**	**56.1**	**26.7**	**22.5**	**43.3**

Table 5.2: Accuracy of the coarse-to-fine viewpoint refinement for the Pascal3D and ImageNet3D datasets that contain occlusions and truncated object instances.

Impact of dimensionality reduction For the results shown from Table 5.1 to 5.4, we reduced the dimensionality of the convolutional feature maps. Since in most of the experiments $D > M + N$, we employ randomised singular value decomposition to reduce the dimensionality for efficiency. Figure 5.11 shows that deep features from convolutional layers can be strongly reduced. While the performance of the HOG features start to decrease with less than 40% of the feature dimensionality, the dimensionality of the AlexNet and VGG CNN-pool5 features can be reduced without significant loss in accuracy by 33% and 25%, respectively.

	3DObjCat		Multi-View Car							KITTI	
	HOG										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	74.1	**73.0**	69.4	58.9	52.0	37.1	21.0	9.2	6.4	**32.4**	27.1
ATI-λ	**86.9**	**73.0**	**77.3**	**70.5**	**63.7**	**46.7**	**27.4**	**11.6**	**7.2**	27.5	**27.5**
	AlexNet CNN-pool5										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	95.5	81.8	80.7	69.6	66.7	52.1	29.7	10.7	5.1	**62.1**	**44.5**
ATI-λ	**98.8**	**84.7**	**86.1**	**82.5**	**75.3**	**62.2**	**37.0**	**14.5**	**6.6**	44.9	44.4
	VGG CNN-pool5										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	93.5	81.8	81.4	70.5	62.3	48.9	25.2	7.5	3.8	**60.5**	**41.0**
ATI-λ	**98.2**	**85.1**	**87.4**	**78.6**	**71.8**	**61.1**	**36.7**	**15.7**	**8.9**	53.7	38.4
	VGG CNN-fc7										
views	8/car	8/bike	8	16	24	36	72	180	360	8	16
w/o DA	61.3	47.8	**56.4**	**35.9**	**24.0**	**17.9**	**8.6**	**3.9**	1.2	**31.8**	**18.6**
ATI-λ	**69.1**	**48.4**	38.9	23.6	18.0	12.7	6.0	2.8	**1.3**	26.4	14.3

Table 5.3: Accuracy of the viewpoint refinement without annotated coarse viewpoints in the target domain on the 3D Object Categorization, Multi-View Car and KITTI datasets.

		PASCAL3D											
		AlexNet CNN-pool5											
views		aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
8	w/o DA	42.1	56.5	**24.2**	**47.1**	**70.8**	52.3	**29.8**	62.7	42.2	**30.6**	**30.6**	44.4
	ATI-λ	**47.7**	**59.8**	22.1	43.7	64.4	**56.9**	29.7	**65.8**	**57.2**	24.3	18.8	**44.6**
16	w/o DA	**27.4**	**36.6**	10.2	**52.8**	49.8	31.9	**8.7**	49.0	27.8	**28.1**	**18.9**	31.0
	ATI-λ	24.8	33.8	**11.0**	47.0	**53.5**	**44.6**	7.1	**58.4**	**30.4**	17.1	17.1	**31.4**
24	w/o DA	**17.7**	20.4	**6.7**	**46.5**	38.1	31.9	15.0	28.2	31.0	**9.4**	**13.9**	23.5
	ATI-λ	13.0	**35.5**	6.2	30.1	**38.8**	**36.6**	**15.1**	**31.7**	**41.1**	4.7	13.0	**24.2**

		ImageNet3D											
		AlexNet CNN-pool5											
		aero	bike	boat	bus	car	chair	table	mbike	sofa	dtrain	tv	Avg.
8	w/o DA	**42.2**	57.3	**23.0**	**62.5**	76.1	56.2	30.4	68.4	**72.3**	37.2	**50.9**	52.4
	ATI-λ	33.2	**65.2**	20.8	60.2	**80.4**	**65.3**	**35.5**	**72.6**	67.3	**38.0**	44.8	**53.0**
16	w/o DA	**30.8**	35.0	**14.0**	**54.1**	60.6	46.1	17.2	42.0	53.0	9.3	**20.7**	34.8
	ATI-λ	21.0	**39.1**	9.3	45.7	**65.6**	**52.4**	**22.1**	**56.7**	**54.5**	**15.1**	19.7	**36.5**
24	w/o DA	**20.8**	27.6	**8.9**	**40.8**	49.4	42.6	11.6	24.9	**41.8**	11.8	25.2	27.8
	ATI-λ	16.0	**32.0**	6.9	36.2	**54.9**	**44.2**	**14.7**	**33.2**	37.8	**12.9**	**26.5**	**28.7**

Table 5.4: Accuracy of the viewpoint refinement without annotated coarse viewpoints in the target domain on the Pascal3D and ImageNet3D datasets.

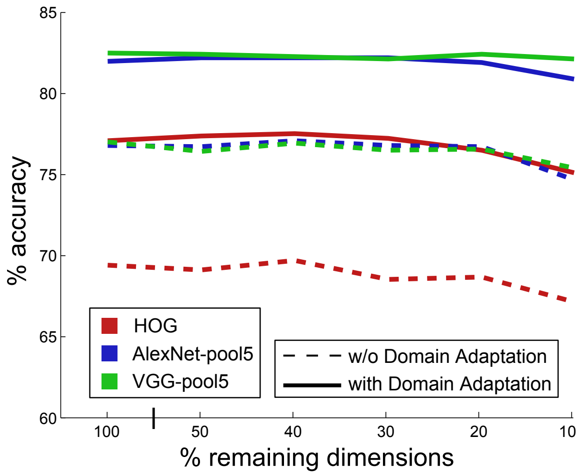


Figure 5.11: Impact of dimensionality reduction using randomised singular value decomposition for different feature descriptors on the Multi-View Car dataset with a 24-viewpoint refinement setting.

5.3.2 Viewpoint Estimation

We then evaluate the accuracy of the pose estimation on the real test images. To this end, we train the viewpoint estimator described in Section 5.2.3 on the synthetic data (*syn*), the real training data (*real*) with refined viewpoint labels or on both datasets (*joint*). For the refinement, we use our approach with $K = N$ (*with DA*) and compare it to the refinement without domain adaptation (*w/o DA*). We report the results for the datasets with non-occluded object instances in Table 5.5, where we also compare the accuracy of the pose estimator when the fine ground-truth viewpoint annotations of the real training images (*gt*) are used for training. This serves as an upper bound of the accuracy in comparison to the setting with only weak supervision.

When comparing the results of the domain adaptation for the synthetic, real or both training sets with the results without domain adaptation, we observe that the domain adaptation improves the viewpoint estimation for all scenarios with HOG and CNN-pool5 features, with the exception of the KITTI dataset with 16 viewpoint refinement, since it mainly contains cars facing coarse directions. On the contrary, the CNN-fc7 features only obtain minor improvements for some of the settings, which is consistent with the previous results.

Using refined real target images (*with DA real*) for training is in most cases sufficient. The adapted synthesized training data, however, performs better for fine-grained viewpoints $V \geq 72$ since the real images do not necessary provide enough samples for each viewpoint. Combining the real and synthetic data for training (*with DA joint*) also works very well for any viewpoint discretisation.

Table 5.6 reports the accuracies for the Pascal3D and ImageNet3D datasets using CNN-pool5 features from the VGG model. On these datasets the adapted synthesized training data performs already better than the real data for $V \geq 16$ fine viewpoints. As

		3DObjCat		Multi-View Car							KITTI	
						HOG						
	views	8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*100.0*	*100.0*	*77.7*	*69.1*	*61.1*	*53.3*	*35.0*	*13.1*	*1.7*	*85.8*	*82.5*
w/o DA	syn	77.1	84.7	67.8	61.1	53.4	35.9	19.8	6.9	4.2	58.8	54.8
	real	**100.0**	99.1	76.2	64.5	53.5	41.3	20.8	2.7	0.6	77.4	64.8
	joint	87.5	97.7	74.1	65.7	55.5	43.7	22.5	6.5	4.3	**80.1**	**66.8**
with DA	syn	86.1	94.0	73.1	66.3	59.5	43.7	22.6	**8.5**	4.5	68.1	46.3
	real	**100.0**	**99.5**	**76.5**	**68.1**	**63.1**	**48.5**	22.8	7.8	1.1	76.9	61.7
	joint	91.0	98.2	74.2	67.9	62.0	45.4	**23.3**	8.0	**4.9**	77.8	62.9
						AlexNet CNN-pool5						
		8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*100.0*	*98.2*	*82.6*	*74.8*	*68.1*	*57.1*	*33.5*	*12.0*	*1.7*	*92.5*	*85.0*
w/o DA	syn	91.0	81.0	84.5	66.8	55.8	44.5	23.7	8.5	4.5	60.3	46.2
	real	**100.0**	97.2	81.4	71.5	62.5	46.9	24.4	2.2	0.1	75.7	64.0
	joint	96.5	93.1	80.6	70.5	62.8	47.5	25.7	**10.6**	5.1	77.8	**66.4**
with DA	syn	98.6	95.4	82.4	73.2	61.4	50.2	26.7	9.3	**5.2**	69.5	36.0
	real	**100.0**	**97.7**	82.9	73.4	**66.3**	**53.5**	26.7	7.0	0.5	78.0	61.4
	joint	98.6	94.9	**83.0**	**74.8**	64.1	52.4	**27.3**	10.4	4.8	**78.5**	62.8
						VGG CNN-pool5						
		8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*100.0*	*99.1*	*85.0*	*75.7*	*70.0*	*56.5*	*34.2*	*10.1*	*1.0*	*87.6*	*82.0*
w/o DA	syn	91.7	83.3	77.3	64.9	53.6	40.9	18.6	5.3	2.9	63.6	42.2
	real	**100.0**	97.2	83.3	73.3	65.2	45.4	20.8	2.8	1.2	75.4	63.7
	joint	97.9	95.8	81.9	73.3	61.9	48.5	20.9	7.2	2.8	79.0	**66.8**
with DA	syn	**100.0**	97.2	82.8	74.1	62.8	49.1	22.6	8.6	3.2	76.9	39.8
	real	**100.0**	**99.5**	**84.5**	74.5	**69.5**	**53.6**	**27.7**	**10.5**	1.6	77.1	61.7
	joint	**100.0**	98.6	83.9	**74.9**	63.9	50.9	23.6	10.1	**3.4**	**81.5**	62.9
						VGG CNN-fc7						
		8/car	8/bike	8	16	24	36	72	180	360	8	16
	gt	*88.2*	*93.1*	*71.2*	*66.0*	*60.6*	*51.2*	*34.2*	*14.7*	*0.8*	*81.5*	*71.1*
w/o DA	syn	76.4	78.2	67.3	61.2	55.0	44.7	26.0	**11.9**	4.5	59.8	49.6
	real	84.7	**91.7**	69.0	**62.0**	55.0	45.9	25.2	**11.9**	0.2	**71.8**	57.3
	joint	84.7	87.5	68.6	**62.0**	54.3	45.0	26.2	9.5	**6.1**	70.6	58.1
with DA	syn	79.9	82.9	67.4	61.0	55.6	44.9	**26.8**	10.2	5.9	62.6	49.0
	real	**87.5**	91.2	68.5	61.8	**55.8**	**46.0**	26.0	10.6	0.4	71.1	57.3
	joint	**87.5**	88.0	**69.6**	61.6	53.9	44.6	25.9	9.5	5.6	70.4	**58.5**

Table 5.5: Pose estimation accuracy on unlabelled test data using real training data, synthetic data or both training sets. All datasets contain non-occluded object instances.

before, combining the refined real data and the adapted synthesized data for training performs well for any viewpoint discretisation $V = 8, 16, 24$. It is interesting to note that our weakly supervised approach (*with DA joint*) even outperforms the fully supervised approach (*gt*) due to the training data augmentation by the adapted synthetic images.

			PASCAL3D											
			VGG CNN-pool5											
views			aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
8	w/o DA	*gt*	*49.0*	*46.0*	*29.4*	*36.6*	*43.2*	*44.4*	*25.0*	*52.8*	*26.4*	*21.6*	*18.6*	*35.7*
		syn	**40.7**	47.9	22.7	42.0	37.2	36.1	**22.9**	48.4	37.4	**32.3**	31.3	36.3
	with DA	syn	34.6	51.7	18.9	45.5	41.5	41.1	16.7	51.7	43.3	31.6	**35.7**	37.5
		real	34.8	**52.1**	**26.8**	33.8	41.6	42.1	18.0	**56.4**	27.1	18.4	24.4	34.1
		joint	35.3	50.9	19.6	**47.8**	**43.9**	**42.3**	17.7	51.4	**46.2**	30.7	34.4	**38.2**
16	w/o DA	*gt*	*29.7*	*23.6*	*16.0*	*21.7*	*28.5*	*25.7*	*12.4*	*28.7*	*18.8*	*15.5*	*10.3*	*21.0*
		syn	**22.2**	23.9	11.9	**30.4**	25.6	22.1	10.6	28.5	24.0	23.3	15.9	21.7
	with DA	syn	20.9	25.1	**13.0**	28.8	27.5	**26.3**	10.1	30.5	**25.5**	20.9	18.4	22.5
		real	21.3	23.8	12.0	22.8	28.2	22.6	12.4	30.6	17.4	17.2	12.8	20.1
		joint	21.4	**26.4**	11.8	29.4	**29.3**	25.1	**16.6**	**31.2**	25.0	**23.9**	**25.7**	**24.2**
24	w/o DA	*gt*	*23.1*	*16.1*	*10.7*	*17.9*	*21.9*	*18.9*	*7.7*	*16.1*	*12.8*	*13.6*	*11.0*	*15.4*
		syn	14.8	18.0	7.8	21.0	18.6	15.6	**8.0**	20.0	18.0	14.9	9.9	15.1
	with DA	syn	16.0	**18.9**	8.0	22.9	19.8	16.7	7.7	20.8	18.2	**16.6**	**14.4**	16.4
		real	15.6	16.7	8.3	21.0	**21.5**	15.1	7.6	**21.2**	13.6	12.1	8.2	14.6
		joint	**18.4**	18.6	**8.7**	**24.1**	20.7	**17.0**	7.6	**21.2**	**18.3**	15.4	14.2	**16.7**

			ImageNet3D											
			VGG CNN-pool5											
			aero	bike	boat	bus	car	chair	table	mbike	sofa	train	tv	Avg.
8	w/o DA	*gt*	*59.2*	*66.4*	*55.4*	*51.4*	*87.6*	*42.9*	*38.8*	*66.6*	*31.7*	*22.9*	*33.8*	*50.6*
		syn	40.3	62.9	27.2	65.7	75.4	60.4	2.1	60.1	46.6	30.0	22.4	46.6
	with DA	syn	40.8	69.3	35.8	72.7	80.8	59.7	39.0	**64.1**	60.8	27.5	47.5	54.4
		real	**43.3**	67.4	**40.7**	58.4	**84.6**	48.8	33.3	64.0	28.5	21.0	39.2	48.1
		joint	43.1	**71.4**	39.1	**77.1**	83.6	**61.1**	**44.6**	63.7	**61.4**	**36.8**	**48.6**	**57.3**
16	w/o DA	*gt*	*42.3*	*44.4*	*39.0*	*35.6*	*71.2*	*29.4*	*24.1*	*41.0*	*22.3*	*22.8*	*16.2*	*35.3*
		syn	**30.1**	35.5	13.0	46.8	60.6	37.7	12.3	35.8	**36.3**	15.1	10.3	30.3
	with DA	syn	26.4	**47.3**	20.4	52.8	66.4	33.0	23.5	42.3	36.1	25.1	31.3	36.8
		real	24.5	42.8	**23.9**	39.3	**68.2**	24.3	16.5	34.8	26.2	16.7	18.2	30.5
		joint	29.0	46.1	23.3	**54.8**	67.5	**42.1**	**23.8**	**44.1**	35.0	**27.7**	**34.7**	**38.9**
24	w/o DA	*gt*	*28.5*	*32.1*	*30.8*	*31.7*	*62.6*	*2.7*	*18.9*	*28.1*	*13.6*	*12.1*	*13.1*	*26.8*
		syn	**20.4**	28.0	10.8	34.3	50.1	35.4	13.3	23.8	18.4	14.6	3.5	23.0
	with DA	syn	18.6	36.1	15.1	35.1	54.0	**37.2**	**20.4**	**39.6**	26.5	**21.1**	15.1	29.0
		real	16.7	15.9	3.5	30.3	57.0	26.1	13.2	28.3	21.1	13.8	9.5	21.4
		joint	18.6	**37.1**	**15.4**	**38.0**	**57.7**	36.7	19.6	31.5	**36.5**	18.2	**16.5**	**29.6**

Table 5.6: Pose estimation accuracy for the Pascal3D and ImageNet3D datasets that contain occlusions and truncated object instances.

5.3.3 Occlusion

In order to measure the actual impact of occluded instances, we also compare the viewpoint refinement for the Pascal3D and ImageNet3D datasets when we only take non-occluded object instances for training and testing (*non-occ*). As shown in Table 5.7, the accuracies for the setting with non-occluded instances in comparison to the complete dataset (*all*) are higher as expected. This is especially the case for Pascal3D since it contains a smaller portion of fully visible samples, i.e. 38% vs. 75%. The gain of our approach compared to the baseline, however, remains similar for *all* and *non-occ* with +6.7% and +5.0%, respectively. This shows that our approach is robust to occlusions.

For completeness, we also evaluate the scenario for viewpoint estimation. Table 5.8 reports the accuracies of all four combinations depending if the training or test data

		PASCAL3D											
		VGG CNN-pool5 - 24 views											
		aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
	non-occ/all	*0.68*	*0.32*	*0.61*	*0.55*	*0.34*	*0.19*	*0.08*	*0.36*	*0.15*	*0.33*	*0.52*	*0.38*
all	w/o DA	24.3	30.7	18.2	43.3	36.1	26.2	22.2	32.7	25.6	30.0	29.6	29.0
	with DA	**29.0**	**39.4**	**26.5**	**46.1**	**40.4**	**30.8**	**27.7**	**38.9**	**38.2**	**37.8**	**37.4**	**35.7**
non-occ	w/o DA	29.2	33.3	15.4	**49.1**	**56.6**	32.5	20.0	31.1	41.1	**50.6**	35.1	35.8
	with DA	**32.2**	**43.9**	**24.5**	44.6	54.0	**45.0**	**21.0**	**53.5**	**50.6**	39.9	**40.1**	**40.8**

		ImageNet3D											
		VGG CNN-pool5 - 24 views											
		aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
		0.91	*0.56*	*0.93*	*0.95*	*0.94*	*0.94*	*0.31*	*0.50*	*0.44*	*0.81*	*0.95*	*0.75*
all	w/o DA	33.5	40.4	26.0	53.9	**63.5**	44.1	33.2	34.2	42.0	22.1	22.1	37.7
	with DA	**35.2**	**50.1**	**30.2**	**57.1**	63.2	**47.4**	**44.5**	**43.1**	**56.1**	**26.7**	**22.5**	**43.3**
non-occ	w/o DA	34.3	43.1	27.1	55.6	**64.4**	47.4	32.7	34.0	**44.9**	25.5	22.6	39.2
	with DA	**36.2**	**50.4**	**30.5**	**57.3**	64.1	**49.0**	**45.1**	**49.7**	41.9	**43.2**	**26.5**	**44.9**

Table 5.7: Accuracy of the coarse-to-fine refinement with 24 fine viewpoints for the Pascal3D and ImageNet3D datasets. We compare the performance of our domain adaptation technique when taking all (*all*) or only non-occluded samples (*non-occ*).

		PASCAL3D											
target		VGG CNN-pool5 - 24 views											
train	test	aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
non-occ	all	14.0	**19.8**	**9.8**	21.0	18.1	14.1	**16.0**	20.0	**24.8**	15.7	11.8	**16.8**
all		**18.4**	18.6	8.7	**24.1**	**20.7**	**17.0**	7.6	**21.2**	18.3	15.4	**14.2**	16.7
non-occ	non-occ	16.8	18.1	**11.3**	**38.9**	33.1	**22.6**	**11.1**	**34.9**	18.3	**21.5**	**16.2**	**22.1**
all		**19.2**	**24.0**	8.7	24.1	**39.0**	17.0	7.6	21.0	18.3	15.4	14.2	19.0

		ImageNet3D											
		VGG CNN-pool5 - 24 views											
		aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
non-occ	all	18.6	32.2	**15.4**	**38.1**	**57.3**	**37.0**	18.1	27.6	23.2	**18.9**	15.5	27.4
all		18.6	**37.1**	**15.4**	38.0	**57.3**	36.7	**19.6**	**31.5**	**36.5**	18.2	**16.5**	**29.6**
non-occ	non-occ	18.6	35.3	**16.3**	36.9	59.1	38.3	**25.5**	35.0	30.4	**27.6**	15.6	30.8
all		19.0	**40.6**	15.8	**38.1**	**59.3**	**38.5**	24.0	**38.2**	36.7	22.0	**16.1**	**31.7**

Table 5.8: Pose estimation accuracy of our approach (*with DA joint*) for 24 fine viewpoints on the Pascal3D and ImageNet3D datasets. We compare the impact of training and testing with or without object occlusions.

contain occluded and truncated objects (*all*) or only fully visible objects (*non-occ*). For Pascal3D, the best average accuracies are obtained if occluded and truncated objects are discarded from the training data although the impact varies strongly among the object categories. For ImageNet3D, which contains by far less occluded samples, the best accuracy is achieved by taking all training samples. A major gain can be observed for the categories *bike*, *motorbike*, and *sofa*, which are the categories with the highest ratio of occluded or truncated samples.

5.3.4 Viewpoint Estimation using CNNs

In order to demonstrate that our approach not only works with linear SVMs but also with other methods for viewpoint estimation, we use our approach to train a state-of-the-art CNN approach for viewpoint estimation [Tulsiani and Malik, 2015], which also models viewpoint estimation as a classification task. In addition, we modify the CNN for viewpoint regression by using Huber loss H of the azimuth angle θ in a continuous representation $F(\theta) = [cos(\theta), sin(\theta)]$ as in Massa et al. [2016]. We augment the training data with mirrored samples and jittered ground-truth bounding boxes that overlap with the annotated bounding box with IoU > 0.7. We run a total of 40000 iterations for the CNNs trained with only real data and 60000 for those that include the synthetic data. In both cases, we start with a learning rate of 0.001 and decrease it by a factor of 10 each time a third of the iterations are completed.

The results for the Pascal3D dataset are given in Table 5.9 where we report the viewpoint estimation accuracy for 24 views as in the previous tables and the median error (*MedError*) as it was used in Tulsiani and Malik [2015]. When we train the CNN with classification loss on the training data with ground-truth labels, we achieve a lower median error and higher accuracy compared to the regression loss. This was already observed in Massa et al. [2016].

When the CNN is trained not on the ground-truth but on the refined viewpoint labels, our proposed approach with domain adaptation (*with DA*) outperforms the baseline (*w/o DA*) for all settings. Training on the synthetic and refined real training images (*joint*) also improves the accuracy and reduces the error compared to using the real training images only (*real*). We finally compare the CNN-based viewpoint classification Tulsiani and Malik [2015] with the linear SVMs (*DA LSVM*), which have been previously used for viewpoint estimation in Table 5.6. Using [Tulsiani and Malik, 2015] instead of linear SVMs improves the viewpoint accuracy by +8%. The results for ImageNet3D are reported in Table 5.10.

5.3.5 Cross-dataset Viewpoint Estimation

We finally perform a cross-dataset evaluation as in Sedaghat and Brox [2015]. We evaluate the viewpoint estimation of cars from the Multi-View Car Dataset, Pascal3D, ImageNet3D and the dataset [Sedaghat and Brox, 2015], denoted as *Freiburg*, whose bounding boxes and viewpoints of cars were annotated in a fully unsupervised manner. The *Freiburg* dataset contains recorded scenes of 47 cars for a total of 5836 training images, on a full 360° rotation. For viewpoint estimation, we use the CNN approach by Tulsiani and Malik [2015] as in Section 5.3.4 trained on the refined training data (*with DA*) and compare it with the approach by Sedaghat and Brox [2015]. The results reported in Table 5.11 show that our approach performs very well across datasets. Our approach outperforms Sedaghat and Brox [2015] for 11 out of 13 configurations. For some dataset combinations, the mean absolute error is reduced by about 14 degrees compared to Sedaghat and Brox [2015].

			PASCAL3D											
			MedError											
			aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
regression	real	*gt*	*20.6*	*23.9*	*42.6*	*8.1*	*18.0*	*21.2*	*22.6*	*20.9*	*15.3*	*15.8*	*15.1*	*20.4*
		w/o DA	**26.5**	29.9	58.0	13.0	26.3	25.7	31.9	27.3	**19.7**	24.8	**16.1**	27.2
		with DA	32.2	**25.2**	**57.5**	**12.5**	**25.8**	**24.0**	**29.2**	**25.3**	21.6	**22.5**	17.9	**26.7**
	joint	w/o DA	**28.4**	25.0	**53.9**	**10.3**	23.8	25.8	**30.5**	22.5	18.6	23.2	15.5	25.2
		with DA	31.8	**20.6**	56.5	10.6	**22.3**	**24.9**	31.4	**20.1**	**15.9**	**19.0**	**13.8**	**24.3**
classification	real	*gt*	*17.7*	*20.3*	*47.8*	*5.8*	*18.1*	*21.0*	*12.1*	*18.0*	*13.8*	*14.7*	*17.2*	*18.8*
		w/o DA	35.3	28.1	**52.3**	18.4	22.8	32.8	45.0	23.4	24.9	27.5	23.2	30.3
		with DA	**32.8**	**21.3**	70.4	**8.6**	**20.8**	**26.9**	**30.0**	**20.6**	**18.7**	**16.8**	**17.5**	**25.9**
	joint	w/o DA	**24.5**	20.0	**53.7**	7.2	18.1	25.6	**30.0**	21.3	**15.0**	15.0	20.5	22.8
		with DA	26.8	**17.5**	54.7	**6.9**	**16.5**	**23.3**	**30.0**	**18.3**	15.1	**14.6**	**16.3**	**21.8**
			24 views											
			aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
regression	real	*gt*	*13.4*	*17.3*	*8.6*	*18.9*	*23.5*	*18.1*	*6.6*	*19.3*	*13.3*	*11.7*	*11.6*	*14.8*
		w/o DA	**14.3**	13.9	7.6	10.7	19.0	14.7	6.8	16.4	**19.3**	**9.4**	12.8	13.2
		with DA	13.2	**19.1**	**8.0**	**16.6**	**20.2**	**15.0**	**10.2**	**19.5**	18.9	**9.4**	**13.5**	**14.9**
	joint	w/o DA	13.2	19.1	**7.9**	16.6	20.2	15.0	**10.2**	19.5	**18.9**	9.4	12.5	14.8
		with DA	**13.6**	**19.8**	7.8	**21.6**	**22.1**	**17.3**	7.8	**24.8**	13.3	**10.8**	**18.4**	**16.1**
classification	real	*gt*	*25.1*	*19.4*	*12.3*	*30.3*	*29.2*	*23.8*	*15.4*	*22.5*	*16.9*	*15.6*	*15.4*	*20.5*
		w/o DA	14.8	19.1	**8.8**	8.9	**26.2**	16.8	11.9	**22.0**	18.0	11.1	7.8	15.0
		with DA	**16.2**	**20.3**	5.0	**21.4**	24.9	**20.6**	**12.2**	19.4	**23.5**	**15.5**	**9.6**	**17.1**
	joint	w/o DA	**19.2**	25.2	10.8	33.7	**27.9**	23.7	16.2	21.4	21.6	18.4	12.8	21.0
		with DA	**19.2**	**27.2**	**14.9**	**35.7**	**27.9**	**24.8**	**18.9**	**27.4**	**33.8**	**19.8**	14.0	**24.0**
		DA LSVM	18.4	18.6	8.7	24.1	20.7	17.0	7.6	21.2	18.3	15.4	**14.2**	16.7

Table 5.9: Pose estimation accuracies for the Pascal3D dataset using Tulsiani and Malik [2015] for regression and classification.

			ImageNet3D											
			CNN-classification											
			aero	bike	boat	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
MedError	real	*gt*	*8.3*	*8.7*	*11.1*	*4.2*	*4.4*	*4.7*	*5.2*	*10.6*	*4.0*	*5.7*	*7.6*	*6.8*
		w/o DA	**20.2**	12.5	26.6	5.7	5.8	9.4	22.5	14.2	**6.9**	9.0	15.7	13.5
		with DA	22.5	**11.4**	**22.5**	**5.2**	**5.7**	**7.4**	**16.0**	**12.6**	7.5	**8.4**	**15.0**	**12.2**
	joint	w/o DA	**19.7**	10.1	22.7	5.6	5.6	8.3	20.6	12.5	**6.6**	8.9	14.4	12.3
		with DA	20.9	**8.7**	**21.6**	**5.1**	**5.5**	**6.9**	**15.2**	**11.5**	7.2	**8.5**	**13.5**	**11.3**
24 views	real	*gt*	*41.8*	*41.9*	*35.4*	*57.6*	*69.9*	*42.0*	*46.2*	*38.0*	*26.2*	*23.2*	*24.9*	*40.6*
		w/o DA	**24.8**	**36.6**	17.8	42.2	**60.7**	32.1	21.0	27.7	23.4	23.0	**24.9**	30.4
		with DA	22.4	36.3	**18.9**	**47.0**	59.7	**41.6**	**26.4**	**31.0**	**24.0**	**28.7**	23.3	**32.7**
	joint	w/o DA	**26.2**	42.6	**21.8**	46.2	62.2	29.7	21.3	37.7	30.5	27..8	25.1	33.7
		with DA	25.9	**46.8**	20.9	**51.0**	**62.3**	**47.9**	**26.8**	**39.3**	33.5	**28.0**	**30.5**	**37.5**
		DA LSVM	18.6	37.1	15.4	38.0	57.7	36.7	19.6	31.5	**36.5**	18.2	16.5	29.6

Table 5.10: Pose estimation accuracies for the ImageNet3D dataset using Tulsiani and Malik [2015] for classification.

		test			
train		Freiburg	Multi-View Car	Pascal3D	ImageNet3D
Freiburg	Sedaghat	-	34.4	61.5	38.0
	with DA		**21.5**	**58.0**	**27.3**
Multi-View Car	Sedaghat*	34.6	-	71.6	53.2
	with DA	**20.7**		**70.9**	**39.8**
Pascal3D	Sedaghat*	26.9	37.0	-	29.3
	with DA	**15.4**	**22.6**		**17.9**
ImageNet3D	Sedaghat	10.6	**17.4**	**47.7**	12.3
	with DA	**8.1**	18.7	51.0	**11.5**

Table 5.11: Viewpoint estimation across datasets. The mean absolute error of viewpoint estimation (in degrees) is reported. In the cases denoted by *, Sedaghat and Brox [2015] uses the entire dataset for training while we use only the training data of the dataset.

5.4 Summary

In this work, we have presented an approach for weakly supervised domain adaptation for the task of viewpoint estimation. It uses synthetic data to refine the viewpoint annotations of the coarsely labelled training images. Using coarse viewpoint annotations of real images as weak supervision together with accurately annotated synthesized images is not only a very efficient approach to collect training data for fine-grained viewpoint estimation, it also allows to achieve an accuracy that goes beyond the abilities of human annotators. An extensive evaluation on five datasets for viewpoint estimation showed that our approach outperforms generic domain adaptation methods, proves effective for a large number of object classes and presents a considerable tolerance against occlusions.

Chapter 6

Joint Viewpoint and Keypoint Estimation with Real and Synthetic Data

Contents

6.1 Introduction

In Chapter 5 we showed that the quality of training data for viewpoint estimation tasks highly improves if we combine real images with coarse viewpoint annotations, using minimal human supervision, and synthetic images from graphics models, which lack realism, but with accurate viewpoints. The domain shift that is present in the features extracted from both types of datasets is reduced by applying an unsupervised domain adaptation technique that aligns the synthetic data to the domain of real images. In consequence, the resulting training data naturally produces better viewpoint estimations. In this chapter, we present another approach that improves the accuracy of viewpoint estimation without increasing the time spent collecting new training data and the consequent labelling effort. Specifically, we exploit the spatial correlations between object keypoints and viewpoints and introduce a multi-task learning approach that also uses keypoint annotations to improve the quality of viewpoint estimations.

Many camera-based applications need to identify and analyse certain object classes for a better understanding of their surroundings. While 2D object detection is often a starting point, it is usually required to extract more detailed information from the detected objects. For instance, 2D keypoints provide additional details regarding the shape of an object and the 3D viewpoint provides the information about the orientation of an object. Both tasks, however, are correlated since the locations of the 2D keypoints depend on the orientation of the object and the 2D keypoints are a cue for the 3D orientation. Based on this implicit correlation, we introduce a joint model for 3D viewpoint and 2D keypoint estimation. The proposed network generalises the human pose estimator by Wei et al. [2016] to multiple objects and it is trained jointly for the two tasks. For the 3D viewpoint estimation, we propose a simple yet effective multi-granular viewpoint classification approach.

The labelling process for training our network requires nonetheless large amounts of accurate labelled data. While human annotations excel in annotating object instances by bounding boxes, they fail to accurately estimate fine 3D viewpoints (see Chapter 5). The same applies for annotating keypoints, which require pixel precision and a correct handling of occlusions. In order to alleviate the collection of training data, we propose two solutions. Firstly, we design our network such that it can be trained with images from different datasets. The datasets can provide annotations for only viewpoints, keypoints or both. Secondly, we make use of synthetic data to increase the amount of training samples since computer generated images are a quick way to collect many training samples, as well as precise ground truth. Specifically, we introduce a novel synthetic dataset that includes not only viewpoints, but also accurate keypoints.

We evaluate our method on 12 popular classes of the *ObjectNet3D* [Xiang et al., 2016] dataset, which contains both viewpoint and keypoint annotations. We demonstrate that our method outperforms current well established methods for multi-class viewpoint and keypoint estimation.

6.2 Joint Viewpoint and Keypoint Estimation

In this work, we propose a multi-task network that leverages 3D viewpoint and 2D keypoint estimation. We assume that an object has been already detected and our goal is to estimate the keypoints as well as the viewpoint. Our network is trained for all object classes $C = \{c_1, \ldots, c_{|C|}\}$ where the number of keypoints per object class K_c varies. A second important aspect of the network is that it can be trained on various types of data including real and synthetic data at the same time. Since the data might be annotated for only one of the two tasks, $\mathcal{M}$ denotes the set of training samples with viewpoint and 2D keypoint annotations, $\mathcal{N}$ denotes the set with only viewpoint annotations and $\mathcal{O}$ the set with only keypoint annotations. An overview of the proposed CNN architecture is presented in Figure 6.1. We first discuss the parts that are relevant for keypoint estimation.

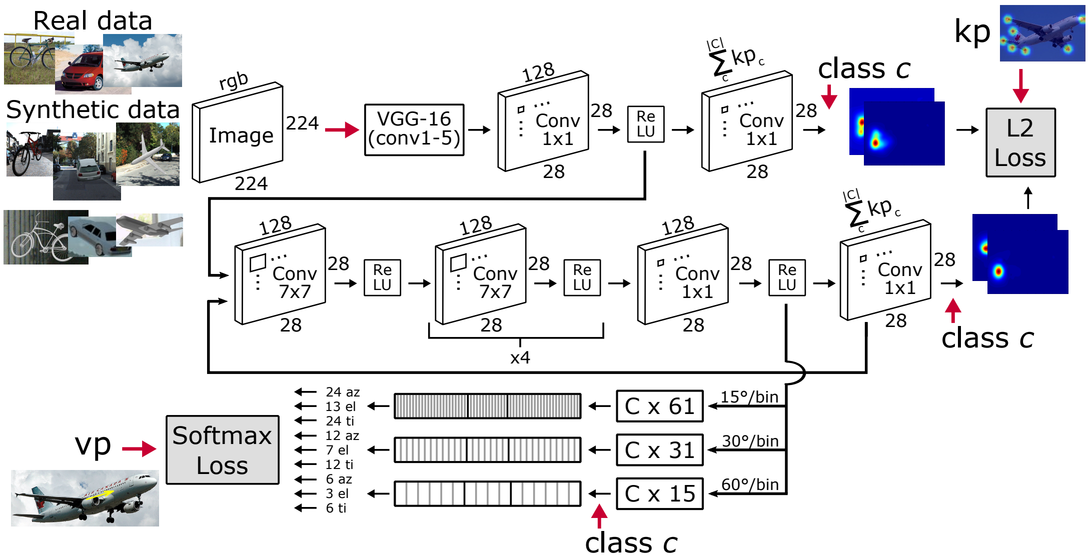


Figure 6.1: Overview of the proposed multi-class CNN for joint viewpoint and keypoint estimation. The network uses a multi-stage architecture. The first row shows the first stage, which predicts for each keypoint per class a heatmap. For the later stages (second row), the features of the first and the previous stage after the last ReLU are used as input. At each stage, an L2-loss is used, which compares the predicted heatmaps for the class of the training sample to the ground truth heatmaps. After the last stage, additional layers for viewpoint estimation are added (third row). We use a multi-resolution loss where fully connected layers map the $128 \times 28 \times 28$ features to nine vectors corresponding to three different discretisations (15°, 30°, 60°) of azimuth (az), elevation (el) and tilt (ti).

6.2.1 Keypoint Estimation

The proposed network is a multi-stage architecture with intermediate loss functions after each stage and the first part is similar to the convolutional pose machines proposed by Wei et al. [2016], which is a multi-stage network for 2D human pose estimation. The cropped image of an detected object is fed to a VGG-16 model [Simonyan and Zisserman, 2014] and additional convolutional layers are used to generate heatmaps for each keypoint and each object class. In total, we have $\sum_{c \in C} K_c$ heatmaps, where K_c denotes the number of keypoints of the c-th class. Since the object class c is known for an image during training, the L_2-loss is computed only for the heatmaps of the corresponding class. At the first stage $s = 1$, the loss is therefore given by

$$\mathcal{L}_{kp_s} = \sum_{x_i \in \{\mathcal{M}, \mathcal{O}\}} \frac{1}{K_{c_i}} \sum_{k=1}^{K_{c_i}} \|y_{i,k} - f_s(x_i)_{c,k}\|_2^2 \,, \tag{6.1}$$

where x_i denotes a training sample from the set $\mathcal{M}$ or $\mathcal{O}$ and $f_s(x_i)$ denotes all heatmaps that are predicted for the stage s. The estimated heatmap for the k-th keypoint of class c is then denoted by $f_s(x_i)_{c,k}$ and $y_{i,k}$ is the corresponding ground-truth heatmap for the training sample x_i. The L2-loss is computed over all pixels in the heatmap, but we write $\|a-b\|_2^2$ instead of $\sum_{\omega\in\Omega}\|a(\omega)-b(\omega)\|_2^2$.

As in Wei et al. [2016], we do not use one stage but 6 stages. For each stage except of the first one, we use the heatmaps of the previous stage and the feature maps of the first stage after the last ReLU layer as input. Since heatmaps are computed at each stage s, we sum the loss functions (6.1) over all stages, i.e. $\sum_s \mathcal{L}_{kp_s}$.

6.2.2 Viewpoint Estimation

As shown in Figure 6.1, the proposed network not only predicts the 2D keypoints but also the 3D viewpoint encoded by the three angles $\{\phi,\psi,\theta\}$, which denote azimuth ($\phi\in[0°,360°]$), elevation ($\psi\in[-90°,90°]$) and in-plane rotation ($\theta\in[-180°,180°]$), respectively. We opt for a classification-based approach to estimate the viewpoints and discretise each angle using a bin size of 15°. We obtain the probabilities for each bin by a fully connected layer and a softmax layer for each angle. The cross-entropy loss for bin size $b=15°$ is then given by

$$\mathcal{L}_{vp_b}=\sum_{x_i\in\{\mathcal{M},\mathcal{N}\}}\sum_{v\in\{\phi,\psi,\theta\}}-\log\left(f_b(x_i)_{c,v,v_i}\right), \tag{6.2}$$

where x_i denotes a training sample from the set $\mathcal{M}$ or $\mathcal{N}$, v_i denotes the ground-truth bin for angle v and $f_b(x_i)$ denotes the vector with the bin probabilities for all classes and angles. The estimated probability for the v_i-th bin of class c and angle v is then denoted by $f_b(x_i)_{c,v,v_i}$.

In addition, the network predicts during training the viewpoint for each class for two coarser discretisations of the angles, namely for 60° and 30°. In this way, the coarse discretisations guide the network to the correct bin of the finer discretisation and improve the accuracy as we will show as part of the experimental evaluation. The multi-task loss for the network is then expressed as

$$\mathcal{L}=\sum_s\mathcal{L}_{kp_s}+\sum_b\mathcal{L}_{vp_b}. \tag{6.3}$$

Since we aim at a finer viewpoint prediction than 15°, we upsample the estimated viewpoint probabilities to an angular resolution of 1° during inference. To this end, we interpolate the probabilities by applying a cubic filter [Keys, 1981] as illustrated in Figure 6.2. For the azimuth and the in-plane rotation, we convolve the discrete bins as a circular array.

6.3 Experiments

In this section we evaluate the performance of our method, denoted as JVK (*Joint Viewpoint and Keypoints*), and compare its results with several popular viewpoint and keypoint estimation algorithms. We train our network for 12 popular object categories, i.e. $|C|=12$, namely: *airplane*, *bicycle*, *boat*, *bottle*, *bus*, *car*, *chair*, *diningtable*,

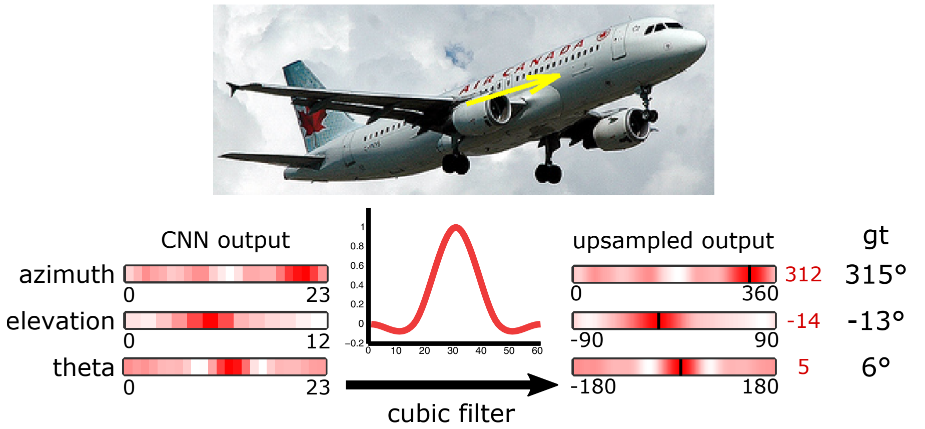


Figure 6.2: Using a cubic filter, the probabilities of the viewpoint quantised at 15° are upsampled to an angle resolution of 1°. Note that we have 24 bins for azimuth and θ since they are circular, but only 13 bins for elevation where the 7th bin is centred at zero elevation and the outer bins have only 7.5°.

motorbike, *sofa*, *train* and *tvmonitor*. We then evaluate our method on the test images of the *ObjectNet3D* [Xiang et al., 2016] dataset.

6.3.1 Datasets

ObjectNet3D [Xiang et al., 2016]

Large dataset that contains real images of 100 object categories. From all of them, the 12 classes that we selected include not just viewpoints from aligned 3D shapes, but also manually annotated keypoints. The selected subset is evenly separated between training and test data with 11421 and 11327 images, respectively. Most of the classes contain between 500 and 1000 samples in every set. The classes *bottle* and *diningtable* are above 1000 samples and *car* above 2000 samples.

ShapeNet [Chang et al., 2015]

Large-scale dataset of 3D shapes whose most relevant subset contains the 12 object categories, providing a considerable amount of models for each class. Although this setting allows for an extensive image dataset with a great variety of object orientations, the low quality of the renderings produce training samples that greatly differ from real images. This dataset only provides 3D viewpoints, automatically generated from the camera parameters in the image rendering. For our experiments, we make use of all models and generate 100000 images per class with random camera viewpoints, i.e. 1200000 images in total.

New Synthetic Data

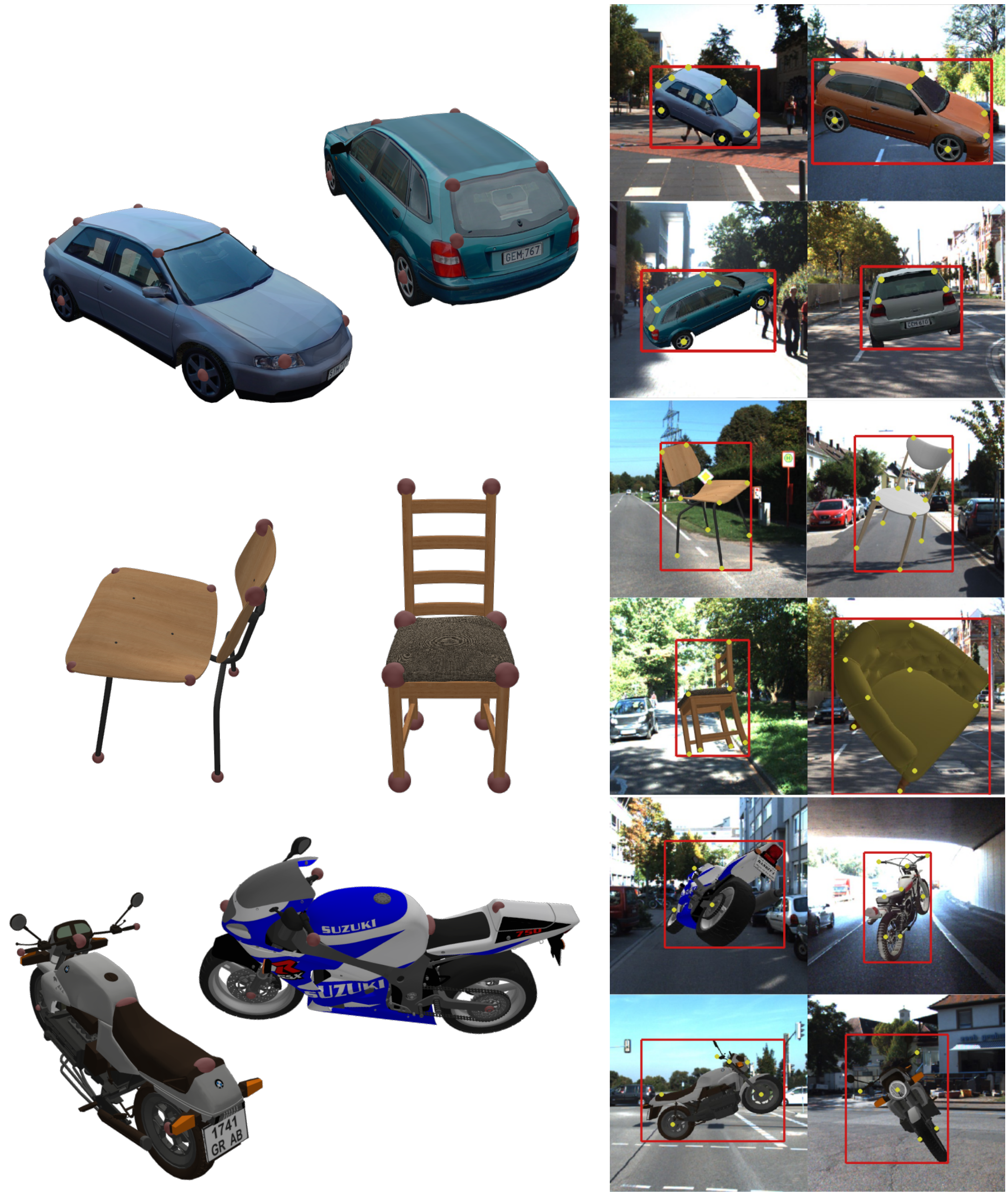
In this work, we introduce a new synthetic dataset from 3D graphics models for the 12 object categories. For each class, we collect 10 graphics models with higher levels of realism and more detailed meshes compared to ShapeNet. In addition to the 3D viewpoint annotations that are directly extracted from the camera rotation, we go one step further and introduce automatically generated 2D keypoints. In order to easily obtain keypoints from synthetic data, we firstly set deformable spheres in the 3D rendered model locations that we consider to be valid using the keypoints from ObjectNet3D as reference. Figure 6.3a shows some 3D graphics models with spheres placed as keypoints. Then, we project the centre of each sphere to pixel coordinates for a given camera orientation to create the 2D keypoints. For the projection, we take occlusions into account. We generate synthetic data with 10000 samples per class with random orientations. Examples of rendered images are illustrated in Figure 6.3b with the 2D bounding boxes and the visible 2D keypoints. The resulting images also include a background image from the KITTI dataset [Geiger et al., 2012].

6.3.2 Network Configuration

We train the proposed CNN model for a total of 150000 iterations when using only real images for training, 250000 iterations when including one of the two synthetic datasets and 350000 iteration for all 3 datasets. The weight decay is set to 0.0005 and the learning rate to 0.00005, which is multiplied by 0.1 every 100000 iterations. The input image will be cropped in all experiments to 224x224 pixels while preserving the aspect ratio. The batch contains 20 samples per iteration where we sample uniformly across the datasets if we use more than one for training. In addition, standard data augmentation techniques are employed during the training of the network: flipping, in-plane rotation $[-45°, 45°]$, image scaling (0.4,1.0) and translation. However, we only add the transformed image if the intersection over union of the transformed bounding box compared to the original one is above 0.8.

For the test phase, we will extract the samples of each object class using their annotated 2D bounding boxes, i.e. without any prior object detector. We run 5 passes with different scaling factors and average all of them to obtain the final confidence map of keypoints and 3D viewpoints.

From our model, we analyse two modifications. In JVK-KP, we only train the keypoint estimation, ignoring the viewpoint extension. Then, JVK denotes the standard network for both keypoint and viewpoint sections. We also modify the training datasets that we utilise, combining the real samples from ObjectNet3D [Xiang et al., 2016] with manually labelled viewpoints and keypoints (Re), ShapeNet [Chang et al., 2015] images with only viewpoints (Sh) and our novel synthetic dataset with generated viewpoints and keypoints (Sy).



(a) Rendered models with spheres as keypoints

(b) Generated 2D images

Figure 6.3: In (a) we show renderings of our graphics models with spheres that represent each keypoint for *cars*, *chairs* and *motorbikes*. In (b) we provide some examples of automatically generated images with their 2D bounding boxes and the projected keypoints that are visible.

ObjectNet3D (12 classes)		aero	bike	boat	bottle	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
PCK $\alpha = 0.1$	VpKp (192)	74.4	80.6	60.7	81.9	80.7	89.6	71.1	52.4	78.0	76.2	57.4	47.1	70.8
	VpKp (384)	80.1	88.6	70.7	90.0	93.7	96.5	76.7	65.4	85.2	89.1	68.7	78.7	82.0
	VpKp (192-384)	84.1	90.0	74.4	91.3	94.4	97.5	84.9	73.3	87.4	91.0	71.3	80.1	85.0
	VpKp (pLike)	82.7	90.7	69.2	92.6	95.8	95.6	89.5	76.3	85.9	92.5	72.0	80.3	85.3
	JVK-KP (Re)	85.7	92.7	74.8	**94.5**	98.1	98.4	89.4	83.9	89.7	93.8	73.4	75.7	87.5
	JVK (Re-Sh)	87.9	94.7	75.3	94.3	**98.6**	**98.5**	89.6	**84.5**	90.6	**94.0**	75.0	77.0	88.3
	JVK-KP (Re-Sy)	87.7	95.2	73.6	93.9	97.8	**98.5**	90.1	81.5	91.3	93.5	**75.2**	83.4	88.5
	JVK (Re-Sy)	88.8	95.2	75.1	93.6	98.0	**98.5**	90.9	83.6	91.2	93.8	73.3	82.3	88.7
	JVK (Re-Sy-Sh)	**89.5**	**95.9**	**77.1**	93.9	98.2	**98.5**	**91.5**	83.3	**93.0**	93.9	74.2	**84.0**	**89.4**

Table 6.1: Keypoint estimation on the ObjetNet3D dataset [Xiang et al., 2016] for 12 object classes. We report the keypoint localisation metric (PCK) introduced by Yang and Ramanan [2011].

6.3.3 Keypoint Estimation

To measure the quality of our keypoint localisation, we use the PCK[$\alpha = 0.1$] evaluation introduced by Yang and Ramanan [2011]. An estimated keypoint is valid if the Euclidean distance with respect to the corresponding ground truth is below $\alpha \times max(h, w)$, where h and w are the height and width of the object's bounding box, respectively.

As a baseline, we compare our method with the popular keypoint estimation for rigid objects [Tulsiani and Malik, 2015] (VpKp). We report the results of VpKp with 192x192 input resolution (192), 384x384 input resolution (384), both resolutions trained one after the other (192-384) and in a setting where the viewpoint is first estimated for the low resolution and used as input to refine the keypoints for the higher resolution (pLike).

We report the results in Table 6.1. Firstly, we observe that JVK-KP (Re), which uses the same real data as in VpKp, already outperforms all variations of VpKp. For instance, our method has +2.2% accuracy compared to VpKp (pLike). In contrast to VpKp that requires several sequential steps and higher resolutions, we only require a small amount of forward passes of our network with rescaled images. If we compare our modifications, we see a comparable improvement when including synthetic images with only keypoints, JVK-KP (Re-Sy), or only viewpoints, JVK (Re-Sh). This shows the benefits of estimating 3D viewpoint and 2D keypoints jointly. The network trained with all three training datasets (Re-Sy-Sh) obtains the best overall PCK accuracy, which is +0.7% higher compared to the result without Shapenet (Re-Sy).

6.3.4 Viewpoint Estimation

We evaluate our viewpoint estimation using two widely used metrics. The first metric [Tulsiani and Malik, 2015] is the geodesic distance between the ground truth and predicted rotation matrices from ϕ, ψ and θ, which is given by

$$\Delta(R_{gt}, R_{pred}) = \frac{||log(R_{gt}^T R_{pred})||_F}{\sqrt{2}}. \tag{6.4}$$

ObjectNet3D (12 classes)		aero	bike	boat	bottle	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
$Acc\frac{\pi}{6}$	Regression (Re)	0.799	0.810	0.667	0.933	0.928	0.967	0.908	0.793	0.830	0.961	0.949	0.897	0.870
	VpKp (Re)	0.887	0.794	0.743	0.917	0.967	0.963	0.922	0.823	0.808	0.954	0.957	0.831	0.880
	Render4CNN (Sh)	0.710	0.761	0.451	0.837	0.863	0.899	0.885	0.630	0.684	0.904	0.823	0.923	0.781
	Class-15 (Re)	0.836	0.770	0.719	0.896	0.954	0.950	0.904	0.848	0.766	0.954	0.935	0.791	0.860
	Class-15-30-60 (Re)	0.858	0.815	0.719	0.924	0.961	0.959	0.927	0.855	0.811	0.951	0.946	0.837	0.879
	Class-15-30-60 up. (Re)	0.867	0.825	0.735	0.928	0.959	0.966	0.931	0.857	0.816	0.960	0.945	0.852	0.887
	Class (Re-Sy)	0.884	0.858	0.765	0.945	0.969	0.968	0.956	0.865	0.885	0.965	0.943	0.875	0.907
	Class (Re-Sh)	**0.915**	0.854	0.803	0.945	0.976	0.973	0.975	0.868	0.866	0.978	0.955	0.899	0.917
	Class (Re-Sy-Sh)	0.907	0.857	**0.810**	0.938	0.980	0.971	**0.979**	0.883	0.885	0.979	0.946	0.903	0.920
	JVK (Re)	0.863	0.851	0.790	0.945	0.985	0.978	0.922	0.877	0.875	0.971	0.951	0.875	0.907
	JVK (Re-Sy)	0.898	**0.889**	0.786	**0.955**	0.983	0.974	0.935	0.873	0.905	0.972	0.940	0.889	0.917
	JVK (Re-Sh)	0.877	0.868	0.806	0.951	0.978	**0.983**	0.962	**0.892**	**0.913**	0.981	0.945	0.920	**0.923**
	JVK (Re-Sy-Sh)	0.878	0.870	0.798	0.950	**0.987**	0.975	0.960	0.866	0.907	**0.983**	**0.958**	**0.927**	0.922
MedError	Regression (Re)	13.4	16.7	18.6	8.2	4.3	4.8	9.9	11.5	16.4	9.1	6.4	13.0	11.0
	VpKp (Re)	12.2	16.0	15.4	12.7	6.8	8.9	11.6	11.1	16.8	12.3	8.0	14.0	12.2
	Render4CNN (Sh)	14.9	18.6	35.5	11.4	8.2	7.5	9.5	17.4	20.1	12.9	13.0	14.6	15.3
	Class-15 (Re)	13.0	17.0	15.8	10.0	5.9	8.1	10.3	9.3	18.1	11.7	8.1	15.0	11.9
	Class-15-30-60 (Re)	11.7	15.2	15.2	9.3	5.8	8.0	9.7	9.5	17.3	11.3	8.0	14.1	11.3
	Class-15-30-60 up. (Re)	9.8	13.8	13.6	8.6	4.5	5.5	7.6	7.3	15.6	9.4	6.9	13.2	9.7
	Class (Re-Sy)	9.0	12.5	12.5	8.0	4.2	5.1	7.2	6.8	13.0	8.6	6.1	11.4	8.7
	Class (Re-Sh)	**8.0**	11.5	11.2	8.4	4.2	4.9	6.9	6.7	13.0	8.3	6.0	10.5	8.3
	Class (Re-Sy-Sh)	8.3	10.9	**10.8**	**7.4**	4.2	4.4	6.9	6.5	12.3	7.9	6.0	10.2	8.0
	JVK (Re)	8.5	11.2	12.3	7.5	4.1	**3.7**	7.3	6.1	12.4	8.1	**5.5**	9.7	8.0
	JVK (Re-Sy)	8.3	**10.0**	12.0	**7.4**	**3.6**	**3.7**	**6.5**	6.0	**11.5**	7.7	5.6	**8.9**	**7.6**
	JVK (Re-Sh)	8.4	10.4	11.2	**7.4**	4.0	3.9	**6.5**	**5.6**	12.1	**7.5**	5.7	9.6	7.7
	JVK (Re-Sy-Sh)	8.1	10.7	11.4	7.6	4.0	3.8	7.2	6.0	11.7	7.7	5.9	9.5	7.8

Table 6.2: Viewpoint estimation on the ObjectNet3D dataset [Xiang et al., 2016] from ground truth bounding boxes. We report the percentage of estimated viewpoints with a geodesic error below $\pi/6$ rad ($Acc\frac{\pi}{6}$) and the median error (MedError).

The viewpoint is considered to be correct if the distance is below $\frac{\pi}{6}$ rad ($Acc\frac{\pi}{6}$). The second measure is the median error (MedError).

For this evaluation against other CNN-based approaches, we take as baseline a standard regression approach by Massa et al. [2016], where continous angles are seen as a circular array and represented in $\mathbb{R}^2$. VpKp [Tulsiani and Malik, 2015] proposes a classification-based viewpoint with also several discretisation levels. Then, Render4CNN [Su et al., 2015] presents a very fine discretisation with Gaussian filters to leverage the neighbouring bins by using millions of synthetic images. Finally, we re-train a VGG-16 [Simonyan and Zisserman, 2014] model for testing different classification-based configurations (Class): with only one level of discretisation (15°), our proposed approach with 3 quantisations with 15°, 30° and 60°, and including the upsampling with cubic filtering (up.).

The evaluation results for all the presented baselines and our configurations are shown in Table 6.2. Generally, we observe that the regression technique obtains similar results compared to other classification-based techniques. However, the cubic interpolation provides a significant reduction in median error and accuracy that favors classification approaches. Compared to the same configuration without upsampling, the error is reduced by $-1.6°$ and the accuracy increases by $+0.8\%$. The fine discretisaton of Render4CNN fails to compute robust viewpoints and ends up being the worst performing method by a large margin. Using real images from ObjectNet3D would not solve the problem, since the amount of training samples is too scarce for the large number of bins per angle. Class-15-30-60 outperforms Class-15, showing that learning several angle quantisations at the same time provides better results. When we compare JVK with Class, we observe that including a specific network for keypoint estimation allows for better viewpoint accuracies and reduced angle errors. JVK (Re) demonstrates to be superior compared Class upsampling (Re) by +2% in accuracy and $-1.7°$ in the median error. Although the gap is significantly smaller when training the networks with synthetic data, JVK trained with additional synthetic data achieves the best overall results. Specifically, the results of JVK trained on our new synthetic data are comparable to the ones using ShapeNet, but employing 10 times less samples. The better quality and additional labelled data of our dataset play an important role in improving the overall results.

6.3.5 Experimental Results on Pascal3D+

We also evaluate our method on the popular *Pascal3D+* dataset [Xiang et al., 2014], which also contains the same 12 classes evaluated on the *ObjectNet3D* dataset [Xiang et al., 2016]. Compared to *ObjectNet3D*, this dataset provides object instances that are not centred in the middle of the image and can thus be found in any image location with different resolutions, producing more challenging scenarios. We follow the standard protocol when the keypoint and viewpoint accuracies are jointly evaluated. That is, for training our network we make use of all 5790 samples included in the training subset together with 28769 additional samples from the ImageNet dataset [Deng et al., 2009]. We only report our results on the fully visible objects of the test dataset, for a total of 2136 samples, with roughly 200-300 samples per class.

We report our trained network JVK with only real images (Re) and with real images together with the synthetic examples from the ShapeNet dataset [Chang et al., 2015] (Re-Sh). For both keypoint and viewpoint estimation results, we compare our method with the popular work by Tulsiani et al. (VpKp) and the State-of-the-Art method by Zhou et al. [2018b] (StarMap).

In Table 6.3, we show that JVK (Re) outperforms VpKP by a large margin and obtains comparable results to StarMap, but in a single joint pass and using a worse performing base CNN (a comparison between VGG-16 and ResNet-152 is shown in Zhou et al. [2018b]). JVK (Re-Sh) obtains the best overall results, where additional synthetic data with only viewpoint annotations improves the keypoint accuracy (+1.8% with respect to JVK (Re)).

In Table 6.4, we show that JVK (Re) outperforms StarMap in both viewpoint accuracy and median error by +1.3% and -0.3, respectively. The best results are again obtained by JVK (Re-Sh), which improves by +1.9% the viewpoint accuracy and by -0.3 its median error compared to JVK (Re).

Pascal3D+ (12 classes)		aero	bike	boat	bottle	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
PCK $\alpha = 0.1$	VpKp	66.0	77.8	52.1	83.8	88.7	81.3	65.0	47.3	68.3	58.8	72.0	65.1	68.8
	StarMap	75.2	83.2	54.8	87.0	94.4	90.0	**75.4**	58.0	68.8	**79.8**	54.0	**85.8**	78.6
	JVK (Re)	77.3	87.0	**68.1**	90.1	97.9	95.4	63.6	75.8	84.9	75.9	57.7.	70.7	78.7
	JVK (Re-Sh)	**80.4**	**89.9**	63.7	**90.5**	**98.0**	**96.8**	70.2	**78.9**	**86.7**	77.4	**63.6**	68.9	**80.5**

Table 6.3: Keypoint estimation on the Pascal3D+ dataset [Xiang et al., 2014] for 12 object classes.

Pascal3D+ (12 classes)		aero	bike	boat	bottle	bus	car	chair	dtable	mbike	sofa	train	tv	Avg.
$Acc_{\frac{\pi}{6}}$	VpKp	0.81	0.77	0.59	0.93	0.98	0.89	0.80	0.62	0.88	0.82	0.80	0.80	0.808
	StarMap	0.82	0.86	0.50	0.92	0.97	0.92	0.79	0.62	0.88	0.92	0.77	0.83	0.823
	JVK (Re)	0.82	0.79	0.61	0.97	0.98	**0.95**	0.83	**0.65**	**0.89**	**0.97**	**0.83**	0.78	0.836
	JVK (Re-Sh)	**0.83**	**0.89**	**0.63**	**0.97**	**0.99**	0.94	**0.89**	**0.65**	0.86	0.90	0.82	**0.84**	**0.855**
MedError	VpKp (pLike)	13.8	17.7	21.3	12.9	5.8	9.1	14.8	15.2	14.7	13.7	8.7	15.4	13.6
	StarMap	**10.1**	14.5	30.0	9.1	**3.1**	6.5	11.0	23.7	14.1	11.1	7.4	13.0	10.4
	JVK (Re)	12.2	12.5	19.0	6.8	4.6	5.0	11.1	**9.4**	12.6	**10.6**	**5.9**	12.3	10.1
	JVK (Re-Sh)	11.3	**12.2**	**18.1**	**6.7**	3.8	**4.7**	**9.6**	11.6	**11.9**	11.1	6.3	**10.3**	**9.8**

Table 6.4: Viewpoint estimation on the Pascal3D+ dataset [Xiang et al., 2014] for 12 object classes.

6.3.6 Qualitative Results

For completeness, we also show some qualitative results in Figure 6.4. For each class, we show the results for the first three test images of ObjectNet3D [Xiang et al., 2016]. We observe that the predicted 2D keypoints and 3D viewpoints are in alignment. The majority of the few wrongly estimated keypoints and viewpoints are due to lateral symmetries of objects.

Our deep neural network also failed in some especial object configurations. As depicted in Figure 6.5, ambiguous image symmetries were not correctly handled by our technique in some cases. Additionally, the vast intra-class variation of many objects promotes misplacements in special samples. An attached refinement to dynamically adapt the resulting keypoints based on the test image might be of great interest.

6.4 Summary

In this chapter we have presented an approach for joint viewpoint and keypoint estimation for multiple rigid object classes. The approach includes a simple yet effective branch for viewpoint estimation with different discretisation levels and cubic upsampling that produce more accurate results. In contrast to previous methods that train a separate approach for each task, we have shown that viewpoint and keypoint estimation benefit from each other. Our approach also handles different kinds of training datasets containing real or synthesized images, as well as datasets where only one of the tasks is annotated. While the main strength of the approach presented in Chapter 5 relies on the accurate refinement of training data, this approach does not require additional labels, but uses annotations from a correlated task. We evaluated our approach on the ObjectNet3D and Pascal3D datasets, where it outperforms previous approaches.

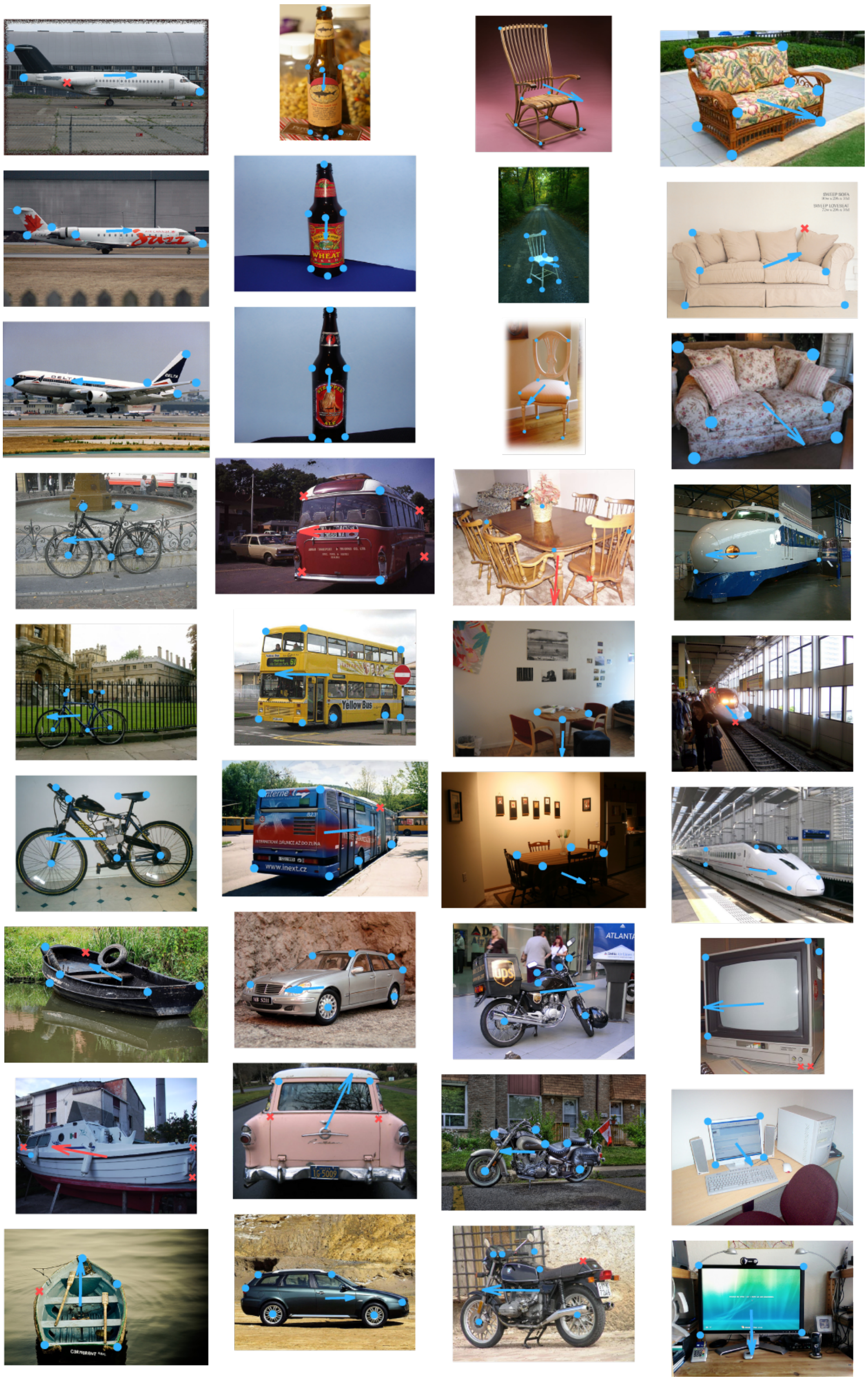
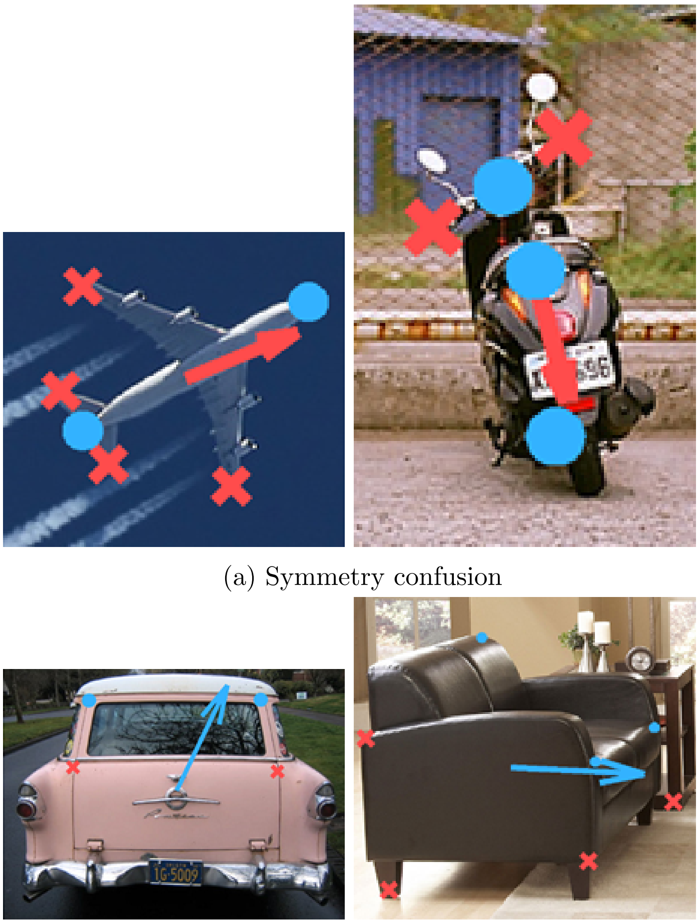


Figure 6.4: Qualitative results for the proposed approach JVK (Re-Sy-Sh). The directional arrow represents the projected 3D viewpoint. Blue (dots) and red (crosses) denote correct and wrong estimations based on the PCK$[\alpha = 0.1]$ or Acc$(\pi/6)$ measure, respectively.



(a) Symmetry confusion

(b) Class unexpected variations

Figure 6.5: Common failure test cases that could not handle symmetry (a) and sample-specific keypoint variations (b).

Chapter 7

Conclusions

Contents

7.1 Overview and Discussion

Currently, image recognition techniques require large amounts of images with reliable annotations in order to train their image classifiers. However, the gathering of meaningful images that best generalise the test scenario becomes quite frequently a challenging task. Besides, the labelling process is very time consuming, expensive and prone to errors. This means that the access to fast, cheap and accurate labelled data arises as one of the main challenges in the field of computer vision. Even in some special situations, no data collection for training purposes is possible at all. Several solutions have been proposed in the last years to solve this critical situation. The most popular approaches include (1) the usage of similar datasets from previous projects that have already been annotated, (2) the generation of synthetic data from 3D computer graphics models and (3) the simultaneous training of datasets with heterogeneous labels but with a correlated task, i.e. the understanding of the object class that must be recognised. In this dissertation, we proposed 3 contributions that optimise and improve the classification performance for each of these solutions. We introduce a novel domain adaptation method that acclimates to a variety of applications, including open sets (see Section 7.1.1), we refine coarsely annotated object viewpoints with fine labels from domain adapted synthetic images (see Section 7.1.2) and combine the estimation of viewpoints and keypoints on a single end-to-end deep neural network to increase the accuracies of both tasks at the same time (see Section 7.1.3), respectively.

7.1.1 Open Set Domain Adaptation for Image and Action Recognition

We introduced in this dissertation a novel domain adaptation algorithm that linearly transforms the source data towards the target data in an iterative approach. We exploit the association between source classes and target samples, treated as a bipartite matching problem, to guide the computation of the linear transformation in every iteration. This layout is extremely flexible to modifications. Therefore, we also extend standard adaptation techniques to open sets, namely open set domain adaptation, and include irrelevant samples in the target domain, i.e. they do not belong to any of our classes of interest, presenting a more realistic scenario. Introducing an outlier handling in the assignment step of our algorithm, we reject uninteresting target samples for the optimisation of the source data transformation. We also attach unknown samples in the source dataset, showing that using them in the pipeline as unknown class might still provide marginal accuracy increments. Another major strength of our approach is that it reports better classification accuracies for a wide variety of problems and input data, including object classification (image data), action recognition (video data) and sentiment analysis (words), and different feature descriptors. An extensive validation on popular evaluation datasets shows that our algorithm not only outperforms well-established domain adaptation methods that are directly applied to feature descriptors, but also achieves similar or even better results than state-of-the-art domain adaptation approaches embedded in deep neural networks.

7.1.2 Viewpoint Refinement and Estimation with Adapted Synthetic Data

Another proposal of this thesis is the alleviation of fine viewpoint labelling by humans and the utilisation of synthetic data to refine the labels of real training images. We have evaluated our approach in the context of pose estimation, where the real images are manually labelled by only four coarse views, but finer viewpoint estimates are required. Due to the differences between the real and the synthetic data, we apply domain adaptation to align both domains and improve the viewpoint refinement. For domain adaptation, we consider the real images as weakly labelled data and use the coarse views to constrain the learning of the transformation from the synthetic data to the real data. The results have shown that 3D generated models can be successfully used to refine labels in real images and therefore overcome the cumbersome annotation of real images by accurate and fine viewpoints. In particular, our approach leverages the abilities of humans of estimating coarse viewpoints and the pose accuracy of synthetic data.

7.1.3 Joint Viewpoint and Keypoint Estimation with Real and Synthetic Data

We finally presented an end-to-end multi-task neural network that jointly trains viewpoints and keypoints of rigid objects. This architecture utilises a state-of-the-art human pose estimation as backbone [Wei et al., 2016] to replicate a multi-stage model and extends it by appending a new branch for viewpoint estimation with different levels of discretisation. The combined training of tasks that are highly correlated, i.e. both seek a better understanding of the object's positioning, outputs better viewpoint accuracies and more precise keypoints. This finding is clearly shown when the keypoint estimation improves by simply adding more training data with only viewpoint annotations. Nonetheless, some failure cases appeared in objects that produced a keypoint layout with symmetries, which led to mirrored results.

Another advantage of our model is that it can be trained with multiple datasets at the same time. For instance, we introduce a new synthetic dataset with automatically generated viewpoint and keypoints that further improves the results of both tasks. This method thus fits into applications where no additional labelling is allowed and the re-utilisation of other datasets with different labelling specifications, e.g. viewpoints, keypoints or both, is necessary. Hence, it becomes a better choice than our proposed method for viewpoint refinement and estimation of Chapter 5, whose coarse viewpoint annotations by humans is a fundamental step to support the unsupervised domain adaptation that follows for each coarse view. However, a drawback of our network is that fine human annotations are not reliable and might lead to worse accuracies than our method with the viewpoint refinement. A potential combination of both methods is later detailed in Section 7.2.3.

7.2 Future Work

7.2.1 Advances in Domain Adaptation

The domain adaptation algorithm presented in this work allows for some interesting extensions that go beyond the scope of this dissertation.

Open Set Domain Adaption with Deep Convolutional Neural Networks

The breakthrough of powerful programmable graphics cards and efficient deep neural networks also had a significant impact in the performance of domain adaptation in object classification tasks. The transfer from source to target domain is, at some extent, implicitly learned in the training of the network models and Siamese networks are perfectly suited to add adaptation loss functions across multiple datasets. Therefore, the next natural step of our open set domain adaptation is its transition to deep neural networks and consequently the exploitation of its usability in more powerful supervised learning techniques. A first proposal has already been introduced in the recent publication by Saito et al. [2018b].

Beyond Object Classification Tasks

Standard domain adaptation algorithms, including the one presented in this work, expect as input data a set of features $X \in \mathbb{R}^{N \times D}$ of sample size N and feature dimensionality D. We consider of great importance the investigation of domain adaptation given any type of feature descriptor, even at image level. At this point in time, some applied works in the field of semantic segmentation [Zhang et al., 2017b] have been published. More focused on our work, domain adaptation with image sequences processed in long short-term memories, i.e. recurrent neural networks, arise as a plausible extension for action recognition tasks.

7.2.2 Synthetic Data from 3D Graphics Models

In this thesis, we introduced a novel synthetic dataset with automatised generation of 3D viewpoints and 2D keypoints for any available 3D mesh. Synthetic data has, nevertheless, a lot of potential and still has many new topics of research.

Generation of 3D Annotations

Many new datasets are very rich in labelled data and provide many types of annotations for different tasks. However, they do not provide higher levels of detail in the annotation of object classes. Our framework for generating synthetic data is well suited with just a few extensions to automatically provide 3D keypoints and depth maps. For animated meshes, the inclusion of scene flow also provides relevant information and the camera can be easily modified to support any sensor specification. All these increments are of great interest when developing multi-task neural networks and real data is scarce.

Generation of New Models from Real Data

The time spent in curating 3D graphics models for the generation of synthetic data becomes a tedious bottleneck. Therefore, another interesting future work is the automatised creation of 3D meshes from real images. Using a set of existing 3D models and a few images of the real objects in all possible viewpoints might suffice to provide interesting results. In extreme cases, partial annotations of the visible viewpoints on single images may also be very helpful in the data collection. An early work has already been presented by Krause et al. [2013].

Impact of Photorealistic Synthetic Data and Dataset Size

There are two parameters that play an important role in the usage of synthetic data for training object classifiers that did not receive enough attraction. Firstly, the impact of investing more time in extremely realistic renderings for better results in the classifiers. How much level of realism is enough to get an improvement? Can photorealistic virtual worlds replace real data if properly tuned for the expected test environment? In the same direction, some application might request the minimum number of 3D graphics models and images per model that are required to get the best possible results. Assuming that generic tasks with large class variations demand more data: how many additional images are enough for training our object classifier?

7.2.3 Viewpoint and Keypoint Estimation

In spite of presenting in this work state-of-the-art viewpoint and keypoint estimators, we still consider a few improvements in our pipeline.

Domain Adaptation from Synthetic to Real Domain

Compared to the methods presented in Chapters 4 and 5, our multi-task learning approach does not employ any domain adaptation technique to bridge the gap between synthetic and real data. Based on the current research, which already reports promising results when domain adaptation is applied to deep neural networks [Ganin and Lempitsky, 2015, Saito et al., 2018a, Long et al., 2018, Kang et al., 2019], the inclusion of additional losses that minimise the discrepancy between both domains, while maximising the discrepancy among discretised viewpoints, become an interesting extension.

In order to reduce the risks of inaccurate human viewpoint annotations, we can coarsely re-label the viewpoints used in the training phase and apply the viewpoint refinement presented in Chapter 5. This step does not even require re-annotation, since the manually-annotated fine viewpoints can be transformed into coarse views and then further refined with the synthetic dataset. Furthermore, the refinement can also be easily integrated into the CNN network by simply providing coarse annotations and use each coarse view as independent cluster of the aforementioned discrepancy losses, in order to adapt the fine viewpoints from the synthetic data into the coarse ones from the real data.

Inclusion of Poses in Object Detectors

A straightforward extension of our viewpoint and keypoint estimator is its inclusion into a state-of-the-art object detector and exploiting its 3D information. This means that a 3D bounding box is jointly trained with our estimators. This extension is similar in spirit with the work published by Braun et al. [2016].

From 2D to 3D Pose Estimation Techniques

2D Keypoint estimation implicitly contains 3D spatial information based on the layout of their detected keypoints. This information is, however, ambiguous is some configurations. Therefore, we propose the training of 3D keypoint estimators in case 3D synthetic annotations, i.e. depth information, is available.

Bibliography

Tobias Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009. (Cited on page 39.)

Y. Aytar and A. Zisserman. Tabula rasa: model transfer for object category detection. In *IEEE International Conference on Computer Vision*, pages 2252–2259, 2011. (Cited on page 28.)

M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *IEEE International Conference on Computer Vision*, pages 769–776, 2013. (Cited on page 27.)

Peter L Bartlett and Marten H Wegkamp. Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9(6):1823–1840, 2008. (Cited on page 29.)

Vasileios Belagiannis and Andrew Zisserman. Recurrent human pose estimation. In *IEEE International Conference on Automatic Face & Gesture Recognition*, pages 468–475, 2017. (Cited on page 31.)

Abhijit Bendale and Terrance E. Boult. Towards open set deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1563–1572, June 2016. (Cited on pages 29 and 37.)

Yoshua Bengio, Grégoire Mesnil, Yann Dauphin, and Salah Rifai. Better mixing via deep representations. In *International Conference on Machine Learning*, volume 28, pages 552–560, 2013. (Cited on pages 37 and 57.)

John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Conference on empirical methods in natural language processing*, pages 120–128, 2006. (Cited on page 28.)

John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *Annual Meeting of the Association of Computational Linguistics*, pages 440–447, 2007. (Cited on pages xii, 37, 61 and 63.)

Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. (Cited on page 15.)

Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3722–3731, 2017. (Cited on page 29.)

Markus Braun, Qing Rao, Yikang Wang, and Fabian Flohr. Pose-rcnn: Joint object detection and pose estimation using 3d object proposals. In *International Conference on Intelligent Transportation Systems*, pages 1546–1551, 2016. (Cited on page 107.)

L. Bruzzone and M. Marconcini. Domain adaptation problems: a DASVM classification technique and a circular validation strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):770–787, 2010. (Cited on page 28.)

Fabio Maria Cariucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Autodial: Automatic domain alignment layers. In *IEEE International Conference on Computer Vision*, pages 5077–5085, 2017. (Cited on page 7.)

Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Autodial: Automatic domain alignment layers. In *IEEE International Conference on Computer Vision*, pages 5077–5085, 2017. (Cited on pages 28 and 59.)

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4733, 2017. (Cited on page 56.)

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3D model repository. *CoRR*, abs/1512.3012, 2015. (Cited on pages 32, 93, 94 and 98.)

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. (Cited on page 42.)

Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3339–3348, 2018. (Cited on page 29.)

S. Chopra, S. Balakrishnan, and R. Gopalan. DLID: Deep learning for domain adaptation by interpolating between domains. In *ICML workshop on challenges in representation learning*, 2013. (Cited on page 35.)

Xiao Chu, Wei Yang, Wanli Ouyang, Cheng Ma, Alan L. Yuille, and Xiaogang Wang. Multi-context attention for human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5669–5678, 2017. (Cited on page 31.)

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016. (Cited on page 33.)

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20 (3):273–297, 1995. (Cited on pages v, 3, 4 and 15.)

Gabriela Csurka, Boris Chidlowskii, Stéphane Clinchant, and Sophia Michel. Unsupervised domain adaptation with regularized domain instance denoising. In *IEEE European Conference on Computer Vision*, pages 458–466, 2016. (Cited on page 27.)

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision*, pages 1–22, 2004. (Cited on page 22.)

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005. (Cited on pages v, v, vi, 5, 23, 30 and 74.)

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. (Cited on pages 25, 73 and 98.)

Gilad Divon and Ayellet Tal. Viewpoint estimation—insights & model. In *IEEE European Conference on Computer Vision*, pages 252–268, 2018. (Cited on pages 30 and 65.)

Andrei Dobrescu, Mario Valerio Giuffrida, and Sotirios A Tsaftaris. Leveraging multiple datasets for deep leaf counting. In *IEEE International Conference on Computer Vision*, pages 2072–2079, 2017. (Cited on page 3.)

Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *IEEE International Conference on Computer Vision*, pages 2051–2060, 2017. (Cited on page 7.)

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 *International Conference on Machine Learning*, pages 647–655, 2014. (Cited on page 28.)

Mohamed Elhoseiny, Tarek El-Gaaly, Amr Bakry, and Ahmed Elgammal. A comparative analysis and study of multiview CNN models for joint object categorization and pose estimation. In *International Conference on Machine Learning*, pages 888–897, 2016. (Cited on page 30.)

M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12): 2179–2195, 2009. (Cited on pages v and 5.)

A. Ess, B. Leibe, K. Schindler, , and L. van Gool. A mobile vision system for robust multi-person tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. (Cited on pages v and 5.)

M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. (Cited on page 73.)

Gabriele Fanelli, Matthias Dantone, Juergen Gall, Andrea Fossati, and Luc Van Gool. Random forests for real time 3D face analysis. *International Journal of Computer Vision*, 101(3):437–458, 2013. (Cited on page 30.)

L. Fei-Fei, R. Fergus, and A. Torralba. Recognizing and learning object categories, 2005. URL `http://people.csail.mit.edu/torralba/shortCourseRLOC/`. (Cited on pages vi and 22.)

P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. (Cited on pages 30 and 31.)

Michele Fenzi, Laura Leal-Taixe, Bodo Rosenhahn, and Jorn Ostermann. Class generative models based on feature regression for pose estimation of object categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 755–762, 2013. (Cited on page 30.)

B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *IEEE International Conference on Computer Vision*, pages 2960–2967, 2013. (Cited on pages 42, 45, 59, 75, 77 and 78.)

Sanja Fidler, Sven Dickinson, and Raquel Urtasun. 3D object detection and viewpoint estimation with a deformable 3D cuboid model. In *Advances in Neural Information Processing Systems*, pages 611–619, 2012. (Cited on page 30.)

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015. (Cited on pages 28, 29, 35, 43, 59 and 107.)

Efstratios Gavves, Thomas Mensink, Tatiana Tommasi, Cees G. M. Snoek, and Tinne Tuytelaars. Active transfer learning with zero-shot priors: Reusing past datasets for future tasks. In *IEEE International Conference on Computer Vision*, pages 2731–2739, 2015. (Cited on page 29.)

A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012. (Cited on pages viii, 33, 68, 69, 72, 73, 77 and 94.)

Muhammad Ghifary, W Bastiaan Kleijn, and Mengjie Zhang. Domain adaptive neural networks for object recognition. In *Pacific Rim International Conference on Artificial Intelligence*, pages 898–904, 2014. (Cited on page 28.)

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *IEEE European Conference on Computer Vision*, pages 597–613, 2016. (Cited on page 37.)

Amir Ghodrati, Marco Pedersoli, and Tinne Tuytelaars. Is 2D information enough for viewpoint estimation? In *British Machine Vision Conference*, pages 1–12, 2014. (Cited on page 30.)

Behnam Gholami, Ognjen Rudovic, and Vladimir Pavlovic. Punda: Probabilistic unsupervised domain adaptation for knowledge transfer across visual categories. In *IEEE International Conference on Computer Vision*, pages 3601–3610, 2017. (Cited on page 27.)

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014. (Cited on page 30.)

Daniel Glasner, Meirav Galun, Sharon Alpert, Ronen Basri, and Gregory Shakhnarovich. Aware object detection and pose estimation. In *IEEE International Conference on Computer Vision*, pages 1275–1282, 2011. (Cited on page 72.)

Daniel Glasner, Meirav Galun, Sharon Alpert, Ronen Basri, and Gregory Shakhnarovich. Aware object detection and continuous pose estimation. *Image and Vision Computing*, 30(12):923–933, 2012. (Cited on page 30.)

B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073, 2012. (Cited on pages xii, 27, 35, 37, 43, 44, 59, 60, 62, 67, 75, 77 and 78.)

B. Gong, K. Grauman, and F. Sha. Reshaping visual datasets for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1286–1294, 2013a. (Cited on pages vii, xi, xi, xi, xii, xii, 28, 43, 44, 47, 50, 59, 60 and 62.)

B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 222–230, 2013b. (Cited on pages xii, 28, 61 and 63.)

R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 999–1006, 2011. (Cited on pages 27 and 35.)

Alexander Grabner, Peter M. Roth, and Vincent Lepetit. 3D pose estimation and 3D model retrieval for objects in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2018. (Cited on page 31.)

Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems*, pages 513–520, 2006. (Cited on page 28.)

Chunhui Gu and Xiaofeng Ren. Discriminative mixture-of-templates for viewpoint classification. In *IEEE European Conference on Computer Vision*, pages 408–421, 2010. (Cited on page 30.)

Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey Vision Conference*, pages 10–5244, 1988. (Cited on page 21.)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. (Cited on page 24.)

Kun He, Leonid Sigal, and Stan Sclaroff. Parameterizing object detectors in the continuous pose space. In *IEEE European Conference on Computer Vision*, pages 450–465, 2014. (Cited on pages 30 and 68.)

M. Hejrati and D. Ramanan. Analysis by synthesis: 3D object recognition by object reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2449–2456, 2014. (Cited on pages 30 and 32.)

Samitha Herath, Mehrtash Tafazzoli Harandi, and Fatih Porikli. Learning an invariant hilbert space for domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3956–3965, 2017. (Cited on page 27.)

J. Hoffman, E. Rodner, J. Donahue, K. Saenko, and T. Darrell. Efficient learning of domain-invariant image representations. In *International Conference on Learning Representations*, 2013. (Cited on pages 28, 67 and 76.)

J. Hoffman, E. Rodner, J. Donahue, B. Kulis, and K. Saenko. Asymmetric and category invariant feature transformations for domain adaptation. *International Journal of Computer Vision*, 109(1-2):28–41, 2014. (Cited on page 35.)

Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5001–5009, 2018. (Cited on page 29.)

Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. Multi-class open set recognition using probability of inclusion. In *IEEE European Conference on Computer Vision*, pages 393–409, 2014. (Cited on page 29.)

S. G. Johnson. The NLopt nonlinear-optimization package, 2007–2010. URL `http://ab-initio.mit.edu/nlopt`. (Cited on pages 41 and 70.)

Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. (Cited on pages 28 and 107.)

L Kaufman and Fernand Broeckx. An algorithm for the quadratic assignment problem using bender's decomposition. *European Journal of Operational Research*, 2(3):207–211, 1978. (Cited on page 40.)

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017. (Cited on pages viii, xiii, xiii, 38, 55, 56, 57, 125, 126 and 127.)

Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *IEEE International Conference on Computer Vision*, 2017. (Cited on page 30.)

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, 2018. (Cited on page 7.)

Robert G. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 29(6):1153–1160, 1981. (Cited on page 92.)

Piotr Koniusz, Yusuf Tas, and Fatih Porikli. Domain adaptation by mixture of alignments of second-or higher-order scatter tensors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7139–7148, 2017. (Cited on page 29.)

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision*, pages 554–561, 2013. (Cited on page 106.)

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. (Cited on pages 20, 24, 28, 30, 43, 45, 57, 59, 67 and 74.)

H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. (Cited on page 72.)

B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: domain adaptation using asymmetric kernel transforms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1785–1792, 2011. (Cited on page 28.)

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2169–2178, 2006. (Cited on page 22.)

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. (Cited on pages vi, 18 and 20.)

Bastian Leibe, Ales Leonardis, and Bernt Schiele. Combined object categorization and segmentation with an implicit shape model. In *IEEE European Conference on Computer Vision*, pages 17–32, 2004. (Cited on page 30.)

Fayin Li and Harry Wechsler. Open set face recognition using transduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1686–1697, 2005. (Cited on page 29.)

J. Liebelt and C. Schmid. Multi-view object class detection with a 3D geometric model. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1688–1695, 2010. (Cited on pages 30 and 72.)

An-An Liu, Yu-Ting Su, Wei-Zhi Nie, and Mohan Kankanhalli. Hierarchical clustering multi-task learning for joint human action grouping and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):102–114, 2016. (Cited on page 32.)

Jonathan L Long, Ning Zhang, and Trevor Darrell. Do convnets learn correspondence? In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2014. (Cited on page 31.)

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015. (Cited on pages 28, 43, 59 and 60.)

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016. (Cited on pages 28, 43 and 59.)

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1640–1650, 2018. (Cited on pages 29 and 107.)

David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. (Cited on pages vi and 21.)

Hao Lu, Lei Zhang, Zhiguo Cao, Wei Wei, Ke Xian, Chunhua Shen, and Anton van den Hengel. When unsupervised domain adaptation meets tensor representations. In *IEEE International Conference on Computer Vision*, pages 599–608, 2017. (Cited on page 29.)

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems*, pages 1041–1048, 2009. (Cited on page 6.)

J. Marín, D. Vázquez, D. Gerónimo, and A. López. Learning appearance in virtual scenarios for pedestrian detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 137–144, 2010. (Cited on pages v, 5, 32 and 66.)

Francisco Massa, Mathieu Aubry, and Renaud Marlet. Convolutional neural networks for joint object detection and pose estimation: A comparative study. *CoRR*, abs/1412.7190, 2014. (Cited on page 30.)

Francisco Massa, Renaud Marlet, and Mathieu Aubry. Crafting a multi-task CNN for viewpoint estimation. In *British Machine Vision Conference*, 2016. (Cited on pages 30, 65, 85 and 97.)

K. Matzen and N. Snavely. Nyc3dcars: A dataset of 3D vehicles in geographic context. In *IEEE International Conference on Computer Vision*, pages 761–768, 2013. (Cited on page 32.)

Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015. (Cited on page 33.)

Tzu Ming Harry Hsu, Wei Yu Chen, Cheng-An Hou, Yao-Hung Hubert Tsai, Yi-Ren Yeh, and Yu-Chiang Frank Wang. Unsupervised domain adaptation with imbalanced cross-domain data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4121–4129, 2015. (Cited on pages 28 and 35.)

Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *IEEE International Conference on Computer Vision*, pages 5716–5726, 2017. (Cited on pages 29, 37 and 60.)

R. Mottaghi, Y. Xiang, and S. Savarese. A coarse-to-fine model for 3D pose estimation and sub-category recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 418–426, 2015. (Cited on pages 30 and 66.)

David M Mount. Ann: A library for approximate nearest neighbor searching. *http://www. cs. umd. edu/˜ mount/ANN/*, 2010. (Cited on page 15.)

Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *IEEE European Conference on Computer Vision*, pages 483–499, 2016. (Cited on page 31.)

Andrew Y Ng and Michael I Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems*, pages 841–848, 2002. (Cited on page 14.)

J. Ni, Q. Qiu, and R. Chellappa. Subspace interpolation via dictionary learning for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 692–699, 2013. (Cited on page 27.)

Timo Ojala, Matti Pietikainen, and David Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *IEEE International Conference on Pattern Recognition*, pages 582–585, 1994. (Cited on page 24.)

M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 778–785, 2009. (Cited on pages v, 6, 30, 72 and 77.)

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. In *International Jont Conference on Artifical Intelligence*, pages 1187–1192, 2009. (Cited on pages 44 and 59.)

P. Panareda Busto and J. Gall. Open set domain adaptation. In *IEEE International Conference on Computer Vision*, pages 754–763, 2017. (Cited on pages xix, 8, 9 and 32.)

P. Panareda Busto and J. Gall. Viewpoint refinement and estimation with adapted synthetic data. *Computer Vision and Image Understanding*, 169:75–89, 2018. (Cited on pages xix and 10.)

P. Panareda Busto and J. Gall. Joint viewpoint and keypoint estimation with real and synthetic data. In *German Conference on Pattern Recognition*, pages 107–121, 2019. (Cited on pages xix, 10 and 11.)

P. Panareda Busto, J. Liebelt, and J. Gall. Adaptation of synthetic data for coarse-to-fine viewpoint refinement. In *British Machine Vision Conference*, pages 14.1–14.12, 2015. (Cited on pages xix, 8, 10, 73, 74 and 76.)

P. Panareda Busto, A. Iqbal, and J. Gall. Open set domain adaptation for image and action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. (Cited on pages xix and 9.)

Eunbyung Park, Xufeng Han, Tamara L Berg, and Alexander C Berg. Combining multiple sources of knowledge in deep cnns for action recognition. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1–8, 2016. (Cited on page 3.)

Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *IEEE International Conference on Robotics and Automation*, pages 2011–2018, 2017. (Cited on page 31.)

Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2 (11):559–572, 1901. (Cited on page 71.)

Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *IEEE International Conference on Computer Vision*, pages 1278–1286, 2015. (Cited on pages 5 and 32.)

Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *CoRR*, abs/1710.06924, 2017. (Cited on pages viii, xii, 37, 57 and 58.)

Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755*, 2018. (Cited on page 32.)

B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3D geometry to deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3362–3369, 2012. (Cited on pages 30, 31 and 72.)

Bojan Pepik, Michael Stark, Peter Gehler, Tobias Ritschel, and Bernt Schiele. 3D object class detection in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition: Workshops*, pages 1–10, 2015. (Cited on page 30.)

P. J. Phillips, Hyeonjoon Moon, S. A. Rizvi, and P. J. Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1090–1104, 2000. (Cited on page 29.)

Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975. (Cited on page 68.)

L. Pishchulin, A. Jain, C. Wojek, M. Andriluka, T. Thormählen, and B. Schiele. Learning people detection models from few training samples. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1473–1480, 2011. (Cited on pages 32 and 66.)

Carolina Redondo-Cabrera, Roberto López-Sastre, and Tinne Tuytelaars. All together now: Simultaneous object detection and continuous pose estimation using a Hough forest with probabilistic locally enhanced voting. In *British Machine Vision Conference*, 2014. (Cited on page 30.)

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. (Cited on page 29.)

Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):801–814, 2018. (Cited on page 28.)

K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *IEEE European Conference on Computer Vision*, pages 213–226, 2010. (Cited on pages v, xi, xi, xii, xii, xii, xii, xii, 3, 4, 27, 28, 35, 37, 42, 45, 46, 52, 53, 55, 59, 60 and 61.)

Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018a. (Cited on pages 28 and 107.)

Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *IEEE European Conference on Computer Vision*, pages 153–168, 2018b. (Cited on pages 29 and 105.)

Enver Sangineto. Statistical and spatial consensus collection for detector adaptation. In *IEEE European Conference on Computer Vision*, pages 456–471, 2014. (Cited on page 28.)

S. Savarese and L. Fei-Fei. 3D generic object categorization, localization and pose estimation. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007. (Cited on pages 73 and 77.)

Walter J. Scheirer, Anderson Rocha, Archana Sapkota, and Terrance E. Boult. Towards open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1757–1772, 2013. (Cited on pages xi, 29, 37 and 47.)

Walter J. Scheirer, Lalit P. Jain, and Terrance E. Boult. Probability models for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2317–2324, 2014. (Cited on pages 29, 37 and 45.)

J. Schels, J. Liebelt, and R. Lienhart. Learning an object class representation on a continuous viewsphere. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3170–3177, 2012. (Cited on page 30.)

Nima Sedaghat and Thomas Brox. Unsupervised generation of a viewpoint annotated car dataset from videos. In *IEEE International Conference on Computer Vision*, pages 1314–1322, 2015. (Cited on pages xiii, 32, 73, 85 and 87.)

S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa. Generalized domain-adaptive dictionaries. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 361–368, 2013. (Cited on page 27.)

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. (Cited on pages viii, 24, 57, 58, 59, 67, 74, 91 and 97.)

Alex J Smola and Bernhard Schölkopf. *Learning with kernels*. Citeseer, 1998. (Cited on page 16.)

Richard Socher, Brody Huval, Bharath Bath, Christopher D Manning, and Andrew Y Ng. Convolutional-recursive deep learning for 3d object classification. In *Advances in Neural Information Processing Systems*, pages 656–664, 2012. (Cited on page 7.)

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012. (Cited on pages viii, xiii, xiii, 38, 55, 57, 125, 126 and 127.)

M. Stark, M. Goesele, and B. Schiele. Back to the future: learning shape models from 3D CAD data. In *British Machine Vision Conference*, 2010. (Cited on page 30.)

Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. In *IEEE International Conference on Computer Vision*, pages 2686–2694, 2015. (Cited on pages 5, 30, 32 and 97.)

B. Sun and K. Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *British Machine Vision Conference*, 2014. (Cited on pages 27 and 66.)

Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI Conference on Artificial Intelligence*, pages 2058–2065, 2015a. (Cited on pages 27, 28, 42, 45, 46, 59, 60, 75, 76, 77 and 78.)

Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. *CoRR*, abs/1502.00873, 2015b. (Cited on page 29.)

K. Svanberg. A class of globally convergent optimization methods based on conservative convex separable approximations. *SIAM Journal on Optimization*, 12(2): 555–573, 2002. (Cited on pages 41 and 70.)

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. (Cited on page 24.)

T. Tommasi and B. Caputo. Frustratingly easy NBNN domain adaptation. In *IEEE International Conference on Computer Vision*, pages 897–904, 2013. (Cited on page 28.)

T. Tommasi and T. Tuytelaars. A testbed for cross-dataset analysis. In *IEEE European Conference on Computer Vision: Workshop on Transferring and Adapting Source Knowledge in Computer Vision*, pages 18–31, 2014. (Cited on pages xi, xi, xii, 37, 52, 53, 55 and 63.)

Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. In *German Conference on Pattern Recognition*, pages 504–516, 2015. (Cited on pages 52, 60 and 61.)

Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. (Cited on page 31.)

Marwan Torki and Ahmed Elgammal. Regression from local features for viewpoint and pose estimation. In *IEEE International Conference on Computer Vision*, pages 2603–2610, 2011. (Cited on pages 30 and 31.)

Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660, 2014. (Cited on page 31.)

Jonathan Tremblay, Thang To, and Stan Birchfield. Falling things: A synthetic dataset for 3d object detection and pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2038–2041, 2018. (Cited on page 32.)

Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015. (Cited on pages xiii, xiii, 30, 31, 65, 74, 85, 86, 96 and 97.)

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014. (Cited on pages 28, 52 and 60.)

Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *IEEE International Conference on Computer Vision*, pages 4068–4076, 2015. (Cited on page 60.)

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2962–2971, 2017. (Cited on pages 29, 37 and 59.)

Paul Upchurch, Jacob R. Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Q. Weinberger. Deep feature interpolation for image content changes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6090–6099, 2017. (Cited on pages 37 and 57.)

Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 109–117, 2017. (Cited on page 32.)

D. Vázquez, A. López, D. Ponsa, and J. Marín. Cool world: domain adaptation of virtual and real worlds for human detection using active learning. In *Advances in Neural Information Processing Systems: Workshop on Domain Adaptation: Theory and Applications*, 2011. (Cited on page 66.)

D. Vázquez, A. López, J. Marín, D. Ponsa, and D. Gerónimo. Virtual and real world adaptation for pedestrian detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):797–809, 2014. (Cited on page 66.)

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5385–5394, 2017. (Cited on pages 29 and 59.)

Xiaoyu Wang, Tony X Han, and Shuicheng Yan. An hog-lbp human detector with partial occlusion handling. In *IEEE International Conference on Computer Vision*, pages 32–39, 2009. (Cited on pages vi and 24.)

Yaming Wang, Xiao Tan, Yi Yang, Xiao Liu, Errui Ding, Feng Zhou, and Larry S. Davis. 3d pose estimation for fine-grained object categories. In *IEEE European Conference on Computer Vision: Workshops*, 2018. (Cited on page 32.)

Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016. (Cited on pages 31, 90, 91, 92 and 105.)

Christian Wojek, Stefan Walk, and Bernt Schiele. Multi-cue onboard pedestrian detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 794–801, 2009. (Cited on pages v and 5.)

Jiajun Wu, Tianfan Xue, Joseph J. Lim, Yuandong Tian, Joshua B. Tenenbaum, Antonio Torralba, and William T. Freeman. Single image 3d interpreter network. In *IEEE European Conference on Computer Vision*, pages 365–382, 2016. (Cited on page 31.)

Pengcheng Wu and Thomas G Dietterich. Improving svm accuracy by training on auxiliary data sources. In *International Conference on Machine Learning*, page 110, 2004. (Cited on page 6.)

Y. Xiang, R. Mottaghi, and S. Savarese. Beyond Pascal: A benchmark for 3D object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82, 2014. (Cited on pages viii, viii, xii, xiii, xiii, 32, 57, 58, 66, 69, 73, 98 and 99.)

Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3D object recognition. In *IEEE European Conference on Computer Vision*, pages 160–176, 2016. (Cited on pages xiii, xiii, 90, 93, 94, 96, 97, 98 and 99.)

H. Xu, J. Zheng, and R. Chellappa. bridging the domain shift by domain adaptive dictionary learning. In *British Machine Vision Conference*, pages 96.1–96.12, 2015. (Cited on page 27.)

Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 945–954, 2017. (Cited on page 28.)

J. Yang, R. Yan, and A. G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *ACM International Conference on Multimedia*, pages 188–197, 2007. (Cited on page 6.)

Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1385–1392, 2011. (Cited on pages xiii and 96.)

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018. (Cited on page 32.)

Jing Zhang, Wanqing Li, and Philip Ogunbona. Joint geometrical and statistical alignment for visual domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5150–5158, 2017a. (Cited on page 27.)

Rong Zhang and Dimitris N Metaxas. Ro-svm: Support vector machine with reject option for image categorization. In *British Machine Vision Conference*, pages 1209–1218, 2006. (Cited on page 29.)

Yang Zhang, Philip David, and Boqing Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *IEEE International Conference on Computer Vision*, pages 2020–2030, 2017b. (Cited on pages 7 and 106.)

Xingyi Zhou, Arjun Karpur, Chuang Gan, Linjie Luo, and Qixing Huang. Unsupervised domain adaptation for 3d keypoint estimation via view consistency. In *IEEE European Conference on Computer Vision*, pages 141–157, 2018a. (Cited on page 31.)

Xingyi Zhou, Arjun Karpur, Linjie Luo, and Qixing Huang. Starmap for category-agnostic keypoint and viewpoint estimation. In *IEEE European Conference on Computer Vision*, pages 318–334, 2018b. (Cited on pages 31 and 98.)

M. Z. Zia, M. Stark, B. Schiele, and K. Schindler. Detailed 3D representations for object recognition and modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2608–2623, 2013. (Cited on page 30.)

Christian Zimmermann and Thomas Brox. Learning to estimate 3D hand pose from single rgb images. *CoRR*, abs/1705.1389, 2017. (Cited on page 31.)

Appendix A

Action Recognition Open Set Protocol

The *action recognition open set protocol* for domain adaptation defines as shared classes between two different domains all actions that reproduce the same movement for a well-defined activity. If one dataset contains more than one class label that is identified as the same action in the other dataset, these will be grouped in one single class.

The list of shared and unknown action classes for the open set domain adaptation protocol between the *Kinetics Human Action Video Dataset* [Kay et al., 2017] (Kinetics) and the *UCF101 Action Recognition Dataset* [Soomro et al., 2012] (UCF101) are given in Table A.1 and Table A.2, respectively.

Action recognition open set protocol for Kinetics → UCF101 (Shared classes)					
Kinetics	UCF101	Kinetics	UCF101	Kinetics	UCF101
archery	archery	dribbling basketball, playing basketball, shooting basketball	basketball	bench pressing	bench press
biking through snow, riding a bike, riding mountain bike	biking	blowing out candles	blowing candles	bowling	bowling
brushing teeth	brushing teeth	canoeing or kayaking	kayaking	catching or throwing baseball	baseball pitch
catching or throwing frisbee	frisbee catch	clean and jerk	clean and jerk	climbing a rope	rope climbing
crawling baby	baby crawling	cutting pineapple, cutting watermelon	cutting in kitchen	diving cliff	cliff diving
dunking basketball	basketball dunk	filling eyebrows	apply eye makeup	getting a haircut	haircut
golf driving	golf swing	hammer throw	hammer throw	high jump	high jump
hula hooping	hula hoop	javelin throw	javelin throw	jetskiing	skijet
contact juggling, juggling balls	juggling balls	juggling soccer ball	soccer juggling	jumping into pool	trampoline jumping
knitting	knitting	long jump	long jump	lunge	lunges
making pizza	pizza tossing	marching	band marching	massaging persons head	head massage
mopping floor	mopping floor	playing cello	playing cello	playing cricket	cricket shot
playing drums	drumming	playing flute	playing flute	playing guitar	playing guitar
playing piano	playing piano	playing tennis	tennis swing	playing violin	playing violin
playing volleyball	volleyball spiking	pole vault	pole vault	pull ups	pull ups
punching bag	boxing punching bag, boxing speed bag	punching person	punch	push up	push ups, wall pushups
riding or walking with horse	horse riding	rock climbing	rock climbing indoor	salsa dancing	salsa spins
scuba diving	diving	shot put	shotput	skateboarding	skate boarding
skiing, skiing crosscountry, skiing slalom	skiing	skipping rope	jump rope	skydiving	sky diving
kicking soccer ball, shooting goal	soccer penalty	squat	body weight squats	surfing water	surfing
swimming breast stroke	breaststroke	swing dancing	swing	tai chi	tai chi
throwing discus	throw discus	trimming or shaving beard	shaving beard	walking the dog	walking with a dog

Table A.1: Definition of shared classes for the open set protocol between the Kinetics [Kay et al., 2017] and the UCF101 [Soomro et al., 2012] action recognition datasets.

Action recognition open set protocol for Kinetics → UCF101 (Unknown classes)
Kinetics
abseiling, air drumming, answering question, applauding, applying cream, arm wrestling, arranging flowers, assembling computer, auctioning, baby walking up, baking cookies, balloon blowing, bandaging, barbequing, bartending, beatboxing, bee keeping, belly dancing, bending back, bending metal, blasting sand, blowing glass, blowing leaves, blowing nose, bobsledding, bookbinding, bouncing on trampolin, braiding hair, breading or breadcrumbing, breakdancing, brush painting, brushing hair, building cabinet, building shed, bungee jumping, busking, celebrating, capoeira, carrying baby, cartwheeling, carving pumpkin, catching fish, catching or throwing softball, changing oil, changing wheel, checking tires, cheerleading, chopping wood, clapping, clay pottery making, cleaning floor, cleaning gutters, cleaning pool, cleaning shoes, cleaning toilet, cleaning windows, climbing ladder, climbing tree, contry line dancing, cooking chicken, cooking egg, cooking on campfire, cooking sausages, counting money, cracking neck, crossing river, crying, curling hair, cutting nails, dancing ballet, dancing charleston, dancing gangnam, dancing macarena, deadlifting, decorating the christmas tree, digging, dining, disc golfing, dodgeball, doing aerobics, doing laundry, doing nails, drawing, drinking, drinking beer, drinking shots, driving car, driving tractor, drop kicking, drumming fingers, dying hair, eating burger, eating cake, eating carrots, eating chips, eating doughnuts, eating hotdog, eating ice cream, eating spaghetti, eating watermelon, egg hunting, exercising arm, exercising with an exercise ball, extinguishing fire, faceplanting, feeding birds, feeding fish, feeding goats, finger snapping, fixing hair, flipping pancake, flying kite, folding clothes, folding napkins, folding paper, front raises, frying vegetables, garbage collecting, gargling, getting a tattoo, giving or receiving award, golf chipping, golf putting, grinding meat, grooming dog, grooming horse, gymnastics tumbling, headbanging, headbutting, high kick, hitting baseball, hockey stop, holding snake, hopscotch, hoverboarding, hugging, hurdling, hurling, ice climbing, ice fishing, ice skating, ironing, jogging, juggling fire, jumpstyle dancing, kicking field goal, kissing, kitesurfing, krumping, laughing, laying bricks, making a cake, making a sandwich, making bed, making jewerly, making snowman, making sushi, making tea, massaging back, massaging feet, massaging legs, milking cow, motorcycling, moving furniture, mowing lawn, news anchoring, opening bottle, opening present, paragliding, parasailing, parkour, passing American football, passing American football, peeling apples, peeling potatoes, petting animal, petting cat, picking fruit, planting trees, plastering, playing accordion, playing badminton, playing bagpipes, playing bass guitar, playing cards, playing chess, playing clarinet, playing controller, playing cymbals, playing didgeridoo, playing harmonica, playing harp, playing ice hockey, playing keyboard, playing kickball, playing monopoly, playing organ, playing paintball, playing poker, playing recorder, playing saxophone, playing squash or racquetball, playing trombone, playing trumpet, playing ukulele, playing xylophone, presenting weather forecast, pumping fist, pumping gas, pushign cart, pushing car, pushing wheelchair, reading book, reading newspaper, recording music, ridin a camel, riding elephant, riding mechanical bull, riding mule, riding scooter, riding unicycle, ripping paper, robot dancing, rock scissors paper, roller skating, running on treadmill, sailing, sanding floor, scrambling eggs, setting table, shaking hands, shaking head, sharpening knives, sharpening pencil, shaving head, shaving legs, shearing sheep, shining shoes, shoveling snow, shredding paper, shuffling cards, side kick, sign language interpreting, singing, situp, ski jumping, slacklining, slapping, sled dog racing, smoersaulting, smoking, smoking hookah, snatch weight lifting, sneezing, sniffing, snorkeling, snowboarding, snowkiting, snowmobiling, spinning poi, spray painting, spraying, springboard diving, stickin tongue, stomping grapes, stretching arm, stretching leg, strumming guitar, surfing crowd, sweeping floor, swimming backstroke, swimming butterfly stroke, swinging legs, swinging on something, sword fighting, taking a shower, tango dancing, tap dancing, tapping guitar, tapping pen, tasting beer, tasting food, testfying, texting, throwing axe, throwing ball, tickling, tobogganing, tossing coin, tossing salad, training dog, trapezing, trimming trees, triple jump, tying bow tie, tying know, tying tie, unboxing, unloading truck, using computer, using remote controller, using segway, vault, waiting in line, washing dishes, washing feet, washing hair, washing hands, water skiing, water sliding, watering plants, waxing back, waxing chest, waxing eyebrows, waxing legs, weaving basket, welding, whistling, windsurfing, wrapping present, wrestling, writing, yawning, yoga, zumba
UCF101
apply lipstick, balance beam, billiards shot, blow dry hair, fencing, field hockey penalty, floor gymnastics, front crawl, hammering, handstand pushups, handstand walking, horse race, ice dancing, jumping jack, military parade, mixing batter, nun chucks, parallel bars, playing daf, playing dhol, playing sitar, playing tabla, pommel horse, rafting, rowing, still rings, sumo wrestling, table tennis shot, throw discus, typing, uneven bars, writing on board, yo yo

Table A.2: Definition of unknown classes for the open set protocol between the Kinetics [Kay et al., 2017] and the UCF101 [Soomro et al., 2012] action recognition datasets.

Erklärung

Ich versichere, dass ich die von mir vorgelegte Dissertation selbständig angefertigt, die benutzten Quellen und Hilfsmittel vollständig angegeben und die Stellen der Arbeit-einschlielich Tabellen, Karten und Abbildungen –, die anderen Werken im Wortlaut oder dem Sinn nach entnommen sind, in jedem Einzelfall als Entlehnung kenntlich gemacht habe; dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat; dass sie noch nicht veröffentlicht worden ist sowie, da ich eine solche Veröffentlichung vor Abschluss des Promotionsverfahrens nicht vornehmen werde. Die Bestimmungen dieser Promotionsordnung sind mir bekannt. Die von mir vorgelegte Dissertation ist von Prof. Dr. Juergen Gall betreut worden.

Unterschift:

____

Datum:

____