Domain Adaptation in Context of Visual Factors

Vom Fachbereich
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades
eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von

Sebastian Schrom, M.Sc.

geboren am 05. August 1989 in Miltenberg

Referent:	Prof. Dr.-Ing. J. Adamy
Korreferent:	Prof. Dr. H. Wersing
Tag der Einreichung:	19. Oktober 2021
Tag der mündlichen Prüfung:	20. Januar 2022

D17
Darmstadt 2022

Schrom, Sebastian - Domain Adaptation in Context of Visual Factors
Darmstadt, Technische Universität Darmstadt
Jahr der Veröffentlichung auf TUprints: 2022
URN: urn:nbn:de:tuda-tuprints-203758
Tag der Disputation: 20. Januar 2022

Vorwort

Die vorliegende Dissertation entstand im Zuge eines PhD Projektes, welches auf einer Kooperation zwischen dem Fachgebiet Regelungsmethoden und Robotik der TU Darmstadt und dem Honda Research Institute Europe GmbH in Offenbach beruhte.

Besonders bedanken möchte ich mich bei dem Leiter des Fachgebietes Regelungsmethoden und Robotik, Prof. Adamy, welcher mir die Möglichkeit gab, als wissenschaftlicher Mitarbeiter an seinem Fachgebiet zu promovieren. Die Forschung und Mithilfe in der Lehre dort hat immer Spaß gemacht und einem die nötigen Werte für das zukünftige Berufsleben vermittelt. Besonderer Dank gilt hierbei auch Volker Willert, welcher stets für wissenschaftliche Fragestellungen zur Verfügung stand. Dies trifft ebenso auf alle meine Kollegen am Fachgebiet zu, ohne die die Arbeit nur halb so viel Spaß gemacht hätte. Auf Seiten des Honda Research Institutes möchte ich mich vor allem bei meinem wissenschaftlichen Betreuer Stephan Hasler bedanken, welcher mir immer für ergebnisreiche Diskussionen zur Seite stand, und welcher meine Kompetenzen für wissenschaftliche Vorgehensweisen erweitert hat. Auch gilt mein Dank dem Management von Honda, welches dieses PhD Projekt sowohl wissenschaftlich, durch die Einbindung in verschiedene wissenschaftliche Austauschmöglichkeiten, als auch mit der für die Forschung notwendigen Hardware unterstützt hat.

Zu guter Letzt möchte ich mich bei meinem engsten privaten Kreis bedanken. Hierbei möchte ich besonders meine beiden Geschwister Eva-Maria und Florian und meinen Schwager Oliver nennen, welche für mich jederzeit ein offenes Ohr hatten. Dies trifft ebenso auf meine Freundin Verónica zu, die ich während meiner Promotion kennen und lieben gelernt habe. Sie hat mir geholfen auch in den anstrengendsten Zeiten stets den Sinn für das Wichtige im Leben zu wahren. Schließlich gilt der größte Dank meinen Eltern, welche mich in meinem gesamten privaten und akademischen Werdegang jederzeit bedingungslos unterstützt haben, und ohne die diese Promotion nie möglich gewesen wäre.

Darmstadt, Oktober 2021 Sebastian Schrom

Contents

Abbreviations and Symbols

Abbreviations

AP	Average Precision
BS	Batch-Size
CNN	Convolutional Neural Network
DA	Domain Adaptation
DSLR	Digital Single Lens Reflex (Camera)
FN	False-Negative
FP	False-Positive
FP-DA	Factor Preserving Domain Adaptation
GAN	Generative Adversarial Network
GPS	Global Positioning System
Grad-CAM	Gradient-weighted Class Activation Mapping
GRL	Gradient Reversal Layer
HSV	Hue Saturation Value (Color Space)
PCA	Principal Component Analysis
ReLU	Rectified linear unit
RGB	Red Green Blue
SuCo	Subset Confusion
TN	True-Negative
TP	True-Positive
ce	cross-entropy
e. g.	exempli gratia
i. e.	id est
sv	supervised
t-SNE	t-distributed stochastic neighbor embedding
usv	unsupervised
w/	with
w/o	without

Symbols

Notation

x	Scalar
$\mathbf{x}$	Vector
$\mathbf{x}'$	Predicted Vector
$\mathbf{X}$	Matrix
$\lvert\mathcal{X}_d\rvert$	Cardinality of set $\mathcal{X}_d$
$P(\cdot)$	Probability Distribution or Value
$\tilde{P}(\cdot)$	Approximated Probability Distribution or Value
'hand'	Factor named hand

Latin Uppercase Letters

C	Classes of given Classification Task
Ch	Channels of Neural Network Layer
$D_s = \{\mathcal{X}_s, P_s(\mathbf{x})\}$	Source Domain
$D_t = \{\mathcal{X}_t, P_t(\mathbf{x})\}$	Target Domain
$\mathbf{F}$	Chosen Factorization
$G(\cdot)$	Transfer Function
H	Height of Feature Map
$\mathbf{I}$	Collection of multiple Images
L	Loss
$L_{ce}(\cdot)$	Cross-Entropy Loss Function
M	MNIST-M data
$M(\cdot)$	Mixer Function that generates Image b
S	MNIST data
$\mathcal{T}_s = \{\mathcal{Y}_s, P_s(\mathbf{y}\lvert\mathbf{x})\}$	Task of Source Domain
$\mathcal{T}_t = \{\mathcal{Y}_t, P_t(\mathbf{y}\lvert\mathbf{x})\}$	Task of Target Domain
W	Width of Feature Map
$\mathcal{X}$	Feature Space
$\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$	Set of n Feature Vectors
$\mathcal{Y}$	Label / Class Space
$\mathcal{Y}_d$	Subset of Classes from Domain d that was involved supervised during training
$\bar{\mathcal{Y}}_d$	Subset of Classes from Domain d that was involved unsupervised or not at all during training
$\mathbf{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\}$	Dataset with n Label Vectors

Latin Lowercase Letters

$\mathbf{b}_i$	Image of Scene i
$\mathbf{d}$	Domain Label Vector
$\mathbf{d}'$	Predicted Domain Label Vector
f_r	Single visual Factor
$\mathbf{f}_i = \{f_{1_i}, f_{2_i}, ..., f_{l_i}\}$	Description of a Scene i by l factors
g	Group Label (for FP-DA)
h	Activation of a Neuron
p	Training Progress
q	Cardinality of a Subset of Classes, e. g. $\lvert\mathcal{Y}_M\rvert$
$\mathbf{x}$	Feature Vector
$\mathbf{y}$	Class Label Vector
$\mathbf{y}'$	Predicted Class Label Vector
$\mathbf{z}_{g=A}$	Group Dependent Gradient Multiplier Vector

Greek Letters

θ	Parameters of Deep Learning Model
λ	Adaptation Parameter for Domain Classifier
μ	Learning Rate

Abstract

The number of application areas of deep neural networks for image classification is continuously growing. A general desired attribute of these networks is to generalize well to test data that visually differs from the training data, but still shows the relevant features of the classes to be discriminated. Reasons for such a difference in data could be related to a change in background, illumination, or camera properties. The research area of Domain Adaptation (DA) deals with the transferability of classification models between such datasets, called domains, with the target to maximize the transferability.

Typically, the differences and similarities of domains are described by the notion of general data distributions. This method, however, does not allow to identify and describe sufficiently the actual cause of a reduced performance on a new domain. To tackle this, in this thesis a novel description of domains, based on a theory of visual factors that describes the characteristics of domains will be introduced. As it will be shown, it can also be used to explain the targets and effects of existing DA approaches more understandable, which ultimately can be used to improve those even further.

When it comes to the application of classification models in context of domains, several generalization cases can occur. In literature the most relevant ones are the cases where the application domain is the same domain as the training domain or the application domain is a completely new domain. The case that the application domain was one of multiple training domains is usually neglected, but will be investigated in this thesis as well, since it has high relevance for the usage of pre-trained classification models on own image data. As it will be shown further, the awareness about the domains for all three generalization cases is important for a well performing classification model in the application domain. The novel investigations in this context will be introduced under the term Effects of Domain Awareness. Different cases of domain awareness are investigated in combination with different domain constellations within the training and test data using the simple DA method of RGB mean normalization. The results on a road segmentation task show the importance to treat a

domain during training and test always in the same way, since otherwise a significantly reduced performance can be observed.

A typical assumption in current DA research is that each training domain includes samples for all classes that should be discriminated. However, thinking of distributed camera systems with a shared classification model, where each system potentially represents a domain, this assumption is too restricted. The more realistic assumption here is that not all classes are covered by samples from each domain during training of the classifier. The aforementioned scenario, which is overlooked in literature, will be extensively investigated under the term Domain Mixture scenario in this thesis. The experiments on MNIST and real-world object classification data show that, given the Domain Mixture scenario, the application of an approach from DA is essential, since otherwise the classification model is not capable to perform well on domain-class combinations that were not represented by supervised samples during training.

A common DA approach to obtain a classification model that performs invariant of a domain well, is to remove all factors from the internal class feature representation that allow a discrimination of domains. This, however, can be harmful if at the same time task-informative factors are removed. To prevent this negative effect, the novel approach of Factor-Preserving DA (FP-DA) will be introduced which allows to preserve a selected factor during training with an adversarial DA approach. The experiments in this context will first show on real-world data that this negative effect exists and afterwards how factors worth preserving can be identified and subsequently be preserved through FP-DA in a multi-domain setting. The results show that FP-DA is capable to achieve the highest average and minimum performance in such a setting compared to the used baseline method.

In summary, this thesis introduces a novel description of domains and based on that, investigates multiple highly relevant constellations for DA and additionally proposes a novel DA approach.

Kurzfassung

Die Anzahl der Anwendungsgebiete von tiefen neuronalen Netzen für die Bildklassifikation wächst kontinuierlich. Eine grundsätzliche, gewünschte Eigenschaft solcher Netze ist es, auf Testdaten zu generalisieren die sich zwar optisch von den Trainingsdaten unterscheiden, aber dennoch die relevanten Merkmale der zu unterscheidenden Klassen aufweisen. Die Gründe für solch einen Unterschied in den Datensätzen können mit einer Änderung des Hintergrundes, der Beleuchtung oder der Kameraeigenschaften zusammenhängen. Das Forschungsgebiet der Domänen Adaptation (DA) beschäftigt sich mit der Übertragbarkeit von Klassifikationsmodellen zwischen solchen Datensätzen, genannt Domänen, mit dem Ziel die Übertragbarkeit zu maximieren.

In der Regel werden die Unterschiede und Gemeinsamkeiten von Domänen anhand von allgemeinen Datenverteilungen beschrieben. Diese Methodik erlaubt es allerdings nicht den eigentlichen Grund für eine reduzierte Leistungsfähigkeit auf einer neuen Domäne zu identifizieren und ausreichend zu beschreiben. Um dieses Problem anzugehen, wird in dieser Thesis eine neue Beschreibung von Domänen eingeführt. Diese basiert auf einer Faktor Theorie, welche die Charakteristiken von Domänen beschreibt. Es wird gezeigt, dass diese auch dafür genutzt werden kann um die Erwartungen und Effekte bestehender DA Ansätze verständlicher darzulegen, was wiederum dafür genutzt werden kann um diese weiter zu verbessern.

Bei der Anwendung von Klassifikationsmodellen im Kontext von Domänen können verschiedene Generalisierungsfälle auftreten. Die relevantesten Fälle in der Literatur sind die, bei welchen die Anwendungsdomäne die gleiche Domäne wie die Trainingsdomäne ist, oder die Anwendungsdomäne eine vollständig neue Domäne ist. Der Fall, dass die Anwendungsdomäne eine von mehreren Trainingsdomänen ist, wird üblicherweise nicht betrachtet. In dieser Thesis wird dieser jedoch untersucht, da er eine hohe Relevanz für die Benutzung von vor-trainierten Klassifikationsmodellen auf eigenen Bilddaten hat. Wie zudem gezeigt wird, ist für ein performantes Klassifikationsmodell in der Anwendungsdomäne das Bewusstsein über Domänen in allen drei Generalisierungsfällen wichtig. Die neuen Untersuchungen in diesem Zusammenhang werden unter dem

Begriff Effekte des Domänen Bewusstseins vorgestellt. Unterschiedliche Fälle des Domänen Bewusstseins werden in Kombination mit verschiedenen Domänen Konstellation innerhalb der Trainings- und Testdaten unter Verwendung der einfachen DA Methode der RGB Mittelwert Normalisierung untersucht. Basierend auf einer Straßensegmentierungsaufgabe zeigen die Versuchsergebnisse die Bedeutsamkeit, eine Domäne während des Trainings und des Testens stets gleich zu behandeln, da andernfalls eine deutlich reduzierter Leistungsfähigkeit beobachtet werden kann.

In der aktuellen DA Forschung wird typischerweise angenommen, dass jede Trainingsdomäne Beispiele für alle zu unterscheidenden Klassen enthält. Bei verteilten Kamerasystemen mit einem gemeinsamen Klassifikationsmodell, wobei jedes Kamerasystem eine Domäne darstellt, ist diese ursprüngliche Annahme aus der Literatur allerdings zu beschränkt. Die realistischere Annahme ist, dass während des Trainings des Klassifikators nicht alle Klassen durch Beispiele aus jeder Domäne abgedeckt sind. Dieses in der Literatur unberücksichtigte Szenario wird in dieser Thesis unter dem Begriff Domain Mixture ausführlich untersucht. Wie die Experimente auf MNIST Daten und realen Objektlassifikationsdaten zeigen, ist eine Anwendung von DA unerlässlich, wenn das Domain Mixture Szenario vorliegt, da andernfalls das Klassifikationsmodell nicht in der Lage ist auf Domänen-Klassen Kombinationen zu generalisieren, welche während des Trainings nicht mit gelabelten Daten repräsentiert waren.

Ein gängiger DA Ansatz, um ein Klassifikationsmodell zu erhalten, das unabhängig von der Domäne gut funktioniert, besteht darin, alle Faktoren von der internen Klassenmerkmals-Repräsentation zu entfernen, welche eine Unterscheidung von Domänen erlauben. Dies kann allerdings nachteilig sein, wenn gleichzeitig aufgaben-relevante Faktoren entfernt werden. Um diesen negativen Effekt zu verhindern, wird der neue Factor-Preserving DA (FP-DA) Ansatz vorgestellt, welcher es ermöglicht einen ausgewählten Faktor während des Trainings mit einem Adversarial DA Ansatz zu erhalten. Die Experimente in diesem Zusammenhang werden zunächst anhand von realen Daten zeigen, dass dieser negative Effekt existiert und anschließend, wie erhaltenswerte Faktoren identifiziert und durch FP-DA in einem Multi-Domänen Setting erhalten werden können. Die Versuchsergebnisse zeigen, dass FP-DA in der Lage ist in solch einem Setting die höchste durchschnittliche und minimale Leistung im Vergleich zur verwendeten Baseline-Methode zu erzielen.

Zusammenfassend führt diese Arbeit eine neuartige Beschreibung von Domänen ein, untersucht darauf aufbauend mehrere hochgradig relevante Konstellationen für DA und stellt zusätzlich einen neuen DA Ansatz vor.

1 Introduction

In recent years computer vision approaches based on deep neural networks have shown outstanding results by achieving performances at and above human-level on selected computer vision benchmarks. The most common application areas of neural networks in computer vision range from tasks like image segmentation, object detection, image classification up to image transformation. The scope of this thesis will be image classification where the goal is to classify an input image as a whole. The major breakthrough of deep neural networks in the area of image classification can be dated back to the convolutional neural network architecture called AlexNet, that was introduced by Krizhevsky et al. in 2012 [48]. In their work they combined an advanced neural network architecture with an efficient utilization of graphics processing units, which allowed to optimize the parameters of the architecture on the ImageNet [13] database with 1.2 million images in a comparatively small amount of time. Various improved architectures have been published in the following years [32, 37, 91], however, until today, such architectures still require large amounts of training images to obtain a classification model that is capable to generalize well to test images that represent the same class but with different visual characteristics.

Therefore, if being limited to little amounts of image data, for example due to costly image collection and labeling, ideally the images used for the optimization should have similar visual characteristics as the images that are expected in the final application area of the classification model. The areas of application, however, might vary a lot, and with it the visual characteristics of the images there. Typical practical application areas where camera based systems are already used can be found in the industrial robotic context or the area of autonomous vehicles. Today's cars for example are usually already equipped with cameras to increase safety and thus to pave the way to fully-autonomous driving. In the near future it is further conceivable that humans will have different types of robots in and around their house that support them in their daily life. Most of them will be equipped with camera sensors to perceive their environment. Already now there are autonomous systems like vacuum cleaner robots or lawn mower robots that are equipped with camera based systems that

are used for example for trajectory planning based on visual perception. The visual characteristics of images acquired in these different application areas can differ significantly. While for distributed household robots the images are mostly captured under artificial light combined with typical indoor background clutter, for outdoor systems natural lighting conditions combined with outdoor environments in the background can be observed. A typical background for images captured by a lawn mower is likely to be grass, while for cars this would be the asphalt of a street. In literature, as well as in this thesis, a set of images with such dominant visual characteristics throughout most of the contained images is usually called a domain. An example where the images from different camera systems can be described as an individual domain is depicted in Fig. 1.1.

In the described constellation of multiple camera based systems the underlying image classification models would profit if they were connected, since often the goal is to discriminate the same classes. For object classification that means that the same objects can potentially be observed in each domain. Using instead an individual classifier for every domain can be useful to handle the typical visual characteristics of each domain very well. However, this requires many training images from all involved classes from each domain, which on the one hand requires a lot of storage space and on the other hand assumes that all image data is already available when setting up a new system. To circumvent the need of massive amounts of storage space and to quickly integrate a new system into an existing network of camera systems, a shared classification model that shows a reasonable classification performance along all domains would be desirable.

Naively transferring a model that was optimized with standard methods on a certain domain to a completely new domain often leads to poor performance on the new domain, due to the high dependence on the visual characteristics of the original training domain. A shared classifier therefore has to be optimized in a way to become domain invariant. The research area of Domain Adaptation (DA), which is the research topic of this thesis, deals with this target. Using an approach from DA to optimize a deep classification model can help to increase the general usability of a classifier along various domains compared to standard optimization algorithms.

Despite many well established DA approaches, this research area can still be considered as a new area with many open research topics, especially when it comes to different domain constellations that have not been considered yet. Furthermore, an often neglected topic is so called negative transfer. Negative transfer describes the situation where the application of DA leads to an even weaker performance on a new domain than simply transferring

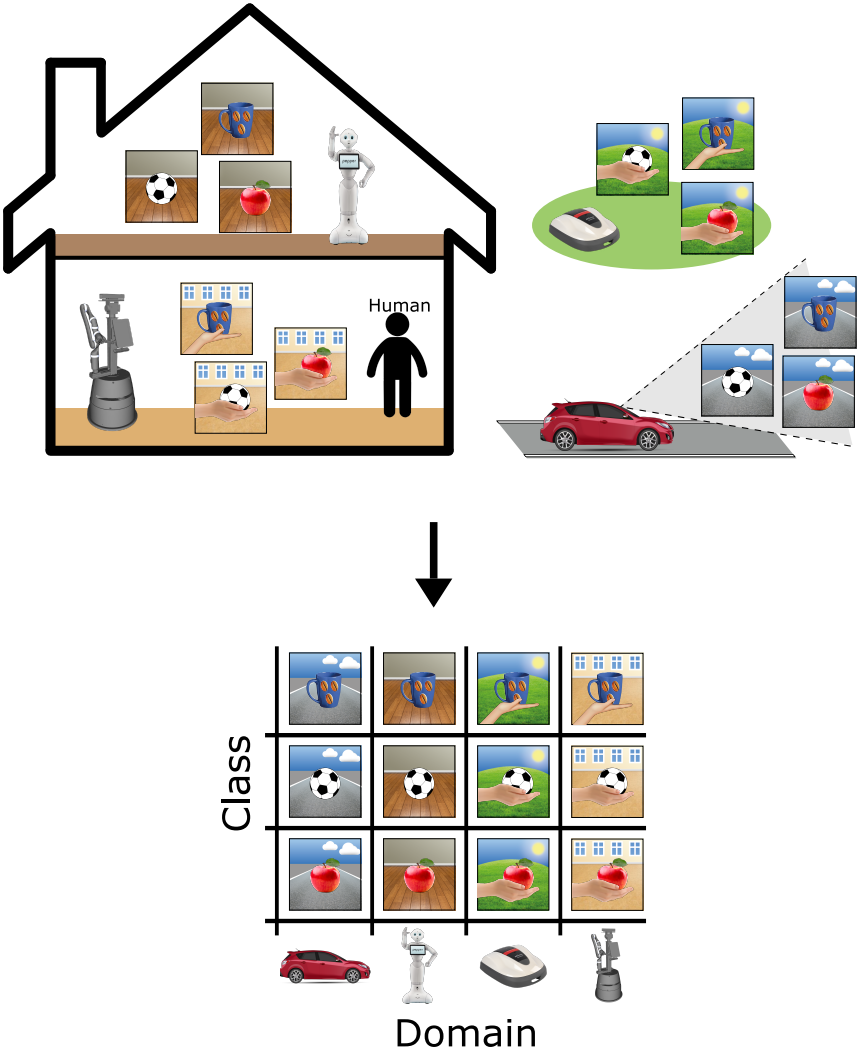


Figure 1.1: Exemplary constellation of multiple camera based systems. While the objects captured by all systems are the same, the visual characteristics introduced by each system's working environment differ. This allows to interpret each camera system as an individual domain in which all captured images have a certain consistent visual characteristic.

the model that was trained on the original domain(s) with standard methods. I. e. DA causes here the opposite effect of what it is designed for. In the investigations of this thesis multiple of such cases will be shown, and ways to prevent negative transfer will be proposed.

In particular this thesis deals with a new investigation about the awareness of involved domains during training and test with respect to input normalization as a DA method, a new domain constellation scenario with high relevance for the robotics community and a new approach to preserve a chosen factor during DA, which can help to reduce negative transfer significantly.

All three parts will be based on a newly introduced factor theory that allows to analyze the differences of domains in more detail than it is mostly done in recent literature. Usually, the investigations of datasets for their similarity are carried out through a distribution analysis of their data samples in a selected feature space. If the data distributions differ significantly, the different datasets are categorized as domains and DA is recommended to be used for a classifier. For many use cases this way of analyzing the datasets might be sufficient, however, it does not give any detailed, human interpretable insights of what caused the mismatch of the distributions. In contrast, the factor theory introduced in this thesis allows to describe domain differences in more detail through factors that have consistent values within a domain, but might possibly have a different value in another domain. Typical factors can be the lighting, the background, or the camera that was used for acquisition. With the factor theory it is possible to describe characteristics of datasets more formal and understandable. Furthermore, it allows to increase the predictability of the classification performance on a new application domain and can help to design DA approaches that consider factors, as it is done in this thesis. Fig. 1.2 shows an example case where the differences and similarities between a set of domains is described more precisely through factors instead of data distributions. The factor theory will be introduced as the first contribution of this thesis and will be used throughout it to interpret the experimental results.

Based on the factor theory the first experimental section of this thesis treats the topic of awareness of involved domains in the training set and the relation to an application domain. In general, an easy way to adapt a model to a new domain can be the method of dataset based input normalization. In the context of image classification with neural networks this is usually done by simply subtracting the RGB mean over images from the domain on which the network is applied before forwarding it through the network. This

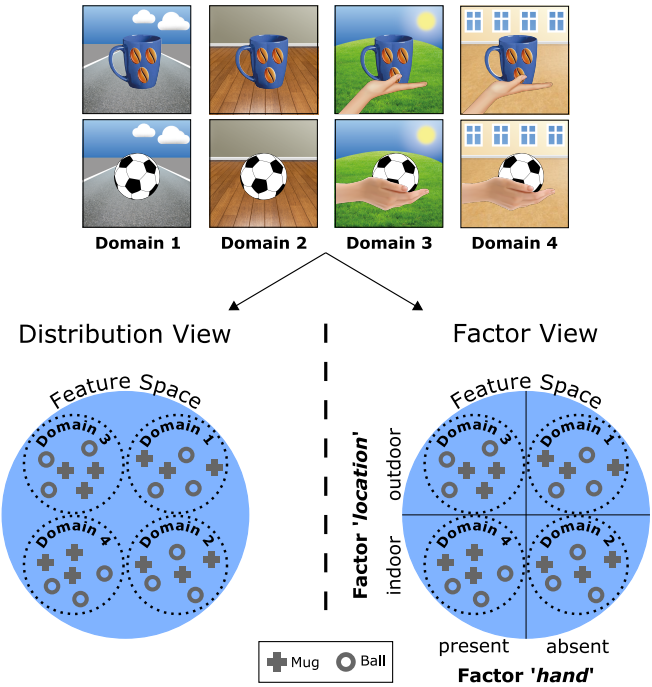


Figure 1.2: Description of the characteristics of a set of domains by data distributions and factors. Using only the data distributions of the involved domains as an explanation for the differences between domains might lead in the exemplary feature space to little human interpretability with no similarity between domains. Using the factors *'location'* ∈ {indoor, outdoor} and *'hand'* ∈ {present, absent} allows to describe similarities and differences in more detail. Aspects like the visible hand allow a better interpretation of transferability between domains and thus can help to predict how well a model trained on three domains would perform on a fourth domain. Note, the visualization style of the lower plots is used throughout this thesis and is referred to as the feature space plot.

is an efficient DA technique if the domain difference is primarily reflected in the RGB mean of the images of each domain. Nowadays, this procedure is a common way to adapt a pre-trained neural network to own image data. Such pre-trained networks have usually been optimized on huge image databases where potentially multiple hidden domains might have been included. Due to the missing domain label, such underlying domains are usually ignored during the initial optimization of the original network.

Specifically this would imply that the RGB mean that is subtracted from the training data is generated over all involved domains and not for each domain individually. As part of the findings of this thesis it will be shown along multiple extensive evaluations that the naive standard workflow of normalization based on the application domain can lead to negative transfer here. This is the case if the test domain was one of the underlying domains in the training set and the images of these domains were not normalized individually. An example where inconsistent treatment of domains during training and test can cause negative transfer is depicted in Fig. 1.3. The experimental findings of the effects of domain awareness investigations in this thesis are based on a patch-wise street scene segmentation task where domains are defined by different camera types with different sensitivities in the color channels.

The second main experimental findings are based on DA scenarios that have a high relevance for the robotic context but are neglected in current research. A typical assumption in DA research is that each training domain contains labeled images, i. e. supervised samples, for all classes, while the application domain is usually only represented by few labeled images or only images without label, i. e. unsupervised samples. This standard scenario is depicted in Fig. 1.4a, where the lawn mower represents the application domain. However, the fact that this scenario does not apply to most real-world systems with distributed camera systems becomes clear when considering the introductory example of Fig. 1.1. The existence of multiple distributed systems does not only lead to different visual characteristics, but also to different object encounters over time and environment. The probability to encounter certain objects might differ across the application environments. While street scenes have higher chances of vehicle encounters, indoor environments are more likely to include typical household objects. Furthermore, since labeling requires human input, it can not be assumed that all classes of a domain have labeled samples when the shared classifier is trained. With all these constraints, the more realistic situation is described by a scenario where certain classes of single domains are not covered by samples and further that the labeled domain-class combinations are non-uniformly distributed in the domain-class space. The scenario is exemplary depicted in Fig. 1.4b. In the context of this thesis it is called the Domain Mixture scenario.

Since the biggest challenge of a domain invariant classifier here is to generalize to the domain-class combinations that are not represented at all or at maximum by unsupervised samples during training, the focus of the investigations will be on the maximization of the classification rate on these

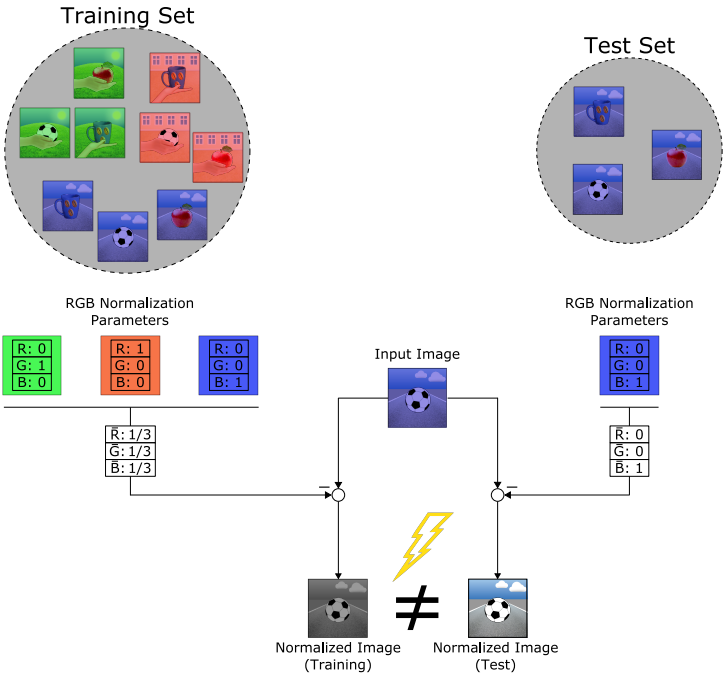
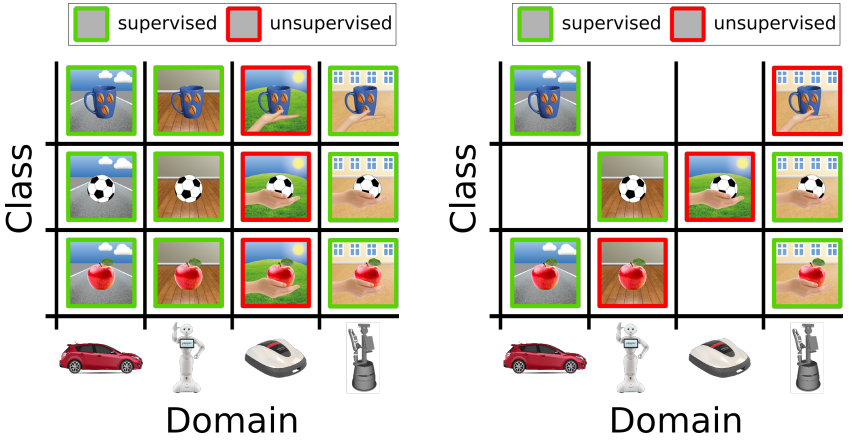


Figure 1.3: Example of a case where the unawareness of multiple domains in the training set can lead to negative transfer on the test domain. During training the RGB mean used for normalizing the input image through mean subtraction is generated from a mix of the three underlying domains, while during test the mean is generated only from the test data, where only a single domain is represented. Here, the test domain is one of the underlying training domains. The depicted scenario shows that an input image from the test domain is thus normalized differently during training and test. For a neural network the normalized sample appears as two different samples and might cause a false classification.



(a) Standard Scenario **(b)** Domain Mixture Scenario

Figure 1.4: Different domain scenarios. (a) Shows the typical scenario from literature. The main assumption is that all classes of the involved domains are covered by samples. Research investigates the transfer to a single domain, here the lawn mower, of which no or only unsupervised samples are available. (b) Shows the more realistic Domain Mixture scenario investigated in this thesis. It is highly relevant for multi-camera environments where not all classes have necessarily been seen in all domains, causing gaps in the domain-class space, and domains might only partially be labeled. Here, the classification performance of interest is on the domain-class combinations that are not labeled, independent of the domain.

domain-class combinations. These evaluations are of particular interest when considering that the shared image database for a common classifier of multiple systems is only built up over time. It will be shown that for the Domain Mixture scenario the application of DA is essential, since otherwise the neural network takes the easiest path and learns to classify the training samples based on values of domain specific factors. As a consequence, it fails to classify samples of unsupervised or not represented domain-class combinations where the values of these domain specific factors differ.

The third main contribution proposes an improved adversarial DA approach that can help to prevent negative transfer in a constellation of multiple supervised training domains. It is based on the adversarial DA approach from [22] which uses an additionally attached domain classifier

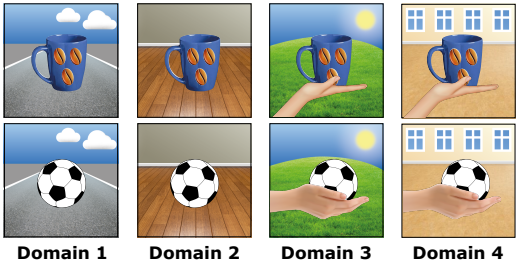


Figure 1.5: The application of a domain classifier based DA approach on the shown domain constellation would remove all factors from the feature representation that allow a discrimination of domains. The hand would be removed here as well, since it is only present in domain 3 and 4. However, here the removal of it is harmful for the object classification task, since it provides information about the object by its posture. A flat posture indicates the mug, while a hollow posture the ball.

to remove the bias that discriminates domains. The goal of it is to classify the domain origin of each sample during training, while being attached via a gradient reversal layer. The combination of the domain classifier and the gradient reversal layer targets to remove all factors from the feature representation that allow a discrimination of domains. However, as it will be discussed and experimentally shown, this can also lead to the removal of task relevant information and therefore cause a reduced classification performance, i. e. negative transfer. Fig. 1.5 shows an exemplary domain constellation where the hand would be such a factor that helps to distinguish domains. With the adversarial DA approach the hand would therefore be removed. However, within domains where the object is presented in a hand, the hand posture, which is flat or hollow, is also task-informative, since it indicates the actual object class. A removal of the hand from the feature representation therefore can cause negative transfer.

To avoid this negative transfer, Factor-Preserving DA will be introduced. It allows to preserve a chosen factor during DA by switching off competition in the domain classifier between domains that differ in this chosen factor. At the example of extensive experiments, including visualizations of learned features and attention areas, it will be demonstrated that the preservation of factors with the introduced Factor-Preserving DA can help to prevent negative transfer in a leave-one-domain-out setting. With this method significant performance improvements for the average accuracy as well as for the minimum accuracy, which can be considered as of great importance

for safety critical systems, can be observed.

1.1 Contribution

The overall contribution of this thesis comes with a strong focus on in-depth investigations of special cases in the DA context, which are highly relevant for distributed camera based systems, e. g. multi-robot environments. Subresults of this thesis were published in [84], [85], and [86]. Some passages of the mentioned publications have been quoted verbatim in this thesis. The overall contributions can be summarized as follows:

- Introduction of a factor theory, which allows the interpretation of differences in domains more fine-grained than the unspecific notion of data distributions [86].
- Extensive in-depth investigations and discussions of neural network behavior in context of DA [85, 86].
- Prove that the naive application of input normalization can cause negative transfer, if the application domain was already involved in the training set [84].
- Investigations of multiple variants of the Domain Mixture scenario with high importance for real-world applications [85].
- Development of a novel DA approach, named Factor-Preserving DA, that allows to preserve a chosen factor in DA with multiple involved domains [86].
- Extensive evaluations and analyses of the novel Factor-Preserving DA method on two real-world datasets, that show in which cases negative transfer can be reduced and standard adversarial DA be outperformed [86].

1.2 Thesis Outline

The structure of the remainder of this thesis is depicted in Fig. 1.6, where chapters involving novel contributions that resulted from this PhD project are marked in blue.

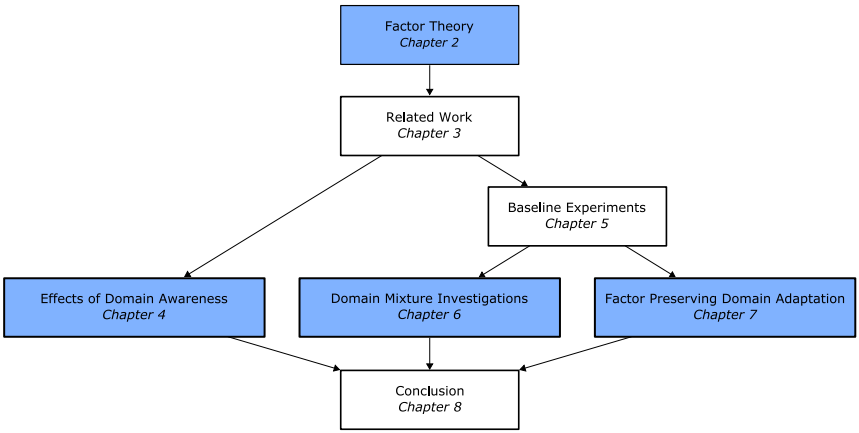


Figure 1.6: Thesis structure. Chapters containing the main contributions of this thesis are colored in blue.

The subsequent Chapter 2 starts with theoretical fundamentals that include a formal definition of domains, as well as the typical data shift types that cause the differences between domains. The factor theory, which generally allows to describe scenes captured in images by a superposition of factors, and thus to describe the characteristics of domains by domain factors, will be introduced there as the main part of the chapter. The principles of this theory will be used throughout the thesis to explain and predict the results of experiments.

Chapter 3 introduces related work and general capabilities of deep neural networks when it comes to feature manifestation and generalization between domains. The first section will give an overview of different image datasets that are commonly used for benchmarking. This is followed by a short recap on convolutional neural network architecture principles and their capabilities to generalize. After that, two main ways of DA will be introduced with example approaches from literature and a review on current research in the context of negative transfer will be given. All explanations in both sections come with a strong focus on the introduced

factorization view. Finally, this chapter ends with a discussion of typical feature manifestations of deep classification models under specific example domains and details on the attributes of factors in context of classification tasks.

Chapter 4 introduces the importance of awareness about domains during training and test of a neural network at the example of patch-based road terrain classification on camera images. Three different cameras, each with its own RGB sensitivity characteristics, are treated here as individual domains. Using the parameter based DA method of RGB mean subtraction it will be shown how negative transfer can be avoided when being aware about the treatment of multiple domains during training. Additionally general recommendations will be given on how to best apply downloadable pre-trained networks on own data.

The following chapters are concerned with effects of adversarial DA approaches. First in Chapter 5 the architecture of [22] will be described, which will be used as a baseline approach in two variants. Then the MNIST domain dataset, that is relevant for Chapter 6, and the CORe50 dataset, that is relevant for both, Chapter 6 and Chapter 7, will be presented. Additionally, for the CORe50 dataset transfer experiments will be carried out with the introduced architecture, that serve as a baseline for the following two chapters.

Chapter 6 starts with the introduction of two relevant variants of the Domain Mixture scenario and proceeds with experiments of these on the introduced MNIST benchmark dataset. The experiments are further extended to the CORe50 real-world object classification dataset to underline the high relevance of this scenario for real-world problems. For the evaluations a novel measure called 'subset confusion' will be introduced, which helps to understand the learned features of the neural network better. As the major outcome of this chapter, the results will reveal that the application of DA is strictly necessary for the investigated Domain Mixture scenarios to obtain a high classification performance.

In Chapter 7 the novel approach of Factor-Preserving DA for datasets with multiple domains will be introduced. Extensive evaluations will show how error causing factors can be identified from one-to-one transfer experiments and later be used for FP-DA to prevent negative transfer in the multi domain setting. The investigations of such error causing factors will not only be carried out on domain level, but also on class level, which provides potentials for improvements of the proposed approach in future works. The effectiveness of Factor-Preserving DA will further be substantiated by the two visualization methods t-SNE and Grad-CAM.

By using the OpenLORIS object dataset as a second dataset, it will be shown how factors must be represented for FP-DA to work and to result in leave-one-domain-out experiments in the highest average and highest minimum performance.

In Chapter 8 the main contents of this thesis will be summarized and an outlook to future research directions will be given.

2 Theory of Visual Factors

This chapter begins with the introduction of formal mathematical notations that are used throughout the DA research community. Further contents are the introduction of the most common types of data shifts and with it the notion of data distributions that is mostly used to explain the differences of domains. As main part of this chapter, with high relevance for the remainder of this thesis, the theory of visual factors will be introduced. It allows to describe certain aspects of domains more detailed than data distributions and was developed as part of this PhD project and published in [86].

2.1 Domain Adaptation Fundamentals

Supervised machine learning algorithms usually optimize the parameters of a classification model based on a labeled dataset

$$(\mathbf{X}, \mathbf{Y}).$$

A dataset consists of a set of n multi-dimensional feature vectors $\mathbf{x}_i$,

$$\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\},$$

together with a set of corresponding label vectors,

$$\mathbf{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\}.$$

In image classification a sample $\mathbf{x}_i$ usually describes a vector of feature values that have been extracted from an image $\mathbf{b}_i$. The feature space $\mathcal{X}$ from which the feature vectors are drawn could be given by a raw RGB image with a certain dimensionality, or a feature space with a manually pre-defined dimensionality where the non-linear mapping is defined by the layers of a neural network. In supervised machine learning, each $\mathbf{x}_i$ has a corresponding label vector $\mathbf{y}_i$, drawn from a label space $\mathcal{Y}$, that assigns one or multiple ground-truth classes to the sample i. A given dataset $(\mathbf{X}, \mathbf{Y})$

comes with the joint probability distribution

$$P(\mathbf{x}, \mathbf{y}) = P(\mathbf{y}, \mathbf{x}),$$

which can be reformulated, using the product rule of probabilities [6] as

$$P(\mathbf{x}, \mathbf{y}) = P(\mathbf{y}|\mathbf{x})P(\mathbf{x}), \tag{2.1}$$

or respectively as

$$P(\mathbf{x}, \mathbf{y}) = P(\mathbf{x}|\mathbf{y})P(\mathbf{y}). \tag{2.2}$$

Following this definition, a difference, i. e. a shift, between datasets is ultimately expressed in different joint distributions $P(\mathbf{x}, \mathbf{y})$, which in turn can be traced back to differing conditional and marginal distributions of the datasets. If such a shift between datasets exists, those are usually called domains. Specifically, the domains that are used for the optimization of a classifier are called *source* domains and the ones on which testing or application is carried out are called *target* domains. Respectively, when considering only a single source and a single target domain, there are two datasets $(\mathbf{X}_s, \mathbf{Y}_s)$ and $(\mathbf{X}_t, \mathbf{Y}_t)$ with their corresponding joint distributions $P_s(\mathbf{x}, \mathbf{y})$ and $P_t(\mathbf{x}, \mathbf{y})$. Note, in the following it is only referred to a single source and a single target domain, however, this does not represent a restriction to only two domains. The assumptions can easily be extended to multiple domains.

There are special types of data shifts between domains if only one of the two factors in (2.1) or (2.2), i. e. either only the conditional or only the marginal distribution, differ [47]. The most common reasons that lead to such a special type of data shift are a sample selection bias or non-stationary environments. The three most relevant special data shift types will be described in more detail in the following, using exemplary a single source and a single target domain. Note that any two datasets are equally applicable here. An overview of the three types is shown in Fig. 2.1.

Prior Shift. This data shift type exists when only the marginal probabilities of the classes differ between the domains, i. e.

$$P_s(\mathbf{y}) \neq P_t(\mathbf{y}),$$

and the conditional probabilities are identical, i. e.

$$P_s(\mathbf{x}|\mathbf{y}) = P_t(\mathbf{x}|\mathbf{y}).$$

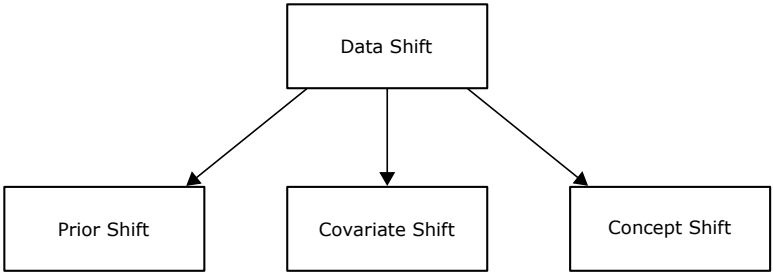


Figure 2.1: Special types of data shift.

Prior shift is mainly caused by a sample selection bias, meaning that the ratio of samples belonging to a certain class differs between the source and the target domain. A simple example for this shift could be the task of classifying images of products from a production process into faulty and correctly produced products, where the occurrence of faulty products is below 5%. If, however, the source domain that is used for training an image classification model is composed of images where faulty and correctly produced product image samples are represented equally, the classifier might be incorrectly biased towards faulty products. Consequently this would cause problems during application which can be interpreted as the target domain dataset.

In general, given prior shift, the optimized classification model might be biased towards a certain distribution among classes from the source domain that does not hold for the target domain. If the shift is explicitly known, typical methods that are used to avoid such a bias are sample re-weighting or oversampling. In re-weighting the influence of overrepresented classes on the parameter optimization is reduced by giving these samples a smaller weight during the parameter update, or vice versa for underrepresented classes. In oversampling, samples from the underrepresented classes are copied and added to the source dataset, to obtain an artificial correction of the class sample ratio to the desired ratio of the target dataset.

Covariate Shift. This data shift type occurs when only the feature distributions of the samples between the domains differ, i. e.

$$P_s(\mathbf{x}) \neq P_t(\mathbf{x}), \tag{2.3}$$

while the conditional distributions are identical, i. e.

$$P_s(\mathbf{y}|\mathbf{x}) = P_t(\mathbf{y}|\mathbf{x}). \tag{2.4}$$

The reason for this shift type can also be a sample selection bias. This time however, it is not caused by unevenly distributed samples to the classes in the dataset, but rather by an imbalance with regard to the features that represent a certain class in the dataset. Besides a sample selection bias, missing data can also be a reason. An example here could be be a face detection system where images of elderly people are completely excluded from the source domain dataset that is used for the training of the system. Thus typical features that occur with elderly people are completely unknown to the system. In real-world applications, which can be seen as the target domain, however, such features can occur and faces of elderly people might therefore not be detectable. Another typical example, that is mostly relevant for this thesis, are changes related to the visual characteristics of the environment between the domains in images of an object classification task as it was depicted by the introductory example of Fig. 1.1.

The main problem that this type of data shift can cause is that a classification model trained by standard methods embeds unintentionally irrelevant features from the source domain that do not exist in the target domain. In the remainder of this thesis several DA approaches will be presented that target to avoid this behavior.

Concept Shift. This type of shift is also often referred to as concept drift [21]. Here one of the conditional distributions between the domains differs, i. e.

$$P_s(\mathbf{y}|\mathbf{x}) \neq P_t(\mathbf{y}|\mathbf{x}) \quad \text{or} \quad P_s(\mathbf{x}|\mathbf{y}) \neq P_t(\mathbf{x}|\mathbf{y}), \tag{2.5}$$

while the corresponding prior probabilities are identical,

$$P_s(\mathbf{x}) = P_t(\mathbf{x}) \quad \text{or} \quad P_s(\mathbf{y}) = P_t(\mathbf{y}). \tag{2.6}$$

The main reason for such a shift between datasets are non-stationary environments. This means that the mapping from feature vectors to classes, or vice versa, changes over time. An example where this shift occurs could be a production process where at some point in time certain features of work pieces were not considered to classify them as defective, while at a later point in time they would be used to classify them as defective. This change of interpreting features could occur due to a change in production

quality requirements. The consequence of such a data shift is that decision boundaries that were learned to solve a classification task based on the source domain might not hold anymore on the target domain and need to be adjusted or have to be learned completely new.

Of the three data shift types presented, this thesis deals mainly with covariate shift occuring between domains in image classification tasks. The covariate shift of the investigated cases here is mainly related to changing recording environments and different cameras that make the captured images look different.

In recent literature [12, 70, 95, 106], source and target domains are typically formally defined by the domain sets

$$D_s = \{\mathcal{X}_s, P_s(\mathbf{x})\} \quad \text{and} \quad D_t = \{\mathcal{X}_t, P_t(\mathbf{x})\} \tag{2.7}$$

with the corresponding task sets

$$\mathcal{T}_s = \{\mathcal{Y}_s, P_s(\mathbf{y}|\mathbf{x})\} \quad \text{and} \quad \mathcal{T}_t = \{\mathcal{Y}_t, P_t(\mathbf{y}|\mathbf{x})\} \tag{2.8}$$

where $\mathcal{X}_s$ and $\mathcal{X}_t$ are potentially different feature spaces and accordingly $\mathcal{Y}_s$ and $\mathcal{Y}_t$ different label spaces. In DA literature [12, 106] it is generally distinguished between heterogeneous DA, where the feature spaces differ, i. e. $\mathcal{X}_s \neq \mathcal{X}_t$, and homogeneous DA, where $\mathcal{X}_s = \mathcal{X}_t$. This work will only consider the latter case. Regarding the tasks there is a differentiation between open set DA, i. e. $\mathcal{Y}_s \neq \mathcal{Y}_t$ and closed set DA, i. e. $\mathcal{Y}_s = \mathcal{Y}_t$ [10]. Thus, open set DA describes the case where the involved classes in the classification task of the source domain and the target domain differ. The scope of this work will only be closed set DA. As in this thesis only covariate shift (see (2.3) and (2.4)) will be considered, the difference between domains therefore is only caused by the accompanying marginal distributions.

In general, the goal of a classifier is to predict the label vector $\mathbf{y}_i$ that belongs to a given sample $\mathbf{x}_i$. To do this it needs to have an internal model that approximates the conditional probability distribution so that the approximation $\tilde{P}(\mathbf{y}|\mathbf{x})$ satisfies

$$\tilde{P}(\mathbf{y}|\mathbf{x}) \overset{!}{=} P_s(\mathbf{y}|\mathbf{x}) = P_t(\mathbf{y}|\mathbf{x}).$$

Typically, this approximation is based on a supervised dataset, here the source domain dataset $(\mathbf{X}_s, \mathbf{Y}_s)$, that is used for the optimization of the internal parameters of the classification model. However, since the approximation is learned only based on the source dataset, the classification model

might be biased towards the source domain, i. e. it might include features that exclusively occur in the source domain to solve the task. Consequently a decreased performance on the target domain dataset $(\mathbf{X}_t, \mathbf{Y}_t)$ can be expected, since $P_s(\mathbf{x}) \neq P_t(\mathbf{x})$. Further details and illustrative examples on this topic will be given in Chapter 3. Approaches from the research area of DA target to make the approximation $\tilde{P}(\mathbf{y}|\mathbf{x})$ less dependent on the characteristics of the source domain dataset, i. e. $P_s(\mathbf{x})$, and thus closer to the actual conditional probability distribution $P_s(\mathbf{y}|\mathbf{x}) = P_t(\mathbf{y}|\mathbf{x})$. Selected recent approaches that show how this can be achieved in context of deep neural networks will be presented in Chapter 3.

2.2 Domains in Context of Visual Factors

As shown in the previous section, in literature the difference between domains is usually mainly explained by different feature distributions of source and target domain, i. e. $P_s(\mathbf{x}) \neq P_t(\mathbf{x})$. However, using only this notion, especially in the context of image classification, does not allow to model the rich constellation of multiple domains comprehensively. In this section the theory of visual factors will be introduced, which targets to describe domains as combinations of a set of shared underlying factors instead. Using the relations between factors in the domains, the effects of DA approaches can better be predicted and thus influenced.

The theory of visual factors is based on the assumption that each high-dimensional image $\mathbf{b}_i$ of a scene i can be generated by a mixing function

$$M(\mathbf{f}_i) = \mathbf{b}_i,$$

from an activation of factors

$$\mathbf{f}_i = \{f_{i1},\ f_{i2},\ ...,\ f_{il}\}.$$

Each factor can either be a continuous or categorical variable. Generally, there is no fixed combination of the mixer M and the factors $\mathbf{F}$ to model the same image data $\mathbf{I}$, rather there are infinite possibilities for modeling the image data. The goal of this factor theory is not to analytically solve the factorization for a given optimality criterium. The goal is more to discuss properties of factors in relation to image classification tasks in context of domain transfer. As it will be shown in this thesis, the theory can be used to describe attributes of image datasets more detailed than only the notion of data distributions. Furthermore, it allows to predict and influence the effects of DA approaches.

When it comes to the classification of an image $\mathbf{b}_i$, usually a feature extractor is used to transform the image from the original image space, e. g. the RGB image space, to a different multi-dimensional feature space that mostly has a lower dimension than the original input space. The transformation function of the feature extractor can be defined as

$$\mathbf{x}_i = G_e(\mathbf{b}_i; \theta_e),$$

which depends on the internal parameters θ_e. The target of $G_e(\cdot)$ is to transform the image to a space where classes can easily be separated. The result of this transformation is the feature vector $\mathbf{x}_i$. The actual

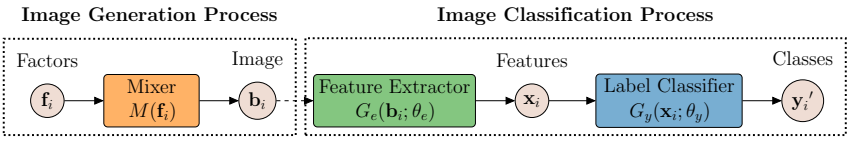


Figure 2.2: An image $\mathbf{b}_i$ can be generated by a complex mixing function M from an activation of factors $\mathbf{f}_i$. For image classification, a feature vector $\mathbf{x}_i$ can be extracted from $\mathbf{b}_i$, and the label vector $\mathbf{y}_i{}'$ can be predicted by a label classifier.

classification of the image $\mathbf{b}_i$ is then realized in a second step by a label classifier with the transfer function

$$\mathbf{y}_i{}' = G_y(\mathbf{x}_i;\theta_y)$$

which assigns a vector of class probabilities $\mathbf{y}_i{}'$ to the feature vector $\mathbf{x}_i$ using its internal parameters θ_y. A visualization of the overall process that includes image generation and classification is given in Fig. 2.2.

Considering a classification task, the factors $\{f_{i1}, f_{i2}, ..., f_{il}\}$ can be categorized as task-informative or not task-informative. Accordingly, the value of a factor that is task-informative is strongly correlated to a specific class. This can be direct or more indirect with respect to the target class. An example for an indirect correlation would be given for the classification of objects that are presented in a human hand. Here the factor *'hand posture'* is not directly related to the object, but can still be task-informative, since objects are usually held in different ways which can give hints about the ground-truth object class.

Given multiple domains, factors can also be categorized by their property of being domain-informative. A factor is domain-informative if its value has a correlation with a certain domain, i. e. it has a value that mainly shows up in a certain domain. In this work, however, a more strict definition of factors is used. This definition restricts the domain-informative factors to those that have a constant value for all images within a domain, independent of the ground-truth class, and change their value across domains. In real-world image domain datasets such factors can typically be related to fixed recording locations, which induce changing lighting or background related factors across domains. According to this definition, a domain can be described by multiple of these factors. Throughout this thesis such factors will be referred to as domain factors.

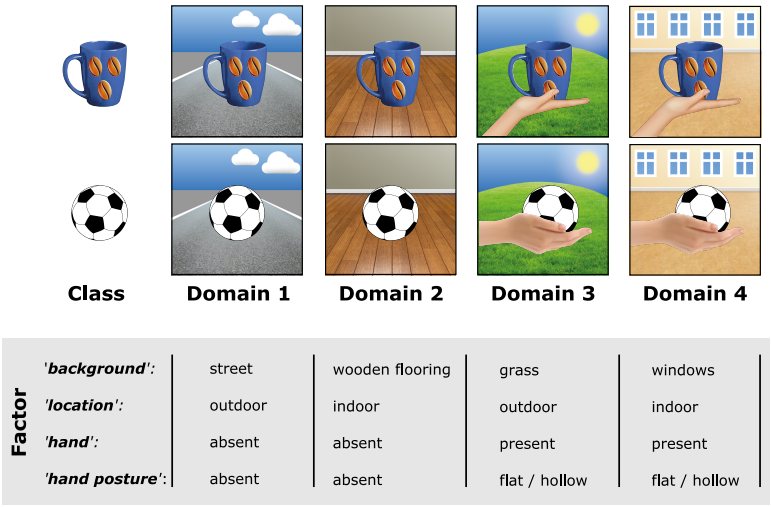


Figure 2.3: The variations in the shown set of domains can be described by different categorical factors, as e. g. *'background'* ∈ {street, wooden flooring, grass, windows}, *'location'* ∈ {indoor, outdoor}, *'hand'* ∈ {present, absent}, *'hand posture'* ∈ {absent, flat, hollow}. A factor is called a domain factor, if it has a constant value within each domain, but different values across domains. All of the mentioned factors are domain factors, except the *'hand posture'*, whose value changes here also along classes. Since its values correlate in Domain 3 and 4 with the same classes it can be categorized as task-informative.

Fig. 2.3 shows an exemplary factorization, including the categorization of the factors, for the four domains of the introduction chapter.

Using the definition of domain factors it becomes obvious that a domain-invariant feature extractor should not learn features that are related to domain factors. If such features are learned, it can lead to misclassification in a target domain, where the values of the domain factors potentially change. However, a domain factor can also be a task-informative factor, or it can be close to a task-informative factor. Neglecting domain factors in these cases can have a negative effect. Further details on this negative effect will be given in Chapter 3.

In general there are various possibilities on how to factorize scenes by visual factors. One way is for example to choose the factors manually, i. e. by human design, as it was done in Fig. 2.3. This has the advantage that the number of factors is actively chosen and the nameability of the

factors is always guaranteed. On the other hand, when using a factorization that is estimated from a component analysis method, the likelihood of a good nameability of the determined factors is clearly reduced, since those factors might not necessarily be human interpretable. Other aspects like the mutual dependence of individual factors, which accordingly influences their values, and the controllability of a factor in a scene are also dependent on the chosen factorization. In the end, a well-chosen factorization might provide increased predictability and explainability of classification models when handling new factor values or new combinations of known values. However, this also depends on the fineness of the factorization. While for some constellations of domains the choice of a factor *'location'* $\in$ {indoor, outdoor} is sufficient to explain certain effects, for others a more fine-grained factorization that additionally includes other factors as for instance *'lighting'* might be required.

Exemplary views on how to factorize scenes can be found in scene rendering or photography. In the rendering view the factorization is based on human interpretable factors:

- Object factors: *'type'*, *'shape'*, *'color'*, *'texture'*, *'viewpoint'*, ...
- Light source factors: *'type'*, *'positioning'*, *'emission power'*, ...
- Camera factors: *'sensor'*, *'filter'*, *'processing'*, *'viewpoint'*, ...

The factorization and thus the number of factors is pre-defined by the rendering program. The factor values and their mutual dependence are fully controllable by a rendering artist. The artist for instance ultimately decides how chosen object colors appear in combination with placed light sources in the rendered scene.

The photographer view differs from the rendering view in having increased flexibility between more and less control over factors and their mutual dependence. While in natural images, the object in focus and the viewing angle are usually controlled by the photographer, there is less control over natural lighting conditions and context objects. Here, certain objects usually induce other context objects, just as a car would most likely induce asphalt. On the other hand, when considering a pure studio environment with artificial lighting and background, the overall controllability over factor values is significantly increased.

The presented views describe two variants regarding controllability of factor values. Sometimes, real-world image datasets, especially those used for DA, describe a mixture of these. This is for example the case when

objects are artificially placed in environments where they usually would not be observed. Here the target object related factors and the environmental background factors are highly controlled as in a studio or a rendering program, but factors like the natural lighting might be less controlled.

3 Related Work and Classifier Habits

The first section of this chapter will introduce different datasets that are commonly used for general image classification tasks, and in specific those that are used for DA research. All datasets will be analyzed in relation to the introduced concept of visual factors. Afterwards, a brief recap on the basic principles of Convolutional Neural Networks (CNNs) will be given, which forms the foundation for understanding the subsequent introduction of recent DA approaches and the analysis of network behavior for different domains. Most parts of this chapter are based on the contents of [84–86] that were published in context of this PhD project.

3.1 Image Datasets in Context of Visual Factors

For the evaluation of image classification approaches there are multiple image datasets with labeled images publicly available. Many of them are based on internet search results, like for example the very well-known ImageNet database [13] or the Caltech-256 dataset [29]. Generally using images from search engines provides a large variation within the individual class representations, like for example the object instances or the surroundings in which the objects are present. This is a good foundation to optimize a model that generalizes well, however, even such large datasets are afflicted by dominant dataset biases. Typical biases here are for example a western world bias [15, 89], that is expressed by images mainly showing scenes from the western world, or a capture bias [98], which can be expressed by either only showing professional or only amateur photos. Such biases ultimately influence the control over different factor values like for example certain object types or the viewing angles.

In datasets that are commonly used for DA research, like [46, 51, 75, 76], specific factors are usually more controlled regarding their values. In

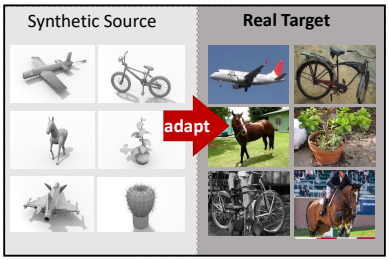
[5, 74, 82, 103] the increased control is for example achieved by introducing constraints on internet search engines that limit the results for instance to only images from Amazon or only clipart images. The search results based on one of such constraints then usually describes one of multiple predefined domains. Such domains can be found for example in the well-known Office-31 DA dataset [82] (see Fig. 3.1b) or the DomainNet dataset (see Fig. 3.1c) that also includes domains solely based on sketches or paintings. Another example here is a restriction to synthetic or real images within a domain only, as it is for example the case in the Syn2Real dataset [76] (see Fig. 3.1d). However, also such restricted search engine based domains show still only little control over certain factors, like for example the actual class instances that are meant to represent a specific class, e. g. the object types. In datasets of robotics research, usually more control over factors in general, and in specific over the class instances is introduced. The datasets there are often composed of domains where the same objects are placed in different environments, meaning that the values of environmental factors change across domains. Two datasets where this was done so are the OpenLORIS object dataset [90] and the CORe50 dataset [56] (see Fig. 3.1e). Note that domains there are usually not named as such but rather as different recording sessions. Due to the changing conditions within these sessions, they still meet the definition of domains and can therefore be used as such. Generally such datasets allow more clear investigations due to less superimposition effects that could be caused by different object instances within the domains. Both presented datasets, CORe50 and OpenLORIS will therefore also be used for the experiments in Chapter 7. Similar control over objects can also be found in the popular Office-31 DA benchmark [82] (see Fig. 3.1b), where the Webcam and the DSLR domain were generated similarly.

Single domains in other DA datasets were generated manually by changing specific factors artificially based on the segmentation of the image. This procedure has been applied, for example, for the handwritten digit dataset MNIST-M [22] which has been generated from MNIST [49] by changing the background artificially (see Fig. 3.1a). In research, both datasets are commonly interpreted as individual domains. Full control over all factor values is generally only found in completely rendered datasets. In the Shapes3D dataset [9] (see Fig. 3.1f) simple object shapes are positioned in an artificial environment with changing values for factors like size, viewing angle, color, and others. Similar datasets can be found in the disentanglement library [55]. Also here, domains are not explicitly defined, however, they can easily be manually generated under the constraint of consistent

(a) MNIST [49] (S) and MNIST-M [22] (M) (2 domains)

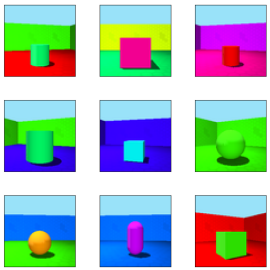
(b) Office-31 [33] (3 domains)

(c) DomainNet [74] (6 domains)



(d) Syn2Real [76] (2 domains)

(e) CORe50 [56] (11 sessions)



(f) Shapes3D [9] (no pre-defined domains)

Figure 3.1: Selected image classification benchmark datasets. a) - d) are commonly used for DA, e) and f) have not been found to be used in the DA context, but provide a clean setting for DA investigations. Images were taken from the listed sources and partially adapted.

domain factor values within a domain.

3.2 Recap - Convolutional Neural Networks

To interpret neural network behavior in context of DA, it is required to have a fundamental understanding of the basic functionalities of the architectural main components. Therefore a brief recap of the basics of CNNs will be given here.

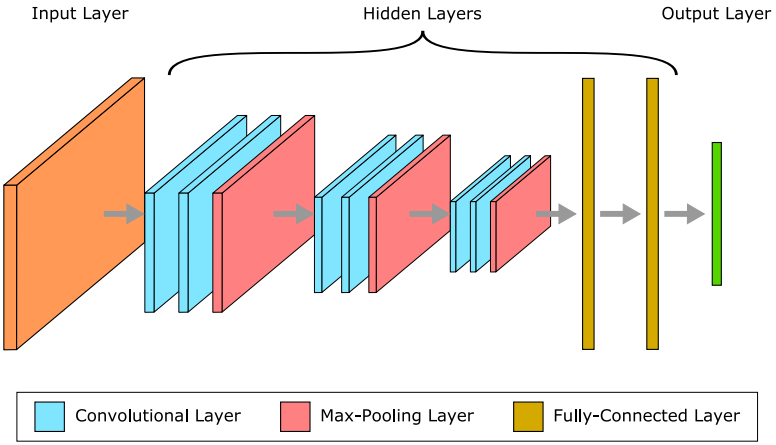


Figure 3.2: Typical structure of a standard CNN architecture. Usually one or multiple convolutional layers are followed by a max-pooling layer. Each convolutional layer can have multiple channels (feature maps) that are not shown in this figure. The extracted features are then usually combined in the fully-connected layers, which are followed by the final output layer.

A typical CNN architecture consists of an input layer, multiple hidden layers, and an output layer. An example of this structure with typical standard layers is shown in Fig. 3.2. The standard layers, like convolutional layers and fully-connected layers, usually consist in their core of classic artificial neurons. The neuron's inputs are multiplied by weights, a bias is added, and the sum of all inputs is forwarded through a chosen activation function to the neurons of the next layer. The model parameters, i. e. the weights and the biases, are updated end-to-end, based on a training dataset and a chosen loss function that is usually targeted to be minimized with respect to the parameters and the given data. Commonly a backpropagation

algorithm is used for this.

The input layer is usually described by the input images that are represented in a pre-defined feature space, like the RGB image space. The hidden layers in CNNs are usually composed of two types, fully-connected layers and convolutional layers. Fully-connected layers describe the classic connections, where each neuron of a layer is connected to each neuron of the subsequent layer via a weight. In convolutional layers only a subset of the neurons, that lie within a receptive field, is connected to a neuron of the subsequent layer. The receptive field can be interpreted as a filter with a limited size, that is moved over the neurons. The weights are shared for all filter positions and will only be updated during backpropagation. An example for this process is shown in Fig. 3.3. The receptive field can be interpreted as a feature detector that generates a 2D feature map and detects specific features invariant of their location. While in the shallow layers of a CNN usually features like specific edges or colors are detected, with increasing depth more comprehensive features are detected.

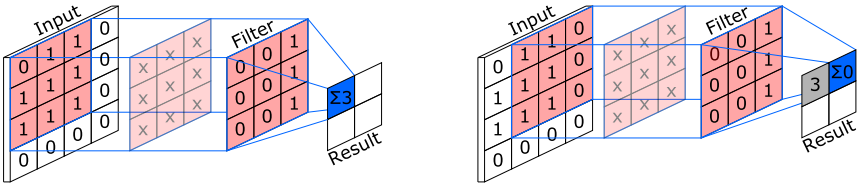


Figure 3.3: Receptive field of a convolutional filter moving over the input with a step-size of 1. The filter weights are multiplied by the corresponding values within the position of the receptive field on the input layer and summed up. Note, the subsequent pass of the resulting values through an activation function is not shown here.

Since for general image classification tasks the exact location of the detected features is mostly not of major importance, commonly max-pooling layers are used. Those layers only forward the maximum value within a small receptive field and thus reduce the feature map size and ultimately also the overall amount of parameters in the network.

Several stacks of convolutional layers and pooling layers are usually followed by one or multiple fully-connected layers that take the extracted features as input, combine them non-linear, and output the classification result. For further details on general neural network architectures and training it should be referred to [6].

In context of DA research the question arises, which conditions must be

met so that such an architecture can be described as domain invariant. As stated earlier, from a theoretical point of view this means that features that are close to the output layer should not contain information about domain factors. However, a standard neural network is not enforced to fulfill this requirement and often uses domain specific information for classification. This means that the network's filters are likely to be biased towards the source domains that are used for optimization. Consequently such filters might fail their actual task on domains that were not involved during training. A highly simplified example for such a situation is shown in Fig. 3.4, where the object appearance is unchanged across domains, but the value of the factor related to the background changes in the target domains. For edge filters in general, there is a high chance of domain factor dependency, since an overlap to the non-relevant background exists and weights and biases are adapted to the given source domains. Generally the learned features in a CNN are mostly more complex than the given example, since numerous filters from multiple layers are connected with each other, directly or indirectly, forming more sophisticated feature detectors. However, if no restrictions are applied, this does not resolve the domain factor dependency problem. DA approaches try to tackle this issue by introducing additional constraints on the feature learning process.

3.3 Domain Adaptation Approaches in Context of Visual Factors

Generally current DA approaches can be divided into two main categories, parametric DA and learned DA. While with parametric DA approaches the adaptation to the target domain is achieved by adapting certain parameters after the model has already been optimized, in learned DA approaches the adaptation is achieved already during optimization by controlling the feature learning process. Both categories will be presented in the following in more detail.

Parametric Domain Adaptation

Parametric DA is mostly based on a simple mathematical layer operation that is designed to normalize out a given aspect of the data explicitly. Usually the adaptation of selected parameters of this layer to the target domain is done after the neural architecture has been optimized on the source domain. If the bias that distinguishes source and target domain

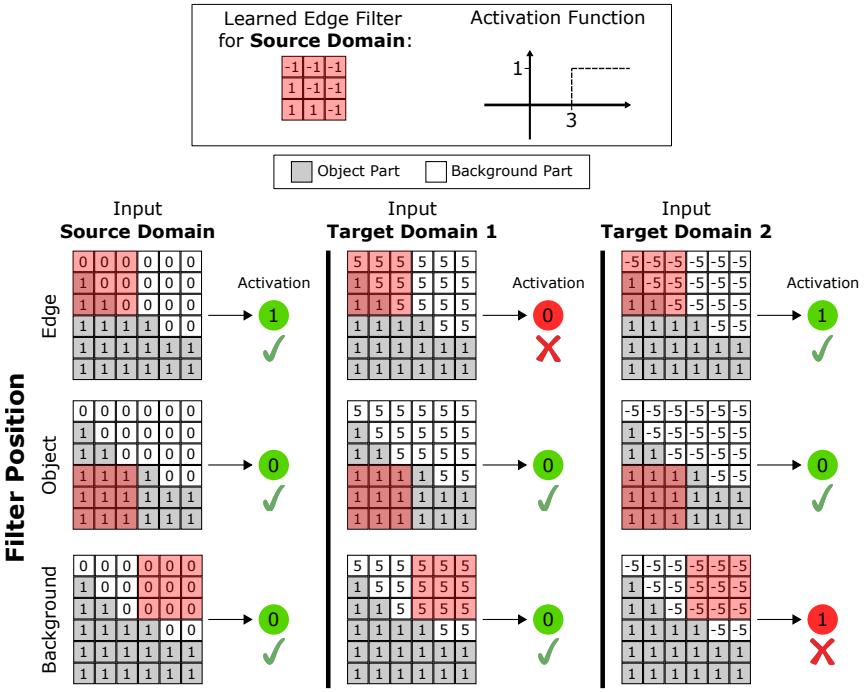


Figure 3.4: Simplified description of the domain transfer problem based on a single edge filter that has been optimized on the source domain and is evaluated on two target domains with changing background values. For simplification the bias parameter is neglected here. Object parts in the input are represented with a value of 1. As desired, in the source domain the filter is activated on the edge shape and not activated when completely on the object or the background. In contrast, in target domain 1 it fails for the edge position and in target domain 2 on the background position by generating false-negative and false-positive activations respectively.

is known, it can be sufficient that such a layer realizes a simple standard normalization method that adapts the target domain to the model trained on the source domain. An example is given in Fig. 3.5.

A common bias is a differing RGB mean between the domains, which can be removed through standard dataset RGB mean normalization. For this, the global mean value for each color channel is computed over the entire dataset and used to normalize each input image. Despite the simplicity of

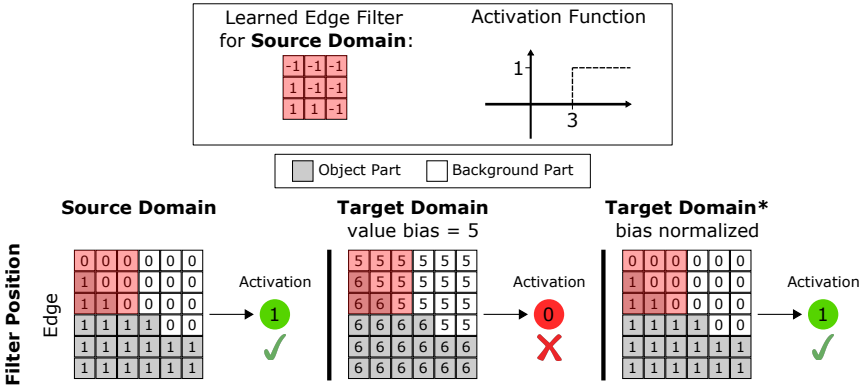


Figure 3.5: Example how parametric normalization can help to adapt a target domain to a classification model optimized on the source domain. The filter kernel that was optimized on the source domain (left) outputs a false-negative activation at the edge position, when applied on the original target domain (center). Normalizing the target domain by subtracting the bias from the input values, here 5, results in the desired behavior of the filter (right).

this method, there are also special constellations in which this procedure can lead to negative transfer, as it will be shown in Chapter 4.

Another parametric method that can be used for DA is batch normalization [38]. Originally it is implemented as an intermediate layer between standard layers like convolutional layers. It normalizes the batch's mean and variance for each training batch individually and like this targets to speed up convergence. During testing on single images the population mean and variance from the training set, i. e. source domain, are used for normalization. The architecture is consequently biased towards mean and variance of the source domain. In [53] they simply replace this mean and variance during test to the one of the target domain and show that competitive adaptation between domains can be achieved.

In terms of visual factors, the parametric DA methods mainly operate well for factors whose values influence the appearance of the entire image, like for example a chosen lighting type, or the camera type as it will be shown in Chapter 4. The presented methods are comparatively simple since they do not require to affect the actual learning process and are fully applicable post-training. Nowadays this is an important aspect, since comprehensive architectures whose parameters have been optimized on large image datasets, where the training process potentially takes up to

several weeks, can be downloaded and with a simple modification adjusted to own datasets. However, this requires that the bias that distinguishes the domains has to be easy to estimate and needs to be modelable.

Learned Domain Adaptation

When it comes to complex or unknown biases, an alternative method is to intervene already in the optimization process of the features. Here the normalization between the domains is learned. Many deep learning approaches target to obtain domain invariant features by adding additional cost functions to specific layers within the architecture. Available unsupervised image data of the target domain can here be exploited to minimize the difference in feature space between source domains and the target domain. If no target domain data is available, the different images of multiple involved source domains can be used to enforce general domain invariance that most likely also positively affects a target domain.

Recent approaches where the normalization of source and target domains is learned can be divided into kernel-based methods and adversarial methods. Both methods are usually implemented in form of an additional loss function besides the main classification loss, that is applied on a shared feature extraction layer to align the distributions of source and target domain there. A typical kernel-based method that is often found in literature [8, 42, 58, 59, 61, 101] is based on so-called maximum-mean discrepancy [28]. During optimization a training batch is composed of supervised data of the source domain and unsupervised data of the target domain. While only the supervised data of the batch is used to optimize the classification loss, the complete batch is used to align the distributions on the selected layer by minimizing the maximum-mean discrepancy loss.

Adversarial DA methods [2, 99] also try to match distributions. However, here the alignment is not based on measuring and minimizing the distance between the different domain distributions directly [23], but rather by modifying the feature activation in a way so that it is not discriminative with respect to the domains. It is mostly realized by a regressor or a domain classifier that is additionally attached to a shared feature extraction layer and connected via a gradient reversal layer [8, 22, 23, 60]. During training the gradient reversal layer acts in the forward phase as an identity function, but reverses the gradient from the domain classifier during backpropagation. An exemplary architecture with a domain classifier based on the proposed architecture of [22] is depicted in Fig. 3.6. Referring to the theory of visual factors, the goal of the domain classifier, together with the gradient

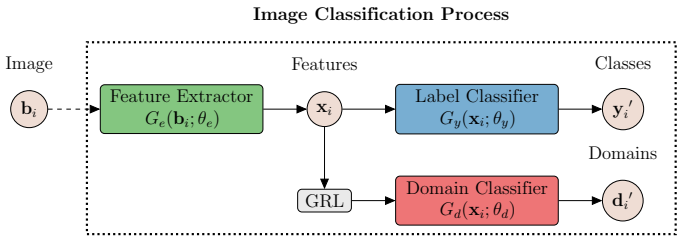
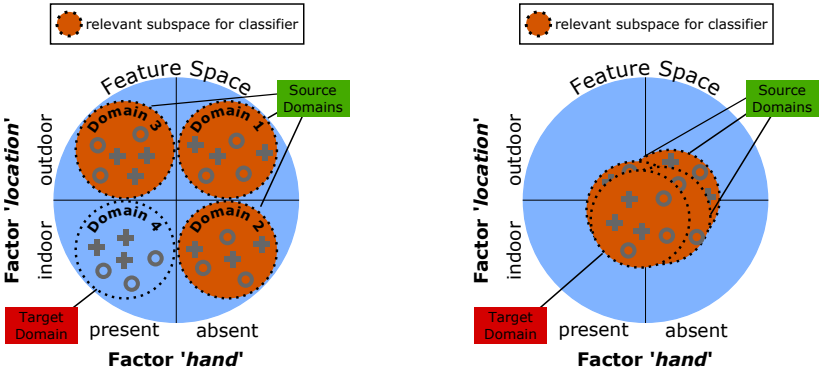


Figure 3.6: Adversarial DA approach from Ganin et al. [22]. An additional domain classifier is connected via a gradient reversal layer (GRL) to the shared feature extractor. The target of the domain classifier is to classify the domains. In combination with the GRL it removes all domain factors from the feature representation given by the parameters θ_e of the feature extractor.

reversal layer, is to remove all information about domains from the features of the layer to which the domain classifier is connected. Thus all domain factors should be removed, whereby the overall classification model is meant to become domain invariant for the domains used during training. A visualization of this concept in the feature space plot is given in Fig. 3.7. However, a drawback of this method is that also task-informative factors might be removed from the feature representation, which can aggravate the classification task. Details on this issue and an improved DA method to preserve a chosen factor will be presented in Chapter 7. Other adversarial DA approaches like [100] are based on the principle of generative adversarial networks [26]. Given a single source and target domain, their approach trains first supervisedly a feature extractor only on the source domain. Afterwards the unsupervised data of the target domain is used to train a separate target feature extractor that is enforced to generate similar outputs as the source feature extractor. This is achieved with the help of a discriminator that should not be able to distinguish the domains based on the outputs of the two feature extractors. In the end, the classification model for the target domain is formed by combining the target feature extractor with the source label classifier. A visualization of this procedure is shown in Fig. 3.8. In case of multiple source domains this procedure is carried out for each source-target pair individually and the classification of target domain images is based on an ensemble of weighted classification models. The weights are based on a score corresponding to the similarity of source and target domain that is evaluated in the training process of the target feature extractor [111]. With such a method, the domain factors of



(a) No DA **(b)** Adversarial DA [22]

Figure 3.7: Simplified visualization of the effect of an adversarial DA approach based on an attached domain classifier. (a) Without DA the feature subspace that is relevant for the classification model is only based on the source domains. Here, the target domain is out of the scope and a classification is more likely to fail. (b) Applying DA with a domain classifier, shifts the embedding of all domains to a common space, where the domain factors *'location'* and *'hand'* are not represented anymore. An improved classification performance on the target domain can be expected.

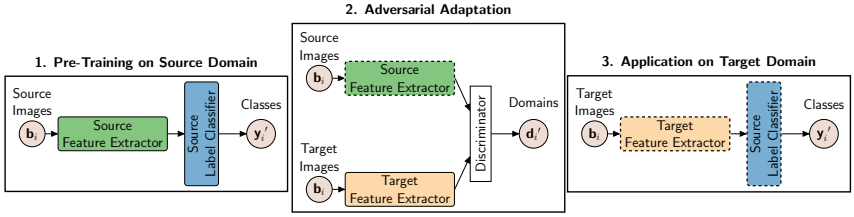
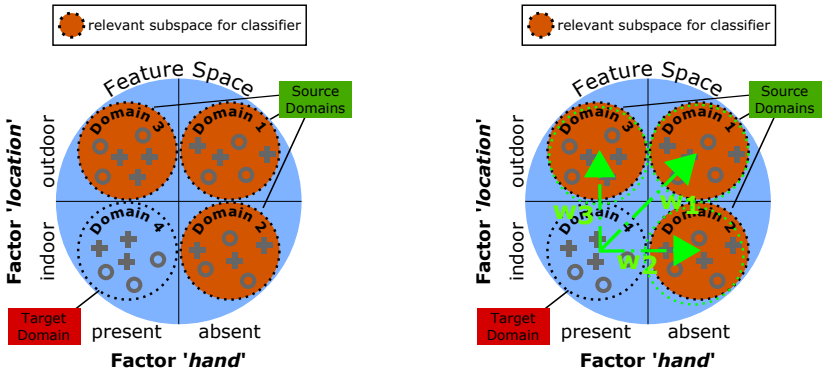


Figure 3.8: Generative Adversarial Network based DA approach from [100]. Dashed boxes represent fixed internal parameters. In a first step a feature extractor and a label classifier are trained end-to-end on the source domain. Then a new feature extractor for the target domain is optimized with the help of a discriminator to generate similar outputs as the source feature extractor for the unsupervised target domain images. For application on the target domain, the optimized target feature extractor is combined with the classifier of the source domain.

the source domains are still present, since the target domain is mapped to each source domain. The domain factor values of the target domain can be imagined to be artificially changed to the values of the paired source domain. A visualization of this process for the approach with multiple source domains is shown in Fig. 3.9.



(a) No DA **(b)** GAN DA [111]

Figure 3.9: Generative Adversarial Network (GAN) based DA approach [111] for multi-source DA. (a) The target domain is not in a relevant feature subspace of the classifier. (b) The target domain is mapped independently to each source domain. The overall classification model is based on an ensemble of weighted (w_1, w_2, w_3) source-target classifiers. The domain factor values of the target domain can be imagined to be artificially changed to the values of the source domains.

Negative Transfer

Commonly most of the approaches in current DA literature mainly report results where the performance on a target domain is improved through the application of DA. However, sometimes DA can also have a negative effect, i. e. the classification rate on the target domain drops through the application of DA compared to the baseline, where no DA is applied. This effect is called negative transfer [80, 106] and its investigation represents a major focus of this thesis. So far, in literature it has only rarely been addressed [11, 24, 41, 105]. In [11] for example it was shown that it can be caused by different label spaces between the domains, i. e. $\mathcal{Y}_s \neq \mathcal{Y}_t$, which, however, describes a case of open-set domain adaptation and was

neglected for this thesis. Other works identify latent domains or underlying multimodal structures within pre-defined domains as a potential cause for negative transfer. Latent domains in a single source domain, studied also in [25, 34, 108], were found in [64] to be one main reason for negative transfer. There, they tackled the negative transfer by aligning each of the identified latent domains of the source domain individually to the target domain. In this work the idea is not to find underlying sub-domains, but more to describe multiple domains data by their superposing factors. Another stated reason is that global domain alignment is generally too inaccurate [14, 40, 54, 71, 107] and can therefore lead to negative transfer. Classes within the globally aligned distributions of the involved domains might for example still be mapped incorrectly to each other. This underlying mismatch is tackled in [73] by using individual domain classifiers for each class instead of a single domain classifier as in [22]. Further in [36] global and local domain alignment is combined in a hierarchical manner.

In context of domain factors, the removal of all of them from the feature representation can cause negative transfer as well. This is for example the case when the domain factors are simultaneously task-informative. In a setting with multiple domains even more domain factors are expected to be removed, potentially leading to an increased negative transfer. In [16, 24] they observed such increased negative transfer with multiple source domains being involved, and traced this back to irrelevant source domains that harm the adaptation process.

3.4 Classifier Habits in Context of Visual Factors

This section contributes with a qualitative analysis of how classification models typically handle domain factors and how this would effect the performance on a target domain, with and without DA. The theoretical analyses are discussed with the help of simplified example domains.

Clarification of Image Segment Notations

For the subsequent discussions and the remainder of this thesis, specific notations in context of the image content description have to be clarified for better understanding. In object classification from camera images, which is the main considered classification task of Chapter 5, 6 and 7, the target is to classify for a certain object visible in the image, while presenting the

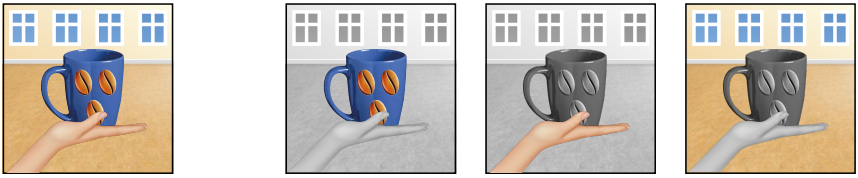


Figure 3.10: Clarification of image segment notations in this thesis. Areas corresponding to the depicted image segments are colored.

image as a whole, meaning without any segmentation applied. The object to classify for, which is ideally focused and centered within the image, will be called target object. On natural images, there are many other objects visually close to the target object that make up the scene. Some of them might have a close interrelation with the target object, those will be called context objects, while others that are not closely interrelated will be called background objects. An example is given in Fig. 3.10.

Domain-dependent Feature Manifestation

In general, during the optimization process CNNs tend to learn the simplest features that allow to solve the classification task. If for example a binary object classification task is described by the discrimination of two objects with differently pronounced colors that always show up in front of a white background, it is very likely that the learned features are solely based on these colors. In the opposite case where no color difference of the objects is represented, e. g. due to only grayscale images or bad lighting conditions, the model is forced to learn more comprehensive shape based features to solve the task. In general, this shows that the values of certain factors especially domain factors, here for example the factor *'lighting'*, can put a strong constraint on the general type of features that are learned during optimization. A visualization of the feature manifestation of two domains with different values for the domain factor *'lighting'* is shown in Fig. 3.11.

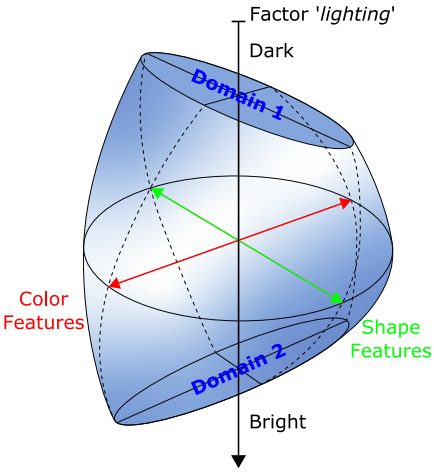


Figure 3.11: Feature manifestation. The type of learned features, here exemplary only color or shape features, highly depends on the value of the domain factor *'lighting'*. With less light the learned features are more biased towards different shapes than colors. Note, the assumption here is that the factor *'lighting'* is a continuous variable.

New Factor Values or new Factor Value Combinations in Test Set

A classification model should generalize well to data that has not been involved during training. In terms of visual factors this consequently means first, that the model is desired to be capable of handling new, unseen factor values of test images that show the same classes, and second that the model should be able to generalize to new combinations of factor values that were already seen during training. An example constellation of training and test set for both cases is shown in Fig. 3.12.

In object classification tasks, new values for factors in the application domain could be found for factors related to the target object, for instance different object types or changing object colors. Furthermore, new values could be observed for factors related to context objects or background object factors, like new surrounding objects or changing lighting conditions. In theory a well generalizing classification model therefore should not embed such factors into its internal class representations that potentially change their value.

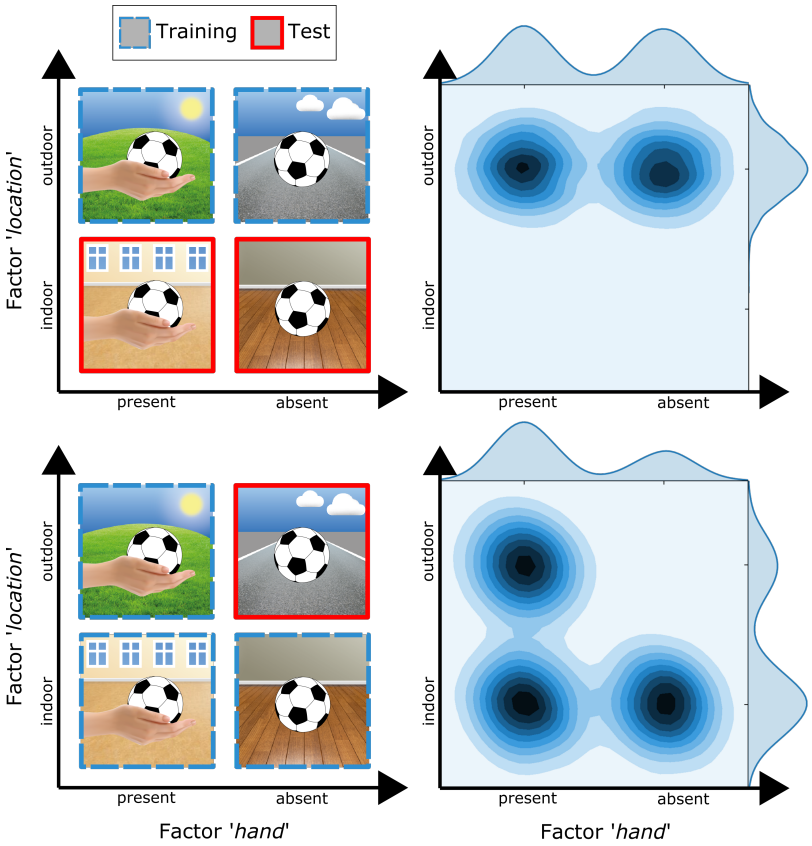


Figure 3.12: Training (blue) and test set (red) constellations depicted by visual factors. *Top:* The test set introduces a new value for the factor *'location'*. *Bottom*: All factor values are already shown during training but a new combination of those shows up during test. In both cases a classification model is expected to be able to generalize to the changes introduced in the test data.

The use of standard machine learning algorithms can not prevent the exploitation of such factors, since the knowledge about the change of the factor values in the application dataset is not provided during training. Unsupervised DA approaches like [22, 100] introduce the new values by showing unsupervised image samples from the target domain already during training. Skillful integration of the samples in the optimization process can

prevent the classification model to put too much focus on the factors that change their value.

The problem that occurs with new combinations of factor values that have already been seen during training is that classification models tend to learn a fixed combination of factor values as a single representative feature. This can be the case if the involved factor values always show up exclusively in certain constellation during training. Therefore, if only one of the factor values is different in the test set, the learned feature detector is not activated anymore and the whole classification chain might fail. If one of the involved factors is a domain factor, DA like [22] might remove them with the help of the unsupervised samples from the target domain already while training, and thus prevents that such factors are embedded in the features.

Task-informative Factor

As stated in Chapter 2, some factors can be classified as task-informative. Considering only a single domain, a domain factor can generally not be task-informative since it is constant within the domain by definition. However, if in a constellation of multiple domains, a domain has a general prior for the appearance of a certain class, a domain factor can be categorized as task-informative. In such a case, a deep classification model will exploit this and uses such a factor as a prominent feature that helps to solve the classification task. An example for this case is depicted in Fig. 3.13, where the occurrence probability for the class 'ball' is higher within domains where the domain factor *'location'* has the value outdoor. With the removal of task-informative domain factors through DA, the network is consequently forced to learn different features. However, such features possibly need to be more complex and can in consequence also reduce the classification performance on the source domains. Other factors that are neither domain factors nor related to the target object, can also be task-informative. This is for example the case for the factor *'hand posture'* that refers with its value to the held object class, or for a factor *'traffic light'* in street scenes, which indicates a higher probability for pedestrians. Without DA such factors are likely to be used to improve classification. If those appear in a potential target domain as well, correct classification is then more likely. Generally the application of DA should not remove those factors, since they are no domain factors. However, there is a chance that such factors are removed as well, if they are related to domain factors. Removing for example the domain factor *'hand'* would consequently also remove features related to

the task-informative factor *'hand posture'*. Due to network architectural reasons regarding the filter size, this might further apply for factors that are visually close to the domain factors. In general, if domain factors close to the target object are removed, then edge features of the target object may also no longer be used. Both depicted cases can ultimately cause negative transfer.

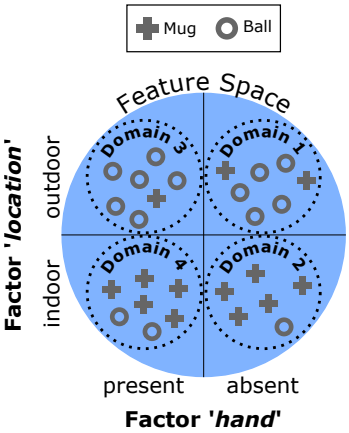


Figure 3.13: Task-informative factor: The domain factor *'location'* is task-informative for the depicted domains. Indoor indicates a higher probability for the mug class, while outdoor a higher probability for the ball class.

Exemplary Domain Constellations and Classifier Habits

The extent to which a factor is relevant to solve a classification task, i. e. its property of being task-informative, in general depends on the given domains and how classes can be discriminated there based on certain factors or their combinations. While in a source domain a factor might be task-informative, it may no longer be so in a target domain. Furthermore, due to network architectural reasons (compare Fig. 3.4) certain non task-informative factors might be inevitably also represented in the learned features for a specific class. An example where a factor changes its property from being task-informative to not task-informative is given by the source domain depicted in Fig. 3.14a and the target domain in Fig. 3.14b. While in the source domain the factor *'coffee beans'* with its values present/absent is task-informative, the additional occurrence of the beans in the background

(a) Source **(b)** Target 1 **(c)** Target 2 **(d)** Target 3 **(e)** Target 4

Figure 3.14: Different constellations of a source domain and different target domains that a classification model could potentially face.

of the ball in the target domain changes this attribute. Without DA, an application of the classification model trained solely on the source domain to the target domain might cause false-positive predictions for the mug class if the learned features rely on the presence of the beans. Since the class-invariant presence of beans can be interpreted as a domain factor when considering source and target domain, DA is likely to inhibit to learn the presence of the beans as a standalone feature. Instead, more complex features that are for example only activated through the combination of the presence and a certain location of the beans could be learned.

Another case is depicted with the second target domain in Fig. 3.14c, where a formerly task-informative factor might change its value. While in the source domain the value for the factor of the ball's global shape can be described as round, in the target domain this value has changed to a different shape due to the occluding hand. If without DA, certain features are based on the ball's global shape, this could lead to false-negative classifications of images showing the ball in the human hand, as those features might not be activated anymore. Using DA in this case could remove also such features, since those help here to discriminate source and target domain. This depicts a nice example where not only domain factors are removed through DA, but also class specific factors that change their value across domains.

As depicted before at the example in Fig. 3.11 with the factor *'lighting'*, domain factors can strongly influence the type of features learned on a single domain. The target domain in Fig. 3.14d, which differs from the source domain only by being based on grayscale images, describes a domain combination where this can cause problems. Training on the source domain

might most likely also include color based task-informative factors, which are absent in the target domain. The application of DA could help here to remove features related to color based factors.

The target domain in Fig. 3.14e depicts the classic case where only the background related domain factor value changes. Here the main reason for a potential performance drop on the target domain without DA is related to filters that overlap with the background (compare Fig. 3.4). In context of convolutional filters, DA could again inhibit that such filters are used as individual features in the higher layers. Assuming the idealized case that the positions of objects are always the same within the image, a fully-connected layer could simply exclude irrelevant image areas by assigning a weight of 0 to neurons that correspond to those.

3.5 Summary

The first part of this chapter showed how the characteristics of current research datasets for image classification can be described in more detail by visual factors. It was shown that the control over factors that is applied during collection or acquisition is a major aspect that differentiates datasets. The real-world datasets with most control over factors are usually found in the robotics community. There, the values of only few factors are changed purposely between recording sessions, while most of the factors are kept constant within a recording session. This characteristic makes such recording sessions highly suitable to treat them as individual domains for DA research.

A short recap on CNN architectures showed how their standard working principles can lead to a poor generalization between domains. This was followed by an overview of current DA approaches, which can generally be divided into two main categories, the approaches with a parametric adaptation between domains and the ones with an adaptation that is learned during optimization of the classification model. Parametric adaptation is comparatively easy to use, but has the disadvantage that the difference between domains needs to be easily modelable by hand. Most DA approaches based on a learned normalization between domains are capable to handle more complex differences between domains. A common approach there is the attempt to remove all domain factors from the feature representation with the help of an attached adversarial domain classifier.

Negative transfer was introduced as a common unwanted drop in performance through the application of DA. This topic also represents a core of

this thesis, where the goal is to investigate in detail the reasons for this effect and reduce it in a subsequent step. It was discussed that the removal of all domain factors from the feature representation, to obtain a domain invariant model, can be harmful if such factors are task-informative or have visual closeness to task-informative factors.

Further details on the feature manifestations within neural networks and their potential difficulties when it comes to generalization to target domains where domain factors can have different values were introduced. It was shown that neural networks tend to solve the classification task with the most simple features as possible, where the features do not necessarily have to be related to the target object in the captured scene. The type of features learned, like for instance shape or color features, may be heavily dependent on a certain factor, like as it was shown for the factor *'lighting'*. The learned features ultimately influence the ability of an architecture to generalize, which was discussed at simple exemplary domain constellations where hypothetical considerations about the usage of DA were made. Moreover, the properties of task-informative factors were discussed.

4 Effects of Domain Awareness

This chapter contributes with an extensive investigation of different domain constellations during training and test of a multi-domain scenario, which have been neglected in literature so far. The topic is highly relevant when considering the increasing number of pre-trained downloadable neural networks that can be used on own data. Special focus will be put in these investigations on the awareness about domains during training and test, which can have direct influence on the treatment of those. The used domains for the investigations mainly differ in the values of the domain factor *'camera'* which are described by different camera types that were used for recording the images. The investigations of the CNN's capability to generalize between domains are based on a road detection task on street scene images. As an exemplary parametric DA method RGB mean normalization will be used here. Nowadays, when using CNNs, RGB mean normalization is a standard pre-processing step, however, there are multiple possible ways on how to carry such a normalization out depending on the awareness about domains. The results show that generally the awareness of the camera type during test can improve the performance for an unseen camera, but can lead to negative transfer when the test camera was already one of multiple cameras during training.

The quantitative numbers presented in this chapter are a result of the previous master's thesis project [83], while the evaluations of these in context of DA and the subsequent publication [84] were part of this PhD project. The main contents of this chapter are therefore based on [84].

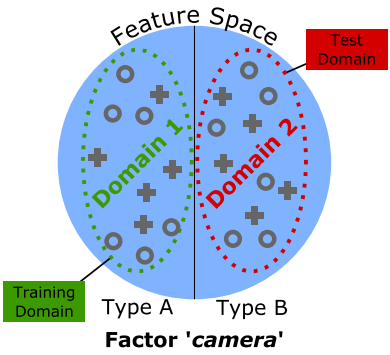
The chapter is structured as follows. In Section 4.1 the different generalization cases on which the effects of domain awareness are investigated will be introduced. Furthermore, the exemplary road segmentation task that is used for the investigations will be presented there. Section 4.2 gives a brief overview of related work. In Section 4.3 the domains defined by three different cameras, as well as the detailed CNN architecture that is used for the segmentation task will be introduced. Subsequently, in Section 4.4 the experiments that were carried out to demonstrate the effects of domain awareness will be presented. The chapter will close with a brief summary of the results and research directions for future work in Section 4.5.

4.1 Problem Statement and Classification Task

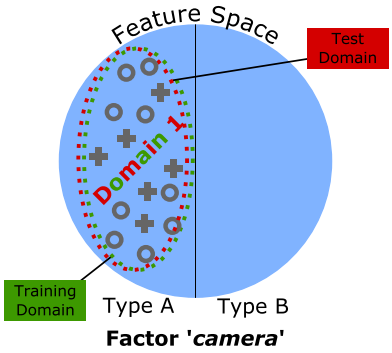
Nowadays CNNs have become state-of-the-art tools for image classification tasks. Networks that have been pre-trained on large-scale object recognition benchmarks [48, 91] are easily accessible and an application on own data can be achieved with only little effort. Downloading such a pre-trained network from the internet can save in certain cases up to several days of training time. If the classification tasks are the same, one common adaptation to own data that is applied is the RGB mean normalization of the input images, which can be seen as a parametric DA method as stated in Chapter 3. However, even with this simple DA method, in certain cases the performance on own data, i. e. potentially different domains, can be significantly reduced for various reasons. First, as stated earlier, the bias that distinguishes the datasets might not be easily modelable and requires a learned normalization, or second, that the bias is modeled incorrectly or inconsistently. In this chapter the latter will be considered and it will be investigated in which generalization cases combined with different normalization strategies such a situation can occur.

Three different generalization cases will be investigated. The first case describes the classic investigation of DA approaches, where the transfer from one or multiple source domains to a new target domain is investigated only [53, 101]. This case will be called the *new*-case. It is visualized in Fig. 4.1a at the example of two domains, where the value of the domain factor *'camera'* is assumed to be defined by a certain camera type that was used for acquisition and influences the appearance of the images. This case is the most challenging generalization case that will be investigated here, since the classification model needs to handle a new value for the factor *'camera'* in the test data. A less challenging case is the *same*-case (see Fig. 4.1b). This describes the typical case from machine learning literature [96, 98], where a dataset is split up into a training and a test set, i. e. when training and test data stem from the same domain. Therefore, a trained CNN is expected to perform well here. A case that is mostly overlooked and only rarely investigated in literature (e. g. [43]) is the *part*-case (see Fig. 4.1c), where the test domain is one of multiple domains that were already involved during training. This case is highly relevant considering networks that were trained on image databases that potentially include multiple latent domains and the own test data describes one of them. As a major contribution of this chapter it will be shown how this *part*-case can

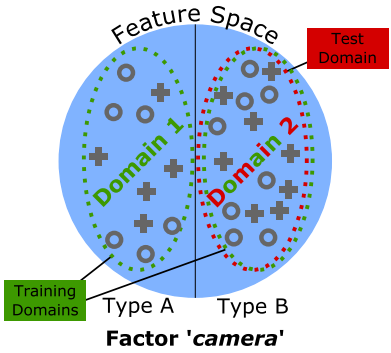
lead to negative transfer.



(a) *new*-case



(b) *same*-case



(c) *part*-case

Figure 4.1: Generalization cases between a training and a test domain. The value of the domain factor *'camera'* is taken as an example to distinguish domains. Note, the terms *source* and *target* domain are not used, since here the definition that no supervised data of the target domain is available does not hold for all cases. In the *same*-case (b) and the *part*-case (c), supervised training data of the test domain is involved during training.

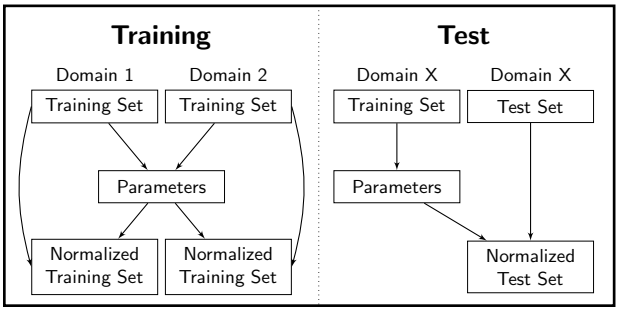
Each of the introduced generalization cases will be investigated for three different domain awareness cases that influence the way how the input normalization parameters are determined during training and test.

Usually the training set is treated as a single domain and the normalization parameters, e. g. the RGB means as here, are determined based on the complete training set. During test these parameters are then newly determined for the test domain. This case of awareness will be called the *test*-aware case (see Fig. 4.2a), since only during test there is an awareness about a new domain. It largely describes the situation when using a pre-trained network from the internet on own data. As the results will show, for the *test*-aware case in the *part*-case of generalization a weak performance could be observed. This indicates an inconsistent normalization treatment of the same domain during training and test. Therefore, two more awareness cases were investigated, where it is guaranteed that domains are treated consistently. Those cases are the *never*-aware case and the *always*-aware case.

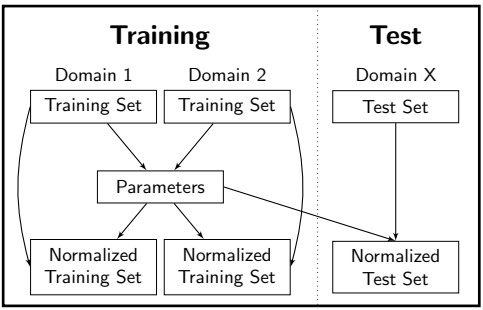
In the *never*-aware case (see Fig. 4.2b) domains are completely neglected, the normalization parameters are only once derived from the entire training set and then reused for the normalization during test on any domain. This case is the most naive and least recommended one since here no adaptation is performed at all, which is likely to result in a poor performance for the *new*-case of generalization.

The *always*-aware case (see Fig. 4.2c) also treats domains consistently during training and test. Here, the test domain is again normalized based on data from this domain. During training, however, domains are normalized individually, leading to the consistent treatment of training and test domains for any case of generalization. This shows an awareness during training and test about domains, leading to the notation of this case. However, often this awareness case is not applicable, since, as stated earlier, large datasets potentially represent a composition of multiple latent domains. Individual treatment of latent domains is generally a challenging task and mostly not considered.

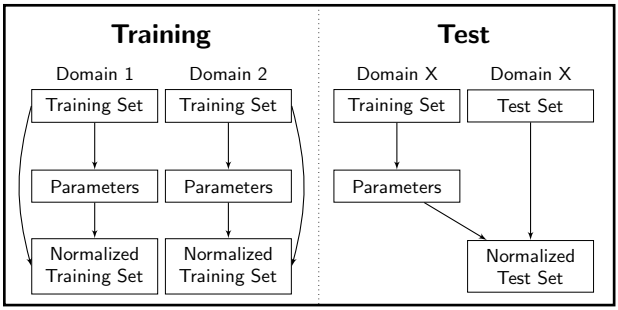
The task on which the awareness cases in combination with the generalization cases are investigated was chosen to be a road segmentation task, where different domains are defined by different camera types that were used for acquisition. For Advanced Driver Assistance Systems this is a highly relevant research topic, since such systems are usually extensively developed using certain types of cameras that might differ to the ones used in production for cost reasons. Different camera types generally might have different values for extrinsic factors like for example the viewing



(a) *Test*-aware



(b) *Never*-aware



(c) *Always*-aware

Figure 4.2: Three different cases of domain awareness.

angle or the focal-length which result in different perspectives on the scene. Furthermore, different mounting positions within the car can influence the field of view. To have a rather controlled domain transfer scenario, with limited other effects for the investigations, here the used cameras mainly differ in their sensitivity of RGB channels, which thus define the dominant bias between domains.

The specific goal of the classification task is to segment a street scene image into *road-like-area* and *non-road-like-area*. Typically, for such tasks complex CNN architectures like [4, 20] are used, that take the whole image as input and provide architecture wise a larger receptive field that is able to take global image semantics into account. The target for the evaluations here, however, is rather to investigate the different domain awareness cases in general, than achieving a highly competitive segmentation result. Therefore, it was decided to base the segmentation on the classification of small image patches combined with a simple CNN architecture (see Section 4.3). On this segmentation task all three generalization cases, the *same*-case, the *part*-case, and the *new*-case with all three cases of domain awareness, the *test*-aware case, the *never*-aware case, and the *always*-aware case are evaluated.

4.2 Related Work

As stated in Chapter 3, many DA datasets like the Office-31 dataset [82] or datasets that are completely based on a web-search as [29], provide a rather uncontrolled setting for in-depth deep learning investigations due to uncontrolled factor values that for example influence the image quality or the types of class representatives. The consequence can be a mixture of effects caused by latent domains that hinder a clear qualitative interpretation of the results. To avoid such effects the choice here was to use three domains composed of typical road scene images that only differ in the value for the domain factor *'camera'*. The values of this factor define the camera type that was used for acquisition. In this setting domains are therefore mainly characterized by the color profile of each camera.

The generalization cases that are investigated here are in literature usually limited to only two of the three introduced cases. Those are the *same*-case, as for example in [96, 98], as the normal training test split, and the *new*-case [53, 101], which describes the standard benchmark case for DA. The *part*-case is only very rarely investigated in recent DA literature. An example where this generalization case was investigated,

but not in context of deep learning architectures, is given in [43]. There they propose an extension for a support-vector machine classifier where an additive bias is learned per source domain. This method outperforms a standard support-vector machine that is not aware of domains during training. These different support-vector machine versions are evaluated on the two awareness cases *never*-aware and *always*-aware. However, for their investigations they also use a mix of web search based benchmarks, whereby latent domains might still persist. Furthermore, in contrast to them the investigations of this thesis also include the relevant *test*-aware case, which a naive user usually does after having downloaded a pre-trained network from an external internet source.

The chosen parametric DA approach of RGB mean normalization is, as stated earlier, one of the easiest and quickest ways to adapt a model to a new domain and therefore is a common pre-processing step used in literature, e.g. [48, 91]. Furthermore, it is generally proven to improve convergence speed during optimization [50]. A methodical similar DA approach, but more complex in application, is the adapted version of batch normalization presented in [53] (see Chapter 3). Despite the popularity of RGB mean normalization, it has not sufficiently been researched regarding the generalization cases with respect to different domain awareness cases. This chapter will close this gap and provide important insights and recommendations on how to handle unknown awareness cases.

There are also more sophisticated normalization techniques that are aware of the meaning and structure of input elements. An example for this is color constancy [77], which was used in [45] to yield invariance to changes in illumination for online learning of color and shape categories. However, this adaptation is done for each image frame individually, in this way lacking the notion of a more stable domain. For the camera DA here, a constant value is subtracted from each RGB channel of all images. This simple parametric DA method is motivated by the strong difference of color sensitivity given by the domains used here, and further, it will prevent that the comparison of different domain awareness strategies is shadowed by a complex method targeting a learned normalization.

4.3 Data and Network Architecture Training

For the evaluation of the effects of domain awareness three different camera type datasets were used. An individual CNN architecture was trained on each single camera, each possible pair and for all cameras together. This

results in overall seven different domain combinations used to train the CNNs. After optimization those architectures were tested independently on each camera test set, covering the three generalization cases, the *same*-case, the *new*-case and the *part*-case. During training and test the three cases of domain awareness (see Fig. 4.2) were applied. Further details on the used data, the architecture, and the optimization procedure are given in the following.

4.3.1 Data

The domains used in the experiments are defined by three image datasets, each recorded by a single different type of RGB camera. Each camera is facing to the front, capturing the street scene in front of the ego-car. The first domain dataset is composed of the KITTI Road Benchmark [19] which is publicly available for download. The other two domains are composed of two street scene image datasets that were acquired by the Honda Research Institute Europe and provided for this work. The cameras that were used for these datasets were the BlackMagic camera[1] (BMAG) and the ELESYS camera that is a special in-house made camera where no external reference is available. Each domain is composed of street scene images that were recorded in different recording sessions where different routes were taken by the car. This consequently also induces changing illuminations and weather conditions. Due to these changing conditions, which lead to many constantly changing factor values within a domain, there is a high likelihood that the camera type is the only constant factor within a domain and thus a domain factor. Details of the three domains are given in Tab. 4.1. There, the channel-wise means over pixels of training patches show a strong difference between the individual domains. These differences justify the treatment of each camera as an individual domain. Moreover, different relations between the channels of one camera can be observed, which justifies the mean normalization of each channel individually.

For each street scene image a manual annotation of one or more *road-like-area* polygons is given. The patches which are extracted from the images and classified by the CNN are labeled as positive samples if their center pixel is part of the annotated *road-like-area*, otherwise it is labeled as a negative sample. The images further come with areas annotated as *exclude-area* that mark areas that can not be clearly defined as *road-like-area*. If a patch has an overlap with such an area, it is excluded from

[1] http://www.blackmagicdesign.com/de/products/blackmagicpocketcinemacamera

Table 4.1: Data statistics of the different camera datasets. The RGB means are based on the samples in each dataset.

		KITTI	BMAG	ELESYS
Image resolution		1242x375	1952x1112	1376x720
Hor. field of view		81.4°	54.9°	64.6°
Red channel mean		96.5	115.3	83.3
Green channel mean		97.0	136.0	89.6
Blue channel mean		92.2	132.0	55.5
Training	#Images	289	113	166
	#Samples	2,312,000	5,085,000	2,822,000
	Pos./neg.	1/1	1/1	1/1
Test	#Images	290	111	166
	#Samples	42,165,099	85,041,497	49,325,077
	Pos./neg.	∼1/1	∼2/1	∼2/1

the training and test data. Moreover, the extraction area is limited to a metric corridor in the so-called Bird's-Eye-View [62]. The determination of this corridor requires the knowledge about the mounting position and angle of each camera, which is given for each dataset. For details on the calculation of this corridor it should be referred to [62]. Here it was limited to 10 meters to the right and to the left of the camera position, and 5 - 45 meters to the front. This ensures negative samples mainly close to the road and the removal of patches above the horizon that mostly contain the sky. Fig. 4.3 shows an exemplary image from the BMAG camera together with its corresponding Bird's-Eye-View transformation of the mentioned corridor.

The images of each camera are split into a training and a test set of similar size. Subsequently patches with a size of 37×37 pixels were extracted from the images, where the size corresponds to the required input size of the chosen CNN architecture. To speed up the training process, only 5% of the available patches were chosen randomly from the training images, while the balance of positive to negative samples was guaranteed. For testing, all negative and positive sample patches inside the Bird's-Eye-View

Figure 4.3: Example street scene image of the BMAG domain. The patches that are extracted from the image in perspective view (left) are limited to the corridor highlighted by colored pixels. The corridor is determined by a limitation of the metric Bird's-Eye-View area (right).

corridor were used. Tab. 4.1 shows the number of used training and test patches from each domain. Between the domains the number of samples are clearly imbalanced. Evaluations showed, however, that the results are unchanged when using more advanced balancing strategies to cope with this imbalance.

4.3.2 Cases of Domain Awareness Implementation

The different awareness cases are all based on a global channel-wise RGB mean normalization. The "parameters" in Fig. 4.2 are here three values, one for each color channel, that are subtracted from all pixels of the corresponding color channel of the input image to obtain the normalized image. Sometimes mean subtraction is done pixel-wise, i. e. without considering the structure of the input elements, and should improve the convergence speed of gradient descent methods. For the experiments here no negative effect on convergence is expected since the small patches are sampled from many slightly different positions in the image, so there is no bias on the presence of a certain pattern inside a patch. This is in contrast to networks trained on image classification data sets like ImageNet, where the target pattern is, due to the photographer bias, usually found in the center of a training sample, resulting in some stronger variation between background and foreground areas. This hypothesis was tested by

Table 4.2: Architecture details for the Effects of Domain Awareness experiments. In all convolutional and fully-connected layers a ReLU activation function was used.

Layer	Kernel Size	Stride	Filters	Parameters	Dimension (HxWxCh)
Input Layer	–	–	–	–	37x37x3
Convolution 1	5x5	1	16	1216	33x33x16
Max-Pool	2x2	1	–	–	32x32x16
Convolution 2	5x5	1	20	8020	28x28x20
Max-Pool	2x2	2	–	–	14x14x20
Convolution 3	5x5	1	20	10020	10x10x20
Max-Pool	2x2	2	–	–	5x5x20
Fully-Conn.	–	–	–	501000	1x1x1000
Output Layer	–	–	–	2002	1x1x2

calculating for each RGB channel the standard deviations over the pixel-wise mean. These are 3.64, 5.44, and 6.75 for the ImageNet LSVRC-2012 competition data, while here, for the used training patches, those are with 1.23, 1.26, and 1.29 substantially lower. The evaluations justify the usage of channel-wise RGB values in this case. Note, in the *test*-aware and in the *always*-aware case the RGB values of the test domain are calculated from the corresponding training data of the same domain to ensure a fair comparison and no bias towards the actual test data.

4.3.3 Network Architecture Training and Evaluation Measure

The task of the CNN model is to assign to each normalized input patch of 37×37 pixels a class label, which can either be *road-like-area* or *non-road-like-area*. The used architecture has three convolutional layers with a filter size of 5×5 and a varying number of channels per layer. Each convolutional layer is followed by a max-pooling layer. After the last convolutional layer, a fully-connected layer with 1000 neurons and a subsequent output layer of two neurons with a softmax function follow. Padding of the feature maps was omitted to be able to use the architecture in a fully-convolutional

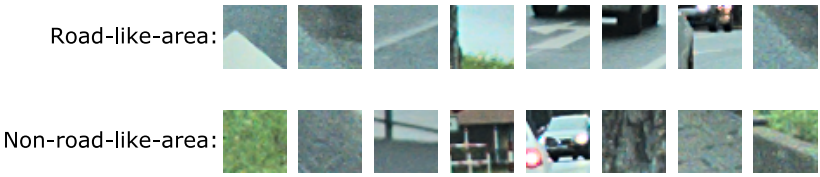


Figure 4.4: Exemplary patches from the BMAG domain that need to be classified by the CNN model. The center pixel of each patch of 37×37 pixels defines the ground-truth label of the sample.

manner [57] during testing, which speeds up the evaluation process since the patch extraction part can be skipped. Since the avoidance of padding results in a stronger spatial reduction of the networks feature map size with increasing network depth, the step-size of the first convolutional was set to only 1, instead of 2 as in the other layers. The idea is to preserve sufficient spatial information after the last convolutional layer in combination with a rather small input image size. An overview of the used architecture is shown in Tab. 4.2, some example patches are given in Fig. 4.4.

The CNN architecture was implemented using the deep learning framework Caffe [39]. As a loss function the cross-entropy classification loss

$$L_{ce} = -\sum_{j}^{C} y_j log(y'_j),$$

was chosen, with the index of summation j, that takes the value of all classes C, here only two classes. The network parameters were optimized with the mini-batch gradient descent algorithm with a batch-size of 50. The number of iterations was fixed to a value so that each sample of the training data was presented on average 20 times to the network. The initial learning rate μ_0 was set to 0.001 and multiplied by 0.5 after every ten percent of the maximum number of iterations. Momentum was applied with a factor of 0.9, weight decay with a factor of 0.0005. Furthermore, dropout regularization with 0.5 was applied on the first fully-connected layer during training, which means that the output value of each neuron there is set to 0 with a probability of 0.5. All hyperparameters were determined based on a manual hyperparameter search.

If the output class *road-like-area* is interpreted as the positive class and *non-road-like-area* as the negative class, the predicted classes of the samples can be categorized as belonging to one of the four categories, true-positive

(TP), false-positive (FP), true-negative (TN), and false-negative (FN). Given a test set where all samples were assigned to a certain class and the ground-truth class is given, the evaluation measures of Precision and Recall [6] can be calculated as

$$\text{Precision} = \frac{TP}{TP + FP},$$

$$\text{Recall} = \frac{TP}{TP + FN}.$$

Precision indicates the proportion of how many of the samples classified as positive, here as *road-like-area*, are actually positive. Recall indicates the proportion of how many samples of the positive class have been classified correctly. Both measures then can be combined in a Precision-Recall curve, where for each activation threshold a specific precision and recall value are depicted. The two measures are typically used in road segmentation task benchmarks [19] in form of the measure of Average Precision (AP) [17]

$$\text{AP} = \frac{1}{11} \sum_{r \in 0, 0.1, ..1} \max_{\widetilde{r}:\widetilde{r} > r} \text{Precision}(\widetilde{r}), \tag{4.1}$$

where r define the recall values. It can be interpreted as the area under the curve of a precision-recall plot approximated by the sections defined by the value of r. The advantage of AP is that it is a measure independent of a specific decision threshold. The main evaluation measure for the experiments in this chapter therefore will be the AP presented in (4.1) with 11 evaluation sections.

4.4 Experiments

For the evaluation of the different awareness cases the value of the output neuron corresponding to the *road-like-area* prediction is interpreted as a confidence score on which the different thresholds are applied to determine the value of the AP as depicted in (4.1). An exemplary qualitative result of a test image that was pixel-wise classified into *road-like-area* and *non-road-like-area* is given in Fig. 4.5. It is an example of the *same*-case of generalization, where the threshold was adjusted to maximize the Quality measure over the entire test set. The details of the Quality measure are of no further relevance for the evaluations here and therefore it should

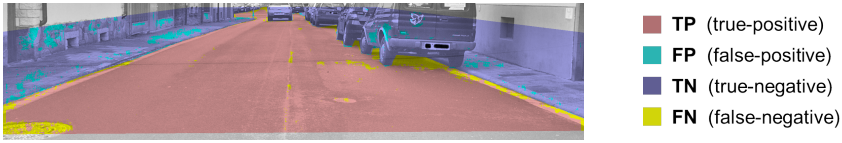


Figure 4.5: Classification of a test image into *road-like-area* (positive class) and *non-road-like-area* (negative class). The area of classification is limited to the Bird's-Eye-View corridor.

be referred to [66]. The qualitative results show that, from a theoretical perspective, the consideration of information from a larger receptive field during training, i. e. using larger input patches, could help to avoid certain false classifications. This applies for example on single yellow pixel areas in the image that are completely surrounded by *road-like-area* predictions. However, the focus here is not to maximize the segmentation accuracy, but rather to investigate the effects of different domain awarenesses in context of the introduced generalization cases.

Note that the AP numbers in the KITTI Road Benchmark [19] are reported for Bird's-Eye-View space, while here the evaluation is conducted in the perspective image space (Fig. 4.3, left).

Tab. 4.3 shows the results in AP for the different experiments. The top rows show the individual domain constellations, while the last three rows show the averaged AP for the three generalization cases. Each result column represents a specific domain awareness case.

***Test*-aware case.** In the *test*-aware case (see Fig. 4.2a) the highest performance can be observed when the training set consists of only one domain and the test data is of the exact same domain, i.e. in the *same*-case (blue). This is expected, since here the classification model does not have to generalize to new values of domain factors. Furthermore, the domain is normalized in the *test*-aware case by the same parameters during training and test which guarantees a consistent treatment of samples from the same domain. For the *part*-case a performance drop to 89.96 AP on average can be observed. In terms of the values for the domain factor *'camera'* this is at first sight surprising, since here no new factor values are presented during test to the classification model. The test domain was already involved during training. The results indicate, however, that this is related to the inconsistent treatment of domains during training and test. During training the normalization parameters are determined over the entire training set,

Table 4.3: Results of different domain awareness cases for different cases of generalization (AP in %). The three cases of generalization are marked by different colors. Blue refers to the *same*-case, gray to the *part*-case and green to the *new*-case. Gray numbers indicate unchanged results in comparison to previous cases of domain awareness. The average results over domain (camera) combinations are shown at the bottom.

Used cameras		**Domain awareness**		
Training	**Test**	*Test*	*Never*	*Always*
KITTI	KITTI	93.94	93.94	93.94
KITTI + BMAG	KITTI	89.66	93.76	93.64
KITTI + ELESYS	KITTI	90.12	94.04	93.93
KITTI + BMAG + ELESYS	KITTI	88.03	93.72	93.82
BMAG	KITTI	85.99	58.57	85.99
ELESYS	KITTI	90.10	86.52	90.10
BMAG + ELESYS	KITTI	78.55	59.99	90.04
BMAG	BMAG	96.29	96.29	96.29
BMAG + KITTI	BMAG	95.65	96.27	96.25
BMAG + ELESYS	BMAG	93.12	96.31	96.24
BMAG + KITTI + ELESYS	BMAG	94.80	96.21	96.28
KITTI	BMAG	93.28	73.68	93.28
ELESYS	BMAG	93.70	89.88	93.70
KITTI + ELESYS	BMAG	80.65	80.08	93.46
ELESYS	ELESYS	95.97	95.97	95.97
ELESYS + KITTI	ELESYS	91.33	96.02	95.97
ELESYS + BMAG	ELESYS	86.68	95.77	95.91
ELESYS + KITTI + BMAG	ELESYS	80.25	95.80	95.88
KITTI	ELESYS	86.67	77.12	86.67
BMAG	ELESYS	84.11	74.35	84.11
KITTI + BMAG	ELESYS	80.66	75.46	80.02
Averages	*Same*	95.40	95.40	95.40
	Part	89.96	95.32	95.32
	New	85.97	75.07	88.60

i. e. by neglecting individual domains within. During test, however, the evaluated domain is normalized by normalization parameters that are determined exclusively from this domain. The assumption is that the CNN overfits too much on the combination of normalization parameters and images from a specific camera and therefore can hardly handle images from the same camera with a different normalization applied. As expected, due to the completely new factor values in the *new*-case the performance is with 85.97 AP on average the worst among all three generalization cases. Note, this drop might not only be related to the RGB sensitivities of the cameras, but also to certain aspects like the image noise or the resolution that change with the camera type that was used.

***Never*-aware case.** In contrast to the *test*-aware case, in the *never*-aware case (see Fig. 4.2b) only one set of normalization parameters is determined during training that is later also used for the test set, independent of the actual test domain. The AP for the *same*-case of generalization is not affected by this since here the same parameters are used. The *new*-case, however, is strongly affected by this way of normalization. The AP here is at only 75.07. Such a result was to be expected since no adaptation at all is made to the new data, which comes with a new domain factor value. Analyzing this case in more detail revealed that many of the predicted confidences on the test domain for *road-like-area* were either very high or significantly low. This is a indicator that the trained networks are driven outside of their working range when a domain is presented that is defined by completely different input ranges, or camera characteristics here. One of the major results in the *never*-aware case is the enhanced performance for the *part*-case of generalization. Although no adaptation to the new domain is made here at all, the performance is better than in the *test*-aware case. This result emphasizes the high importance that domains should be treated consistently during training and test.

***Always*-aware case.** The way of normalization in the *always*-aware case (see Fig. 4.2c) is defined by an independent normalization of all individual domains involved during training and test. This removes the inconsistent treatment in the *part*-case of generalization, since now also during training domains are treated independently. In this case of generalization, the same high performance as in the *never*-aware case can be observed, which confirms the importance of consistent domain treatment. Noteworthy for this case of domain awareness is the individual AP on the KITTI domain

and the BMAG domain in the *new*-case (see last row of first and second result section in Tab. 4.3). Compared to the other two cases of domain awareness the performance has drastically improved, to 90.04 AP and 93.46 AP respectively. A reason for these significant improvements could be that the classification model is now forced to learn more general features for *road-like-area* during training, since the input samples of multiple domains are now more similar due to a better zero-centering of those.

4.5 Summary

As one of the contributions of this thesis, in this chapter the effects of domain awareness were investigated at the example of a road segmentation task. For this a controlled setting with three domains was used. The domain factor here was *'camera'*, whose values are defined by different camera types that were used during acquisition of the street scene images of each domain. The camera types introduce different characteristic RGB means on the recorded images and represent the prominent bias that differentiates the domains. Since this bias is easily modelable, RGB mean normalization was used as a simple parametric DA method that targets to remove this bias.

It was shown that the standard workflow of adapting the normalization parameters to the test domain can lead to negative transfer if the test domain was already one of multiple domains during training and those domains were not normalized individually there. However, when using pre-trained neural networks, e. g. downloadable ones from the internet, it is often not evident if the targeted application domain was part of the original training set and if so, how it was normalized during optimization of the provided network.

The findings from the experiments are highly relevant for the DA community and the general usage of pre-trained neural networks. From the experiments it can be derived that in practice, when handling pre-trained networks for own purposes, it is very beneficial to find out if the test domain was already part of the training data on which the network was optimized. If it was part of the training data, it is further important to know, how it was treated during optimization to adapt the normalization method accordingly to achieve the maximum possible performance.

In general, the use of the controlled investigation setting together with a simple DA method strongly helped to reveal the effects of the awareness about involved domains. Future experiments could investigate less con-

trolled settings or different adaptation methods, like for example batch normalization as an alternative parametric DA method [53], to see whether the awareness about domains influences the results similarly.

5 Setup and Baseline for Adversarial DA Experiments

After parametric DA now the focus will be on adversarial DA. The evaluations carried out in Chapter 6 and 7 are both based on the established unsupervised adversarial DA approach from Ganin et al. [22] that allows to learn a normalization between domains. The basic principle of this architecture was already introduced in context of Fig. 3.6. Here, in Section 5.1 more details on the used architectures, the training procedure, and relevant hyperparameters will be given. Furthermore, Section 5.2 introduces the datasets, MNIST with MNIST-M, and CORe50, on which the main results of the remaining chapters are based. For the latter, baseline experiments relevant for the upcoming chapters will be presented in Section 5.3. Most parts of this chapter are based on the results that were published in the conference paper [85] and in the journal publication [86].

5.1 Implemented Domain Adaptation Architectures

The base adversarial DA approach that was used for a learned normalization between domains was taken from Ganin et al. [22] (see Fig. 3.6). The target of this approach is to remove all domain factors from the feature representation embedded in the parameters θ_e of a shared feature extractor with the mapping function

$$\mathbf{x}_i = G_e(\mathbf{b}_i; \theta_e).$$

The removal is achieved through an additional domain classifier with the parameters θ_d, with the mapping function

$$\mathbf{d}_i{}' = G_d(\mathbf{x}_i; \theta_d),$$

that is positioned in parallel to the label classifier with the parameters θ_y and the mapping function

$$\mathbf{y}_i{}' = G_y(\mathbf{x}_i; \theta_y).$$

The domain classifier is attached to the shared feature extractor via a gradient reversal layer (GRL), which does not influence the forward path of an input sample, but flips the gradient during backpropagation. Like this, the embedding of domain-informative features should be inhibited. The parameters of the entire classification model are updated during backpropagation in the following way for each batch:

$$\theta_e \leftarrow \theta_e - \mu \left(\frac{\partial L_y}{\partial \theta_e} - \lambda \frac{\partial L_d}{\partial \theta_e} \right) \tag{5.1}$$

$$\theta_y \leftarrow \theta_y - \mu \frac{\partial L_y}{\partial \theta_y} \tag{5.2}$$

$$\theta_d \leftarrow \theta_d - \mu \frac{\partial L_d}{\partial \theta_d} \tag{5.3}$$

Here, L is the corresponding loss function, μ the learning-rate and λ an adaptation factor that controls the influence of the domain classifier on the shared feature extractor. λ is smoothly increased during the optimization process, which was evaluated in [22] to lead to more stable optimization. In the original implementation of [22] the domain classifier was used to only discriminate a single source and target domain, nevertheless, it can also be used with multiple source domains, which eventually allows to use it without any unsupervised target data.

With a single supervised source domain and a single unsupervised target domain, the mini-batches used for training are composed of one half randomly chosen supervised samples from the source domain and one half randomly chosen unsupervised samples from the target domain. For the samples in the first half of the batch the class label vector $\mathbf{y}_i$ as well as the domain label vector $\mathbf{d}_i$ are known, while for the second half only $\mathbf{d}_i$ is known. The entire batch is forwarded to train the domain classifier, while only the first half of the batch is forwarded to train the label classifier. The batch-size during training was set to 64 samples for all experiments.

For $\lambda = 1$, the gradients backpropagated from both, the label and the domain classifier, influence the shared feature extractor equally. During DA training of the reported experiments, the adaptation factor λ is increased

smoothly from 0 to 1, following the update rule proposed by [22],

$$\lambda_p = \frac{2}{1 + e^{(-10 \cdot p)}} - 1, \tag{5.4}$$

where p corresponds to the proportion of iterations already completed over the number of maximum iterations. The optimized update rule from [22] for the decrease of the learning rate μ was likewise incorporated in the optimization process of the models. The update rule for μ is

$$\mu_p = \frac{\mu_0}{(1 + 10 \cdot p)^{0.75}}\,, \tag{5.5}$$

where the initial learning-rate μ_0 was chosen depending on the architecture. For pre-trained architectures a comparatively smaller learning-rate was used. For both, the label and the domain classifier, the cross-entropy loss function was chosen, which is commonly used for classification problems with discrete target output values. For the label classifier it is calculated for a single sample as:

$$L_{ce} = -\sum_{j}^{C} y_j log(y'_j)$$

where the index of summation j runs over all C classes. For the domain classifier y is replaced by d. Before the loss is determined, the outputs of the classification model are passed through a softmax function, which transforms the outputs of the neurons to a range between 0 and 1, while all sum up to 1. The transformation of a single neuron i is achieved by

$$y'_i = \frac{e^{h_i}}{\sum_j^C e^{h_j}},$$

where h_i is the activation of the i-th output neuron corresponding to the i-th of C classes. The implementation of the architectures in Python was done with the deep learning framework TensorFlow [1]. For the optimization of the internal parameters the Adam optimizer [44], pre-implemented in TensorFlow, was used.

Adversarial DA architecture for small input images. For the investigations in the upcoming chapters two different network architectures were used. The feature extractor and the label classifier of the first archi-

tecture are similar to [22] inspired by the popular LeNet-5 architecture from [49]. This architecture was designed to solve handwritten digits classification tasks based on low resolution input images with a size of 32×32 pixels. The architecture comes with only two convolutional layers and two fully-connected layers. Similar to [22], a domain classifier was attached to the feature extractor via a GRL layer. It consists of a single fully-connected layer and an output layer with two neurons. Rectified linear unit activation functions (ReLU) [68] were used on all layers. Further details about this architecture can be found in Tab. 5.1. All weights of this architecture were initialized randomly by drawing them from a uniform distribution.

Table 5.1: Adversarial DA architecture for small input images inspired by LeNet-5 [49] and [22].

Layer	Kernel Size	Stride	Filters	Parameters	Dimension (HxWxCh)
Feature Extractor					
Input Layer	–	–	–	–	32x32x3
Convolution 1	5x5	1	32	2432	28x28x32
Max-Pool	2x2	2	–	–	14x14x32
Convolution 2	5x5	1	32	25632	10x10x32
Max-Pool	2x2	2	–	–	5x5x32
Label Classifier					
Fully-Conn. 1	–	–	–	80100	1x1x100
Fully-Conn. 2	–	–	–	10100	1x1x100
Output Layer	–	–	–	1010	1x1x10
Domain Classifier					
GRL Layer	–	–	–	–	5x5x32
Fully-Conn. 1	–	–	–	80100	1x1x100
Output Layer	–	–	–	202	1x1x2

Adversarial DA architecture for large input images. Since the previously presented architecture can only be used for datasets with artificial images of low resolution, another architecture was implemented that is capable to solve real-world image classification problems with images

of higher resolution that ultimately carry more information. For the adversarial DA architecture, the VGG-16 architecture from [91] was used, of which the 14 convolutional layers were interpreted as the feature extractor and the two subsequent fully-connected layers as the label classifier. The convolutional layers used zero padding at the edges of the input feature maps to preserve the feature map dimensions. Thus, the feature map size in this architecture is only reduced through max-pooling layers. In the label classifier, the original number of neurons was reduced to 4000 in the first and 1000 in the second fully-connected layer to speed up convergence. Several pre-studies revealed that this has only a minor influence on accuracy. The additionally attached domain classifier has been extended compared to the previously introduced architecture (Tab. 5.1) and is here composed of two fully-connected layers with 1024 neurons each. Also for this architecture the ReLU activation function was used on all layers. Since the training of all weights would require huge amounts of training data and ultimately lead to long training times, it was decided to use a set of pre-trained weights for the shared feature extractor. The weights are based on a pre-training on the ImageNet database [13]. Details on the architecture can be found in Tab. 5.2.

5.2 Datasets

MNIST & MNIST-M

The Domain Mixture investigations in Chapter 6 are primarily based on the usage of two domains defined by the standard MNIST dataset [49] and the adapted MNIST-M dataset [22]. Throughout the remainder of this thesis the first will be referred to as S and the latter as M. Both domains consist of images showing handwritten digits from 0 to 9. Sample images are shown in Fig. 5.1. M was originally generated by [22] through random extraction of patches from the BSDS500 image dataset [3] with the same size as images from the S dataset. On these patches, the pixels corresponding to the location of the digit pixels in the S samples were inverted to obtain the final samples. In terms of visual factors the two domains can thus mainly be described by the domain factor *'background'*, which can be interpreted to have the value black in S and cluttered in M. Generally, for the classification task, the M domain is due to its nature much more challenging since it can additionally also have digits with dark pixels on a bright background and furthermore with hard edges within

Table 5.2: VGG-16 architecture with attached domain classifier.

Layer	Kernel Size	Stride	Filters	Parameters	Dimension (HxWxCh)
Feature Extractor					
Input Layer	–	–	–	–	224x224x3
Conv 1_1	3x3	1	64	1792	224x224x64
Conv 1_2	3x3	1	64	36928	224x224x64
Max-Pool	2x2	2	–	–	112x112x64
Conv 2_1	3x3	1	128	73856	112x112x128
Conv 2_2	3x3	1	128	147584	112x112x128
Max-Pool	2x2	2	–	–	56x56x128
Conv 3_1	3x3	1	256	295168	56x56x256
Conv 3_2	3x3	1	256	590080	56x56x256
Conv 3_3	3x3	1	256	590080	56x56x256
Max-Pool	2x2	2	–	–	28x28x256
Conv 4_1	3x3	1	512	1180160	28x28x512
Conv 4_2	3x3	1	512	2359808	28x28x512
Conv 4_3	3x3	1	512	2359808	28x28x512
Max-Pool	2x2	2	–	–	14x14x512
Conv 5_1	3x3	1	512	2359808	14x14x512
Conv 5_2	3x3	1	512	2359808	14x14x512
Conv 5_3	3x3	1	512	2359808	14x14x512
Max-Pool	2x2	2	–	–	7x7x512
Label Classifier					
Fully-Conn. 1	–	–	–	100356000	1x1x4000
Fully-Conn. 2	–	–	–	4001000	1x1x1000
Output Layer	–	–	–	1001*Classes	1x1xClasses
Domain Classifier					
GRL Layer	–	–	–	–	7x7x512
Fully-Conn. 1	–	–	–	25691136	1x1x1024
Fully-Conn. 2	–	–	–	1049600	1x1x1024
Output Layer	–	–	–	1025*Domains	1x1xDomains

Figure 5.1: Domains used to evaluate the Domain Mixture scenarios in Chapter 6: Standard MNIST (S) and MNIST-M (M).

the written digits (see digit 5 and 9 in Fig. 5.1). This generally requires more complex features to discriminate the classes. The randomly extracted patches for the generation of M might further have the effect, that S might already be included in great parts in M, which is given when randomly complete black patches are extracted.

Both datasets come with a pre-defined split into a training set of 60,000 samples and a test set of 10,000 samples. The separation of training and test is kept throughout all experiments with these datasets. All samples have a size of 28×28 pixels. For the training of the DA architecture for small input images (see Tab. 5.1), the samples were scaled up to 32×32 pixels. The label space for all classification tasks is defined as $\mathcal{Y} = \{0, 1, ..., 9\}$, where the label of an individual sample is defined by the handwritten digit shown on the image.

CORe50

The CORe50 dataset that was introduced by Lomonaco et al. in [56] is the second main dataset of this thesis. It will be used for the evaluations of the Domain Mixture scenarios (Chapter 6) and the evaluations of the new Factor-Preserving DA approach (Chapter 7). The dataset consists of images of objects from 10 distinct categories with 5 objects each. All objects were recorded in 11 different recording sessions that are mainly defined by different locations where a consistent background clutter can be observed. All objects were held and rotated in a human hand during acquisition. For each object there are 300 images per session, which results in a total of 15,000 images per session and 165,000 images overall. Some example images from all recording sessions are shown in Fig. 5.2.

The CORe50 is mainly used as a benchmark dataset for continuous learning approaches [63, 72], where the challenge is mostly to adapt a

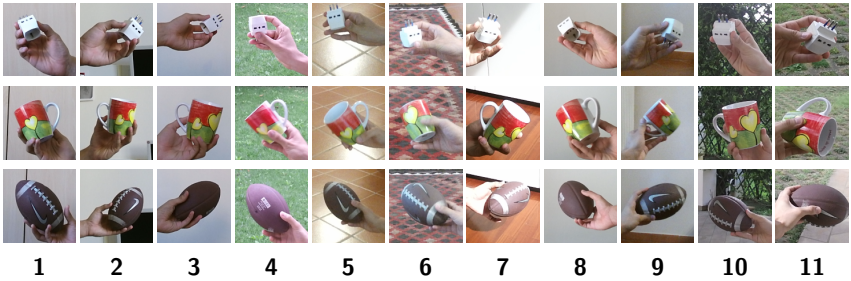


Figure 5.2: Exemplary images for three of 50 objects that were recorded in the 11 different recording sessions of the CORe50 dataset. The background clutter characteristic for each session is representative for all 50 objects. In each session a single hand, i. e. right or left, was used exclusively for presentation. Each session is used as an individual domain in this thesis.

classification model to newly emerging class representatives that occur over time, without loosing the ability to classify samples from previous time steps. However, because of the dataset's controlled and clean recording setting it will be used in this thesis for the evaluation of DA approaches, where each recording session is interpreted as an individual domain. As of today, this research project is the first to use the CORe50 dataset in the context of DA. The chosen classification task here is the discrimination of the 10 object categories represented in the dataset.

In context of the introduced factor theory this dataset corresponds mostly to the photographer view, where often a special object is placed in natural environments. The domain factors here are mostly related to the location where the images were captured. All objects are exactly the same in each domain and therefore all factors directly related to the objects can initially be assumed to have the same value across domains. During acquisition of the domains one hand was used exclusively to present the object, i. e. either the right or the left hand, which makes it an additional domain factor. Furthermore, it was observed that individual objects are held and moved differently across domains, however it cannot be judged if there is a consistent presentation style inside a domain and thus a domain factor.

Due to the originally intended use of the dataset for continuous learning approaches it does not come with a pre-defined separation into training and test data. However, since this split is necessary for the experiments in this thesis, this was done manually by dividing the 300 images per object and session into chunks of 20 images and using the odd chunks as training

data and the even chunks as test data. Note, when evaluating a transfer without DA between source domain A and target domain B, all images of the domain A are used, but the performance is evaluated only on the half of the images of domain B. This is done for reasons of comparability since with DA the other half of B is always used as unsupervised data.

The neural DA architecture that was used in context of the CORe50 dataset was the VGG-16 architecture for large images (see Tab. 5.2). For all experiments the image samples with a size of 128×128 pixels were upscaled to a size of 224×224 pixels to be compatible with the input size of the pre-trained VGG-16 architecture.

5.3 Baseline Experiments on CORe50

In Chapter 6 the approach of Ganin et al. [22] will be used in a special domain setting compared to its original version, while in Chapter 7 the novel Factor-Preserving DA approach will be based on it. Therefore, here some baseline experiments with [22] on the CORe50 domains will presented. The idea is to get a first impression of the general functionality of the standard DA approach from Ganin et al. [22] and further to understand the difficulty of the domains and the transfers of models across them. In a first step multi-source leave-one-out DA experiments will be presented, where a single domain represents the target domain and all others the source domains. After this, single transfers, i. e. one-to-one experiments, between all domains will be investigated, with and without DA. Note, there are much more other constellations that generally can be investigated here, however, this thesis will only consider the leave-one-out and the one-to-one experiments since those are the most meaningful corner cases and also tractable to do. These results have been published in [86].

Leave-one-domain-out Experiments

The first experiments target to investigate how well multiple source domains generalize to a single target domain. The experiments are conducted in a leave-one-out manner without DA and with DA. As stated earlier, with the availability of multiple source domains the DA approach of [22] can also be used without any data of the target domain. This investigation and the investigation where unsupervised data of the target domain is used, will represent two distinct cases of the use of DA here. In the experiments special focus is put on the effects of the removal of the domain factors from the feature embedding of the shared feature extractor, which is the goal of the used adversarial DA approach.

For the experiments here the VGG-16 architecture (see Tab. 5.2) was used, where the number of output neurons was set to 10, corresponding to the number of object categories in CORe50. The CORe50 comes with 11 domains. Therefore, depending on the investigated DA case, the number of output neurons of the domain classifier was set either to 10, when no unsupervised data was used, or to 11, when additionally unsupervised data of the target domain was used. For the experiments without DA the adaptation factor in the GRL layer was set to 0, i. e. $\lambda = 0$. In the experiments with DA it was updated according to Eq. (5.4). The initial learning rate was set to $\mu_0 = 0.0001$ and reduced over iterations

according to Eq. (5.5). To reduce overfitting, dropout [92] was used in both classifiers as a regularizer. The input data was RGB mean normalized following the strategy of the *never*-aware case (see Fig. 4.2b). In the end, the chosen normalization strategy does only introduce a bias on all results, while general effects of the major changes between the experiments are not affected by this. Besides the RGB normalization, no further data modification or augmentation methods were applied. The number of epochs was set to two for all experiments, which turned out to be a good trade-off between convergence and overfitting to the training data.

To get an impression about the trends of the training loss and the training accuracy of the label and the domain classifier with respect to λ and μ during training, an exemplary optimization run is shown in Fig. 5.3 and Fig. 5.4. There, domain 9 was used as the target domain and the remaining ones as the source domains. For completeness the loss and the accuracy of the domain classifier without DA are shown as well, although there was no influence of it on the feature extraction path in this case due to $\lambda = 0$. The plots in Fig. 5.3 show that all accuracy curves follow the desired behavior. Except the domain classifier in case of DA, all curves converge towards 100% training accuracy. When DA is applied, i. e. $\lambda \neq 0$, the classification accuracy of the label classifier can be observed to be significantly less stable. This is obviously due to the changing features it receives as input from the shared feature extractor. The domain classifier in case of DA starts with an increasing accuracy but collapses at some point with a growing λ. This is the expected behavior which shows that it now has difficulties to discriminate the domains based on the output of the shared feature extractor due to the removal of domain factors. The cross-entropy losses in Fig. 5.4 show the corresponding trends for the described characteristics.

For the main evaluation of the multi-source leave-one-domain-out experiments a single result is based on 10 runs with randomly initialized parameters θ_y of the label classifier and parameters θ_d of the domain classifier. The parameters of the feature extractor θ_e were initialized in all runs with the same pre-trained weights. Furthermore, in each run the training data was shuffled randomly, resulting in differently composed mini-batches used during optimization. The reported numbers refer to the average classification accuracy over the 10 runs. Besides the results for each leave-one-domain-out experiment the average and the minimum over all experiments is reported. The minimum performance can be considered an important measure, especially when looking at safety critical systems where it is essential, independent of the specific case, to not fall below a certain performance threshold value that ensures a safe behavior of the

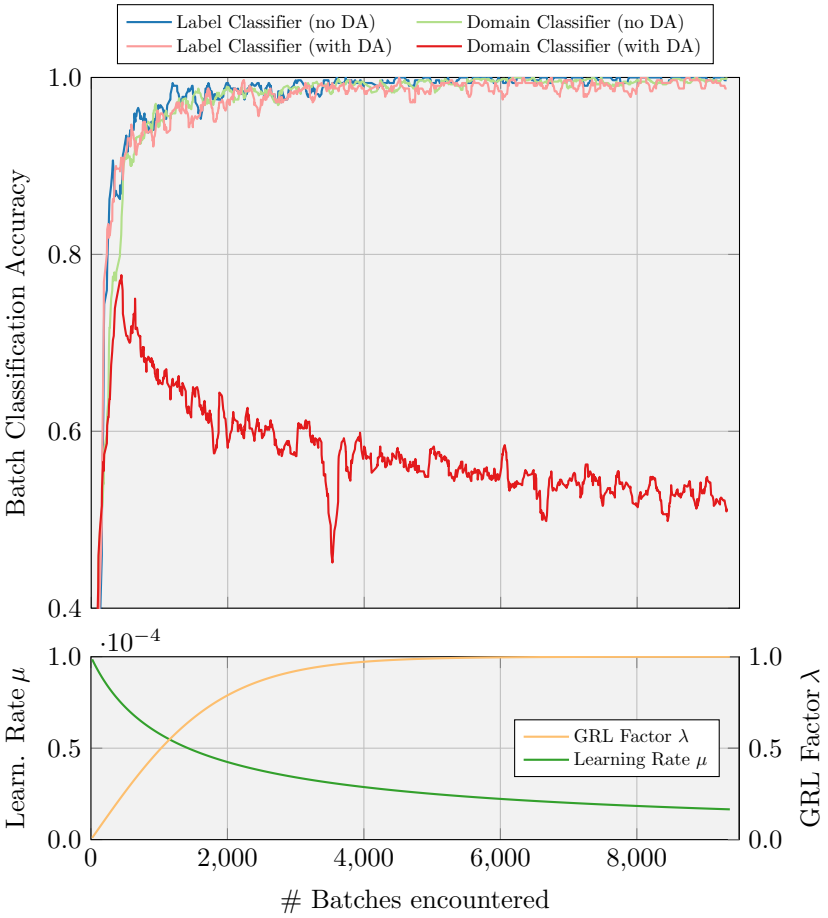


Figure 5.3: *Top*: Accuracy curves for the leave-one-domain-out experiments where domain 9 was the target domain. The plot shows two epochs with a moving average applied to flatten the original curve. *Bottom*: Curve of the learning rate and the GRL adaptation factor λ during training. Without DA, $\lambda = 0$ for the entire optimization process.

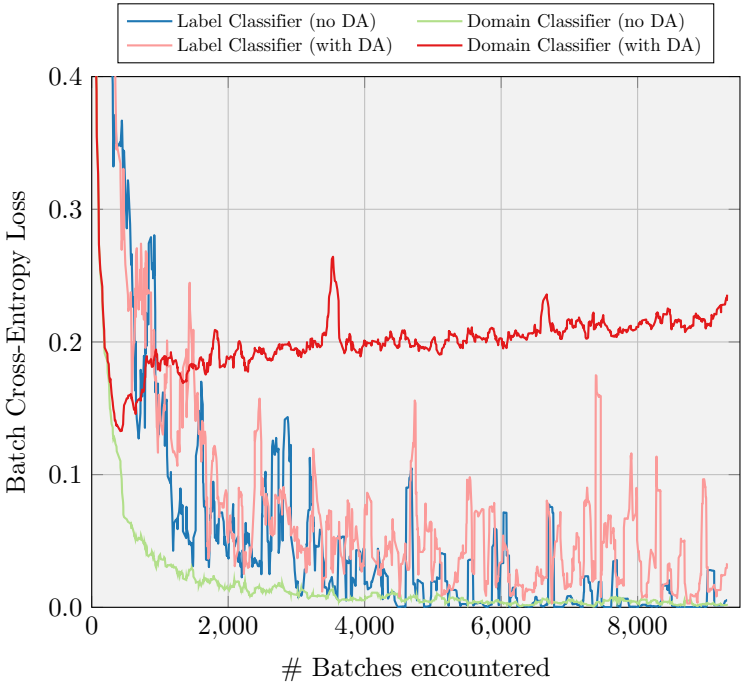


Figure 5.4: Cross-entropy losses during training corresponding to the curves from Fig. 5.3.

system.

The general expectation of the experiments here is that a good performance should be observed even without DA since in the CORe50 dataset there is usually at least one domain that is visually similar to one of the other domains, i. e. most of the relevant factor values should already be covered by the training set. If there are nevertheless new factor values in the target domain, DA with unsupervised data should help to improve the performance in such a case. If there are only new combinations of factor values in the target domain, DA without unsupervised data should already be sufficient to improve the generalization ability of the models.

The results are shown in Fig. 5.5. The general expectation that many domains are very similar and thus the target domain factor values are well covered by other domains is satisfied when looking at the results without DA. There, independent of the target domain, mostly more than 90 %

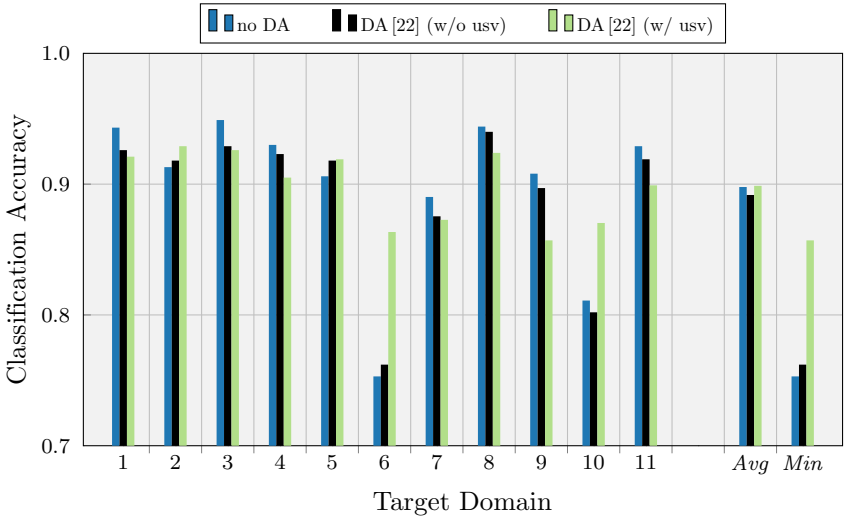


Figure 5.5: Multi-source leave-one-out experiments. The left out domain represents the target domain, while the others are used as the supervised source domains. The experiments were carried out with and without standard DA, while the latter was applied with and without additional unsupervised (usv) samples from the target domain.

classification accuracy can be reached. The only outliers are domain 6 and 10. Analyzing the samples of these domains reveals that those can be expected to be more difficult to transfer to since their background characteristics differ strongly from other domains. Both come with dominant fine-grained edges in the background of the presented object. In domain 6 this is caused by a carpet and in domain 10 by a fence. From theory, considering these appearances as visual factors those can either be new values or new combinations. A clear tendency to one of these cases can not be made only from the experiments without DA.

If DA without unsupervised data is applied for the same source-target constellations, predominantly negative transfer can be observed, i. e. a drop in classification performance occurs compared to no DA. Possible reasons for this drop can be either the additional constraint on the feature extractor that is introduced by the domain classifier or the removal of task-informative factors through DA. Looking at domain 6 and 10 no significant changes in classification accuracy can be observed. This shows that the

main reason for the low accuracy is most likely caused by completely new factor values and not by new combinations of already known values from training.

When DA with unsupervised data is applied, the performance on domain 6 and 10 improves significantly. This confirms the assumption about new factor values in these domains that are now introduced by the unsupervised samples. However, for other target domains, like domain 1, 4, and 8, even a further decrease in performance compared to DA without unsupervised data can be observed, i. e. negative transfer occurs here. A possible reason for this could be the increased number of domains during training, through which possibly even more domain factors are removed, which potentially include task-informative factors as well. Also, compared to the experiments without DA, significant negative transfer can be observed on seven target domains. This drop is most likely also caused by the gradient from the domain classifier and the removal of task-informative factors.

In summary, looking only at the averaged results of the experiments without DA and DA with unsupervised data, it can be seen that those are similar. This is due to the fact that the slight negative transfer on several domains is balanced out with the strong improvement for domain 6 and 10 when unsupervised data is used. When looking at the minimum performance, the clear method of choice would be DA with unsupervised data since this also leads to a high performance for cases where the target domain comes with new factor values. From the leave-one-domain-out experiments, however, it can not be clearly inferred which factors cause the negative transfer, since the factors are based on mutual similarity and difference between individual domains. Consequently one-to-one experiments are required to gain more insights into factors that cause errors when being removed.

One-to-one Transfer Experiments without DA

To get a better impression of the transfers between individual domains of the CORe50 dataset and to identify factors that potentially caused negative transfer in the previous multi-domain experiments, one-to-one experiments were conducted. In a first step this was done without DA between every possible pair of domains. For the implementation this means optimizing the VGG-16 architecture from Tab. 5.2 on each domain with $\lambda = 0$ and testing it on all other domains.

The results for all transfers are given in Tab. 5.3. The table further shows the maximal performance for a given target domain among all source

Table 5.3: One-to-one transfer experiments without DA. The results show the classification accuracy. Good performance is indicated by bright green color, while bad performance by bright red color. Left-hand domains are marked in blue.

Source Domain \ Target Domain	1	2	3	4	5	6	7	8	9	10	11	*Avg*	*Min*
1	-	0.67	0.72	0.74	0.68	0.45	0.64	0.80	0.57	0.48	0.52	0.63	0.45
2	0.69	-	0.80	0.59	0.54	0.55	0.72	0.60	0.76	0.52	0.69	0.65	0.54
3	0.69	0.78	-	0.52	0.51	0.36	0.72	0.60	0.74	0.40	0.66	0.60	0.36
4	0.70	0.46	0.50	-	0.68	0.70	0.51	0.72	0.36	0.68	0.52	0.58	0.36
5	0.67	0.42	0.46	0.68	-	0.62	0.40	0.72	0.37	0.54	0.34	0.52	0.34
6	0.68	0.46	0.44	0.72	0.68	-	0.47	0.67	0.38	0.54	0.49	0.55	0.38
7	0.70	0.75	0.75	0.58	0.52	0.47	-	0.65	0.66	0.47	0.76	0.63	0.47
8	0.83	0.59	0.57	0.76	0.71	0.59	0.57	-	0.48	0.52	0.50	0.61	0.48
9	0.52	0.66	0.74	0.42	0.37	0.30	0.57	0.43	-	0.40	0.66	0.51	0.30
10	0.75	0.69	0.63	0.81	0.75	0.69	0.59	0.74	0.60	-	0.64	0.69	**0.59**
11	0.57	0.66	0.79	0.48	0.42	0.40	0.74	0.53	0.66	0.48	-	0.57	0.40
Avg	0.68	0.61	0.64	0.63	0.59	0.51	0.59	0.64	0.56	0.50	0.58		
Max	**0.83**	0.78	0.80	0.81	0.75	0.70	0.74	0.80	0.76	0.68	0.76		

domains. Comparing the results to those of the corresponding target domains in the leave-one-domain-out experiments in Fig. 5.5, it can be observed that in the single transfer experiments the performance generally is lower. This shows that training on multiple-source domains not only memorizes internally all source domains, but rather builds a more general model that helps to perform better on the target domain.

The overall tendencies of the accuracies are very similar to the multi-source experiments. This can been seen at the example where domain 6 and 10 are the target domains (columns). Reflected in the average accuracy, here the performance is compared to other target domains very low, which was also a major outcome of the multi-source experiments.

However, when domain 10 is the source domain (row), the best average and highest minimum performance can be observed. This indicates that the domain itself covers already a large variety of values for factors related to the background and thus the model is able to handle those for the different target domains. Among the target domains, domain 1 shows both, the highest maximum and the highest average performance. Since

this domain comes with a white and mainly clutterless background, this could be assumed to be the reason since the other domains are more complex and thus already cover this easier domain. However, in Chapter 7 a characteristic of the presenting hand will be identified as contributing mostly to this result.

Looking at the transfer matrix as a whole, reoccurring patterns of weak performance can be observed. Examples for such are the transfers from source domains 4, 5, 6, and 8 to the target domains 2, 3, 7, 9, 11. Analyzing the images of these domains shows that for the mentioned source domains the objects are exclusively presented in the right hand, while for the target domains the left hand was used (see Fig. 5.2). The transfer between these two groups of domains obviously leads to a clear drop in performance. The same can be observed when the mentioned source and the target domains are swapped. In contrast, the transfer of models between domains where the same hands were used works clearly better.

The results showed that the hand might obviously be a prominent factor that is embedded in the feature representation of classes and consequently causes errors for changing hands. Removing the hand from the feature representation through DA could therefore theoretically help to improve the transfer performance.

One-to-one Transfer Experiments with DA

For the application of DA between a single source and a single target domain, following the approach of [22], unsupervised data of the target domain is additionally used. As stated earlier the goal of this DA approach is to remove all domain factors from the feature embedding of the shared feature extractor. In the previous one-to-one transfers without DA it was shown that the hand used for presenting the object is a factor value that changes across certain domains and thus might be the main reason for a reduced performance between domains where a different hand was used. Since the hand is obviously a domain factor in such a transfer constellation, the removal of it through DA could be expected to lead to an improved transfer performance.

The results are given in Tab. 5.4, where the difference in percentage points to the experiments without DA is reported. Here, surprisingly similar patterns can be observed as in the experiments without DA. Obviously the performance on transfers where the hand changes between the domains has decreased even further. This is particularly pronounced for the transfer from domain 3 to 5 with a decrease of 17 percentage points and 7 to 8

Table 5.4: Improvement by DA on one-to-one experiments. The colored cells show the difference in percentage points to Table 5.3. Min and Max refer to absolute accuracies. Left-hand domains are marked in blue.

Target Domain (w/ usv data for training)

Source Domain	1	2	3	4	5	6	7	8	9	10	11	*Avg*	*Min*
1	-	0.08	-0.06	0.04	0.07	0.21	-0.01	0.04	-0.07	0.22	0.09	0.06	**0.49**
2	0.04	-	0.02	-0.02	-0.02	-0.07	0.04	-0.02	-0.04	0.05	0.03	0.00	0.48
3	-0.03	0.02	-	-0.11	-0.17	-0.02	0.02	-0.10	0.01	0.06	0.07	-0.02	0.33
4	0.05	0.00	-0.05	-	0.12	0.06	-0.05	0.07	-0.08	0.05	-0.03	0.01	0.28
5	0.04	0.06	-0.05	0.02	-	0.09	-0.03	0.05	-0.05	0.12	0.00	0.02	0.32
6	0.08	0.06	-0.03	0.10	0.14	-	-0.01	0.11	-0.08	0.17	-0.02	0.05	0.31
7	-0.09	0.04	0.02	-0.09	-0.09	0.02	-	-0.16	-0.01	0.03	0.02	-0.03	0.43
8	0.00	-0.04	-0.08	0.03	0.09	0.08	-0.07	-	-0.08	0.17	-0.04	0.01	0.41
9	0.05	0.10	0.07	-0.08	-0.05	-0.02	0.11	-0.02	-	-0.02	0.11	0.03	0.28
10	0.03	-0.05	-0.13	0.01	0.00	0.01	-0.07	0.01	-0.15	-	-0.05	-0.04	0.45
11	-0.04	0.01	-0.01	0.00	-0.09	-0.03	0.00	-0.11	0.07	-0.02	-	-0.02	0.34
Avg	0.01	0.03	-0.03	-0.01	0.00	0.03	-0.01	-0.01	-0.05	0.08	0.02		
Max	0.83	0.80	0.82	0.83	0.83	0.76	0.75	**0.84**	0.75	0.74	0.78		

with 16 percentage points. The targeted goal to improve the performance through DA can mainly be observed between domains where the hand does not change. Here, the transfers from domain 1 to 10 and 1 to 6 show the biggest improvement with 22 and 21 percentage points respectively. A plausible reason for this, similar to the multi-source experiments, is that the new, unique factor values of domain 6 and 10 are introduced through the unsupervised data during training which ultimately leads to a better classification accuracy. Overall, the average increase for transfers between domains where the same hand was used is at 5.7 percentage points. For changing hands an average decrease of -3.8 percentage points can be observed.

Without DA the highest minimum performance of 59% accuracy could be achieved when domain 10 was the source domain. After the application of DA, the domain in this category is now with only 49% accuracy domain 1, indicating the strong negative transfer introduced through DA.

Contrary to the expectation that the removal of the factor *'hand'* from the feature representation through DA could improve the transfer between domains where different presenting hands were used, the experiments showed the opposite. A possible reason for this could be related to the fact that the hand is not only domain-informative, but can also be task-

informative by giving hints about the object category through its posture when holding the object. Furthermore, a holding hand is visually close to the target object or even overlaps with it. This can also have negative effects since object features overlapping with the hand will not be allowed anymore, as already explained in Chapter 3. In Chapter 7 FP-DA will be proposed to preserve such a valuable factor in the leave-one-domain-out DA setting, reducing the shown negative transfer significantly.

5.4 Summary

This chapter introduced in the first part details on the functionality of the DA approach from [22] and presented the two implemented variants of it that are used as a baseline DA model for the experiments in the subsequent chapters. The two variants differ in their usability for images of different sizes, which require a different number of layers and thus parameters. Furthermore, two relevant DA datasets for the evaluations, the MNIST digit dataset in two variants and the CORe50 object classification dataset, were introduced. Since the latter is relevant as a baseline for both subsequent chapters, baseline experiments were carried out on this dataset here as well. The idea of those is to provide a better understanding of the DA approach and the transferability of models between the pre-defined domains from the CORe50 dataset. The experiments were composed of leave-one-domain-out experiments as well as one-to-one transfer experiments with and without DA. The leave-one-out experiments showed that through the application of DA negative transfer can be observed, which is suspected to be caused by the removal of task-informative factors. The following investigations of one-to-one transfers revealed that generally the transfer between domains with a change in the presenting hand leads to a reduced performance in comparison to similar hands. The assumption that the removal of the domain factor *'hand'* from the feature embedding through DA would lead to a better transferability could not be confirmed in the subsequent experiments, instead an even worse performance was observed. A hypothesis for this observation is that the removal of the hand removes also task-informative factors, like here the hand posture. Furthermore, the visual closeness or overlap of the hand with the target object might have a negative effect due to architectural reasons.

6 Domain Mixture Scenario

In this chapter the commonly neglected Domain Mixture scenario will be introduced and evaluated. The scenario describes a situation where there are domains that only have training samples for a subset of all relevant classes. Two constellations, the complete and the sparse Domain Mixture scenario, will be investigated in this chapter on the two introduced domain datasets MNIST/MNIST-M and CORe50. The results on the MNIST domains were published in [85].

The first section of this chapter discusses commonly investigated scenarios and introduces the two variants of the Domain Mixture that are investigated here. This is followed by a brief overview of related work, where the focus is on the scenarios occurring there, and their distributions of samples in the domain-class space during training. The experiments that are composed of the investigations of a standard scenario and the two variants of the Domain Mixture scenario will be presented in the subsequent section. Finally, a short summary closes this chapter.

6.1 Domain Adaptation Scenarios

The typical scenario that is investigated in DA research is the case that one or multiple source domains and a single target domain are given, while in all source domains usually all classes from $\mathcal{Y}$ are represented with supervised samples and for the target domain no or only unsupervised samples of all classes are available. In this thesis this described DA scenario will be called the *standard scenario*. The sample coverage of the domain-class space in this scenario is exemplary depicted in Fig. 6.1 (I), where the classification task comprises three classes and domain 3 is the unsupervised target domain. The only known label of the target domain samples is the domain label $\mathbf{d}_i$.

However, the standard DA scenario with its highly regular assignment of supervised and unsupervised data is very idealized in context of real-world applications. This is for example the case for a constellation with multiple intelligent camera systems, as it was described in the opening example of

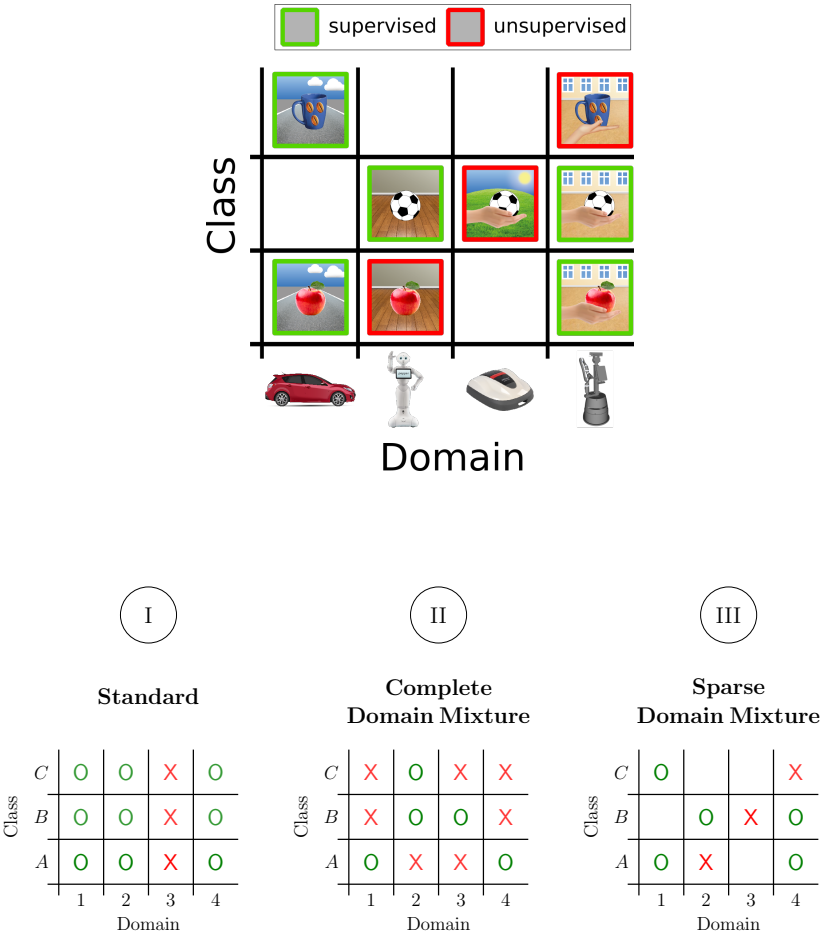


Figure 6.1: *Top*: Distribution of object samples in an interactive object learning setting of different camera systems. Each system represents an individual domain. *Bottom*: For a given domain-class combination, 'O' describes supervised samples, 'X' unsupervised samples, and empty fields missing samples. The investigations of this chapter are based on the shown scenarios, the standard scenario (I) and the overlooked Domain Mixture scenario in the two configurations (II), (III).

this thesis (see Fig. 1.1). There, each system encounters samples of different classes during lifetime and can receive labels for them through interaction with users, as e. g. presented in [31]. The depicted situation most likely leads in an early stage of the system to a sparse distribution of supervised samples for certain classes across domains. Furthermore, due to limited user interaction some classes might also only be covered by unsupervised samples. Such DA scenarios will be called *Domain Mixture scenarios* (see Fig. 6.1 II & III). A distinction is made further between the complete (II) and the sparse (III) Domain Mixture scenario. In the complete scenario the entire domain-class space is covered by samples, while in the sparse scenario all classes are covered, but some domain-class combinations are lacking data completely. For real-world systems the sparse scenario is the closest approximation since databases of such systems usually build up over time. In the depicted situations the source and target data is mixed, therefore, the terms source and target *dataset* will be used in the following instead of the terms source and target domain.

A visualization of the complete Domain Mixture scenario in the feature space plot is given in Fig. 6.2. The Domain Mixture scenario shows a kind of strong sample selection bias, where the selection bias affects classes that are exclusively selected or excluded from certain domains in the overall source dataset.

6.2 Related Work

Related work of the DA research area mostly assume that the source dataset is composed of one or multiple domains, where each provides sufficient samples for all classes defined by $\mathcal{Y}$. In such a case the notation source domain(s) is a valid notation [7, 67, 101].

A categorization of DA approaches that has not been introduced yet in Chapter 3 can be based on the amount of label information that is available from the target domain. Here it is usually differentiated between supervised DA approaches, semi-supervised DA approaches and unsupervised DA approaches.

In supervised DA approaches like [67, 81, 101] the target domain is covered by supervised samples, however, usually only few of such, which makes the optimization of a well generalizing classification model challenging. The limited availability separates this group of approaches from standard network training, where domains are commonly not considered. A typical approach here is to use these limited labeled samples from the

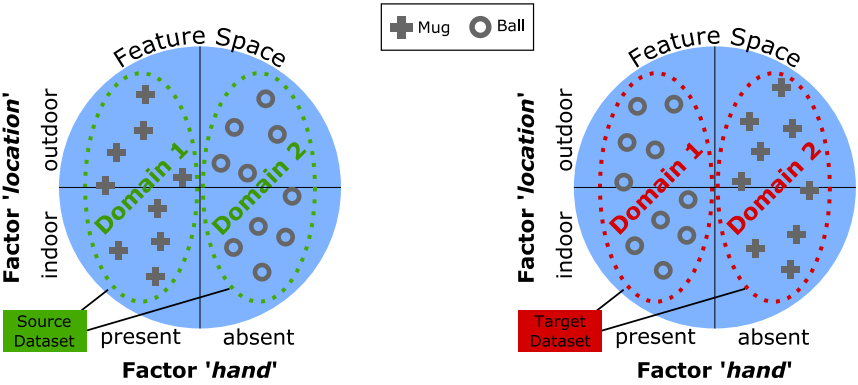


Figure 6.2: Visualization of the source dataset (left) and the target dataset (right) constellation of the complete Domain Mixture scenario at the example of two domains in the feature space plot. The visualization depicts the extreme case of only two domains with only two classes. Note, here the domain factor *'hand'* might mistakenly be interpreted as task-informative when training is carried out only on the source dataset.

target domain to align the distributions not only globally in the feature space but also on a class level, which can reduce hidden class misalignment between domains and thus reduce negative transfer [67]. Semi-supervised DA approaches like [35, 53] investigate more challenging scenarios, where only a subset of all classes are covered by few supervised samples from the target domain. In this case the supervised target data can be used to align at least the represented classes between the domains in the feature space. If additionally unsupervised data of the other classes are available, both, the class alignment and the global alignment, e. g. via a domain confusion loss, can be combined for improved DA [35].

Most DA approaches, e. g. [7, 18, 81, 82, 93], face the most challenging case, where only unsupervised samples of the target domain are available, i. e. the case depicted in Fig. 6.1 (I). Since here no additional class alignments are possible due to the missing labels, the deep learning approaches mainly focus on additional cost functions that target to align the global distributions better [8, 22, 94, 101].

Furthermore, DA approaches can be categorized by how domains are distributed to the source and the target dataset. The classic case investigated is that the source dataset consists of a single domain and the target dataset of a single, different domain, as e.g. [100]. This describes the *new*-case of

generalization from Chapter 4 (see Fig. 4.1a). Also multiple domains can be included in the source dataset, as for example in [27, 34, 51]. However, as stated already in Chapter 4 the *part*-case of generalization, where the target domain is one of multiple source domains, is mostly neglected there.

A general goal that investigations from literature and the investigations related to the Domain Mixture scenarios (II) and (III) have in common is to generalize well to the unseen or unlabeled domain-class combinations. In literature these domain-class combinations are all of a single domain, while here those are distributed over all domains. The entire Domain Mixture investigations of this chapter can be categorized as part of unsupervised DA, since from the domain-class combinations that should be generalized to no supervised samples are assumed to be given. The DA approach of [22] (Fig. 3.6) is used here as an exemplary unsupervised adversarial DA approach for the investigations, but could generally also be replaced by another approach like [8]. The results are expected to be invariant of the chosen unsupervised DA approach.

6.3 Experiments

The datasets that were used for the main investigations of the Domain Mixture scenario are the standard MNIST dataset (S) and the MNIST-M (M) dataset, where each was considered as a pre-defined domain, i. e. $d \in \{S, M\}$ (see Chapter 5). Other typical DA datasets like [5, 69, 82, 97] that provide a more challenging classification task could have been used here as well, however, to get a first general idea of the difficulties of the Domain Mixture scenario, the simple MNIST data was most suitable. Since the investigation required several hundred optimization runs, the dataset choice further allowed to use an architecture with less parameters, which in consequence results in less training time. To show that the results also hold for real-world datasets in combination with a more complex architecture, extended investigations with the CORe50 dataset are presented at the end of this section.

For the MNIST experiments the adversarial DA architecture depicted in Tab. 5.1 was used. The learning-rate μ and the adaptation factor λ were updated during training according to (5.5) and (5.4). The initial learning rate was set to $\mu_0 = 0.01$ and dropout was applied in all fully-connected layers during training. A single optimization run consisted of two epochs.

6.3.1 Standard Scenario (I)

In the first experiments the standard scenario (I), will be investigated using the two MNIST domains with their ten digit classes. The results of these will later be used as a baseline for the newly defined Domain Mixture scenarios. Some experiments reproduce values from the used architecture in [22], however, due to the fact that for the DA approach an own implementation in a different framework was used, slightly different results could be expected.

The results for these experiments are shown in Tab. 6.1. Each reported number is based on an average of 10 optimization runs, where in each all parameters were initialized randomly. For the experiments here the term source and target *domain* are valid notations since for one domain all classes are either completely covered by supervised or unsupervised samples. For each transfer configuration, i. e. $S \rightarrow M$, $M \rightarrow S$, $(S + M) \rightarrow (S + M)$, three results are reported. The 'DA [22]' result and two baselines without DA. 'Source only' refers to the baseline of training only on the source domain. 'Train on Target' to the baseline of training on the training set of the target domain. All reported numbers are based on the test sets of the pre-defined domains.

Table 6.1: Standard DA experiments with the MNIST domains. The results show the accuracy on the target test sets.

	I		
Source:	S	M	$S + M$
Target:	M	S	$S + M$
Source only	0.546	0.976	0.978
DA [22]	0.763	0.963	0.965
Train on Target	0.962	0.992	0.978

Column 1 of Tab. 6.1 shows the classic transfer $S \rightarrow M$ that has also already been investigated in [22], where S is the source domain and M the challenging target domain that comes with new factor values related to the background. 'Source only' shows with the very low accuracy of only 54.6 % that the classification model is biased too much to the monotonous black background in S and therefore is not forced to learn filters that are able to cope with varying background values as represented in M. Using the adversarial DA method [22] leads to a significant gain in classification

accuracy to 76.3 %. The introduction of the new factor values through the unsupervised samples from M obviously helps to learn features that can handle varying background factor values better. Compared to the results for the same experiment presented in [22], where 81.5 % accuracy was reached, the number here is slightly lower. The lower result is assumed to be related to the different deep learning framework that was used. Nevertheless, the general tendencies are similar and thus show the correct implementation of the approach. The last result for $S \rightarrow M$ is the upper baseline 'Train on Target', which shows which performance can be reached if training and test data are both from the M domain. As expected for this case, a high classification accuracy of 96.2 % could be reached.

The transfer $M \rightarrow S$, given in the second column of Tab. 6.1, shows a very high accuracy even without the application of DA, i. e. training only on the training set of M. The most likely reasons for this are on the one hand side that M introduces already a large variety of background related factor values and on the other hand side that the S domain might anyways already implicitly be included in M due the way how M was created (see Chapter 5). Thus the latter explanation would correspond to the *part*-case of generalization that was introduced in Chapter 4 (see Fig. 4.1c). Applying DA for the $M \rightarrow S$ case leads to slight negative transfer. Possible reasons for this could be the removal of task-informative factors related to domain factors and the general influence of the additional loss from the domain classifier that puts in this case a needless constraint on the feature learning process.

The third investigated experiment represents a rather unusual transfer with $(S + M) \rightarrow (S + M)$. It describes the *same*-case of generalization (see Fig. 4.1b). Here, the two baseline results 'Source only' and 'Train on Target' are identical. The accuracy is with 97.8 % approximately in the middle between 'Train on Target' of $S \rightarrow M$ and $M \rightarrow S$. For DA, negative transfer can be observed for probably the same reasons as at $M \rightarrow S$. The result here and the result of the previous transfer $M \rightarrow S$ show the general risk of negative transfer when a DA approach with a learned normalization is applied in the *same*- or the *part*-case of generalization.

6.3.2 Complete Domain Mixture Scenario (II)

In this section the more open Domain Mixture scenario will be investigated in its complete configuration (see Fig. 6.1 (II)). The goal is here to reveal the effects on the classification model with and without DA for the described scenario. Furthermore, it will be investigated how labels should ideally be

distributed in the domain-class space if control over labeling is given, but the amount of labels is limited.

In the experiments here, the source dataset is composed of supervised samples from the subset of classes $\mathcal{Y}_S \subseteq \mathcal{Y}$ from the S dataset and the subset of classes $\mathcal{Y}_M \subseteq \mathcal{Y}$ from the M dataset. It is ensured, that the union of both include all digit classes, i. e. $\mathcal{Y}_S \cup \mathcal{Y}_M = \mathcal{Y}$. Additionally, in the experiments the cardinality of the subsets of classes is always identical, i. e. $q = |\mathcal{Y}_S| = |\mathcal{Y}_M|$. Under the conditions described, the possible cardinalities of $\mathcal{Y}_S, \mathcal{Y}_M$ are in the range of $5 \leq q \leq 10$. For $q > 5$ this means that certain classes are represented by supervised samples from both domains in the source dataset, i. e. partial overlap of supervised classes in the domain-class space. The case $q = 10$ corresponds to full overlap and thus to the standard DA experiment $(S + M) \rightarrow (S + M)$ reported in Tab. 6.1.

The target dataset is composed of unsupervised samples for all domain-class combinations that are not in the source dataset, i. e. $\bar{\mathcal{Y}}_S = \mathcal{Y} \setminus \mathcal{Y}_S$ and $\bar{\mathcal{Y}}_M = \mathcal{Y} \setminus \mathcal{Y}_M$. Since one is generally interested in the generalization to the domain-class combinations for which no supervised data is available, $\bar{\mathcal{Y}}_S$ and $\bar{\mathcal{Y}}_M$ also define the test classes on which the evaluations of the experiments are carried out here. This will be done independently of the application of DA. The experiments were conducted in dependence of the number of supervised classes per domain $q = |\mathcal{Y}_d|$ with $d \in \{S, M\}$. Every result for a specific cardinality is based on 35 runs, while in each run the classes in $\mathcal{Y}_d$ were chosen randomly under the introduced constraints. An exemplary run configuration is shown in Fig. 6.3.

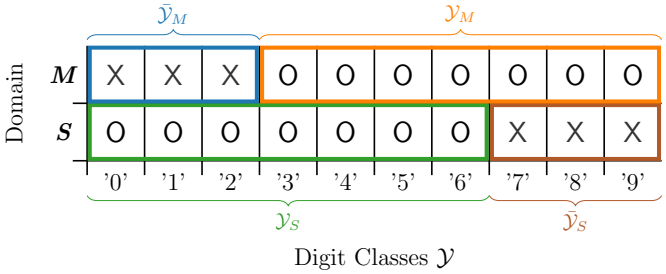


Figure 6.3: Exemplary distribution of the domain-class combinations to $\mathcal{Y}_S$, $\mathcal{Y}_M$, $\bar{\mathcal{Y}}_S$, and $\bar{\mathcal{Y}}_M$, for $q = |\mathcal{Y}_d| = 7$. 'O' describe supervised samples, 'X' unsupervised samples. Note, the supervised and unsupervised domain-class combinations in the experiments are randomly distributed and are not necessarily concatenated as shown here.

The upper two plots of Fig. 6.4 show the results for the 'Source only' and 'DA [22]' evaluations under the conditions introduced. The 'Source only' evaluations reveal a remarkably poor performance for the case of $|\mathcal{Y}_d| = 5$. This is the case where no overlapping supervised class samples from the two domains exist and testing is performed on the domain-class combinations that were not involved during training at all. The average accuracy on the entire test set, i. e. on $\bar{\mathcal{Y}}_S$ and $\bar{\mathcal{Y}}_M$, reaches only 15.8 %. This is surprisingly low, since naively one could have expected the classification model to generalize better, as it has seen all domains and all classes. In terms of domain factors this means there should not be completely new factor values anymore, but rather only new combinations of already known values. Especially, in comparison to the 'Source only' standard DA experiment $S \rightarrow M$ (see Tab. 6.1), where in contrast the model only saw a single domain with all classes, i. e. new factor values are introduced by the target domain, a higher accuracy was expected here. This shows that new combinations of already known factor values seem to be the more challenging task here.

As a consequence of the unexpected poor performance, the errors made by the classification model were investigated more closely. The investigations showed that the classification model matches test data of one domain to classes that have been seen during training of the same domain. Given the exemplary constellation of Fig. 6.3, where from M the digits 3 to 9 were shown during training, testing is performed on the digits 0 to 2 of M. The predicted classes for the test data then mostly are among the digit classes 3 to 9 that were shown during training. This indicates that the classification model obviously uses the value of a domain factor as an additional powerful feature for classification. Doing so, it allows the classifier to split the training data into two groups for the two domains, and then based on that, classify which digit is shown. Overall, this simplifies the classification, since within the pre-classified groups only 5 digit classes have to be discriminated for the case of $|\mathcal{Y}_d| = 5$. This pre-grouping, however, becomes harmful if new domain-class combinations appear and can thus lead to false classifications as a consequence.

To provide a measure for the influence of this pre-grouping, the subset confusion was defined as

$$\mathsf{SuCo}(\bar{\mathcal{Y}}_d) = \frac{|\{i \mid \max(\mathbf{y}_i) \in \bar{\mathcal{Y}}_d,\ \max(\mathbf{y}_i') \in \mathcal{Y}_d\}|}{|\{i \mid \max(\mathbf{y}_i) \in \bar{\mathcal{Y}}_d\}|}, \tag{6.1}$$

for a given test set of the classes $\bar{\mathcal{Y}}_d$ with samples i, the given ground-truth one-hot label vector $\mathbf{y}_i$ and the predicted one-hot label vector $\mathbf{y}_i{}'$. If

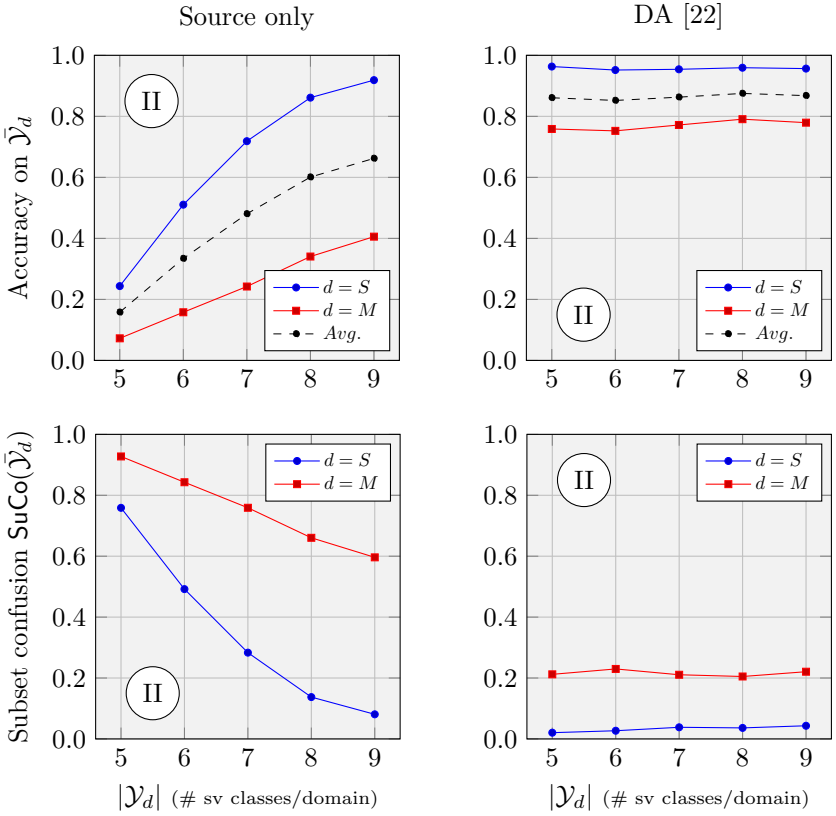


Figure 6.4: 'Source only' and DA results for the accuracy on $\bar{\mathcal{Y}}_d$ and the subset confusion $\mathsf{SuCo}(\bar{\mathcal{Y}}_d)$. During optimization of the architecture, the training set, i. e. source dataset for DA, was composed of supervised (sv) samples from S for the classes $\mathcal{Y}_S$, and sv samples from M for $\mathcal{Y}_M$. For DA all domain-class combinations that were not in the source dataset were involved unsupervised during optimization.

$\mathsf{SuCo}(\bar{\mathcal{Y}}_d) = 0$ it means that all the predicted classes of the test samples are in the correct subset $\bar{\mathcal{Y}}_d$. Note, this does not indicate that the classification was correct. For $\mathsf{SuCo}(\bar{\mathcal{Y}}_d) = 1$ it means that all test samples were mixed up with classes that were included in the source dataset from the same domain. A value of 1 thus indicates that none of the classifications was correct.

The lower plots of Fig. 6.4 show the corresponding subset confusions for the experiments. Without DA the subset confusion shows for $|\mathcal{Y}_d| = 5$ a very high value and thus explains the low classification accuracy of this constellation. Generally, when comparing the curve of the accuracy and the subset confusion, it is evident that they are basically the flipped counterpart of each other. This indicates that the classification errors are mostly caused by the described pre-grouping learned on the given source dataset.

When the amount of supervised classes per domain $|\mathcal{Y}_d|$ increases, the accuracy on the unseen domain-class samples increases as well, and ultimately reaches 66 % over all test samples. The reason for the gain in accuracy is most likely caused by the increasing overlap of supervised classes samples in $\mathcal{Y}_S$ and $\mathcal{Y}_M$ during training. In terms of factors this means that an increased amount of factor value combinations is shown. Showing the same digit supervised with varying background factor values forces the classification model more to learn the factors independent of their combinations rather than using the combination itself as a single feature. Thus, digit shapes and background are represented more independently, from which also the test samples in $\bar{\mathcal{Y}}_S$ and $\bar{\mathcal{Y}}_M$ profit.

Overall, similar to the previous experiments on the standard DA scenario (Tab. 6.1) the results here also showed the increased difficulty of M compared to S, which is reflected in the permanently lower accuracy on M.

When the DA method from [22] is applied in the complete Domain Mixture scenario (II), already at only five supervised classes per domain, i. e. $|\mathcal{Y}_d| = 5$, a very high accuracy can be observed (see Fig. 6.4, top right). The accuracy reaches here without any overlap in supervised classes from the domains a high accuracy of 86 %. This accuracy meets the expectations since it lies directly in the middle of the standard experiments with DA of $S \rightarrow M$ and $M \rightarrow S$ from Tab. 6.1.

The attached domain classifier together with the unsupervised domain-class combination samples obviously helps to inhibit the learned pre-grouping, i. e. it inhibits the domain factors as a feature, making them no longer available for exploitation. In contrast to the experiments without DA, surprisingly here the accuracy does not further increase with a larger

overlap of supervised classes from the domains, instead it remains approximately constant. An increase could have been expected here as well since with more overlap of supervised class samples from the domains, the model could be forced to learn more general features.

With regard to a multi-camera system setting with human interaction it can be concluded from the standard DA experiments (I) and the complete Domain Mixture scenario experiments (II) that it makes most sense to primarily provide labels to domains that come with a high variance in factor values. If this domain property is known, it allows to achieve the better performance, as it was shown in the $S \rightarrow M$ vs. $M \rightarrow S$ experiments. The application of DA is not an essential requirement in this case. However, the domain that provides the greater variance in factor values is sometimes unknown. If for such a case the goal is to keep the minimum performance as high as possible, it makes most sense to distribute labels across the involved domains and combine this with DA, to inhibit that domain factors are used to learn a pre-grouping of the classes.

6.3.3 Sparse Domain Mixture Scenario (III)

The even more realistic and relevant scenario for real-world applications is the sparse configuration of the Domain Mixture scenario (III). In contrast to the complete Domain Mixture scenario, here the domain-class space is not completely filled, meaning that for some domain-class combinations not even unsupervised samples are available. The relevant question for this configuration is whether the unsupervised domain-class samples are actually required for DA and if so, how many are needed to achieve a reasonable performance. Note that in the Domain Mixture scenario the domain classifier can still be trained even without unsupervised data, since data from both domains is always available in the chosen investigation setting.

To answer this questions the evaluation scheme of the experiments was slightly changed. In the previous investigations, $\bar{\mathcal{Y}}_d$ always represented all unsupervised (usv) domain-class combinations during training, i. e. $\bar{\mathcal{Y}}_d = \bar{\mathcal{Y}}_{d_{\text{usv}}}$. For the sparse Domain Mixture scenario investigations is $\bar{\mathcal{Y}}_d = \bar{\mathcal{Y}}_{d_{\text{usv}}} \cup \bar{\mathcal{Y}}_{d_{\text{unseen}}}$, where $\bar{\mathcal{Y}}_{d_{\text{unseen}}}$ represents the domain-class combinations that were not shown during optimization at all, i. e. the gaps in the domain-class space. In the experiments here, the number of supervised classes per domain was fixed to $|\mathcal{Y}_d| = 5$ and the number of unsupervised classes per domain was increased from 0 to 5, i. e. $|\bar{\mathcal{Y}}_{d_{\text{usv}}}| \in \{0, ..., 5\}$, to investigate the necessity for unsupervised data.

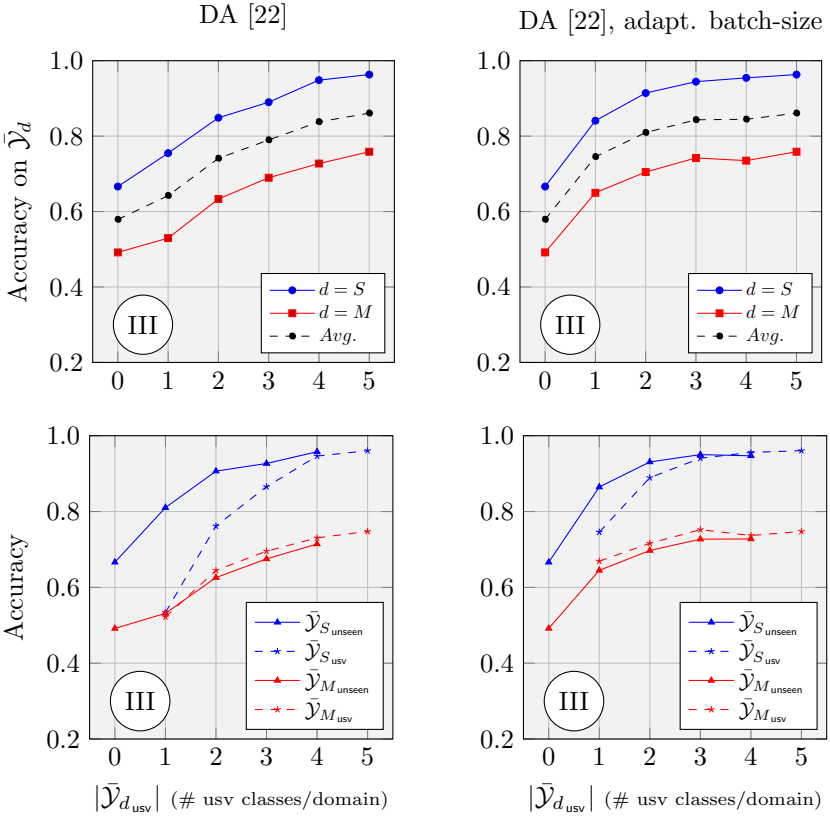


Figure 6.5: Investigation of the necessity of unsupervised samples in the chosen DA approach for a fixed number of supervised classes per domain $|\mathcal{Y}_d| = 5$. The right plots show the result for an adapted batch-size, where the appearance frequency of domain-class combinations in the batch is balanced. *Bottom*: Split up of the upper results into actual unseen and unsupervised (usv) domain-class combinations.

Fig. 6.5 shows the results of the experiments. The results for $|\bar{\mathcal{Y}}_{d_{\text{usv}}}| = 5$ correspond to the result of the complete Domain Mixture scenario with DA for the case of $|\mathcal{Y}_d| = 5$. In the top left plot, $|\bar{\mathcal{Y}}_{d_{\text{usv}}}| = 0$ depicts the case where no additional unsupervised data from the target dataset is added. The average accuracy reached there is with 57.9 % clearly higher than the accuracy for 'Source only' at $|\mathcal{Y}_d| = 5$ in Fig. 6.4, where only 15.8 % with the identical training and test data was reached. This clearly reinforces the general benefits of using DA in the Domain Mixture scenario, even without additional unsupervised data. The domain classifier attached to the shared feature extractor obviously removes the background related factors to a large extent and thus hinders that features are based on the corresponding domain factors. Through this the mentioned pre-grouping of the samples is suppressed in large parts. Since most domain factor values are already sufficiently represented in $\mathcal{Y}_S$ and $\mathcal{Y}_M$, no additional unsupervised data is obviously required to achieve a first positive effect of the domain classifier. With an increasing number of unsupervised classes per domain an increase in accuracy can be observed. This is likely caused by new background related factor values that add to the positive effect of the domain classifier.

Another notable result of the experiments conducted here can be seen in the bottom left plot of Fig. 6.5. In this plot, the test accuracy is further split up into $\bar{\mathcal{Y}}_{d_{\text{usv}}}$ and $\bar{\mathcal{Y}}_{d_{\text{unseen}}}$. The accuracy for the unsupervised samples of M, i. e. $\bar{\mathcal{Y}}_{M_{\text{usv}}}$, is for all $|\bar{\mathcal{Y}}_{d_{\text{usv}}}|$ slightly above the unseen domain-class combinations $\bar{\mathcal{Y}}_{M_{\text{unseen}}}$. This could have been expected, since the model has seen the $\bar{\mathcal{Y}}_{M_{\text{usv}}}$ domain-class combinations already during training and thus should be capable of handling those better. In the S domain this is different. Here the accuracy on $\bar{\mathcal{Y}}_{S_{\text{usv}}}$ is surprisingly constantly below $\bar{\mathcal{Y}}_{S_{\text{unseen}}}$, while this is very pronounced for small $|\bar{\mathcal{Y}}_{d_{\text{usv}}}|$. A hypothesis for this effect is related to the visual 'simplicity' of the S dataset on which already with only training on M a good accuracy can be achieved, as shown in Tab. 6.1 for $M \rightarrow S$ without DA. Adding in such a case an additional constraint by means of the domain classifier can lead to negative transfer on the unsupervised domain-class combinations similar to the experiments $M \rightarrow S$ and $(S + M) \rightarrow (S + M)$ with DA.

For a more detailed study of the effect that the unseen domain-class combinations show a better result than the unsupervised combinations, the composition of batches during training was closer investigated. During training of the DA architecture proposed by [22], the second half of each batch consists of images from the available unsupervised samples, i. e. samples of which only the domain label is known. In the sparse Domain Mixture scenario investigations, however, this can lead to a strong imbalance

among all domain-class combinations when training the domain classifier.

With a batch-size of 64 and ten supervised domain-class combinations, each supervised combination appears on average 3.2 times in the first half of a training batch. Following the procedure of [22] for the experiments here, the second half of each batch is filled up with unsupervised domain-class combinations. This leads to an imbalance regarding the appearance frequency if less unsupervised than supervised domain-class combinations are available. In such a case the unsupervised domain-class combinations are presented more frequently to the domain classifier and can lead to a falsely biased tendency to classify samples. To tackle this issue a heuristic was chosen with which for each experiment an individual batch-size was determined. The batch-size was increased starting from 32 samples by approximately 3.2 samples for each additional unsupervised domain-class combination to keep the overall average appearance of 3.2 times per domain-class combination in the complete batch. With the increase of the number of unsupervised classes per domain $|\bar{\mathcal{Y}}_{d_{\text{usv}}}|$ from 0 to 5 the batch-sizes are $BS \in \{32, 38, 45, 51, 58, 64\}$. Note that this heuristic does not necessarily define the optimum solution. The results for the adapted batch-size experiments are given in the right plots of Fig. 6.5. It shows that the general accuracy for all experiments where the imbalance existed increased with the adapted batch-size. However, the described effect that $\bar{\mathcal{Y}}_{S_{\text{usv}}}$ is constantly below $\bar{\mathcal{Y}}_{S_{\text{unseen}}}$ could only be reduced but not resolved completely.

Theoretically, this adaptation of the batch-sizes would also have been necessary for the previous DA experiments in the complete Domain Mixture scenario (Fig. 6.4). The imbalance exists there as well for $|\mathcal{Y}_d| > 5$. However, generally such an adaptation for real-world applications is not feasible since the classes represented in the unsupervised data are unknown.

6.3.4 Evaluation on CORe50 Dataset

The MNIST domains provide a clean setting that allowed to reveal the actual problem of the Domain Mixture scenario when no DA is applied. To show that these findings generally also hold for real-world applications combined with more complex architectures, the investigations of the complete Domain Mixture scenario were also conducted on the CORe50 dataset (see Fig. 5.2). The architecture that was used here is the VGG-16 architecture depicted in Tab. 5.2. Here a batch-size of 64 was chosen as well and the initial learning was set to $\mu_0 = 0.0001$.

For the evaluations in this thesis only pairs of the CORe50 domains were investigated. Combinations of more than two domains would have been

equally possible and would probably lead to comparable results. Since the baseline experiments in Chapter 5 revealed that the combinations of the used hand within the domain has a major influence on the generalization between them, two constellations are investigated in the experiments here. The first constellation examines the case of same hands, while the second constellation the case of two different hands.

For the constellation of same hands, here the right hand, the CORe50 domains 1 and 4 were chosen, i. e. $d \in \{1, 4\}$. The results for the accuracy and the subset confusion, with and without DA, are shown in Fig. 6.6.

For 'Source only', the general tendency of the accuracy over an increasing $|\mathcal{Y}_d|$ are similar to the MNIST experiments, with a constant gain in performance the more supervised domain-class combinations are available while training. The maximum average accuracy that is reached here without DA is at 44 % for $|\mathcal{Y}_d| = 9$. Similar to the MNIST experiments the subset confusion explains here also most of the errors, which is most likely caused by the dominant green background of domain 4 in contrast to the simple white background of domain 1. In contrast to the MNIST experiments the overall accuracy reached here is lower, which can be explained by the generally more difficult classification task of CORe50. Looking at the difference of the individual results of the two domains involved here, i. e. of $\bar{\mathcal{Y}}_1$ and $\bar{\mathcal{Y}}_4$, the gap between them can be observed to be smaller. Domain 1 shows a slightly better result, which is also in line with the results of the one-to-one transfers of Tab. 5.3 where domain 1 was on average easier to generalize to.

When DA is applied, also a clear improvement of the classification accuracy on the unsupervised domain-class combinations can be observed. The accuracy increases slightly with increasing $|\mathcal{Y}_d|$. Interestingly, here domain 4 shows now a better result than domain 1. Clear reasons for this could not be found within the scope of this thesis.

The other domain constellation that was investigated involved domains 3 and 4, that were recorded presenting the objects in different hands. In domain 3 the left hand was used, while in domain 4 the right hand. As shown in the baseline experiments of Chapter 5, two domains with varying hands generalize clearly worse to each other. Therefore, also in the Domain Mixture scenario significantly lower accuracies can be expected. The results shown in Fig. 6.7 confirm this.

In the case of 'Source only' the average accuracy reaches a maximum of only 16 % accuracy on the domain-class combinations that were not involved during training. Note this low accuracy despite the fact that the overlap in supervised class samples between the domains is here already at

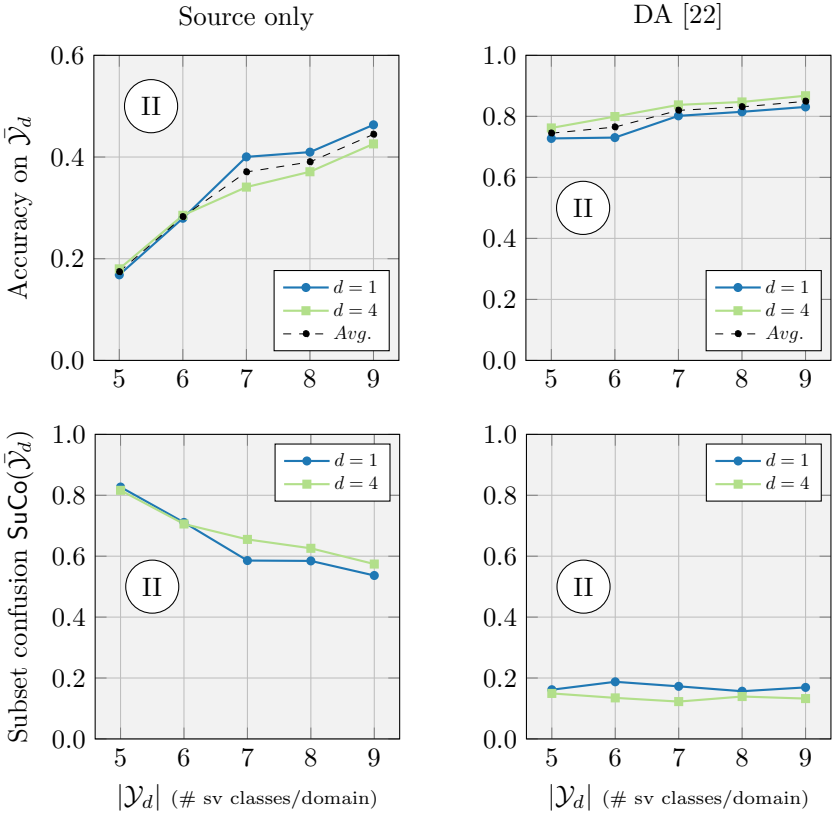


Figure 6.6: Complete Domain Mixture experiments on the CORe50 dataset, here in both involved domains, domain 1 and domain 4, the right hand was used for presenting the objects. Compared to the MNIST experiments the overall tendencies are similar, however, the maximum accuracy is lower and the results of the individual domains are closer. Note the different scaling of the y-axis in the top left plot.

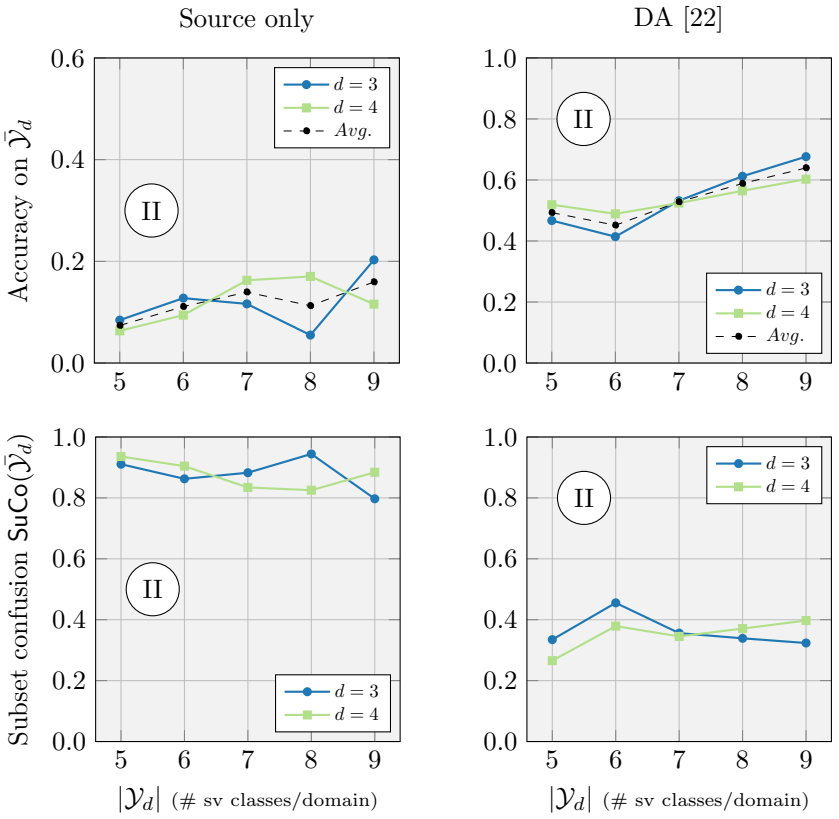


Figure 6.7: Complete Domain Mixture experiments on the CORe50 dataset, here with a change of the presenting hand between the involved domains. In domain 3 the left hand was used, while in domain 4 the right hand was used. A significantly lower performance compared to the transfer between domains where the same hand was used can be observed.

its maximum with $|\mathcal{Y}_d| = 9$. Obviously the architecture is in this case still not forced to learn more general features, which could indicate that it has too many free parameters allowing to solve the classification task based on the pre-grouping for the classes in $\bar{\mathcal{Y}}_d$. Furthermore, here a larger variance along $|\mathcal{Y}_d|$ can be observed, which is especially pronounced for $|\mathcal{Y}_d| = 8$. This might be caused by certain object categories that are more difficult to classify than others and that have been randomly assigned to $\mathcal{Y}_d$. A larger number of runs would probably compensate for this. The subset confusion shows for 'Source only' here as well, that most of the errors can be traced back to the pre-grouping.

When DA is applied, a reduced, but still significant overall increase in accuracy can be observed here as well. For $|\mathcal{Y}_d| = 5$ the accuracy is at 49 % which is better than the result of $3 \rightarrow 4$ and $4 \rightarrow 3$ with DA of Tab. 6.1, where 41 % and 45 % respectively were reached with DA. Note that the domain classifier sees in both evaluations exactly the same images during training. The increase can therefore be explained by the label classifier which is more familiar with both domains since it has seen supervisedly some classes of each during training. Also here an increase with more overlap in the supervised classes from both domains can be observed. However, the maximum accuracy reached here is only at 64 %. A hypothesis for this low accuracy compared to the same hand domains is again, that the removal of the factor *'hand'* also removes task-informative components. The subset confusion here is still on a high level, which shows that the removal of the domain factor *'hand'* from the feature representation is more challenging than only the background. If it is still partially embedded in the features, it can still be used for the pre-grouping. With domain 1 and domain 4 it was no domain factor and therefore could not be used for the pre-grouping, resulting in a lower subset confusion.

Further experiments with other combinations of domains can be found in the Appendix A.1. The overall tendencies are all similar. For a combination of domains that are more similar regarding the background and further come with the same presenting hand, like for example domain 1 and domain 8, the performance gain through DA is reduced, but still significant. All in all, the general difficulties of the Domain Mixture scenario without DA and the benefit of using DA there, could be confirmed on the real-world dataset combined with a more complex neural network architecture.

6.4 Summary

This chapter introduced and investigated the Domain Mixture scenario. The scenario is an overlooked constellation in DA that describes the situation where several domains are represented with supervised samples during training, however, no domain is completely represented by supervised samples for all competing classes. It was shown that this scenario has high relevance for real-world systems with multiple entities, e. g. robots, that are equipped with cameras and collect image data over time for which not necessarily a ground-truth label is provided. It was shown at baseline experiments on MNIST and later confirmed on CORe50 that, given the Domain Mixture scenario, a standard CNN uses the domain factors as prominent features to perform some form of pre-grouping that eases the overall classification task. However, this makes the generalization to unseen domain-class combinations almost impossible. In contrast, using a DA method like [22], the domain factors are mostly removed, whereby the model is not capable to apply the pre-grouping in the classification process anymore and a significantly improved generalization to new domain-class combinations is possible. This leads to a clear increase in classification accuracy, even without any overlapping supervised domain-class combinations of the involved domains. For the MNIST experiments, it was further shown that even with gaps in the domain-class space, i. e. with domain-class combinations which are not even represented by unsupervised samples, a significant increase in performance through DA can be achieved.

Generally, based on the evaluations of this chapter, it can be recommended that if being limited to labeling only specific domain-class combinations and the goal is to achieve always the highest minimum performance, then the labels should be distributed among the domains and DA should be used. The experiments showed that this strategy is only harmful if one of the domains implicitly covers already one of the other domains. If this is known beforehand, then the best performance can be achieved by allocating all labels to the classes of the more comprehensive domain and to train the classification model without DA.

7 Factor Preserving Domain Adaptation

In this chapter Factor Preserving DA (FP-DA) will be introduced, a new training method that was developed as part of this PhD project. The idea of this training method is to reduce negative transfer caused by the removal of specific factors through DA. With FP-DA it is possible to preserve a chosen factor during DA with multiple source domains. If such a factor is task-informative, this procedure can help to reduce negative transfer. In the first section of this chapter the theoretical idea of FP-DA will be presented, while in the subsequent Section 7.2 related work will be introduced in brief. The preservation of a factor in FP-DA is only useful if the removal of the chosen factor would influence the transfer performance negatively. To find such factors, it will be shown that the application of PCA on one-to-one transfer experiments between the domains can be a useful means. This procedure will exemplarily be carried out in Section 7.3.1 based on the baseline results of the CORe50 one-to-one experiments of Chapter 5. As an extension, it will be shown in Section 7.3.2 how this PCA procedure can further be used to identify factors on class level that potentially add to a poor generalization between the domains. The effectiveness of the proposed FP-DA method will then be demonstrated on the CORe50 domains quantitatively in Section 7.3.3 and qualitatively with different visualizations methods in Section 7.3.4. Additionally, to show limitations and necessary constraints of FP-DA to work, it will also be applied on the OpenLORIS object dataset [90] in Section 7.3.5.

The main contents of this chapter are based on the published journal article [86].

7.1 FP-DA Approach

In the one-to-one experiments that were carried out in Chapter 5.3 it was shown that negative transfer through DA mainly occurred between domains where different hands are used to present the objects. It is assumed that

this also caused the observed negative transfer within the multi-source setting investigated in the leave-one-out experiments (see Fig. 5.5). As a potential reason of this negative transfer the removal of all domain factors from the feature representation through DA has been identified. As stated before, a domain factor can be semantically related or just visually close to some task-informative factors. In both cases the removal can make the classification task more difficult. To sum up, it is not always beneficial to remove all domain factors, in certain cases it might be useful to preserve some factors despite the fact that they are domain-informative.

With FP-DA it is possible to preserve a chosen factor f_c during DA. The proposed architecture of Ganin et al. [22] (see Fig. 3.6) represents the basis for this approach. The update rules of the parameters of the feature extractor $\boldsymbol{\theta}_e$, the label classifier $\boldsymbol{\theta}_y$, and of the domain classifier $\boldsymbol{\theta}_d$ were introduced in (5.1), (5.2) and (5.3). Using this approach in the way as proposed by [22], all domains compete against each other in the domain classifier, resulting in the removal of all domain factors. In FP-DA this competition is limited to only groups of domains whereby a chosen factor f_c can be preserved. The competition here is switched off between domains that have different values for f_c. This is achieved by decomposing during backpropagation the partial derivative of the loss L_d with respect to the parameters of the domain classifier $\boldsymbol{\theta}_d$,

$$\frac{\partial L_d}{\partial \boldsymbol{\theta}_d} = \frac{\partial L_d}{\partial \mathbf{d}'} \cdot \frac{\partial \mathbf{d}'}{\partial \boldsymbol{\theta}_d}, \tag{7.1}$$

and replacing $\frac{\partial L_d}{\partial \mathbf{d}'}$ by

$$\widetilde{\frac{\partial L_d}{\partial \mathbf{d}'}} = \mathbf{z} \odot \left(\frac{\partial L_d}{\partial \mathbf{d}'} \right). \tag{7.2}$$

Here, $\mathbf{z}$ is a m-dimensional vector of which each element z_j corresponds to one of the m domains in a given multi-domain constellation. The value of each element is either 1 or 0. For $z_j = 1$ it means that the gradient of the domain output neuron d_j is kept and backpropagated to the parameters $\boldsymbol{\theta}_d$. In contrast, if $z_j = 0$, then the gradient from the output neuron d_j is not backpropagated. The elements of $\mathbf{z}$ are determined dependent on a given training sample of domain k the following way:

$$z_j = \begin{cases} 1 & \text{if } f_c(j) = f_c(k) \\ 0 & \text{otherwise,} \end{cases} \tag{7.3}$$

where $f_c(k)$ is the value of the factor f_c in domain k and $f_c(j)$ of domain j. Like this, it is guaranteed that there is only competition between domains that share the same value for the factor f_c as the current training sample. Note that the application of FP-DA requires a factorization based on categorical factor values. Fig. 7.1 visualizes the principle in the domain classifier at an example of four given domains.

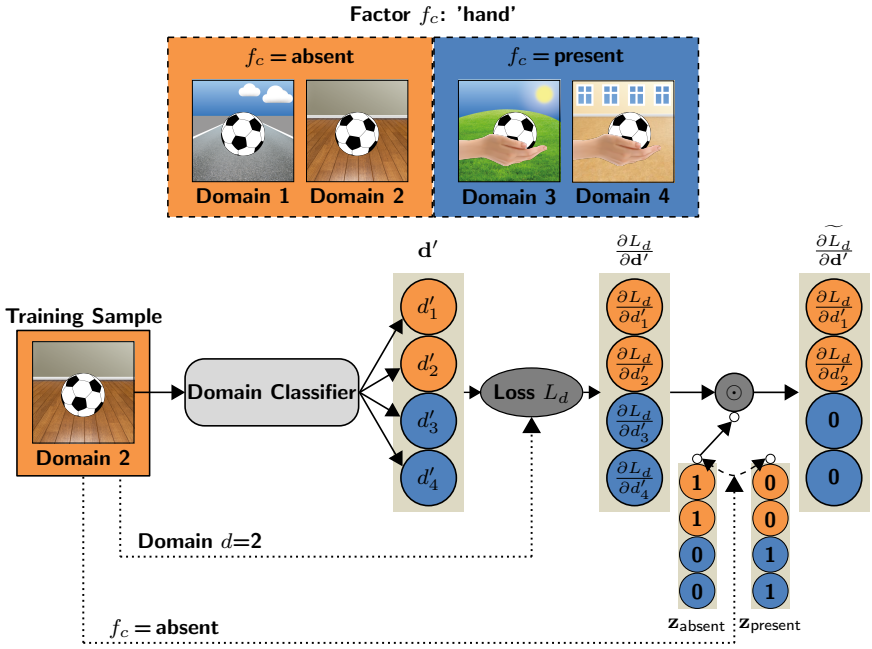
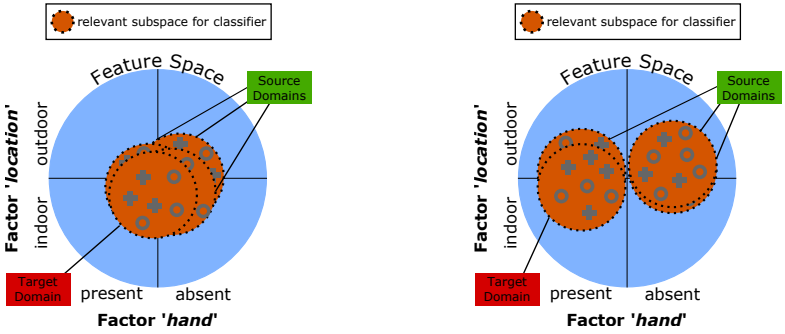


Figure 7.1: The developed FP-DA training method uses given groups of domains, here generated by the values of the chosen factor f_c, *'hand'*, and allows competition only between domains within each group. A training sample from the group where $f_c = \text{absent}$ is forwarded through the domain classifier. The resulting gradient vector $\frac{\partial L_d}{\partial \mathbf{d}'}$ is element-wise multiplied by $\mathbf{z}_{\text{absent}}$ that zeros the gradients of the domains where $f_c = \text{present}$.

To guarantee that still all other domain factors are removed, the domains within each group should contain different values of the other domain factors. If this is not the case, a reduced effect of DA could be expected, which depicts a limitation of FP-DA. The effect of FP-DA compared to the standard approach of [22] is shown in the feature space plot in Fig. 7.2.



(a) Ganin [22] **(b)** FP-DA

Figure 7.2: Comparison of the DA effect in the feature space plot between the standard adversarial DA approach [22] and the novel FP-DA approach. In Ganin [22] both domain factors are removed. With FP-DA it is possible to preserve a chosen factor, here the factor *'hand'*, and to remove only the remaining factors, here *'location'*.

7.2 Related Work

This section is a short extension to the related work that has already been introduced in Chapter 3.

The developed FP-DA method is designed to completely switch off competition between groups of domains that are not sharing the same value for a selected factor. In literature, with focus on multi-source DA, there are other approaches that try to better control or limit the competition between domains to reduce negative transfer.

Often such a limitation is introduced by a weighting strategy of the involved domains. One method is to weight pair-wise the target domain in relation to each single source domain as it was for example done in [30, 65, 79, 109, 111]. The overall classification model is then usually defined by multiple sub-classification models for each pair, that together, weighted by a score of the pair-wise domain similarity, contribute to the final classification of samples from the target domain. An approach of this kind is the GAN based approach already presented in Chapter 3, Fig. 3.9. Another method is to not only consider the source-target relations, but also the relations between multiple source domains and use them to form a weighted classification model [52, 74, 78, 104]. In contrast to FP-DA, none of the methods found in literature considers factor or group based

competition between domains.

The main goal of FP-DA is to reduce negative transfer by preserving a chosen factor. As mentioned in Chapter 3 other approaches target to reduce negative transfer through aligning the domains additionally on class-level. This is for example also done in [112]. There, the alignment of class specific clusters of source and target domain is achieved through an additional cluster adaptation loss that is based on pseudo labels generated during training for the unsupervised target domain samples. In [110] they add additionally to a domain classifier a sliced Wasserstein distance measure on the softmax output of the label classifier with the target to increase the discriminability of classes also for the target domain. Unlike adversarial approaches solely based on a domain classifier, here the unsupervised samples are also forwarded to the label classifier. In contrast to the mentioned approaches, FP-DA considers the effect of class independent factors in multi-domain data, which is a complementary direction.

In [87] they consider the case where the difference between a source and target domain can be described by an accumulation of multiple intermediate domains. Their approach is to adapt the model step-wise to the target domain, by generating pseudo labels and fine-tuning the model on the intermediate domains until the target domain is reached. In terms of factors they consider a single continuous domain factor that changes its value smoothly between the adjacent domains. FP-DA in contrast considers the more general case of multiple heterogeneous factors in a larger multi-domain setting, while further no intermediate domains are assumed to be given.

7.3 Experiments

This section will present the experiments that were done to evaluate the proposed FP-DA approach. Similar to the previous experiments the classification task will be image classification, here mainly based on the domains of CORe50 but in Section 7.3.5 extended by experiments on the OpenLORIS object dataset. The preceding experiments that are relevant for this section are the multi-source leave-one-out experiments and the single-source one-to-one transfer experiments on the CORe50 dataset from Chapter 5.3. Using the results of the one-to-one experiments it will be shown how factors that potentially cause errors in transfer between domains can be identified and used to group the domains as it is required by the FP-DA approach. To show that not only domain factors are

responsible error factors, the investigations will be extended on the class level. This allows to get more detailed insights of domain differences and provides directions for future research. Subsequently it will be shown how preserving a factor through FP-DA is capable of improving the average and minimum performance in the leave-one-out setting. The positive effects of it will be highlighted through different visualization methods. Limits and requirements of FP-DA will then further be investigated in the experiments on the OpenLORIS object dataset. The setup regarding the chosen hyperparameters and the VGG-16 architecture of all optimization experiments was chosen the same as in Chapter 5.3. All reported numbers are based on 10 runs with randomly chosen parameters θ_d, θ_y and randomly composed batches.

7.3.1 Identification of Error Factors on Domain Level

In the experiments in Chapter 5.3 it was already shown how domain factors that potentially cause errors when being removed can be identified from one-to-one experiments manually. There, the hand that changes between the domains was identified as such a factor. In this section the evaluation should be carried out in a more systematic way that also allows to identify less obvious domain factors. Here, Principal Component Analysis (PCA) was selected as a means. It is applied on the one-to-one experiments matrix of Tab. 5.3. Each row there is treated as a sample with 11 dimensions and the gaps on the diagonal of the matrix are replaced for simplicity with the value 1.0. The total variance of the accuracies in Tab. 5.3 is 0.26. With the first PCA component, 60% of the total variance are already explained. The activation of the first component for the different source domains is shown in Fig. 7.3. There, a clear grouping of the domains into high positive and high negative values can be observed. The domains within each of the two groups are all domains where the same hand was used during recording. Moreover, a visual analysis of the domains reveals that the absolute value of the activation further indicates how clear on one side of the target object the hand is positioned within the image. The activation of domain 1 is for example a special case with its value close to zero. In comparison to the other domains, here the hand mostly shows up in a neutral position below the target object (see also Fig. 5.2).

The second principal component explains 12% of the total variance. As shown in Fig. 7.3, the distribution is rather continuous among the activations compared to the first component's activations. The images of domains that show high positive activations have compared to the

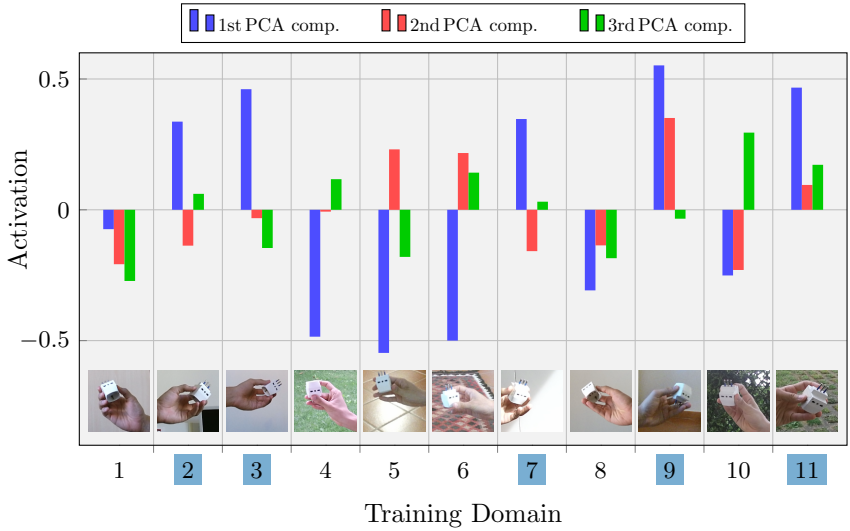


Figure 7.3: PCA analysis of Tab. 5.3: The sign of the 1st component activation is in line with the hand that was used. Positive values refer to left hand domains (marked blue), negative values to right hand domains. Note the low absolute value for domain 1 which indicates the hand presenting the object in a neutral position from the bottom. Positive values in the 2nd component could be related to domains with low contrast, while negative ones to high contrast. The 3rd component might indicate highly textured background by high positive values.

other domains a lower contrast of the objects and the holding hand to the background. This is especially pronounced in the domains 5, 6, 9 and partly 11.

With the third PCA component another 10% of the variance can be explained. Similar to the second component here the distribution of the activations is rather continuous again (see Fig. 7.3). The activation could possibly correlate with the strength of the background texture. The highest values for this component can be observed for the domains 6, 10 and 11, which all have a highly textured background.

The evaluation showed that the PCA components can give hints about possible domain factors. Each component indicated a certain characteristic that is similar in certain domains, but changes compared to other domains. Here the interpretation of the factors in relation to the component activa-

tions was done manually. Another possibility would be to automate this by using a correlation analysis based on the PCA activations and image metadata, like the time of the day, the time of the year, or the GPS location of the recording. Note that in general this approach does not necessarily lead to humanly interpretable factors, since the similarities among domains suggested by the PCA might also be based on a mix of multiple factors. Likewise it would be conceivable to replace PCA with other factorization methods like independent component analysis, that potentially lead to different results which can be more or less meaningful depending on the proposed factorization.

The evaluations proposed here are based on the pre-defined separation of the CORe50 dataset into domains. Therefore this method allows only to detect domain factors that have a strong influence on the transfer performance. Other factors that are individual for each class, like factors related to target objects that have different values between domains, can not be clearly identified like this. However, these can potentially also have influence on the measured transfer performance. Therefore, in the next section PCA will be applied on class level, i. e. here the 10 object categories.

7.3.2 Identification of Error Factors on Class Level

In the previous section the analysis of factors that influence the transfer between domains was only limited to domain factors. However, despite these factors there can also be class specific factors that negatively influence the transfer performance between domains. Such factors can be related for example to target object factors, e. g. the object color, or factors related to the background that are typical for the objects only within a certain domain. An example for the latter could be an object that is placed on a different table in each of the domains.

To identify such factors, the same PCA procedure as in the previous section has been applied here, however, individually for the transfer matrix of each class. The transfer matrix from Tab. 5.3 describes the average over all classes of the CORe50 dataset. Note, the same classification models were used here, while only the evaluation is split up into the accuracies per class. The classes here correspond to the 10 CORe50 object categories {plug adapter, mobile phone, scissors, light bulb, can, glasses, ball, marker, cup, remote control}.

Since in all domains exactly the same object instances are presented, mainly the style of presentation, i. e. how the object is held or moved, or

the state of an object can lead to a poor generalization between domains. Here, a factor in context of hand held objects could be the position at which the object is held, as e. g. the bulb at the glass cover or the socket. The object state could be for example for the scissors described by the factor *'scissor state'* with the values {open, closed}. Exemplary for all classes, the two classes *plug adapter* and *glasses* were picked for the investigations here. While for the first no obvious object states exist, the second comes with the two object states {folded, unfolded}.

In principle when naively assuming that the domain factors are the only reason for a performance drop, one could expect that all sub-transfer matrices show similar classification accuracies, independent of the actual object class. However, the results show that this is not the case. Comparing the transfer matrix for the *plug adapter* class in Tab. 7.1 to the transfer matrix where all classes are represented (Tab. 5.3), similar patterns of weak performance between domains where different hands were used can be observed. Here, however, those transfers perform clearly worse, while some of the transfers between same hand domains perform far above average. Very low performances can for example be observed for the transfers from domain 5 and 6 to the domains 2, 3, 9, and 11. Similar observations can be found in many transfers with domain 9 as the source domain, where for target domain 8 the lowest transfer performance can be observed. Investigating the image samples of domain 9 and domain 8 reveals that one of the plug adapter instances is consistently held in another way. Exemplary samples for the instances from these domains are given in Fig. 7.5. Besides the changing hand, this different way of holding an object is likely to occlude certain features and therefore reduces the transfer performance even further.

Compared to the general average transfer accuracy over all domains for all classes (Tab. 5.3), where 59% accuracy was achieved, only 50% is reached here. This is most likely due to more complex holistic features that need to be learned for this class. In contrast, for the simpler 'cup' class, the average accuracy is at 72%.

Table 7.1: One-to-one transfer experiments for the plug adapter class only.

Source Domain \ Target Domain (only plug adapter class)	1	2	3	4	5	6	7	8	9	10	11	*Avg*	*Min*
1	-	0.53	0.32	0.75	0.85	0.58	0.39	0.98	0.38	0.52	0.22	0.55	0.22
2	0.50	-	0.76	0.51	0.37	0.49	0.72	0.35	0.79	0.45	0.66	0.56	0.35
3	0.54	0.86	-	0.31	0.26	0.24	0.86	0.48	0.82	0.31	0.66	0.53	0.24
4	0.86	0.27	0.21	-	0.83	0.67	0.34	0.77	0.15	0.82	0.52	0.54	0.15
5	0.81	0.05	0.11	0.75	-	0.73	0.23	0.71	0.07	0.63	0.05	0.41	0.05
6	0.66	0.05	0.07	0.63	0.83	-	0.20	0.63	0.03	0.32	0.04	0.35	0.03
7	0.73	0.73	0.74	0.50	0.81	0.69	-	0.71	0.75	0.54	0.78	0.70	**0.50**
8	0.99	0.41	0.20	0.83	0.83	0.80	0.36	-	0.31	0.76	0.19	0.57	0.19
9	0.09	0.72	0.71	0.07	0.05	0.04	0.59	0.02	-	0.24	0.71	0.32	0.02
10	0.93	0.63	0.56	0.87	0.94	0.74	0.60	0.90	0.68	-	0.34	0.72	0.34
11	0.11	0.57	0.67	0.08	0.04	0.07	0.61	0.10	0.65	0.18	-	0.31	0.04
Avg	0.62	0.48	0.43	0.53	0.58	0.51	0.49	0.56	0.46	0.48	0.42		
Max	**0.99**	0.86	0.76	0.87	0.94	0.80	0.86	0.98	0.82	0.82	0.78		

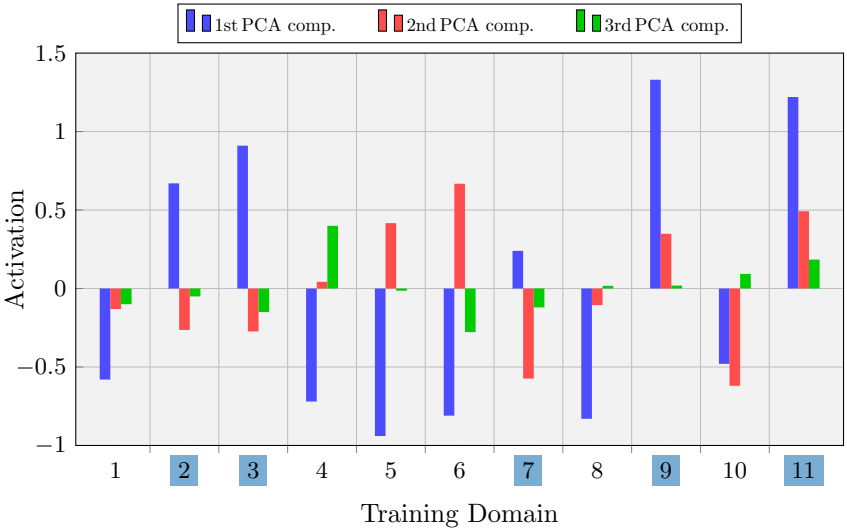


Figure 7.4: PCA result for the analysis of the one-to-one domain transfers of the plug adapter class in Tab. 7.1.

Figure 7.5: Samples of plug adapter instances of domain 9 and domain 8. Instance 1 in domain 8 is held throughout the entire recording stream at the blue part.

As expected, since a similar pattern of weak performance could be observed, the application of PCA (see Fig. 7.4) groups the domains based on the sign of the first component in the same way as when all classes are involved. The absolute values, however, differ more significantly. Overall, the first component explains 73 % of the total variance. The second component explains 17 % of the variance and also results in the same grouping as before, different absolute values can be observed here as well. The third component suggests a different grouping than before, but can be neglected due to its low importance here, which is expressed by explaining only 3 % of the total variance.

All in all, apart from the very low and very high accuracies on single transfers, the overall results of only the plug adapter class are similar to the results where all classes were considered. The patterns of weak performance caused by the changing hand can be observed here as well, indicating that the domain factors here are the factors that influence the generalization between domains for the plug adapter class the most.

For the transfers of the 'glasses' class, given in Tab. 7.2, the patterns of low classification accuracy are not as pronounced as in the transfer matrix for all classes anymore. The most prominent low performances can be observed again with domain 9 as the source domain. Obviously the transfer based on the training samples for the glasses to any other domain results for this source domain in a poor accuracy. This is also reflected in the PCA analysis in Fig. 7.6. Here, the first component explains 46 % of the variance. The grouping of the domains by the hand is not given anymore, instead a clear separation of domain 9 from the others can be observed. The second and the third component explain 32 % and 9 % respectively, where neither of the two suggests a hand based grouping indicated by the sign of the activity.

Table 7.2: One-to-one transfer experiments for the glasses class only.

Source Domain \ Target Domain (only glasses class)	1	2	3	4	5	6	7	8	9	10	11	*Avg*	*Min*
1	-	0.91	0.89	0.53	0.80	0.06	0.39	0.61	0.32	0.19	0.10	0.48	0.06
2	0.89	-	0.94	0.46	0.81	0.66	0.86	0.70	0.66	0.38	0.65	0.70	0.38
3	0.92	0.91	-	0.52	0.75	0.16	0.74	0.66	0.51	0.24	0.31	0.57	0.16
4	0.82	0.80	0.90	-	0.70	0.64	0.79	0.66	0.73	0.63	0.64	0.73	**0.63**
5	0.73	0.65	0.58	0.64	-	0.44	0.45	0.65	0.63	0.36	0.51	0.56	0.36
6	0.59	0.43	0.42	0.66	0.58	-	0.53	0.48	0.39	0.65	0.79	0.55	0.39
7	0.85	0.81	0.85	0.74	0.95	0.92	-	0.79	0.53	0.73	0.83	0.79	0.53
8	0.81	0.84	0.73	0.60	0.87	0.51	0.72	-	0.47	0.59	0.63	0.68	0.47
9	0.15	0.34	0.20	0.11	0.14	0.09	0.31	0.26	-	0.12	0.43	0.21	0.09
10	0.49	0.50	0.42	0.87	0.73	0.57	0.56	0.85	0.54	-	0.64	0.69	0.42
11	0.67	0.68	0.86	0.78	0.66	0.52	0.78	0.73	0.67	0.67	-	0.70	0.52
Avg	0.69	0.69	0.68	0.59	0.70	0.46	0.61	0.64	0.54	0.46	0.55		
Max	0.92	0.91	0.94	0.87	**0.95**	0.92	0.86	0.85	0.73	0.73	0.83		

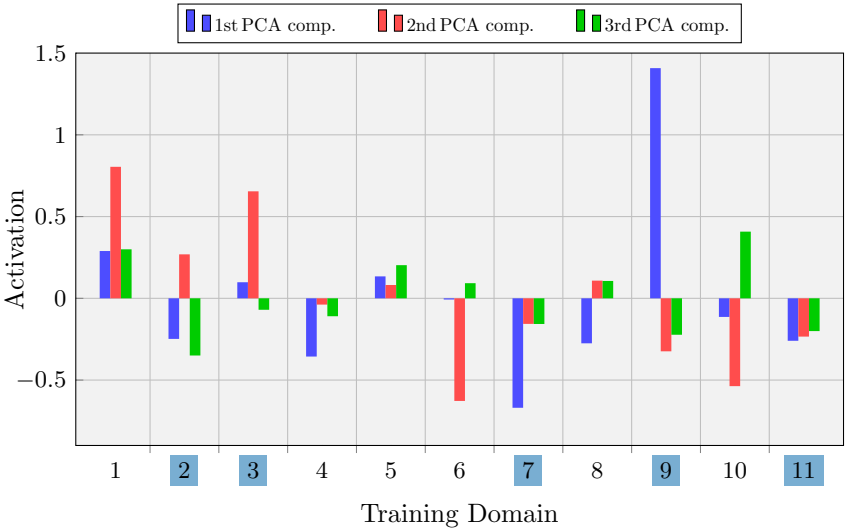


Figure 7.6: PCA result for the analysis of the one-to-one domain transfers of the glasses class in Tab. 7.2.

A detailed investigation of the poor transfer performance when training on domain 9 revealed that the image samples from this domain of all five glasses instances showed the glasses in a folded state. In the other domains those are mostly unfolded or at least partially unfolded. Exemplary image samples of the glasses instances of domain 9 and domain 3 are given in Fig. 7.7. Interestingly the transfer from unfolded glasses to folded glasses performs better, which can be seen in the higher average accuracy of the column where domain 9 is the target domain.

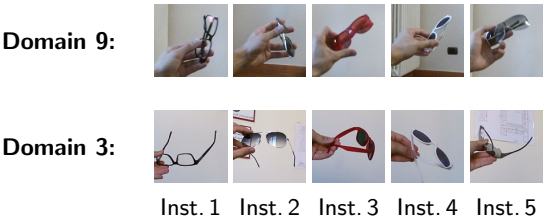


Figure 7.7: Samples of glasses instances of domain 9 and domain 3. In domain 9 all glasses are folded throughout the recording stream, while in domain 3 they are mostly at least partially unfolded.

A similar occurrence was found for the scissors class. Here, the reason was not the state of the scissors, but rather how it was presented by the human. In this specific case most of the domains show the scissor instances held by the blade of the scissors, while in domain 6 three of the instances were held exclusively by the handle.

The conducted investigations on class level demonstrated that not only domain factors are responsible for a drop in performance, but also class specific factors of a certain domain. It was shown that such factors can be related to a state of a certain object that is consistent in one domain, but changes across domains. Furthermore, other factors, like the way of presenting a specific object, which ultimately influences which features can be detected, can have an effect on the transferability of models between domains. Adversarial DA with a domain classifier, like in [22], pre-dominantly removes domain factors, since those are the most significant ones to discriminate the domains. Generally other factors like the class specific factors presented in this section might also be removed, since those can also help to discriminate the domains. However, if such factors are removed depends on the capabilities of the domain classifier. Removing such would require to learn multiple comprehensive feature representations, possibly individ-

ual features for each domain-class combination. The introduced FP-DA approach, which is based on a grouping of the domains by the value of a chosen factor therefore is primarily designed to only preserve domain factors. Class-level factors would be preserved as well, if the domain grouping is in line with the values of those. Generally, in this thesis, more elaborate versions of FP-DA that also consider a factor preservation on class level were not considered, but could point out directions for future work.

7.3.3 Quantitative Evaluations of FP-DA on CORe50

This section will demonstrate how FP-DA is capable of reducing negative transfer by preserving a chosen factor in the leave-one-out experiments of the CORe50 domains. The previous one-to-one evaluations with and without DA in Chapter 5.3 showed that the hand could be a factor that causes negative transfer. Therefore, preserving it and consequently grouping the domains in FP-DA by the used hand seems most promising. This grouping was also suggested by the sign of the first PCA component activation of the domain-level transfer analysis in Section 7.3.1 as a prominent cause of error. Two more groupings, based on the sign of the second and the third PCA component's activations, will also be evaluated here. Generally, to benefit most from FP-DA it is necessary that the factor that is preserved through the grouping is task-informative and that its removal would cause a large error. Note that FP-DA switches off competition between certain domains, which leads to an overall reduced influence of the classifier path on the shared feature extractor. This alone could lead already to reduced negative transfer. To evaluate the influence of this effect, additionally three random factors were preserved, which was achieved through groupings of 5 and 6 random domains per group respectively. The results for these leave-one-out experiments, together with the results from Chapter 5.3 as comparison, are shown in Fig. 7.8.

Using FP-DA and grouping the domains based on the first PCA component, where the factor *'hand'* is preserved, clearly shows the overall highest average and highest minimum classification accuracy among all experiments. This outperforms significantly the original approach of [22] on the given CORe50 domains. Compared to the approach of [22] with unsupervised data, the quantitative improvement is for the average accuracy at 1.9 percentage points and for the minimum performance at 1.1 percentage points. The individual results show that in most of the constellations the negative transfer caused by DA was clearly reduced through FP-DA with a hand based domain grouping. Comparing FP-DA with this grouping to

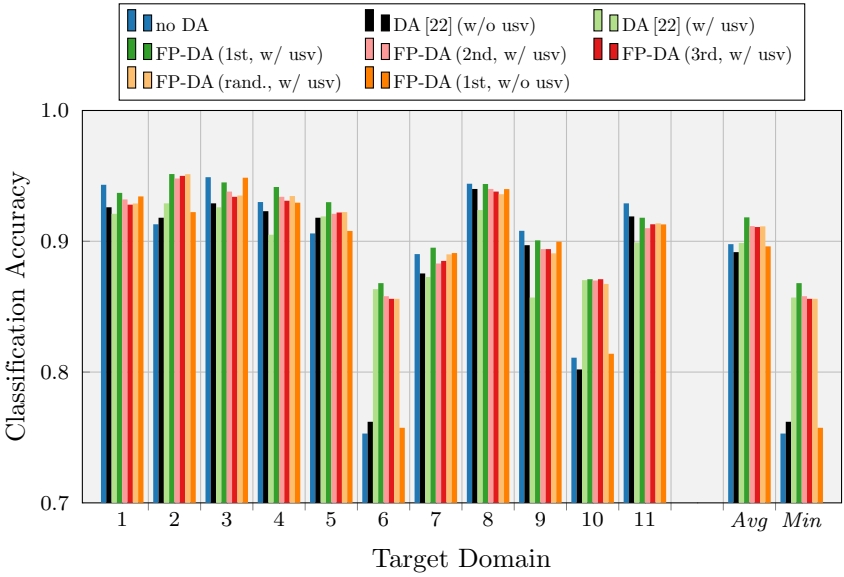


Figure 7.8: Leave-one-domain-out DA on the CORe50 domains using the proposed FP-DA approach. '1st', '2nd', and '3rd' describe the grouping based on the three PCA components shown in Fig. 7.3, 'rand.' a random permutation. FP-DA with the domain grouping based on the used hand ('1st') and usv data shows the highest average and highest minimum performance among all constellations.

the experiments without DA, it shows here in many cases also an improved performance. Domain 4 and domain 7 are examples for this. However, not in all cases this observation is given. Examples here are domain 1 and domain 3, where negative transfer was reduced, but no application of DA still results in the better performance. All in all, here the expectations of FP-DA to reduce negative transfer by preserving a task-informative factor were fulfilled and its effectiveness was shown.

Factors other than the hand are preserved when grouping the domains based on the second or the third PCA component. FP-DA based on both groupings achieves only a reduced gain of approximately 1.2 percentage points on average compared to the experiments with [22]. The effect of reduced negative transfer, that nevertheless still occurs here, can be explained by the results for the random grouping of domains, i. e. neglecting completely the PCA step. For these experiments a similar gain can be

observed. This indicates that in general a reduced competition between domains and thus influence of the domain classifier is beneficial. Further, this shows that there is no particular benefit of preserving the factors suggested by the second or the third PCA component, i. e. the suggested factors are not semantically related to task-informative factors.

Note, a strategy to face the general problem of the hand showing up on the left or the right side of a target object in the domains could also be to use artificial data augmentation methods. Here, a random horizontal and vertical flipping of input images during training could force the network architecture generally to be less dependent on local features that are not part of the target object. However, this is not a method to deal with domain factors in general and could also have a strong negative influence to objects that change their class membership when being flipped horizontally.

7.3.4 Qualitative Evaluations of FP-DA on CORe50

For a qualitative evaluation of the FP-DA approach applied on the CORe50 domains, two common visualization methods were chosen. At first the sample embedding in feature space was investigated. For this the frequently used method of t-distributed stochastic neighbor embedding (t-SNE) [102] was used to visualize the samples mapped from the high-dimensional feature space to a human interpretable 2D feature space. As a second visualization method Grad-CAM [88] was chosen, which allows to determine the areas of sample images that had most influence on the output classification result.

t-SNE Visualizations

A common method to visualize how samples are embedded in feature space is the t-SNE method from [102]. It uses a nonlinear technique that reduces the dimensionality of high-dimensional data to a dimension that is easier to interpret by humans, like 2D or 3D. It is designed in such a way, that data that is close in the higher-dimensional space is also close in the lower dimensional space. The method is often used in context of deep neural networks which allows to analyze how data samples are clustered based on the embedding of a chosen feature layer.

For this thesis a mapping of the high-dimensional data to the 2D feature space was chosen, which allows to visualize the samples in 2D scatter plots that are easy to interpret. The three cases, without DA, DA [22] with unsupervised data, and FP-DA with a grouping based on the first PCA component were investigated. For each case a single trained multi-source

model was chosen from a run where domain 9 represented the target domain. The high-dimensional embeddings of two layers were investigated here separately. The first was the output of the last layer of the shared feature extractor after the max-pooling operation, which has a dimensionality of 25,088. The second was the second fully-connected layer of the label classifier with a dimensionality of 1000.

For the visualizations the amount of data samples was limited to 2000 samples, 1000 randomly chosen samples from the target domain test data and 1000 samples from all source domains. The resulting t-SNE embeddings were visualized with different annotations, in specific source domains vs. target domain, left-hand domains vs. right-hand domains, and all classes individually of domain 9. Other visualizations, where correct and falsely classified samples, and each domain individually are marked are given in Appendix A.2.

When no DA is applied, the feature embedding of the samples from the source domains after the last layer of the shared feature extractor should still be easily separable from the samples of the target domain. The reason for this is that samples from the target domain most likely are out of the scope of the known value distribution of the network and therefore would map to different locations in feature space. For the source domains, generally class based clusters should already be slightly visible in this feature layer, but not yet clearly separated as it is expected in one of the subsequent fully-connected layers, where features are more class specific.

The upper t-SNE plot in Fig. 7.9 (left column) shows for the case where no DA is applied that these expectations are fulfilled. Many samples of the target domain cluster with samples from the source domains, which was expected, since even without DA here an accuracy of about 90% was achieved. However, a separation of source domains and target domain samples is clearly visible, where source domain samples are more frequently found in the lower part of the plot. When DA [22] is applied the desired effect occurs that a differentiation of source domains and target domain is not easily possible anymore. Interestingly, clusters for the ten different classes are now pronounced more clearly as well. With FP-DA, the clusters are less separated as with DA, but still more than without DA. This was expected since FP-DA reduces the overall influence of the domain classifier path. Nevertheless, a separation of source and target domain samples is not given, which shows that the main goal of DA is still achieved.

Looking at the same data with right-hand and left-hand domains labeled in Fig. 7.9 (right column), the result without DA shows as expected that

the used hand can still be distinguished. With DA, this separability is clearly reduced. Here, smaller clusters randomly distributed in space are more dominating, while left- and right- hand clusters mostly overlap. When FP-DA is applied, such clusters persist, however, the overlap of left- and right-hand clusters is reduced, which could indicate the preservation of the information about the hand while simultaneously clustering by the object classes. Note that right-hand domain samples are generally underrepresented in this plot, since half of the samples were chosen from the target domain, which is a left-hand domain. Nevertheless, the discussed observations are not affected by this.

Fig. 7.10 shows a t-SNE mapping for only samples from domain 9, where the ten class labels of the samples have been visualized. Without DA a lot of samples with different ground-truth labels are close together in the center. A separation of these thus is likely to be more complex in a subsequent layer. With DA, the clusters are better separated, however, on the top right a similar area of mixed classes can be observed. When FP-DA is applied, there is still a cluster of samples with mixed ground-truth labels in the center, however, within this cluster they seem to be more cleanly separated. Furthermore, the overall impression compared to the other approaches is that FP-DA provides here the cleanest separation of the samples, despite the fact that clusters seem to be closer together than with DA [22].

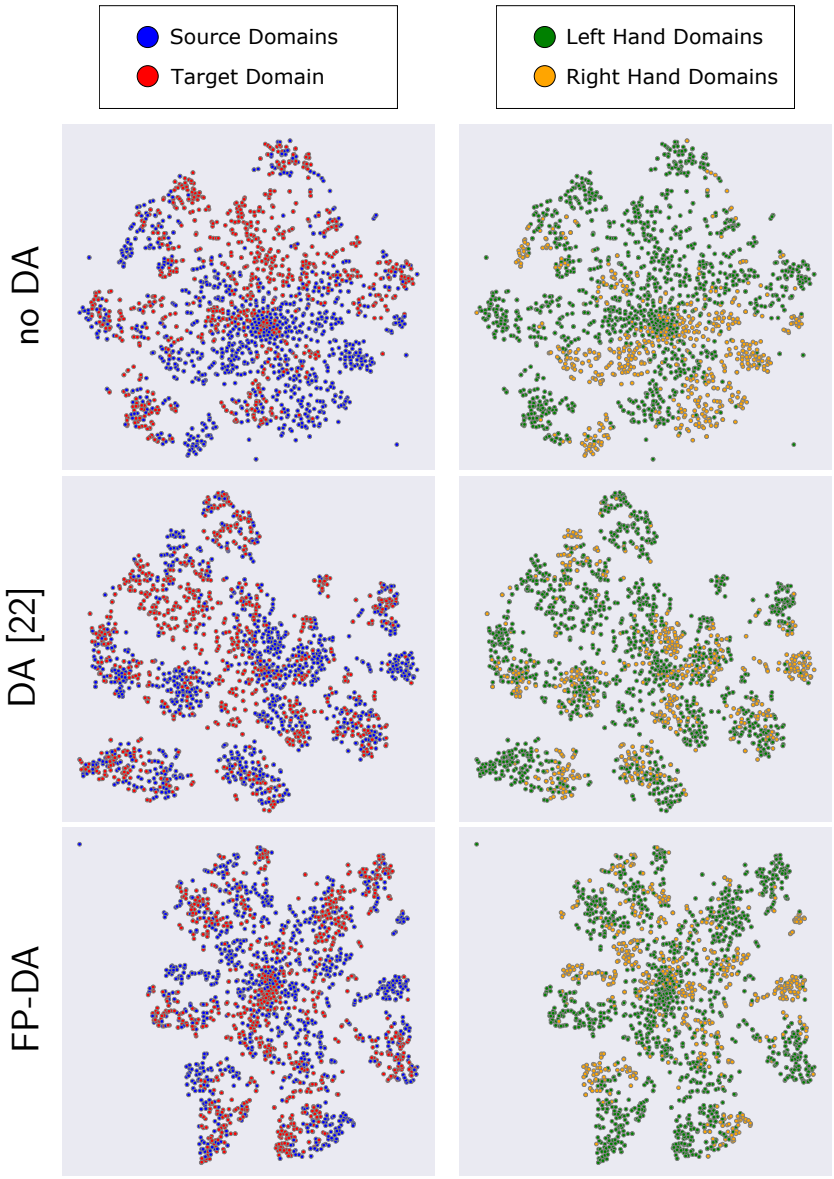


Figure 7.9: t-SNE plots for the last layer of the shared feature extractor.

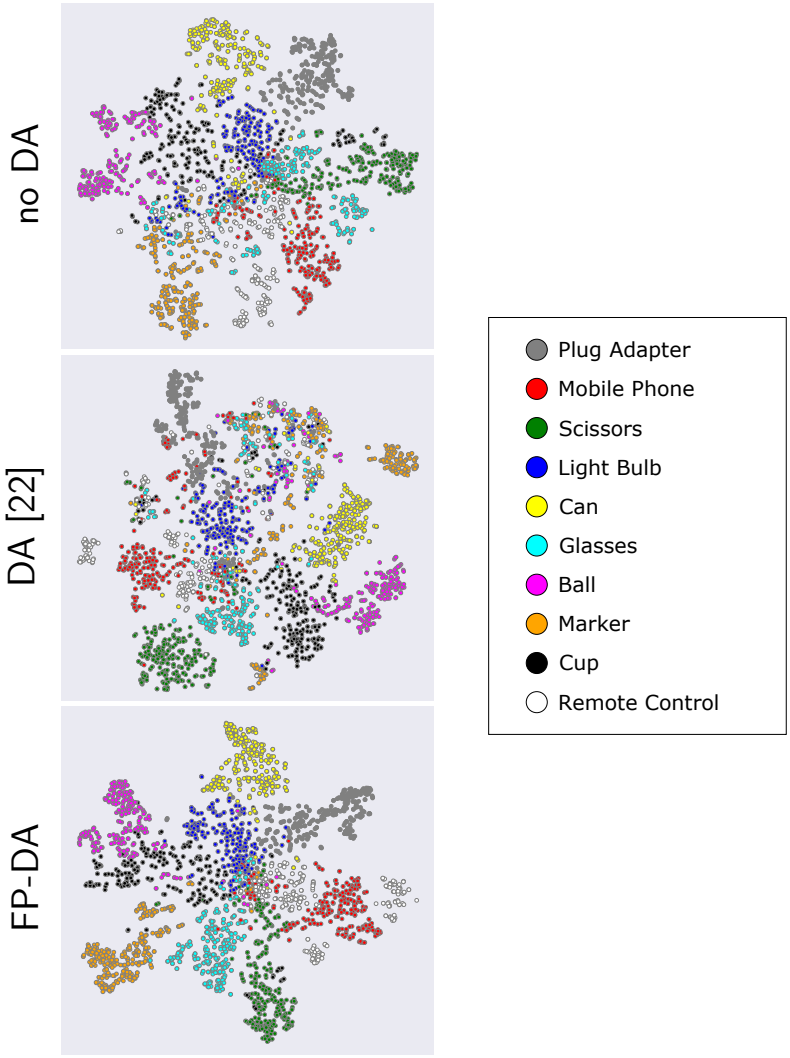


Figure 7.10: t-SNE plots for samples from domain 9, labeled by the ground-truth class. The embeddings are from the last layer of the shared feature extractor.

In the fully-connected layers feature embeddings are usually more class specific, thus better separable clusters should be observable in the t-SNE plots. Nevertheless, samples from uninvolved domains that differ significantly from the training data are expected to still not be assignable to a certain cluster. The t-SNE plots for the second fully-connected layer shown in Fig. 7.11 (left column) confirm these assumptions. Without DA most of the samples from the target domain can be assigned to one of the now very clearly separated clusters, which is in line with the achieved accuracy of about 90% on the target domain data. Within the clusters source and target domains are mostly still separable. As expected, in the center there are some target domain samples that can not be assigned to the clusters. When DA is applied those unassignable samples become less and within the clusters the samples are more evenly distributed. With FP-DA the even distribution within the clusters seems to be kept, but the unassignable target domain samples become more, showing again the decreased influence of the domain classifier path. The results with the used hand being labeled in Fig. 7.11 (right column) show for this layer similar effects as for the last layer of the shared feature extractor. Here, left- and right-hand domains are least separable in the case of the normal DA approach.

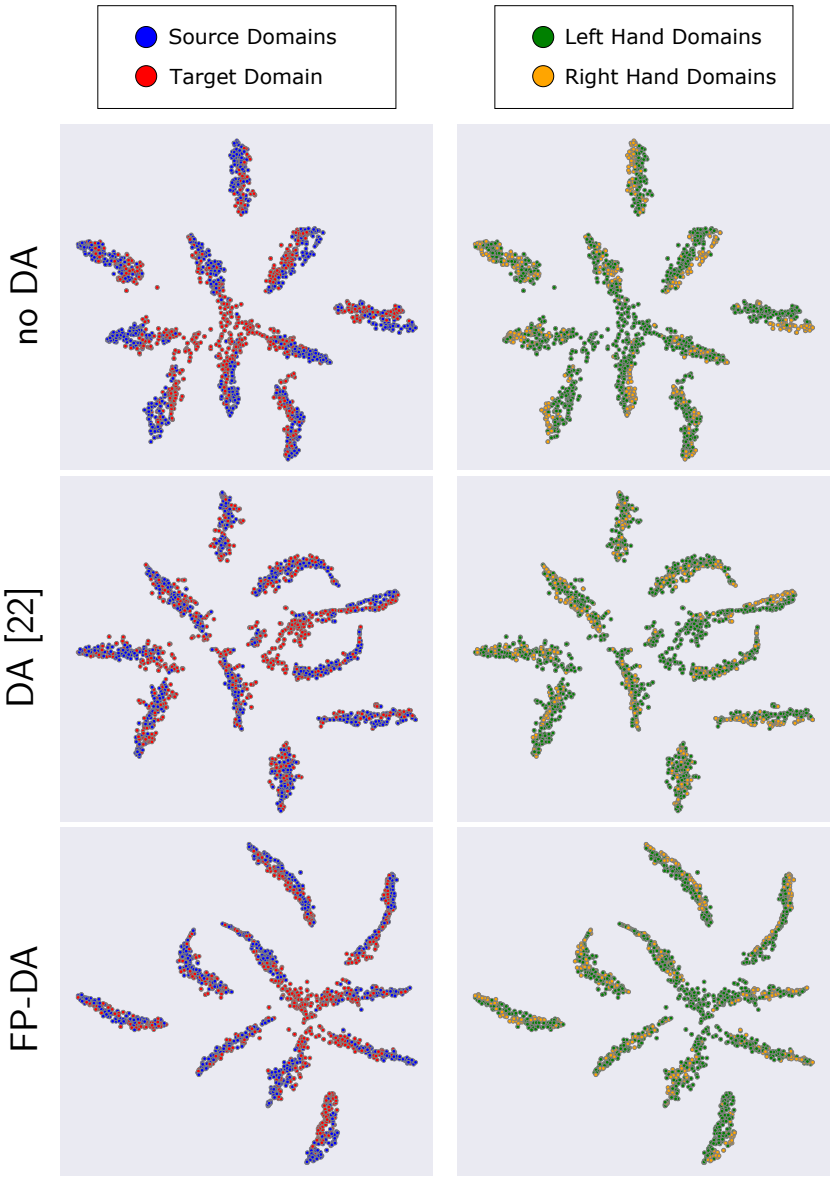
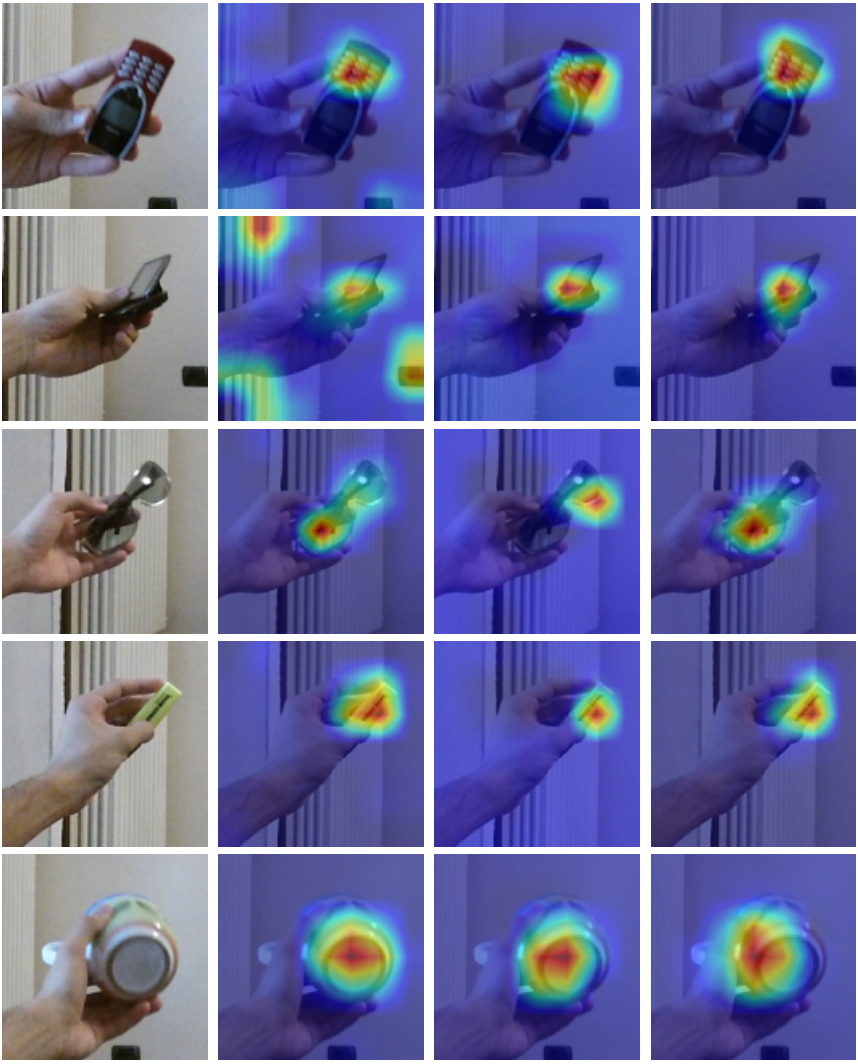


Figure 7.11: t-SNE plots for the second fully-connected layer.

Grad-CAM Visualizations

An additional qualitative evaluation to show the effect of FP-DA was to visualize which areas of an image contribute most to the activation of the output neuron of the target class. This was realized by the method of Gradient-weighted Class Activation Mapping (Grad-CAM) from [88]. To highlight the area of an image that contributes most to the target class, the image is in a first step regularly forwarded through the trained network, whereby each layer's feature activation maps are computed. Then, only the gradient of the output neuron of the target class is backpropagated with respect to the activation maps of a chosen layer, resulting in feature gradient maps. Those gradient maps are averaged and represent a weight for each corresponding feature activation map of the chosen layer. In a next step the feature activation maps are summed up, while each is weighted by the previously determined weight. The resulting summed activation map is then passed through a rectified linear unit, showing only the features that positively contributed to the activation of the target class. The rectified map that highlights specific areas is then rescaled to the original input image size, allowing clearer visual interpretations of those highlighted areas.

For the experiments with the CORe50 data in specific, it should be investigated whether there is visual evidence that the factor *'hand'* is preserved when FP-DA is applied based on a domain grouping by the used hand. For the evaluations here, again a trained model for each of the cases, without DA, DA [22], and FP-DA was chosen, with domain 9 as the target domain. The investigated example images were taken from test data of domain 9. The Grad-CAM visualizations are based on the feature maps after the last max-pooling layer of the shared feature extractor. Fig. 7.12 shows the results for selected images. The images show the general effect that the application of DA puts more focus on the actual target object and removes activations from the background that can be observed in cases where no DA is applied. This is especially pronounced for the mobile phone in the second row. Overall, the highlighted area is also clearly shrunk and shifted away from the hand in comparison to the case where no DA is applied. In contrast, with FP-DA, this area is less shrunk and activations closer to the hand can be observed, while at the same time almost no activation on the background is visible. This indicates that the *'hand'* is preserved and that target object related features close to the hand can be considered for classification.



(a) Input **(b)** no DA **(c)** DA [22] **(d)** FP-DA

Figure 7.12: Grad-CAM visualization for selected images of target domain 9. DA [22] removes background activation (especially row 2), but also shrinks the activation on the object (especially row 4) and shifts it away from the hand (especially row 3). With FP-DA, this area is less shrunk and activations closer to the hand can be observed.

7.3.5 FP-DA on OpenLORIS Object Dataset

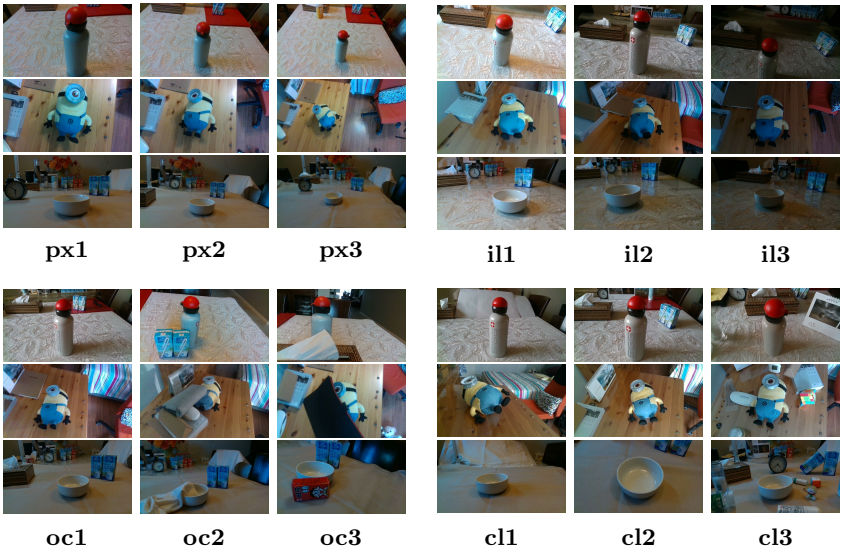


Figure 7.13: Example images of three objects from the 12 domains of the OpenLORIS object dataset. In each domain one of the factors *'object pixelsize'* (px), *'illumination'* (il), *'occlusion'* (oc), and *'clutter'* (cl) has a fixed value, according to one of three levels of difficulty. Level 1 is the easiest, while level 3 is the most difficult level.

To further evaluate the capabilities of the FP-DA approach it was applied on the OpenLORIS object dataset [90]. Similar to CORe50, this dataset is also mostly used in the robotics community when the robustness of models to certain environmental changes should be investigated. The dataset consists of 69 objects that were presented in 12 domains. During acquisition of each domain, one of the four factors *'object pixelsize'* (px), *'illumination'* (il), *'occlusion'* (oc), and *'clutter'* (cl) was set to one of three difficulty values {level1, level2, level3}, while the other three factors are undefined. For px the size of the shown object decreases from level1 to level3 (px1 to px3), for il the illumination reduces, for oc the objects become more occluded, and for cl the background gets more cluttered. Exemplary images of three objects in all domains are given in Fig. 7.13. Despite the fact that the dataset comes with the pre-defined factors, the effectiveness of FP-DA might be strongly limited in preserving one of these factors since

the combinations of the factors are not represented in the dataset. A clear preservation of the factor *'clutter'* would for instance require to group all domains based on the values of this factor. Here, however, images from the px, il, and oc domains are undefined for the factor *'clutter'*. Nevertheless, similar steps as on CORe50 still make sense here, since also other factors worth preserving could be detected through the one-to-one experiments with the related PCA analysis.

The results for the PCA analysis of the one-to-one experiments for the OpenLORIS domains are shown in the top plot of Fig. 7.14. The first PCA component explains 44% of the total variance and clearly separates the difficult level3 domains together with oc2 from the other domains. The second component explains 17% of the variance and slightly suggests a grouping of {px, oc} and {il, cl}. The third component explains 12% of the variance and separates px3 and cl3 from the remaining domains.

The corresponding leave-one-out experiments are shown in the lower plot of Fig. 7.14. For DA [22] the results show similar patterns as for the CORe50 dataset. The difficult domains, here the level3 domains, come with the largest performance increase through DA, while for the easier domains like px2, il1, and oc1 slight negative transfer can be observed. The application of FP-DA based on the first PCA component leads here only to a minimal increase of the average performance. The minimum performance is even decreased. The other groupings, based on the other PCA components and random grouping, are only slightly worse on average and slightly better in the minimum performance. All in all, FP-DA can not outperform DA [22] in this case. It is assumed that a possible reason for this result is the lack of missing combinations of factors in the data.

To investigate whether this is the case, the overall complexity of the setting was reduced by limiting the involved domains to only two of the predefined factors and evaluating all possible pairs of two factors. Thus, there are only 6 domains instead of 12. The results for the corresponding PCA analysis and the multi-source experiments showed very similar patterns as in the previous experiments. FP-DA could not outperform the reference DA method [22] here either. The quantitative results for these experiments are given in the Appendix A.3.

To investigate whether FP-DA would benefit from a combination of factors in the data, i. e. there must be at least two clearly defined factors per domain, the original OpenLORIS domains were overlaid with an additional manually introduced artificial factor *'brightness'*. The possible values of this factor are {normal, dark}. The original OpenLORIS domains that were used before have the value 'normal' for this factor, while new domains

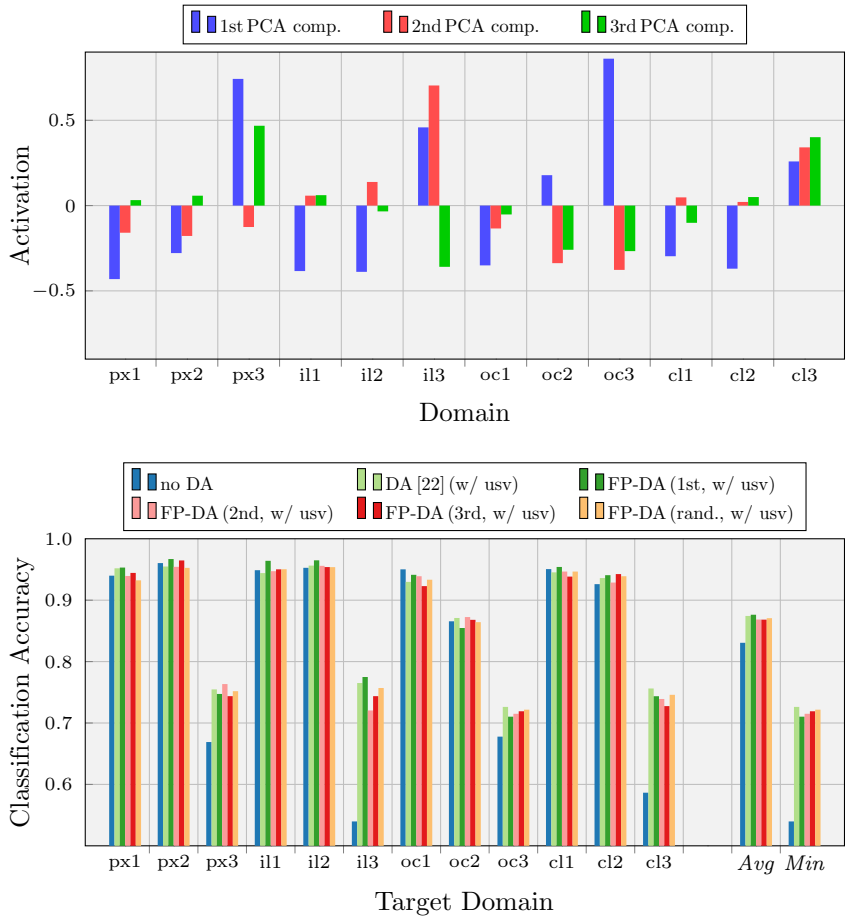


Figure 7.14: PCA and FP-DA on the OpenLORIS dataset. *Top*: PCA identifies the most difficult domains plus oc2 in the first component. The second component suggests the groups {px, oc} and {il, cl}, while in the third component only px3 and cl3 are clearly separated from il3, oc2, and oc3. *Bottom*: DA causes negative transfer for most of the easy domains and positive transfer for the difficult ones. FP-DA increases the performance only slightly in *Avg* and decreases it in *Min*.

Figure 7.15: Exemplary domains from the adapted OpenLORIS dataset. Domains indicated with cl*D were generated manually from the corresponding OpenLORIS domains cl* by darkening the images. Thus, each image has two factor labels, one label for *'clutter'* $\in$ {level1, level2, level3} and one for *'brightness'* $\in$ {dark, normal}.

are generated with the factor value 'dark'. The new domains are generated by multiplying the value channel of all images of an original OpenLORIS domain in the HSV space by 0.3. With this modification each image of the extended dataset has now two labeled factors. The following experiments with FP-DA were carried out such that the new factor *'brightness'* was individually tested in combination with either px, oc, or cl. Exemplary images of the domains that were used for the investigations with cl are shown in Fig. 7.15. Note, the combination of the newly introduced factor with il was omitted, since *'brightness'* and *'illumination'* are of similar nature.

Fig. 7.16 shows the PCA and the leave-one-domain-out results for the investigated domain constellations. The latter are reduced here to the relevant measures of average and minimum performance. The PCA analyses show that in the first component, consistently the 'dark' and 'normal' domains are clearly separated from each other by the sign of the activation. This is an expected result, since the conducted one-to-one experiments showed that a change in the factor *'brightness'* is a more challenging transfer for the model than a change in the difficulty level of the other involved factor. The second PCA component separates for all constellations the level3 domains from the remaining domains since the difficulty level still has major influence on the transfer performance. This is also reflected in the average explained variance for the second component which is at approximately 25 %. For the third component no common patterns among all three investigated domain combinations can be observed. The average

results of the leave-one-out experiments show that here DA [22] mostly leads to slight negative transfer. This also applies for the minimum performance, except on px where a slight increase can be observed. When FP-DA is applied based on a grouping of the first PCA component an even further decrease in comparison to DA [22] occurs. However, using a grouping based on the second component leads to a strong gain in performance and clearly outperforms most of the other experiments. Taking the third component, only the result for oc is similarly good, while on px and cl it is clearly worse.

Analyzing the used groupings in more detail, it becomes obvious, that the first PCA component suggests to preserve the factor *'brightness'*, while the other involved factor, i. e. *'object pixelsize'*, *'occlusion'*, or *'clutter'* is removed. This can be harmful when considering that the latter factors are for the OpenLORIS dataset generally task-informative as well, since the shown target objects constantly show up with individual context objects that vary across classes within a domain and across domains in general. The removal of those therefore complicates the classification task. Note, the described situation also applies for px3 since with the greater distance between camera and object, also more task-informative background noise is visible in the image (see Fig. 7.13). In contrast, the grouping suggested by the second PCA component removes mainly the factor *'brightness'*, since 'normal' and 'dark' domains are represented in each group of domains. Through the separation of the level3 domains in an individual group, this also preserves most of the task-informative background noise and therefore leads to a clear superiority of FP-DA.

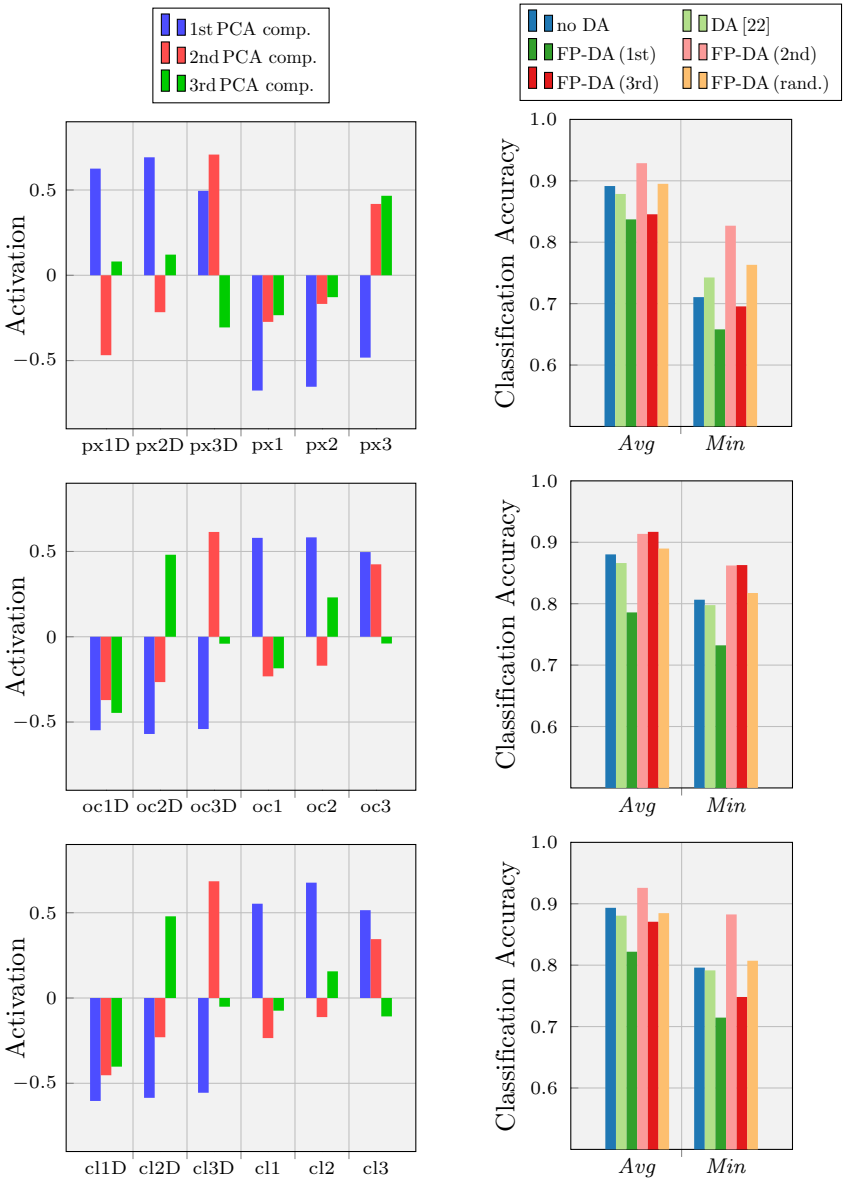


Figure 7.16: PCA and FP-DA on subsets of OpenLORIS domains with artificially darkened domains (*D). All DA experiments use unsupervised data.

7.4 Summary

As discussed in the related work chapter, most adversarial DA approaches that rely on additional domain classifiers, like the approach of [22], mainly target to remove all domain factors embedded in the feature representation of the feature extractor. However, this can be harmful if the domain factors are not only domain-informative but also task-informative or are visually close to task-informative factors. In the baseline experiments of Chapter 5.3 this effect of negative transfer was observed using the adversarial DA approach of [22] in multi-source experiments on the CORe50 dataset. Subsequent evaluations on one-to-one domain transfer experiments with and without DA showed that the domain factor *'hand'* leads generally to a poor transfer performance if it changes its value between domains and that the removal of it through DA causes an even weaker performance. A hypothesis for this weaker performance was that through the removal of the factor *'hand'* also the task-informative factor *'hand posture'* was removed. The assumption was that this also caused the negative transfer observed in the multi-source experiments.

To preserve a chosen factor and thus to reduce negative transfer, FP-DA was proposed in this chapter. FP-DA is a novel adversarial DA method based on [22] which uses during training a domain grouping based on the value of the factor to preserve and switches off competition in the domain classifier between domains from different groups. With FP-DA it was possible to achieve the highest average and minimum performance on the CORe50 domains. The effectiveness was further substantiated through qualitative analyses with t-SNE and Grad-CAM visualizations. Additional experiments on the OpenLORIS object dataset showed that insufficient combinations of factor labels are adverse for FP-DA. Overlaying an additional factor *'brightness'* showed that FP-DA works best here as well.

It was further shown in this chapter that a meaningful factor that is worth to be preserved can either be derived manually or with the help of a PCA analysis on one-to-one transfer experiments. The PCA evaluations showed that one of the first PCA components can be a beneficial factor to be preserved. Furthermore, additional PCA investigations on class level showed that there is increased potential to reduce negative transfer when considering not only the preservation of domain factors, but also a preservation of class specific factors. In the current design, however, FP-DA does not consider such class-level factors. The extension of FP-DA to such factors could therefore be a possible direction for future work.

8 Conclusion

This chapter summarizes the main contents of this thesis which are composed of the Effects of Domain Awareness, the Domain Mixture scenario, and the Factor-Preserving DA method, that all have been evaluated in context of the newly introduced factor theory. Subsequently future research directions for the three presented main topics will be discussed and basic ideas for an implementation of a DA-based 24/7 learning system with multiple camera entities in a smart environment will be presented.

8.1 Summary

This thesis discussed and evaluated different DA approaches and DA constellations in context of visual factors. In this work, the appearance of an image is explained by a mixture of multiple factors that have different values and describe a scene. In contrast to current domain transfer literature, using such factors allows to explain effects like negative transfer more precisely than the unspecific notion of differing data distributions across domains. Furthermore, the better understanding of these effects provides the opportunity to develop new DA approaches that explicitly consider such factors and thus improve the domain transfer capabilities of deep classification models. The introduced factor theory can on the one hand help to interpret and explain the characteristics of image datasets from literature, as well as on the other hand help to interpret the behavior and effects of recent DA approaches from literature. Both aspects were shown in this thesis.

Further, it was shown that general habits of deep classification models, e. g. the type of features that are learned, can more easily be explained by using such factors. The generalization challenges 'new factor values' and 'new combinations of known factors' that occur in potential application domains in context of DA have been discussed here with a particular focus on.

As one of the main parts of this thesis the effects of domain awareness during training and test of a classification model were investigated at

the example of a road segmentation task. For the evaluations, the simple normalization technique of RGB mean subtraction was used as an exemplary method for parametric DA. The results showed that the unawareness about domains during training and test can cause negative transfer if the test domain is normalized individually and was already one of multiple training domains but not normalized there individually. A general recommendation derived from the experiments is to always try to treat the same domains consistently during training and test to achieve the best classification performance.

The second main contribution of this thesis was the investigation of the Domain Mixture scenario, which describes a DA scenario that is usually neglected in current literature. In this scenario the training dataset is composed of multiple domains, of which, however, no domain is completely covered by supervised samples for all classes. The investigations showed the tendency of the classification model to pre-group the training data based on domain factors if no DA is applied, which results in a very poor performance when little overlapping domain-class combinations are given. The experiments showed that the application of DA, in this case the adversarial DA approach of [22], is essential to achieve a good performance on the domain-class combinations that are not represented with supervised data in the training set. Further, it was shown that even without any samples of the domain-class combinations that are not supervisedly given during training, the application of DA [22] is useful to achieve a higher minimum performance on these combinations.

The problem of the standard adversarial DA approach of [22] is that primarily all domain factors are removed from the feature representation, which can lead to negative transfer if those are task-informative or visually close to such. To prevent this negative effect, the novel FP-DA approach was introduced in this thesis. It allows to preserve a chosen factor during DA in a multi-source domain constellation. The effect of negative transfer caused by the approach of [22] was shown in multiple one-to-one and multi-source transfer experiments. An in-depth analysis of the transfer results combined with PCA on one-to-one transfer experiments helped to reveal factors that potentially lead to a decreased performance when being removed and thus are worth to be preserved through FP-DA. The effectiveness of FP-DA was then demonstrated on two different object datasets from the robotics community, the CORe50 dataset and an adapted version of the OpenLORIS dataset, while the latter allowed to reveal the requirement of labeled factor combinations for FP-DA to work best. Overall, FP-DA was able to show the highest average and minimum performance across all constellations in

a leave-one-domain-out setting with multiple domains.

8.2 Future Research

DA is a challenging task when it comes to deep neural networks. Due to the large number of parameters within deep neural networks, the human interpretability of which features a model learned and how they are interconnected within the classification model is generally very limited. This thesis provided a factor based analysis of different scenarios and approaches which helps to better understand and interpret deep neural network behavior. The three main parts of this thesis provided essential insights and ideas how to improve the generalization ability of neural networks, however, there are many other conceivable research directions for each part.

Regarding the investigations of domain awareness, here the experiments were limited to small image samples where simple features are sufficient to classify those. Whether the presented outcomes also hold for more complex datasets, as for example the presented CORe50 dataset, in combination with a more comprehensive architecture could be a direction for future research. An interesting investigation there would be whether the more comprehensive architecture would be capable to overcome a normalization mismatch in the *part*-case of generalization or would suffer even more from the stronger color bias introduced by the backgrounds of certain CORe50 domains. Further, the scalability of the findings to different DA approaches could be worth investigating. Here only a parametrized DA method was used, however, naturally the question arises, whether the results also hold in combination with an additional DA method that uses a learned normalization like [22].

For the generalization investigations of the Domain Mixture scenario, experiments with more than two domains are conceivable. With multiple domains involved, the omission of DA would most likely lead to similar results on the unseen domain class combinations. Of course, here the chosen architecture plays a major role, which defines the complexity and amount of features that can be learned. Furthermore, this can be influenced by the number of classes and their general discriminability. If DA is applied in the presence of multiple domains, also in the Domain Mixture scenario a larger influence of negative transfer similar to the CORe50 baseline leave-one-out experiments could be expected, which potentially shades other effects. The evaluations regarding the imbalance within the training batches also revealed more potential for improvement. As stated, the

adaption of the batch-size does not scale to real-world problems where the number of classes within the unsupervised data is generally unknown. A more advanced heuristic needs to be found to overcome this issue and improve the performance even further.

For FP-DA, further datasets could be investigated to substantiate the effectiveness of it even more. The most promising area of research would be the extension of the architecture to a class-level approach that does not only group domains, but also classes. Generally a hierarchical approach would be conceivable, where in a first instance the domains would be grouped by a chosen domain factor, since this most likely has most influence on the transfer performance, and in a second instance a grouping by classes to preserve certain factors on class-level. This would of course require special treatment of the unsupervised data of the target domain since there no class information is available. Nevertheless, the classes of the different source domains within a group could be grouped according to this, from which the overall feature learning process might already benefit.

An interesting area of research would be to investigate the integrability and importance of all researched topics in this thesis for a 24/7 smart environment with a connected network of multiple camera entities. This could be composed of an evaluation of the relevance of each part's findings for a system where image data is collected 24/7 and an image classification model is frequently updated based on the expanding image data pool. Such a system could be built, as depicted in the introduction, of multiple camera based systems representing domains and therefore make the application of DA algorithms crucial. During runtime all acquisition metadata should further be stored, to be able to use it in subsequent correlation analyses for the identification of factors that highly influence the generalization ability of the classification model shared among the entities.

A Appendix

A.1 Additional Results - Complete Domain Mixture on CORe50

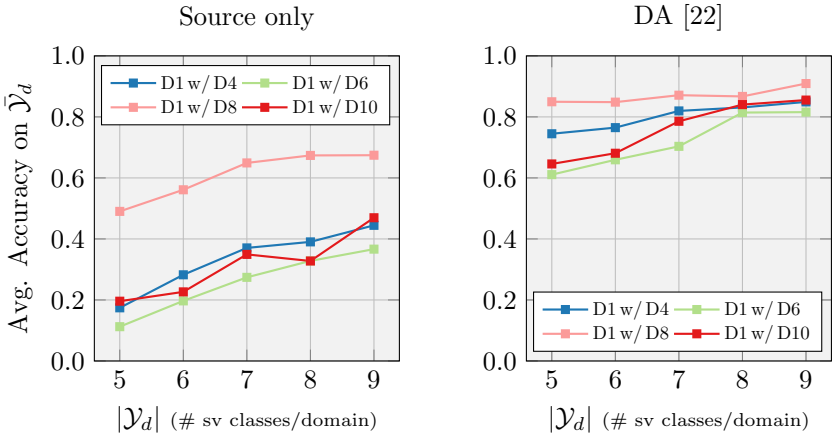


Figure A.1: Complete Domain Mixture scenario on the CORe50 dataset. Here, two domains, each using the right hand to present the object were combined. I. e. D1 w/ {D4, D6, D8, D10}. Note, here only the accuracy on the entire target dataset is shown.

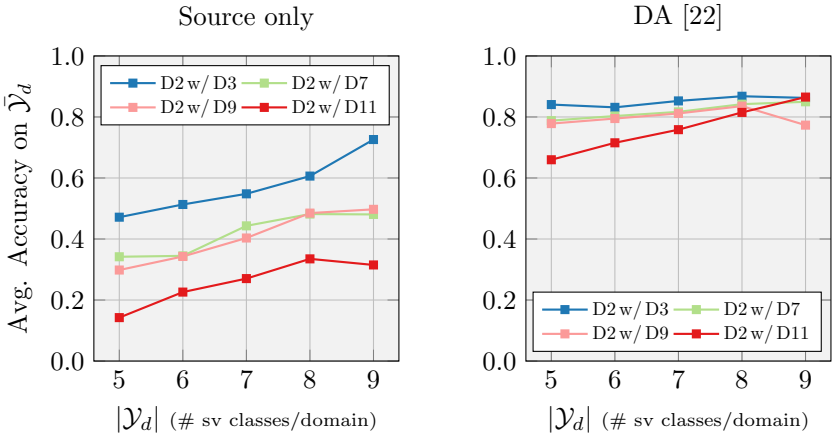


Figure A.2: Complete Domain Mixture scenario on the CORe50 dataset. Here, two domains, each using the left hand to present the object were combined. I. e. D2 w/ {D3, D7, D9, D11}.

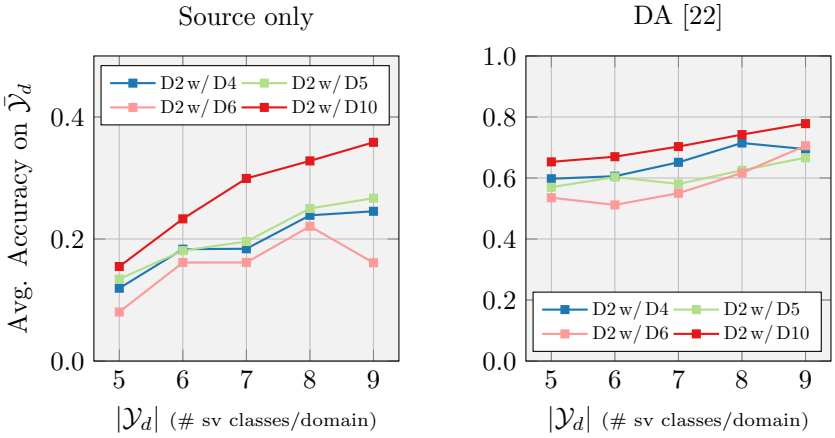


Figure A.3: Complete Domain Mixture scenario on the CORe50 dataset. Here, two domains, each using a different hand for presenting the object were combined. I. e. D2 w/ {D4, D5, D6, D10}.

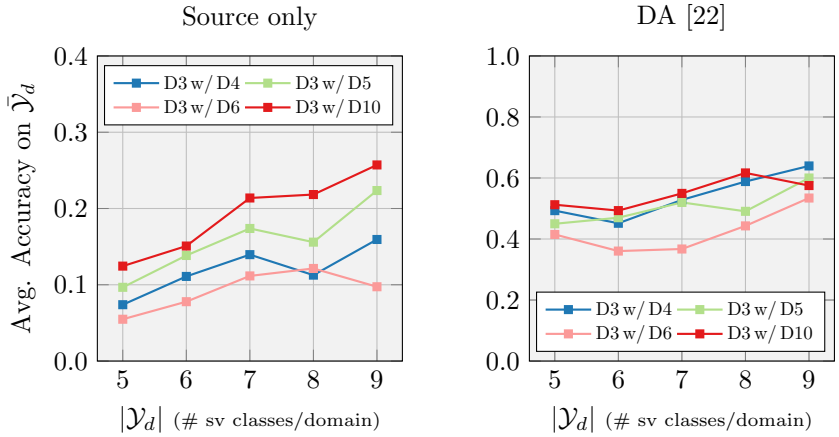


Figure A.4: Complete Domain Mixture scenario on the CORe50 dataset. Here, two domains, each using a different hand for presenting the object were combined. I. e. D3 w/ {D4, D5, D6, D10}.

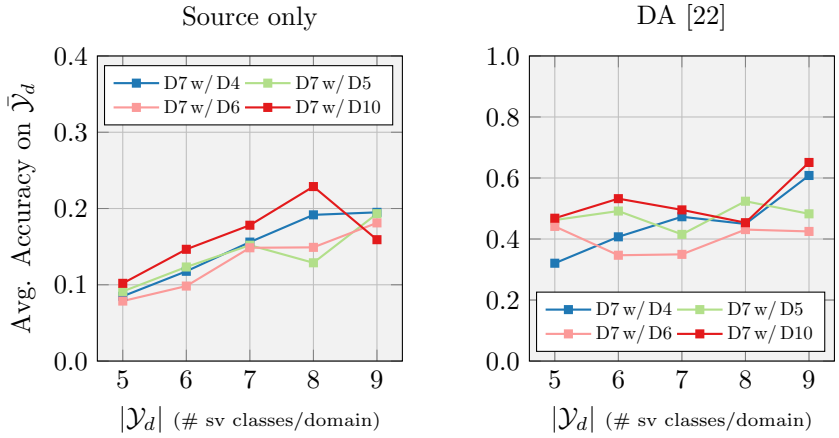


Figure A.5: Complete Domain Mixture scenario on the CORe50 dataset. Here, two domains, each using a different hand for presenting the object were combined. I. e. D7 w/ {D4, D5, D6, D10}.

A.2 Additional t-SNE Plots

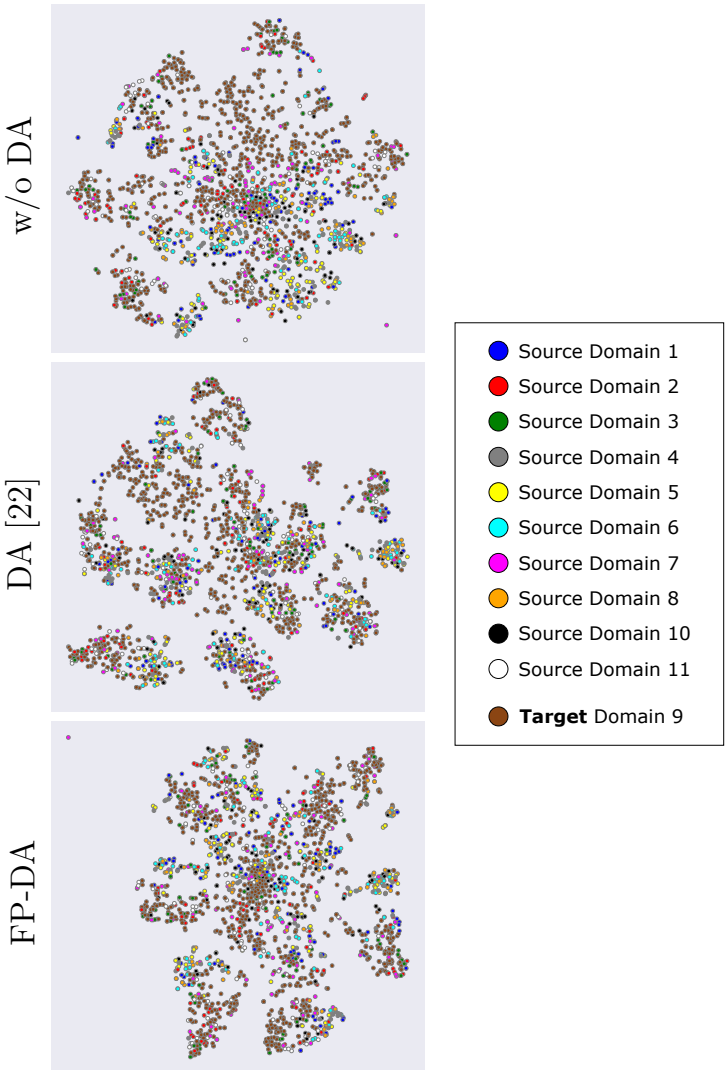


Figure A.6: t-SNE plots for the last layer of the shared feature extractor. Here, the samples are colored by domains.

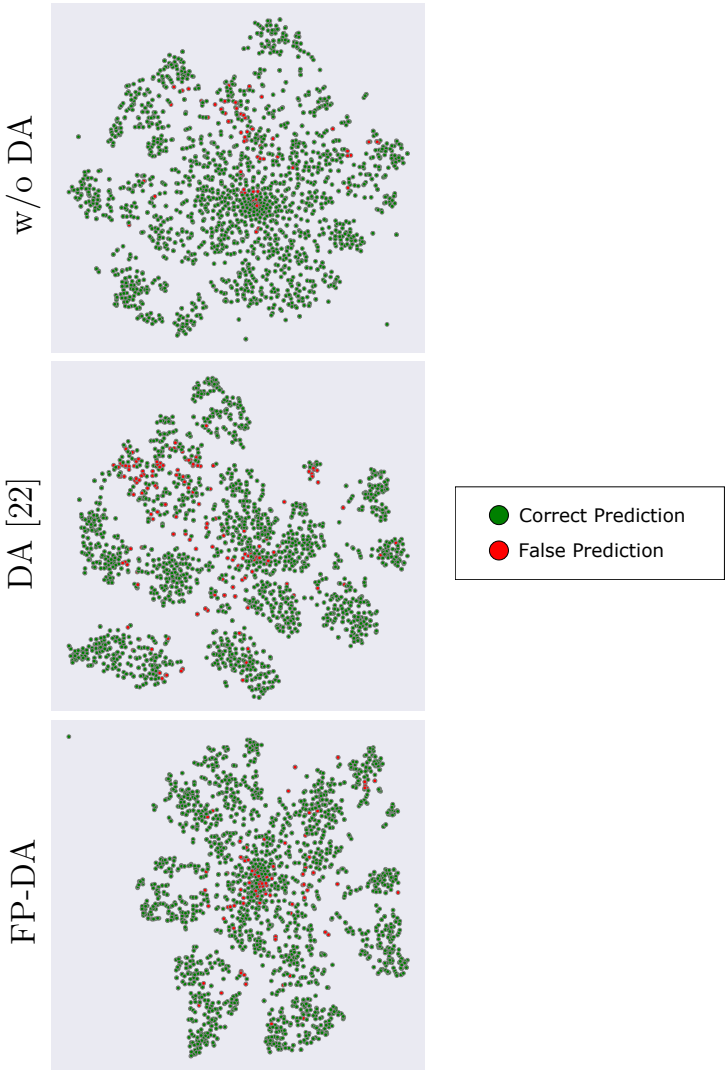


Figure A.7: t-SNE plots for the last layer of the shared feature extractor. Here, the samples are colored by correct and false predictions.

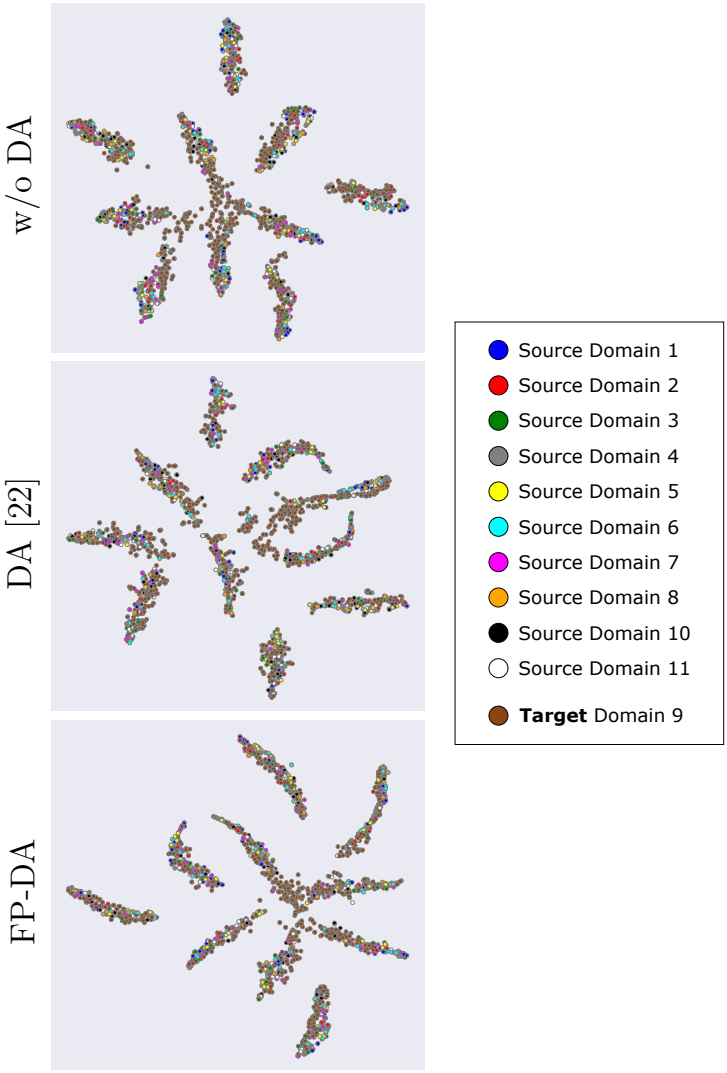


Figure A.8: t-SNE plots for the second fully-connected layer. Here, the samples are colored by domains.

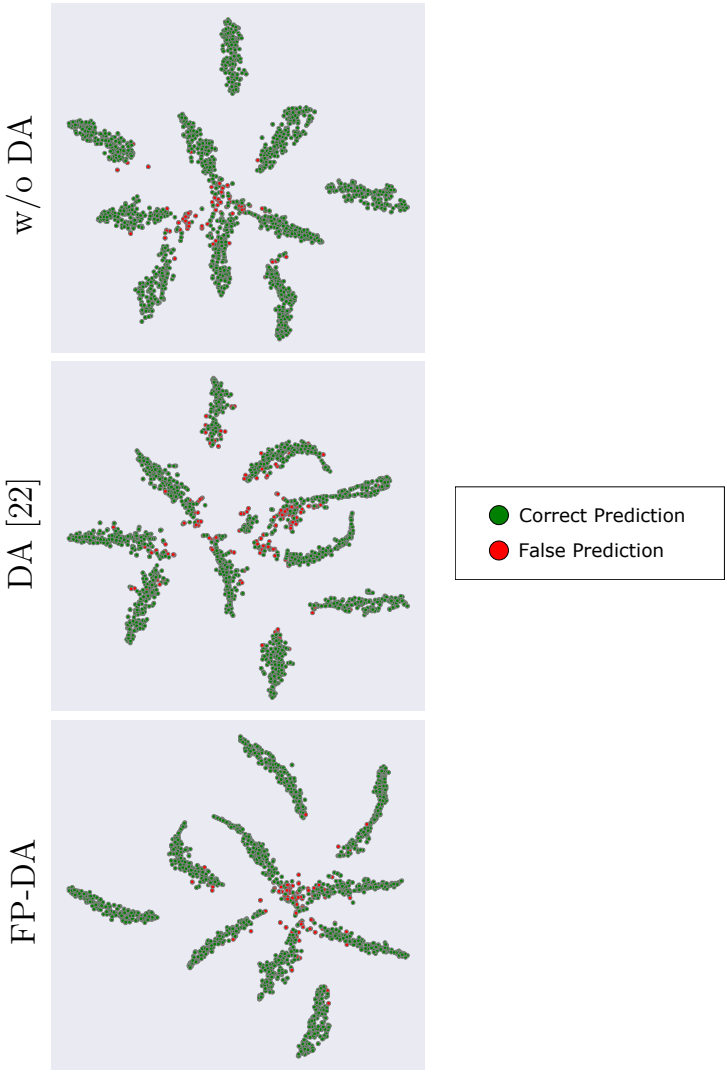


Figure A.9: t-SNE plots for the second fully-connected layer. Here, the samples are colored by correct and false predictions.

A.3 OpenLORIS Domain Subsets Results

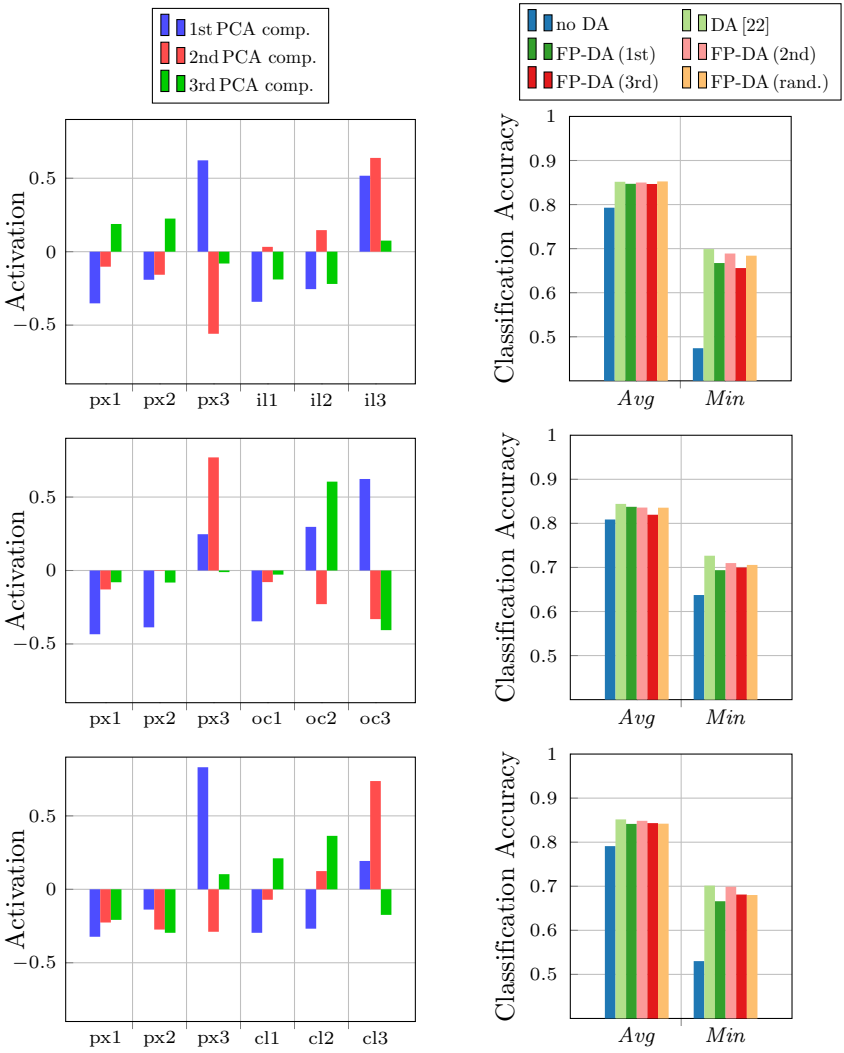


Figure A.10: PCA and FP-DA results on subsets of the OpenLORIS domains. The left column shows the PCA results, the right column the leave-one-out experiments where only the relevant average and minimum performance is shown.

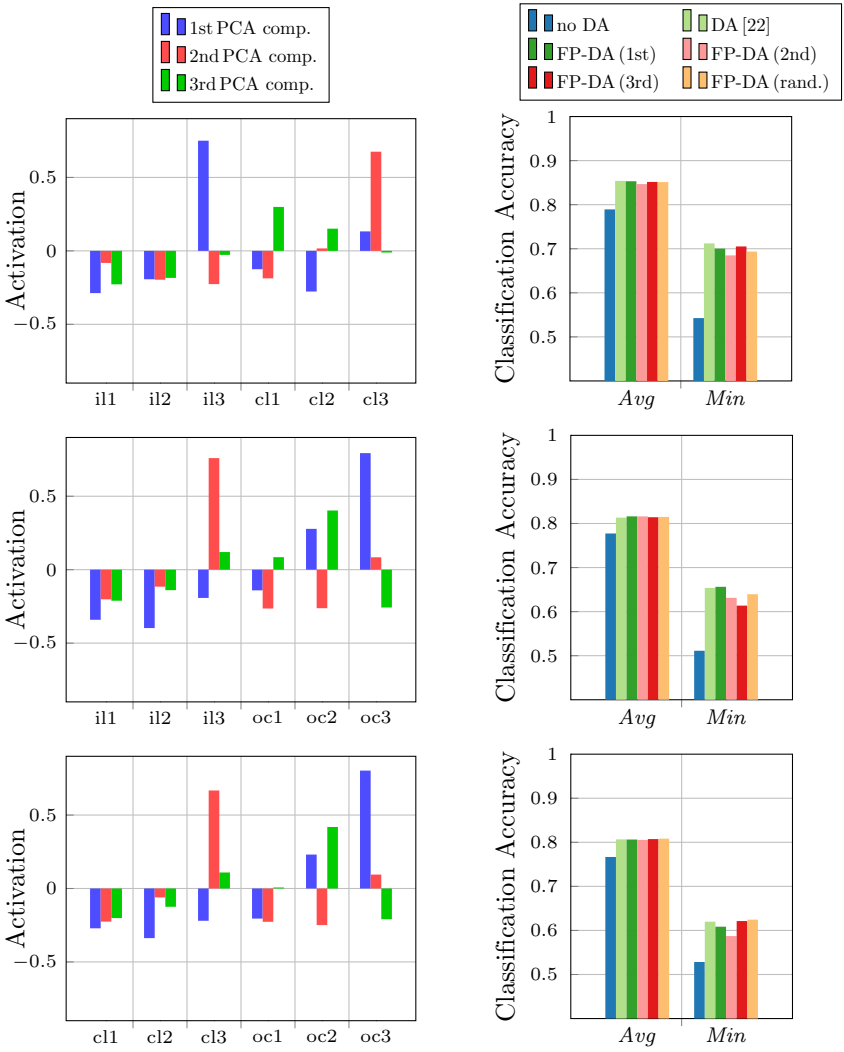


Figure A.11: PCA and FP-DA results on combinations of two factors of the OpenLORIS domains. (Continuation of Fig. A.10)

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, and M. Marchand. Domain-adversarial neural networks. *CoRR*, abs/1412.4446, 2014.

[3] P. Arbelaez, M. Maire, C. C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, 2011.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495, 2017.

[5] A. Bergamo and L. Torresani. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *Conference on Neural Information Processing Systems (NIPS)*, pages 181–189. Curran Associates, Inc., 2010.

[6] C. M. Bishop. *Pattern recognition and machine learning, 5th Edition.* Information science and statistics. Springer, 2007.

[7] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 95–104. IEEE Computer Society, 2017.

[8] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *Conference on Neural Information Processing Systems (NIPS)*, pages 343–351, 2016.

[9] C. Burgess and H. Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

[10] P. P. Busto and J. Gall. Open set domain adaptation. In *International Conference on Computer Vision (ICCV)*, pages 754–763. IEEE Computer Society, 2017.

[11] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang. Learning to transfer examples for partial domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2985–2994. IEEE Computer Society, 2019.

[12] G. Csurka. A comprehensive survey on domain adaptation for visual applications. In *Domain Adaptation in Computer Vision Applications*, pages 1–35. Springer, Cham, 2017.

[13] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE Computer Society, 2009.

[14] Z. Deng, Y. Luo, and J. Zhu. Cluster alignment with a teacher for unsupervised domain adaptation. In *International Conference on Computer Vision (ICCV)*, pages 9943–9952. IEEE, 2019.

[15] T. DeVries, I. Misra, C. Wang, and L. van der Maaten. Does object recognition work for everyone? In *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 52–59. IEEE Computer Society, 2019.

[16] L. Duan, D. Xu, and S. Chang. Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1338–1345. IEEE Computer Society, 2012.

[17] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010.

[18] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *International Conference on Computer Vision (ICCV)*, pages 2960–2967. IEEE Computer Society, 2013.

[19] J. Fritsch, T. Kühnl, and A. Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, pages 1693–1700. IEEE, 2013.

[20] J. Fritsch, T. Kühnl, and F. Kummert. Monocular road terrain detection by combining visual and spatial information. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1586–1596, 2014.

[21] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37, 2014.

[22] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning (ICML)*, pages 1180–1189. JMLR.org, 2015.

[23] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. S. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17:59:1–59:35, 2016.

[24] L. Ge, J. Gao, H. Q. Ngo, K. Li, and A. Zhang. On handling negative transfer and imbalanced distributions in multiple source transfer learning. *Statistical Analysis and Data Mining*, 7(4):254–271, 2014.

[25] B. Gong, K. Grauman, and F. Sha. Reshaping visual datasets for domain adaptation. In *Conference on Neural Information Processing Systems (NIPS)*, pages 1286–1294, 2013.

[26] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014.

[27] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *International Conference on Computer Vision (ICCV)*, pages 999–1006. IEEE Computer Society, 2011.

[28] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13:723–773, 2012.

[29] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. 2007.

[30] J. Guo, D. J. Shah, and R. Barzilay. Multi-source domain adaptation with mixture of experts. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4694–4703. Association for Computational Linguistics, 2018.

[31] S. Hasler, J. Kreger, and U. Bauer-Wersing. Interactive incremental online learning of objects onboard of a cooperative autonomous mobile robot. In *International Conference on Neural Information Processing (ICONIP)*, volume 11307 of *Lecture Notes in Computer Science*, pages 279–290. Springer, 2018.

[32] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. IEEE Computer Society, 2016.

[33] S. Herath, M. T. Harandi, and F. Porikli. Learning an invariant hilbert space for domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3956–3965. IEEE Computer Society, 2017.

[34] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *European Conference on Computer Vision (ECCV)*, volume 7573 of *Lecture Notes in Computer Science*, pages 702–715. Springer, 2012.

[35] J. Hoffman, E. Tzeng, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *Domain Adaptation in Computer Vision Applications*, Advances in Computer Vision and Pattern Recognition, pages 173–187. Springer, 2017.

[36] L. Hu, M. Kan, S. Shan, and X. Chen. Unsupervised domain adaptation with hierarchical gradient synchronization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4042–4051. IEEE Computer Society, 2020.

[37] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269. IEEE Computer Society, 2017.

[38] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[39] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *International Conference on Multimedia (ACM-MM)*, pages 675–678. ACM, 2014.

[40] X. Jiang, Q. Lao, S. Matwin, and M. Havaei. Implicit class-conditioned domain alignment for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 4816–4827. PMLR, 2020.

[41] M. Jiménez-Guarneros and P. Gómez-Gil. A study of the effects of negative transfer on deep unsupervised domain adaptation methods. *Expert Systems with Applications*, page 114088, 2020.

[42] M. Karimpour, S. Noori Saray, J. Tahmoresnezhad, and M. Pourmahmood Aghababa. Multi-source domain adaptation for image classification. *Machine Vision and Applications*, 31(6):44, 2020.

[43] A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *European Conference on Computer Vision (ECCV)*, volume 7572 of *Lecture Notes in Computer Science*, pages 158–171. Springer, 2012.

[44] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[45] S. Kirstein, A. Denecke, S. Hasler, H. Wersing, H.-M. Gross, and E. Körner. A vision architecture for unconstrained and incremental learning of multiple categories. *Memetic Computing*, 1(4):291–304, 2009.

[46] P. Koniusz, Y. Tas, H. Zhang, M. T. Harandi, F. Porikli, and R. Zhang. Museum exhibit identification challenge for the supervised domain adaptation and beyond. In *European Conference on Computer Vision (ECCV)*, volume 11220 of *Lecture Notes in Computer Science*, pages 815–833. Springer, 2018.

[47] W. M. Kouw. An introduction to domain adaptation and transfer learning. *CoRR*, abs/1812.11806, 2018.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

[49] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[50] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[51] D. Li, Y. Yang, Y. Song, and T. M. Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision (ICCV)*, pages 5543–5551. IEEE Computer Society, 2017.

[52] Y. Li, M. Murias, G. Dawson, and D. E. Carlson. Extracting relationships by multi-domain matching. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 6799–6810, 2018.

[53] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou. Revisiting batch normalization for practical domain adaptation. In *International Conference on Learning Representations (ICLR), Workshop Track Proceedings*. OpenReview.net, 2017.

[54] J. Liang, R. He, Z. Sun, and T. Tan. Distant supervised centroid shift: A simple and efficient approach to visual domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2975–2984. Computer Vision Foundation / IEEE, 2019.

[55] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International*

Conference on Machine Learning (ICML), volume 97 of *Proceedings of Machine Learning Research*, pages 4114–4124, 2019.

[56] V. Lomonaco and D. Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning (CoRL)*, volume 78 of *Proceedings of Machine Learning Research*, pages 17–26. PMLR, 13–15 Nov 2017.

[57] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440. IEEE Computer Society, 2015.

[58] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan. Transferable representation learning with deep adaptation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):3071–3085, 2019.

[59] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 97–105. JMLR.org, 2015.

[60] M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 1647–1657, 2018.

[61] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *Conference on Neural Information Processing Systems (NIPS)*, pages 136–144, 2016.

[62] H. A. Mallot, H. H. Bülthoff, J. Little, and S. Bohrer. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biological cybernetics*, 64(3):177–185, 1991.

[63] D. Maltoni and V. Lomonaco. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116:56–73, 2019.

[64] M. Mancini, L. Porzi, S. R. Bulò, B. Caputo, and E. Ricci. Boosting domain adaptation by discovering latent domains. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3771–3780. IEEE Computer Society, 2018.

[65] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *Conference on Neural Information Processing Systems (NIPS)*, pages 1041–1048. Curran Associates, Inc., 2008.

[66] T. Michalke, R. Kastner, J. Fritsch, and C. Goerick. A self-adaptive approach for curbstone/roadside detection based on human-like signal processing and multi-sensor fusion. In *Intelligent Vehicles Symposium (IV)*, pages 307–312. IEEE, 2010.

[67] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Unified deep supervised domain adaptation and generalization. In *International Conference on Computer Vision (ICCV)*, pages 5716–5726. IEEE Computer Society, 2017.

[68] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814. Omnipress, 2010.

[69] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[70] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[71] Y. Pan, T. Yao, Y. Li, Y. Wang, C. Ngo, and T. Mei. Transferrable prototypical networks for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2239–2247. Computer Vision Foundation / IEEE, 2019.

[72] G. I. Parisi, J. Tani, C. Weber, and S. Wermter. Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization. *Frontiers Neurorobotics*, 12:78, 2018.

[73] Z. Pei, Z. Cao, M. Long, and J. Wang. Multi-adversarial domain adaptation. In *AAAI Conference on Artificial Intelligence*, pages 3934–3941. AAAI Press, 2018.

[74] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang. Moment matching for multi-source domain adaptation. In *International*

Conference on Computer Vision (ICCV), pages 1406–1415. IEEE Computer Society, 2019.

[75] X. Peng, B. Sun, K. Ali, and K. Saenko. Learning deep object detectors from 3d models. In *International Conference on Computer Vision (ICCV)*, pages 1278–1286. IEEE Computer Society, 2015.

[76] X. Peng, B. Usman, K. Saito, N. Kaushik, J. Hoffman, and K. Saenko. Syn2real: A new benchmark for synthetic-to-real visual domain adaptation. *CoRR*, abs/1806.09755, 2018.

[77] T. Pomierski and H. Gross. Biological neural architecture for chromatic adaptation resulting in constant color sensations. In *International Conference on Neural Networks (ICNN)*, pages 734–739. IEEE, 1996.

[78] S. Rakshit, B. Banerjee, G. Roig, and S. Chaudhuri. Unsupervised multi-source domain adaptation driven by deep adversarial ensemble learning. In *German Conference on Pattern Recognition (GCPR)*, volume 11824 of *Lecture Notes in Computer Science*, pages 485–498. Springer, 2019.

[79] I. Redko, N. Courty, R. Flamary, and D. Tuia. Optimal transport for multi-source domain adaptation under target shift. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, pages 849–858. PMLR, 2019.

[80] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich. To transfer or not to transfer. In *Conference on Neural Information Processing Systems (NIPS) Workshop on Transfer Learning*, volume 898, pages 1–4, 2005.

[81] A. Rozantsev, M. Salzmann, and P. Fua. Beyond sharing weights for deep domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(4):801–814, 2019.

[82] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision (ECCV)*, volume 6314 of *Lecture Notes in Computer Science*, pages 213–226. Springer, 2010.

[83] S. Schrom. Deep road terrain detection. Master's thesis, Technical University of Darmstadt, 2017.

[84] S. Schrom and S. Hasler. Effects of domain awareness in generalizing over cameras in road detection. In *Machine Learning Reports 03/2017*, 2017.

[85] S. Schrom and S. Hasler. Domain mixture: An overlooked scenario in domain adaptation. In *International Conference On Machine Learning And Applications (ICMLA)*, pages 22–27. IEEE Computer Society, 2019.

[86] S. Schrom, S. Hasler, and J. Adamy. Improved multi-source domain adaptation by preservation of factors. *Image and Vision Computing*, 112:104209, 2021.

[87] M. Schutera, F. M. Hafner, J. Abhau, V. Hagenmeyer, R. Mikut, and M. Reischl. Cuepervision: self-supervised learning for continuous domain adaptation without catastrophic forgetting. *Image and Vision Computing*, 106:104079, 2021.

[88] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision (ICCV)*, pages 618–626. IEEE Computer Society, 2017.

[89] S. Shankar, Y. Halpern, E. Breck, J. Atwood, J. Wilson, and D. Sculley. No classification without representation: Assessing geodiversity issues in open data sets for the developing world. *CoRR*, abs/1711.08536, 2017.

[90] Q. She, F. Feng, X. Hao, Q. Yang, C. Lan, V. Lomonaco, X. Shi, Z. Wang, Y. Guo, Y. Zhang, F. Qiao, and R. H. M. Chan. OpenLORIS-Object: A robotic vision dataset and benchmark for lifelong deep learning. In *International Conference on Robotics and Automation (ICRA)*, pages 4767–4773, 2020.

[91] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[92] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[93] B. Sun and K. Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *British Machine Vision Conference (BMVC)*. BMVA Press, 2014.

[94] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision (ECCV), Workshops*, volume 9915 of *Lecture Notes in Computer Science*, pages 443–450. Springer, 2016.

[95] T. Tommasi. *Learning to Learn by Exploiting Prior Knowledge*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2012.

[96] T. Tommasi, N. Patricia, B. Caputo, and T. Tuytelaars. A deeper look at dataset bias. In *German Conference on Pattern Recognition (GCPR)*, Lecture Notes in Computer Science, pages 504–516. Springer, 2015.

[97] T. Tommasi, T. Tuytelaars, and B. Caputo. A testbed for cross-dataset analysis. In *European Conference on Computer Vision (ECCV), TASK-CV Workshop*, volume 8927 of *Lecture Notes in Computer Science*, pages 18–31, 2014.

[98] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1521–1528. IEEE Computer Society, 2011.

[99] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *International Conference on Computer Vision (ICCV)*, pages 4068–4076. IEEE Computer Society, 2015.

[100] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2962–2971. IEEE Computer Society, 2017.

[101] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014.

[102] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11), 2008.

[103] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5385–5394. IEEE Computer Society, 2017.

[104] H. Wang, W. Yang, Z. Lin, and Y. Yu. Tmda: Task-specific multi-source domain adaptation via clustering embedded adversarial training. In *International Conference on Data Mining (ICDM)*, pages 1372–1377. IEEE Computer Society, 2019.

[105] Z. Wang, Z. Dai, B. Póczos, and J. G. Carbonell. Characterizing and avoiding negative transfer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11293–11302. Computer Vision Foundation / IEEE, 2019.

[106] K. R. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big Data*, 3:9, 2016.

[107] S. Xie, Z. Zheng, L. Chen, and C. Chen. Learning semantic representations for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 5419–5428. PMLR, 2018.

[108] C. Xiong, S. McCloskey, S. Hsieh, and J. J. Corso. Latent domains modeling for visual domain adaptation. In *AAAI Conference on Artificial Intelligence*, pages 2860–2866. AAAI Press, 2014.

[109] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3964–3973. IEEE Computer Society, 2018.

[110] Y. Zhang, N. Wang, and S. Cai. Adversarial sliced wasserstein domain adaptation networks. *Image and Vision Computing*, 102:103974, 2020.

[111] S. Zhao, G. Wang, S. Zhang, Y. Gu, Y. Li, Z. Song, P. Xu, R. Hu, H. Chai, and K. Keutzer. Multi-source distilling domain adaptation. In *AAAI Conference on Artificial Intelligence*, pages 12975–12983. AAAI Press, 2020.

[112] Q. Zhou, S. Wang, et al. Cluster adaptation networks for unsupervised domain adaptation. *Image and Vision Computing*, 108:104137, 2021.

Index