SAPIENZA
UNIVERSITÀ DI ROMA

Broadening Deep Learning horizons: models for RGB and Depth images adaptation

School of Information Engineering, Computer Science and Statistics

Dottorato di Ricerca in Ingegneria Informatica – XXXII Ciclo

Candidate

Paolo Russo

ID number 1297064

Thesis Advisor

Prof. Barbara Caputo

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering in Computer Science

October 2019

Broadening Deep Learning horizons: models for RGB and Depth images adaptation
Ph.D. thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: prusso@diag.uniroma1.it

Acknowledgements

This dissertation sums up three years of hard and satisfying work in which some people have played a big role. First of all, I would like to thank my friend and co-author Fabio for all his collaboration, helping and teaching through the world of Deep Learning. It has been a true pleasure working with you, hoping to repeat the experience! Another special mention is for Arjan: I have lost the count of how many times I have asked his help, probing him about the biggest threats I faced when studying Machine Learning, Python and Linux. If I mastered a tiny fraction of Computer Science, it is thanks to them. I would like to thank all my PhD colleagues: Massimiliano, Valentina, such kind and nice people which shared with me the passion for research; Nizar, a great friend whose presence and deep connection has helped me in the most difficult moments of this journey. Being a researcher of the VANDAL lab has been a wonderful experience which have greatly formed and ironed my professional figure, my thanks go to each of its members. I cannot avoid to mention Antonio, Mirco and Silvia, which shared both research and laughs along the PhD path. I acknowledge as well all the professors which taught me how to do research and how to get passionated about all the interesting arguments of Science, with a particular mention to professor Pirri, which gave me precious hints about this thesis. I would like to thank also every person which hindered my route: they trained my resilience, one of the most important quality to get ahead in research as in life. A big thanks goes to my family, as my father and my brother are the biggest resource which I could ever have; and I will never forget my mother, which is helping me from somewhere with her unique emotional intelligence and wisdom. I praise all my friends which helped preserving my mental sanity: Emanuele, Raffaele and all my tennis mates contributed to my peak climbing. Mara, I cannot sufficiently thank you for pushing me into the research world, with your deep belief in my abilities: I would never started this study without your efforts. I will never forget the support and aid of Claudia in my PhD course as well as in my life, thank you so much. A final, special thanks goes to my cats, whose presence in my life as well as on my keyboard has been invaluable. To all of you, my best acknowledgements. Ad Maiora.

Abstract

Deep Learning has revolutionized the whole field of Computer Vision. Very deep models with an huge number of parameters have been successfully applied on big image datasets for difficult tasks like object classification, person re-identification, semantic segmentation. Two-fold results have been obtained: astonishing performance, with accuracy often comparable or better than a human counterpart on one hand, and on the other the development of robust, complex and powerful visual features which exhibit the ability to generalize to new visual tasks.

Still, the success of Deep Learning methods relies on the availability of big datasets: whenever the available, labeled data is limited or redundant, a deep neural network model will typically overfit on training data, showing poor performance on new, unseen data.

A typical solution used by the Deep Learning community in those cases is to rely on some Transfer Learning techniques; within the several available methods, the most successful one has been to pre-train the deep model on a big heterogeneous dataset (like ImageNet) and then to finetune the model on the available training data. Among several fields of application, this approach has been heavily used by the robotic community for depth images object recognition. Depth images are usually provided by depth sensors (eg. Kinect) and their availability is somewhat scarce: the biggest depth images dataset publicly available includes 50.000 samples, making the use of a pre-trained network the only successful method to exploit deep models on depth data.

Without any doubt, this method provides suboptimal results as the network is trained on traditional RGB images having very different perceptual information with respect to depth maps; better results could be obtained if a big enough depth dataset would be available, enabling the training a deep model from scratch.

Another frequent issue is the difference of statistical properties between training and test data (*domain gap*). In this case, even in the presence of enough training data, the generalization ability of the model will be poor, thus making the use of a *Domain Adaptation* method able to reduce the domains gap; this can improve both the robustness of the model and its final classification performances.

In this thesis both problems have been tackled by developing a series of Deep Learning solutions for Domain Adaptation and Transfer Learning tasks on RGB and depth images domains: a new synthetic depth images dataset is presented, showing the performance of a deep model trained from scratch on depth-only data. At the same time, a new powerful depth→RGB mapping module is analyzed, to optimize the classification accuracy on depth images tasks while using pretrained-on-ImageNet deep models. The study of the depth domain ends with a recurrent neural network for egocentric action recognition capable of exploiting depth images as an additional source of attention.

A novel GAN model and an hybrid pixel/features adaptation architecture for RGB images have been developed: the former on single-domain adaptation tasks, while the latter on multi-domain adaptation and generalization tasks. Finally, a preliminary approach to the problem of multi-source Domain Adaptation on a semantic segmentation task is examined, based on the combination of a multi-branch segmentation model and a adversarial technique, capable of exploiting all the available synthetic training datasets and to increase the overall performance.

The performance obtained by using the proposed algorithms are often better or equivalent with respect to the currently available state of the art methods on several datasets and domains, demonstrating the superiority of our approach. Moreover, our analysis shows that the creation of ad-hoc domain adaptation and transfer learning techniques are mandatory in order to obtain the best accuracy in the presence of any domain gap, with a little or negligible additional computational cost.

Contents

Chapter 1

Introduction

1.1 Context and Motivations

Vision, intended as the process of discovering the world from images, is probably the most important sense of living beings. Our eyes present both an huge optical resolution [36] and an huge throughput of processed data, quantified in ten gigabits of captured informations per second, while our brain is able to process around three million bits per second of this information [4]. With these premises, it is surprising to notice that in the sixties the task of *computer vision* was something that the leading experts in the field of Artificial Intelligence thought to be at the level of difficulty of a summer student's project [61]. Almost 60 years later the problem is still far to be solved: the task of building a general artificial vision system capable of analyze the world as an human being would do is extremely difficult. Nevertheless, the last ten years have seen one of the most powerful Artificial Intelligence revolution with the rise of Deep Learning. Among the several machine learning and AI fields that have been revolutionized by the advent of deep neural networks (Natural Language Processing [122], Reinforcement Learning [108] and many others), the Computer Vision field is the one where the introduction of very deep models has produced the most dramatic effects. In few years, thanks to the use of deep convolutional neural networks, performances on typical object classification tasks like ImageNet Large Scale Visual Recognition Challenge 2012 (Figure 1.1) jumped from 26% top-5 error of XRCE method [74] to 3.1% of the Inception V4 deep model [153]

However, the ability of deep neural networks to build complex visual features and to handle challenging classification tasks relies on the availability of big, labeled images datasets for training; moreover, this training set should share the same statistics of the test one. In all the

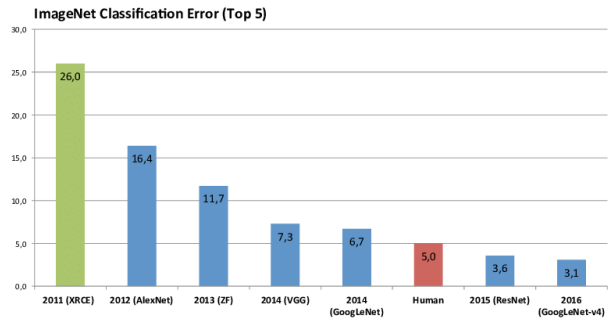


Figure 1.1. ILSCVR 2012 performances over time. If the best *shallow* method in 2011 exhibits 26.1% top-5 error, after 5 years deep models like Inception V4 are able to reach 3.1% top-5 error.

cases in which this training data is lacking, or if there is a statistical gap with respect to test data, the final performance will be poor and suboptimal. In order to solve these issues and to reduce the loss of performance, two very promising family of techniques are *Domain Adaptation* and *Transfer Learning*. While both Domain Adaptation and Transfer Learning already existed in the classical field of machine learning, it is with the rise of Deep learning that they become a key component for optimal performance.

Beside adapting the models to the data currently available, another powerful approach is to synthetically generate data in order to both increase the training set and to decrease the statistical domain gap with respect to the test set. While one straightforward approach for generating realistic data is to use a rendering software like Blender [44], with the rise of Generative Adversarial Networks [53] researches are able to produce any kind of data which mimic the properties of any given domain. Exploiting both these techniques could finally enable or improve the use of deep models in critical tasks like autonomous driving, medical images analysis, person re-identification, and so on.

1.2 Addressed Problems

The problems addressed in this thesis can be listed as follows:

- RGB object recognition in the presence of a strong domain gap between training and test domains;
- Depth images object recognition and first-person RGB-D action recognition;
- Semantic segmentation in urban scenes.

1.3 Contributions of the Thesis

The main contributions of this work is the development and analysis of deep algorithms and models for Computer Vision tasks, under the hypothesis of lack of training data or with a statistical gap between training and test data. The chosen approach to solve those tasks has been to synthetically generate data or to map data from one domain to another one, or to align domains on a feature level. More into details, the contributions can be summarized in the following presentations:

- **A new synthetic depth images dataset (*VANDAL*)** [20] and a deep model trained from scratch on the proposed dataset (*DepthNet*), for depth images object recognition;
- **A novel deep module (*DECO*)[21] for depth→RGB mapping** which is able to learn the optimal colorization for maximizing accuracy on the same dataset as well as on different datasets;
- **A depth enhancing process for RGB-D first person action recognition (*LODEAR*)** on short video sequences, with the use of a deep CNN+LSTM module;
- **A new GAN architecture (*SBADAGAN*)[128] capable of realistic RGB→RGB transformation** for tackling digits recognition tasks in the presence of domains shift;
- **An hybrid pixel-features adaptation approach (*ADAGE*)** [22] for multi-source domain adaptation and generalization on digits datasets;

- **A multi-source domain adaptation algorithm**[129] for semantic segmentation tasks on urban scenes.

The majority of proposed techniques have been implemented by using the *Pytorch* framework, with the exception of two works done in *Caffe* and *Keras*. All the source codes are available on Github or freely distributed by the author upon request.

1.4 Structure of the Thesis

The rest of this thesis is structured as follows. Chapter 2 presents formally the unsupervised domain adaptation and transfer learning problems (Section 2.1 and Section 2.2) and the datasets used in this work (Section 2.5). Being 2.5D and RGB recognition two significantly different tasks, each with their own unique issues and solutions, the presented works have been divided into two main chapters. In Chapter 3 the work done on the depth domains is presented: the synthetically generated VANDAL dataset with the corresponding DepthNet model (Section 3.1), the DECO module for optimal depth→RGB mapping (Section 3.2), and LODEAR, an effective approach for RGB-D first person action recognition (Section 3.3). In Chapter 4 several algorithms for domain adaptation and transfer learning tasks into the RGB domain are introduced: SBADA-GAN (Section 4.1) is a novel adversarial method for digits recognition, ADAGE (Section 4.2) is an hybrid pixel/feature adaptation method for multi source domain adaptation and generalization tasks, and Section 4.3 shows a multi source GAN model for semantic segmentation in urban scenes. Conclusions and possible directions for future work are discussed in Chapter 5.

Declaration This thesis is a presentation of original work of its author. The work was done under the guidance of Prof. Barbara Caputo at Sapienza University of Rome. The ideas and the results presented in this work have all appeared in conference/article proceedings or workshops.

1.5 Peer-reviewed publications

The following list represents the current author peer-reviewed publications in a chronological order and his contribution to each work:

- Carlucci, F. M., Russo, P., & Caputo, B. (2017). A deep representation for depth images from synthetic data. In Robotics and Automation (ICRA), 2017 IEEE International Conference on (pp. 1362-1369). IEEE. The author contributed to the virtual camera setup, the CAD model cleaning phase, the data augmentation phase, as well as to the DepthNet network trainings.

- Carlucci, F. M., Russo, P., & Caputo, B. (2018). DE2CO: Deep Depth Colorization. IEEE Robotics and Automation Letters, 3(3), 2386-2393. (also presented at the IEEE International Conference on Robotics and Automation). The author contributed to the $(DE)^2CO$ module invention and its training on Washington and JHUIT-50 datasets.

- Russo, P., Carlucci, F. M., Tommasi, T., & Caputo, B. (2018). From source to target and back: symmetric bi-directional adaptive GAN. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. The author effectively created the SBADA-GAN modules and its additional losses and contributed in the training phase.

- Russo, P., F. M., Tommasi, T., & Caputo, B. (2019). Towards Multi-Source Adaptive Semantic Segmentation In Proceedings of the International Conference of Image Analysis

and Processing. The author assembled the main architecture and performed all the trainings.

- Carlucci, F. M., Russo, P., Tommasi, T., & Caputo, B (2019) Hallucinating Agnostic Images to Generalize Across Domains in European conference on computer vision, task-CV workshop. The author invented the Hallucinator module while helping on the overall architecture tests.

Chapter 2

Background

In this chapter the unsupervised Domain Adaptation problem and the Transfer Learning problem will be presented and discussed with an approach both informal and formal (Section 2.1 and Section 2.2). Section 2.3 presents the RGB-D recognition setting with its unique properties and issues. In Section 2.4 the current state of the art algorithms and methods will be briefly showed, while in Section 2.5 the datasets used in this work will be listed with their different domains and characteristics.

2.1 Unsupervised domain adaptation problem

One of the main underlying assumption in any model of learning is that both the training samples and the unseen test samples are drawn from the same probability distribution, which assure the correctness of the classifier decisions and enable the generalization error estimation [165]. In many practical applications this assumption does not hold: training and test sets could have different distributions, raising severe performance issues. Among several examples, we can mention the different statistical properties of object recognized by a robot when working in a controlled setting, like in a laboratory, with respect to the same robot working in a new, unseen setting, like in a home. Face recognition algorithms will exhibit inferior performance when trying to recognition faces put on different lights conditions, while autonomous driving system trained on synthetically generated data will perform poorly on real urban scenes data. The unsupervised Domain Adaptation techniques attempts to solve this issue by training a classifier using source samples that can generalize well to the target (test) domain while mitigating the domain shift between the two underlying distributions. In a formal notation, let define a domain as $D = \{\chi, P(X)\}$ made by the feature space χ and the edge probability distribution $P(X)$ where $X = \{x_1, x_2, ...x_n\} \in \chi$, and a task as $\mathcal{T} = \{Y, P(Y|X)\}$ made by label space $Y = \{y_1, y_2, ...y_n\}$ and the conditional probability function $P(Y|X)$. Finally, the source set can be defined (omitting the probability distribution) as $S = \{(x_1, y_1), (x_2, y_2), ...(x_n, y_n) \in X \times Y\}$ while the target set is defined as $T = \{\hat{x_1}, \hat{x_2}, ...\hat{x_n} \in \hat{X}\}$. While the source and the target domains share the same label sets $Y_s = Y_t$, the two related distributions are not identical: $P_s(Y|X) \sim P_t(Y|X)$. Source and target domains mismatch can be seen also in the marginal data distribution: $P_s(X) \neq P_t(X)$ and on a feature level: $X_s \neq X_t$.

The same issue of Domain Adaptation can be cast into two different settings: in the *semi-supervised* case, some few labeled examples from the target domain are available for training, while in the *unsupervised* case no label from the target domain set is available. In the case of Deep Learning applied on Computer Vision tasks, Domain Adaptation techniques can also be differentiated by the chosen approach to the problem: some algorithms work on a *pixel level*, reducing the gap between domains directly on the images, while other algorithms work on a

feature level (where in that case feature means the property learned by a convolutional filter), reducing the discrepancies between distributions on the features extracted from a deep neural network at some convolutional layer. In the case of semantic segmentation, an algorithm at *output level* can reduce the statistical differences directly on the deep model output.

Furthermore, we talk about *Multi Source Domain Adaptation* when multiple sources with different marginal data distributions are available as training set. In that case the training set can be redefined as $S = \{(x_1, y_1, d_1), (x_2, y_2, d_2), ...(x_n, y_n, d_n) \in X \times Y \times D\}$

Finally, a challenging variation of the multi source case is presented as *Domain Generalization*, where the target domain data is not available during the training process. In that case the aim is to build a classifier as much robust and general as possible, based on the available source domains.

The works in this thesis have been mostly based both on single source and multi source domain adaptation, with the *ADAGE* work tackling also the case of domain generalization. The majority of the developed algorithms work on a pixel level, with the exception of the hybrid pixel/feature approach of ADAGE and the multi-source semantic segmentation algorithm which works directly on the output level.

2.2 Transfer Learning

Data dependence is one of the most serious problem in deep learning. The huge deep models developed in the last years, which easily exhibit 10^7~10^8 trainable parameters, require a big amount of labeled data to learn its latent patterns. Unfortunately, in the majority of Computer Vision tasks there is a scarcity of labeled images, leading to overfitting and poor generalization ability. For these reasons, a critical step of a successful deep architecture is the use of a *Transfer Learning* technique. Given a learning task $\mathcal{T}_t$ based on D_t, Transfer learning aims to improve the performance of the conditional probability function $P_t(Y|X)$ for learning task $\mathcal{T}_t$ by discovering and transferring latent knowledge from a different domain D_s and $\mathcal{T}_s$, where $D_s \neq D_t$ and/or $\mathcal{T}_s \neq \mathcal{T}_t$. In addition, in most cases the size of D_s is much larger than the size of D_t. Without any doubt, one of the most used transfer learning techniques is *finetuning*. With the finetuning procedure, the deep neural network is first trained from scratch on a huge, heterogeneous dataset related to $D_s/\mathcal{T}_s$, where will learn robust and general visual features, and then finetuned by an additional training (usually with a lower learning rate) on the target domain D_t with task $\mathcal{T}_t$. In fact, the almost totality of deep models developed in the last years exploit finetuning by pre-training on ImageNet dataset [127], which has shown impressive generalization capabilities. In this thesis, we have dealt with transfer learning on depth domains with the DepthNet method, while we have developed a new transfer learning method on RGB images with DECO. Finally, in the proposed multi-source domain adaptation algorithm for semantic segmentation, we have exploited as a standard transfer learning technique the pre-training on ImageNet of the DeepLabV3 model.

2.3 RGB-D recognition

Vision perception is a critical component for enabling the use of autonomous robots in an unconstrained world. Nowadays, RGB-D sensors are ubiquitous in many robotic systems, thanks to their low cost and their ability to provide additional invariant informations about the objects, with respect to a simple RGB sensor which is limited to appearance and texture. For example, depth sensors are robust to low-contrast images, or images taken with poor light conditions (Figure 2.1); they can also provide unique perceptions as the objects shape and distance. Unfortunately, the use of depth data with Deep Learning models is not trivial due to

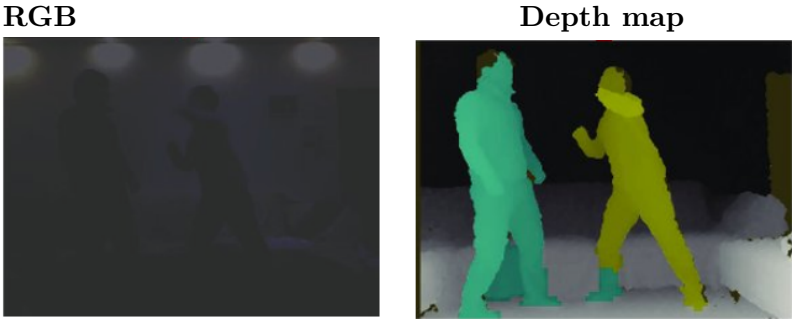


Figure 2.1. Comparison of a RGB image and a depth map taken by a Kinect RGB-D sensor in a low light condition. While RGB sensor require a good source of light in order to capture objects appearance, the respective depth sensor in the same condition can work flawlessly on complicated tasks like instance segmentation.

the small number of labeled depth images available on public datasets. Because of that, any deep model trained from scratch on that kind of data suffers poor generalization ability and strong overfitting. For these reasons, the majority of works which deal with depth or RGB-D data exploits the use of Transfer Learning techniques, in order to map the single channel of depth data into traditional 3 RGB channels and then by performing a finetuning on ImageNet pretrained networks. In this thesis is presented a study for two alternative approaches: the training of a deep model from scratch directly on depth data, and a Transfer Learning method for depth→RGB mapping as a replacement of the standard finetuning procedure.

2.4 State of the Art

In this section we will perform a survey on the current state of the art methods which are most related to the works presented in the thesis:

- Domain Adaptation and Transfer Learning techniques applied on RGB images.
- Object recognition models on RGB, RGB-D or depth only data.
- Semantic segmentation models on urban scenes.

It is worth mentioning that the majority of the RGB and 2.5D works in the current era of Deep Learning exploit some use of Transfer Learning and/or Domain Adaptation techniques, even if not explicity cited. For example, the use of finetuning techniques is ubiquitous among both RGB and depth domains, as well as adopting generative (GAN) methods in any case where there is a sensible shift between training and testing domains. As such, most of the reported papers related to RGB, depth and RGB-D setting can be also seen as particular cases of applied Domain Adaptation and/or Transfer learning methods.

In **single source Domain Adaptation**, *feature adaptation* approaches aim at learning deep domain invariant representations [98, 149, 18, 19, 126, 50, 138, 59, 132]. Other methods rely on adversarial loss functions [46, 161, 135], while the method proposed in [135] is a variant of [46], where the domain difference is still measured at the feature level but passing through an image reconstruction step. Besides end-to-end trained architectures also two-step adaptive networks have shown practical advantages [162, 6]. The family of image-based strategies for domain adaptation exploit powerful *GAN* [52] methods to generate new images or perturb existing ones to resemble the visual style of a certain domain, thus reducing the discrepancy at

pixel level [16, 141]. *Generative Adversarial Networks* are composed of two modules, a generator and a discriminator. The generator synthesizes samples whose distribution closely matches that of real data, while the discriminator distinguishes real from generated samples. GANs are agnostic to the training samples labels, while conditional GAN variants [107] exploit the class annotation as additional information to both the generator and the discriminator.Some works used multiple GANs: in CoGAN [93] two generators and two discriminators are coupled by weight-sharing to learn the joint distribution of images in two different domains without using pair-wise data. Cycle-GAN [186], Disco-GAN [76] and UNIT [95] encourage the mapping between two domains to be well covered by imposing transitivity: the mapping in one direction followed by the mapping in the opposite direction should arrive where it started. For this image generation process the main performance measure is either a human-based quality control or scores that evaluate the interpretability of the produced images by pre-existing models [134, 186]. Most of work based on *image adaptation* aims at producing either target-like source images or source-like target images, but it has been recently shown that integrating both the transformation directions is highly beneficial [128, 1]. In particular [1] combines both image and feature-level adaptation, but the proposed network contains two generators, three discriminators and one classifier for which significantly increase the computational cost at training time.

Multi Source Domain Adaptation was initially studied from a theoretical point of view focusing on theorems indicating how to optimally sub-select the data to be used in learning the source models [33] or proposing principled rules for combining source-specific classifiers and obtain the ideal target class prediction [103]. Several other works followed this direction presenting algorithms based on the *combination of source models* [151, 41, 68]. A different set of approaches is based on learning new *feature representations*, defining a mapping to a latent space shared between domains. Some of them are created for single sources but can be easily extended to multiple ones [35, 54], as the feature replication method of [35] and the unsupervised approach based on Grassman manifolds presented in [54]. Other works developed instead dedicated solutions for multiple sources [65, 71]. When dealing with shallow-methods the naïve model learned by collecting all the source data in single domain without any adaptation usually shows low performance on the target. It has been noticed that this behavior changes when moving to Deep Learning, where the larger number of samples as well as their variability supports generalization and provides good results on the target. Within the context of convnet-based approaches, the vanilla solution of collecting all the source data in a single domain is already quite effective. Only very recently two methods presented multi-source deep learning approaches that improve over this baseline. The method proposed in [172] builds over [46] by replicating the adversarial domain discriminator branch for each available source; these discriminators are also used to get a perplexity score that indicates how the multiple sources should be combined at test time, according to the rule in [103]. A similar multi-way adversarial strategy is used also in [182], but this work comes with a theoretical support that frees it from the need of respecting a specific optimal source combination and thus from learning the source weights.

In the **Domain Generalization setting**, no access to the target data is allowed, thus the main objective is to look across multiple sources for shared factors in the hypothesis that they will hold also for any new target domain. Usually these factors which are searched at *model-level* to regularize the learning process on the sources, or at *feature-level* to learn some domain-shared representation. Deep model-level strategies are presented in [102, 87, 38]; in the first work a weighting procedure on the source models is proposed, while the others aim at separating the source knowledge into domain-specific and domain-agnostic sub-models either with a low-rank parametrized network or through a dedicated learning architecture with a shared backbone and source-specific aggregative modules. A meta-learning approach was recently presented in [88]: it starts by creating virtual testing domains within each source mini-batch

and then it trains a network to minimize the classification loss, while also ensuring that the taken optimization direction leads to an improvement on the virtual testing loss. Regarding feature-based methods, [114] proposed to exploit a *Siamese* architecture to learn an embedding space where samples from different source domains but same labels are projected nearby, while samples from different domains and different labels are mapped far apart. Both works [49, 89] exploit deep *autoencoders* for Domain Generalization still focusing on representation learning, but relying on image reconstruction: multi-task autoencoders are trained so that, given an image from one domain it is possible to reconstruct its analogous for all domains. More recently, new approaches based on *data augmentation* have shown promising results. Both [139] and [166] propose domain-guided perturbation of the input instances in the embedding space, with the second work able to generalize to new targets also when starting from a single source.

Style transfer methods can be seen as **Transfer Learning** techniques: in the seminal work of [48] new images were synthesized to maintain a specific content while replicating the style of one or a set of reference images. Among many others, we can cite *DeepDream* [112] which is the first attempt to produce artistic images by reversing CNN representations. [73] uses perceptual loss defined over deep convolution layers to train a transfer net. [85, 163, 164] train the texture network based on image or neural patch. [25] presents a new method based on patch swap to express style transfer in feature space. In [23], the authors extend a similar patch based method [84] to incorporate a semantic mask into neural patch swap.

Regarding the subfield of RGB-D object classification, recent advances in transfer learning techniques [39, 173, 51] made it possible to use depth data with CNN pre-trained on a database of a different modality. Several authors proposed hand-crafted mappings to colorize depth data, obtaining impressive improvements in classification over the Washington [80] database [137, 43]. At the same time, the heavy use of 2.5D depth sensors on robot platforms has generated a lively research activity on 2.5D object recognition from depth images [80, 146, 28]. Hasan et al [176] pushed on a multi-modal approach, proposing an architecture merging together RGB, depth and 3D point cloud information. In the context of RGB-D object detection, a recent stream of works explicitly addressed cross modal transfer learning through sharing of weights across architectures [67], [66] and [57] or through distillation [64]. While [57] has proved very successful in the object detection realm, it presents some constraints that might potentially be problematic in object recognition, from the requirement of paired RGB-D images, to detailed data preprocessing and preparation for training. Sun et al. [150] propose to use large scale CAD rendered data to leverage depth information without using low level features or colorization. In Asif et al.[10], hierarchical cascaded forests were used for computing grasp poses and perform object classification, exploiting several different features like orientation angle maps, surface normals and depth information colored with *Jet* method. Within the RGB-D classification literature, [13] converts the depth map to surface normals and then re-interprets it as RGB values, while Aekerberg et al.[3] builds on this approach and suggests an effective preprocessing pipeline to increase performance. Gupta, Saurabh, et al.[56] proposed *HHA*: a mapping where one channel encodes the horizontal disparity, one the height above ground and the third the pixelwise angle between the surface normal and the gravity vector. Schwarz et al [136] proposed a colorization pipeline where colors are assigned to the image pixels according to the distance of the vertexes of a rendered mesh to the center of the object. Eitel et al [43] used a simple color mapping technique known as *ColorJet*, showing this simple method to be competitive with more sophisticated approaches. All these works, as well as others like [176, 69], make use of an ad-hoc mapping for converting depth images into three channels.

In the case of **activity recognition** tasks, most of the works deal with RGB and optical flow data, avoiding to exploit corresponding depth images. [91] builds feature representations encoding egocentric knowledge such as hand pose, hand motion, head movement gaze direction and so

forth. Several approaches adopt a double stream network architecture [100, 130, 145, 155] in order to deal with the temporal aspect of the activities, with one stream encoding the processing of RGB images and the other dealing with stacked optical flow. Some of these focus on learning jointly the two streams [100], while others add an input stream encoding information related to the hand position, gaze and head motion of the actor [145]. Recently, it has been proposed with success to further use Conv-LSTM [171] to process the spatio-temporal information contained in the input video [148]. Among researches that exploited RGB-D data for object recognition, the most relevant are [109] which used depth information for modeling hands through low level skin features, and [155] which added a network branch to encode depth data, then combined to the outcome of other two branches encoding RGB knowledge and stacked optical flow.

In the recent years, the important task of **semantic segmentation** has seen a wide development, due to the availability of powerful GPUs and robust pre-trained visual features. The first work to put semantic segmentation under the deep learning spotlight was [97] that showed how fully connected networks could be used to assign a label to each image pixel. Several following works have then extended the interest around this task proposing tailored architectures which involve multi-scale feature combinations [24, 174] or integrate context information [96, 181]. The main issue with deep semantic segmentation remains that of collecting a large amount of images with pixel-based expensive annotations. Some solutions in this sense have been proposed either developing methods able to deal with weak annotations [120, 75], or leveraging on other domain images, as the synthetic ones produced by 3D renderings of urban scenes [40, 123, 125]. To avoid the drop in performance due to the synthetic to real shift, domain adaptation techniques have been integrated with approaches involving different network levels. The most widely used solution consists in adding a domain classifier used adversarially to minimize the gap among different feature spaces [26]. In [160], adversarial learning is used both on the segmentation output and on inner network features. A third family of methods applies adaptation directly at the pixel level with GAN-based style transfer techniques [1]. Other alternative strategies have focused on the introduction of critic networks to identify samples close to the classification boundary and exploit them to improve feature generalization [133], or defined a curriculum adaptation to focus first on easy and then on hard samples during the learning process [179], or even introduced tailored loss functions [187].

2.5 Datasets

The algorithms proposed in this thesis run successfully on several different images datasets. While the most used ones are RGB images datasets for object classification, some methods run on depth maps and/or combined RGB-D data collections, while a preliminary approach on semantic segmentation is developed on the three most used RGB semantic segmentation datasets for urban scenes. The rest of the section illustrate the chosen datasets with their main characteristics by grouping them into RGB, RGB-D and RGB/semantic segmentation groups.

2.5.1 RGB

- **MNIST** [83] contains centered, 28×28 pixel, grayscale images of single digit numbers on a black background, and it has been constructed from NIST's databases which contain binary images of handwritten digits.
- **MNIST-M** [46] is a MNIST tougher variant where the background is substituted by a randomly extracted patch obtained from color photos of BSDS500 [8].
- **USPS** [45] is a digit dataset automatically scanned from envelopes by the U.S. Postal

Service containing a total of 9,298 16×16 pixel grayscale samples. The images are centered, normalized and show a broad range of font styles.

- **SVHN** [116] is the challenging real-world Street View House Number dataset. It contains over $600k$ 32×32 pixel color samples. Besides presenting a great variety of shapes and textures, images from this dataset often contain extraneous numbers in addition to the labeled, centered one. While most previous works simplified the data by considering a grayscale version, instead we applied our methods to the original RGB images.

- **German Traffic Signs Recognition Benchmark GTSRB** [147] consists of $51,839$ 32×32 cropped images of German traffic signs, created for a single-image classification challenge held at the *International Joint Conference on Neural Networks (IJCNN) 2011.*

- **Synthetic Signs collection** [111] contains $100k$ samples of common street signs obtained from Wikipedia and artificially transformed to simulate various imaging conditions, in a 32×32 pixel format, with class labels compatible with GTSRB dataset [147].

2.5.2 RGB-D

- **Washington RGB-D** [80] consists of $41,877$ RGB-D images organized into 300 instances divided in 51 classes. Each object instance was positioned on a turntable and captured from three different viewpoints while rotating. Since two consecutive views are extremely similar, only 1 frame out of 5 is used for evaluation purposes, as in [80].

- **JHUIT-50** [86] is a challenging recent dataset that focuses on the problem of fine-grained recognition. It contains 50 object instances, often very similar with each other (e.g. 9 different kinds of screwdrivers). As such, it presents different classification challenges compared to the Washington database.

- **BigBIRD**[144] is the biggest RGB-D dataset we considered: it contains 121 object instances and 75.000 images. Unfortunately, it is an extremely unforgiving dataset for evaluating depth features: many objects are extremely similar, and many are boxes, which are indistinguishable without texture information. To partially mitigate this, we grouped together all classes annotated with the same first word: for example *nutrigrain apple cinnamon* and *nutrigrain blueberry* were grouped into *nutrigrain*. With this procedure, we reduced the number of classes to 61 (while keeping all of the samples). As items are quite small, we used the object masks provided by [144] to crop around the object. Evaluation-wise, we followed the protocol defined in [86].

- **GUN-71**[124] contains 12,000 RGB-D images divided into 71 categories, representing everyday grasps in natural interactions. It is a challenging dataset, where 28 objects per grasp have been used, for a total of 28x71 = 1988 different hand-object configurations. The grasp actions are performed by 8 different subjects (4 males and 4 females) and are made in 5 different houses.

- **WCVS**[110] contains egocentric RGB-D images collected by 4 different users in 2 different indoor scenarios. The action sequences are manually labeled with 3 different granularity level for activity labels: for example, manipulation actions have been grouped into similar actions according to how the hands and arms are located (from the first-person perspective) while performing them: *Twohands*, *One-hand*, *Pick-up*, and *Others*.

2.5.3 RGB Semantic Segmentation

- **Cityscapes** [32] is a real-world, vehicle-egocentric image dataset collected in 50 cities in Germany and nearby countries. It provides a training set made of 2,993 images as well as 503 images for validation purpose, having 2048×1024 resolution. All the training, validation, and test images are accurately annotated with per pixel category labels by human experts. We followed the VisDA Semantic Segmentation challenge protocol, focusing on 19 labeled classes.

- **GTA5** [123] is composed by 24,966 images with resolution 1914×1052, synthesized from the homonym video game and set in Los Angeles. Ground truth and annotations are compatible with the Cityscapes dataset [32] that contains 19 categories. Depending on the role of the dataset in the experiments we used either all the available images (as source) or a 500 sample subset (as target).

- **Synthia** [125] is made of 9400 images at 1280×760 resolution compatible with the Cityscapes dataset, but covering only 16 object categories. Even if the virtual city used to generate the synthetic images does not correspond to any of the real cities covered by Cityscapes, Synthia shows almost photo-realistic frames with different light conditions and weather, multiple season, and a great variety of dynamic objects. With the same approach of GTA5, we used the full dataset for training and the first 500 images while testing.

Chapter 3

Depth modality

This chapter presents several methods for tackling the problem of domain adaptation and transfer learning in the depth domain. The first one is *DepthNet* (Section 3.1): a deep convolutional neural network trained on a novel depth dataset (VANDAL) made by rendered depth images starting from freely distributed 3D CAD models from the web. The second algorithm is *DECO* (Section 3.2): a deep module capable of learning the optimal depth→RGB mapping for depth images object classification. The third algorithm in Section 3.3, *LODEAR*, is a simple but effective depth integration method for first person action recognition in short videos.

3.1 DepthNet

3.1.1 Introduction

Deep learning has changed the research landscape in visual object recognition over the last few years. Since their spectacular success in recognizing $1,000$ object categories [79], convolutional neural networks have become the new off the shelf state of the art in visual classification. Since then, the robot vision community has also attempted to take advantage of the deep learning trend, as the ability of robots to understand what they see reliably is critical for their deployment in the wild. A critical issue when trying to transfer results from computer to robot vision is that robot perception is tightly coupled with robot action. Hence, pure RGB visual recognition is not enough.

Object recognition from RGB-D data traditionally relied on hand-crafted features such as SIFT [99] and spin images [80], combined together through vector quantization in a Bag-of-Words encoding [80]. This heuristic approach has been surpassed by end-to-end feature learning architectures, able to define suitable features in a data-driven fashion [12, 146, 9]. All these methods have been designed to cope with a limited amount of training data (of the order of $10^3 - 10^4$ depth images), thus they are able to only partially exploit the generalization abilities of deep learning as feature extractors experienced in the computer vision community [78, 140], where databases of 10^6 RGB images like ImageNet [127] or Places [184] are available.

An alternative route is that of re-using deep learning architectures trained on ImageNet [79] through pre-defined encoding [56] or colorization. Since the work of [137] re-defined the state of the art in the field, this last approach has been actively and successfully investigated. Eitel et al [43] proposed a parallel CNN architecture, one for the depth channel and one for the RGB one, combined together in the final layers through a late fusion scheme. Some approaches coupled non linear learning methods with various forms of spatial encodings, and the encoding of an implicit multi scale representation through a rich coarse-to-fine feature extraction approach [29, 27, 28, 86].

All these works build on top of CNNs pre-trained over ImageNet, for all modal channels. Thus, the very same filters are used to extract features from all of them. As empirically successful as this might be, it is a questionable strategy, as RGB and depth images are perceptually very different, and as such they would benefit from approaches able to learn data-specific features (Figure 3.1).

Is this the best we can do? What if one would train from scratch a CNN over a very large scale 2.5D object categorization database, wouldn't the filters learned be more suitable for object recognition from depth images? RGB images are perceptually very rich, with generally a strong presence of textured patterns, especially in ImageNet. Features learned from RGB data are most likely focusing on those aspects, while depth images contain more information about the shape and the silhouette of objects. Unfortunately, as of today a 2.5D object categorization database large enough to train a CNN on it does not exist. A likely reason for this is that gathering such data collection is a daunting challenge: capturing the same variability of ImageNet over the same number of object categories would require the coordination of very many laboratories, over an extended period of time.

With DepthNet we follows an alternative route. Looking at the promising emerging trand of using realistic synthetic data in conjunction with deep learning architectures [119, 105, 170, 60], we propose to use synthetic data as a proxy for training a deep learning architecture specialized in learning depth specific features. To this end, we constructed the VANDAL database, a collection of 4.5 million depth images from more than $9,000$ objects, belonging to 319 categories. The depth images are generated starting from 3D CAD models, downloaded from the Web, through a protocol developed to extract the maximum information from the models. VANDAL is used as input to train from scratch a deep learning architecture, obtaining a pre-trained model able to act as a depth specific feature extractor, called DepthNet. Visualizations of the filters learned by the first layer of the architecture show that the filters we obtain are indeed very different from those learned from ImageNet with the very same convolutional neural network architecture (Figure 3.1). As such, they are able to capture different facets of the perceptual information available from real depth images, more suitable for the recognition task in that domain.

Experimental results on two publicly available databases confirm this: when using only depth, DepthNet features achieve better performance compared to previous methods based on a CNN pre-trained over ImageNet, without using fine tuning or spatial pooling. The combination of the DepthNet features with the descriptors obtained from the CNN pre-trained over ImageNet, on both depth and RGB images, leads to strong results on the Washington database [80], and to results competitive with fine-tuning and/or sophisticated spatial pooling approaches on the JHUIT database [86]. Moreover, this is actually the first work that uses synthetically generated depth data to train a depth-specific convolutional neural network.

3.1.2 The VANDAL dataset

With $4.5M$ synthetic images, VANDAL it is the largest existing depth database for object recognition. Starting from the consideration that CNNs trained on ImageNet have been shown to generalize well when used on other object centric datasets, we defined a list of object categories as a subset of the ILSVRC2014 list [127], removing by hand all scenery classes, as well as objects without a clear default shape such as clothing items or animals. This resulted in a first list of roughly 480 categories, which was used to query public 3D CAD model repositories like 3D Warehouse, Yeggi, Archive3D, and many others. Being aware of the domain shift between the training CAD models and the target real data domain, we adopted a twofold strategy: in a first step we cleaned the model as much as possible, in order to remove irrelevant items (as walls, people standing next to the object, etc), and in a second step we performed a strong data augmentation routine, which increased the variety of object shapes and viewpoints, thus

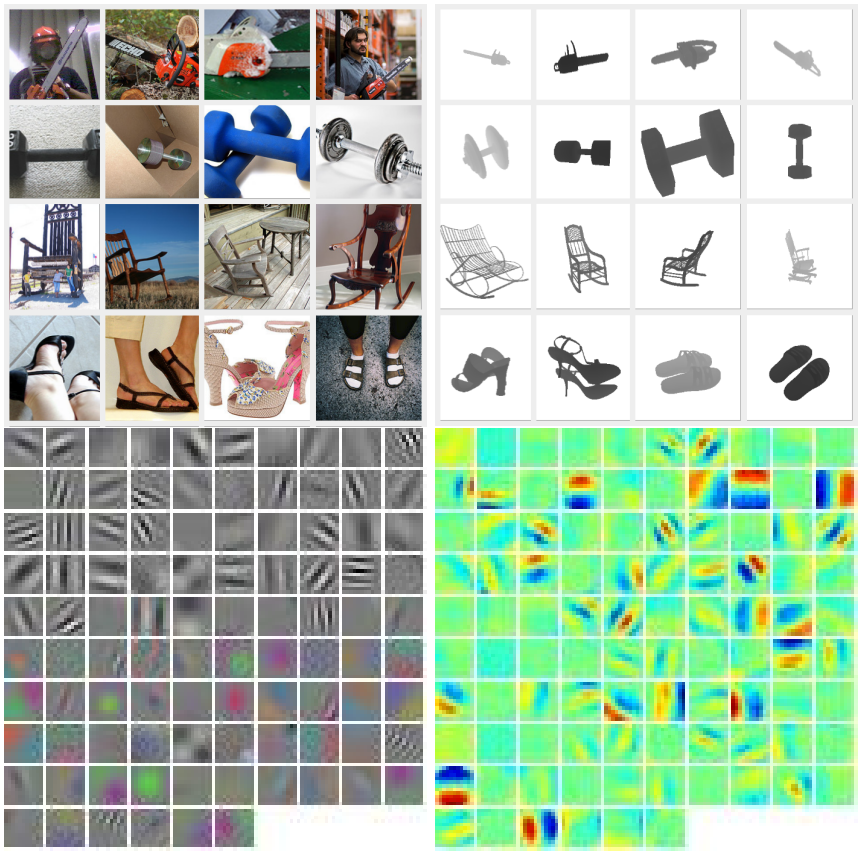


Figure 3.1. Sample images for the classes chainsaw, dumbbell, rocker chair and sandal from ImageNet (a) and VANDAL (d). We show the corresponding filters learned by the very same CNN architecture respectively in (b) and (c) (note that this is colorized for easier viewing). We see that even though the architecture is the same, using 2D rather than 2.5D images for training leads to learning quite different filters. In (c) some of the features appear to be undefined.

Figure 3.2. Sample morphs (center, right) generated from an instance model for the category coffee cup (left).

creating at training timethe most general visual features as possible. In order to do so, five volunteers[1] manually downloaded the models, cleaning the CAD models from any spurious object and supporting surfaces, and running a script to harmonize the size of all models (some of them were originally over 1GB per file). They were also required to create significantly morphed

[1]Graduate students from the MARR program at DIAG, Sapienza Rome University.

(a) The 319 categories in VANDAL **(b)** Some examples from the classes more populated

Figure 3.3. The VANDAL database. On the left, we show a word cloud visualization of the classes in it, based on the numbers of 3D models in each category. On the right, we show exemplar models for the six categories more populated: coffee cup, vase, pencil case, computer mouse, table and chair.

variations of the original 3D CAD models, whenever suitable. Figure 3.2 shows examples of morphed models for the object category coffee cup. Finally, we removed all categories with less than two models, ending up with 319 object categories with an average of 30 models per category, for a total of 9,383 CAD object models. Figure 3.3, left, gives a world cloud visualization of the VANDAL dataset, while on the right it shows examples of 3D models for the 6 most populated object categories.

All depth renderings were created using Blender [44], with a Python script fully automating the procedure, and then saved as 8bit grayscale .png files, using the convention that black is close and white is far.

The depth data generation protocol was designed to extract as much information as possible from the available 3D CAD models. This concretely means obtaining the greatest possible variability between each rendering. The approach commonly used by real RGB-D datasets consists in fixing the camera at a given angle and then using a turntable to get all possible viewpoints of the object [80, 86]. We tested here a similar approach, but we found out using perceptual hashing[2] that a significant number of object categories had more than 50% nearly identical images.

We defined instead a configuration space consisting of:

- (a) object distance from the camera,
- (b) focal length of the camera,
- (c) camera position on the sphere defined by the distance, and
- (d) slight ($< 10\%$) random morphs along the axes of the model.

Figure 3.4 illustrates the described configuration space. This protocol ensured that almost none of the resulting images were identical. Object distance and focal length were constrained to make sure the object always appears in a recognizable way. We sampled this configuration space with roughly 480 depth images for each model, obtaining a total of 4.5 million images.

[2] `http://www.phash.org/`

Preliminary experiments showed that increasing the sampling rate in the configuration space did lead to growing percentages of nearly identical images.

The rendered depth images consist of objects always centered on a white background, as it allows us the maximum freedom to perform various types of data augmentation at training time, which is standard practice when training convolutional neural networks. This is here even more relevant than usual, as synthetically generated data are intrinsically perceptually less informative. The data augmentation methods we used are: image cropping, occlusion (1/4 of the image is randomly occluded to simulate gaps in the sensor scan), contrast/brightness variations (which allows us to train our network to deal with both raw and normalized data), in depth views corresponding to scaling the Z axis and shifting the objects along it, background substitution (substituting the white background with one randomly chosen farther away than the object's center of mass), random uniform noise (as in film grain), and image shearing (a slanting transform). While not all of these data augmentation procedures produce a realistic result, they all contribute as regularizers; we avoided modeling a more complex sensor model noise for efficiency reasons. Figure 3.5 shows some examples of data augmentation images obtained with this protocol.

Once the VANDAL database has been generated, it is possible to use it to train any kind of convolutional deep architecture. In order to allow for a fair comparison with previous work, we

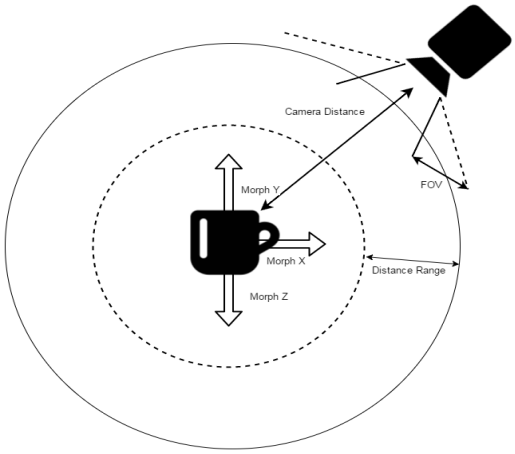


Figure 3.4. Configuration space used for generating renderings in the VANDAL database.

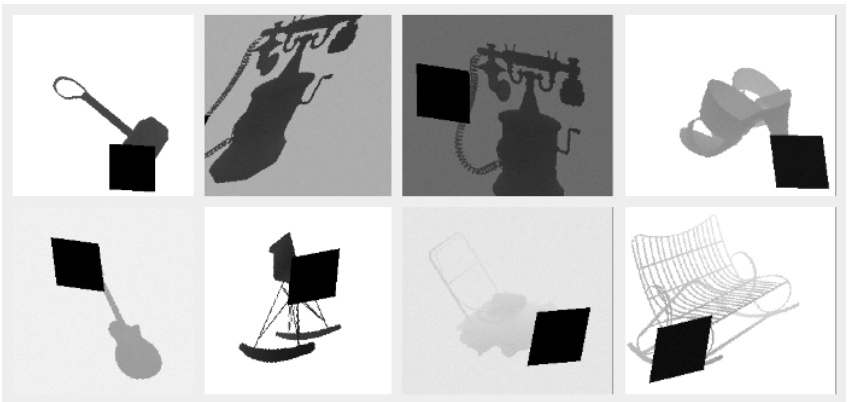


Figure 3.5. Data augmentation samples from various classes (hammer, phone, sandal, guitar, rocker, lawn mower, bench).

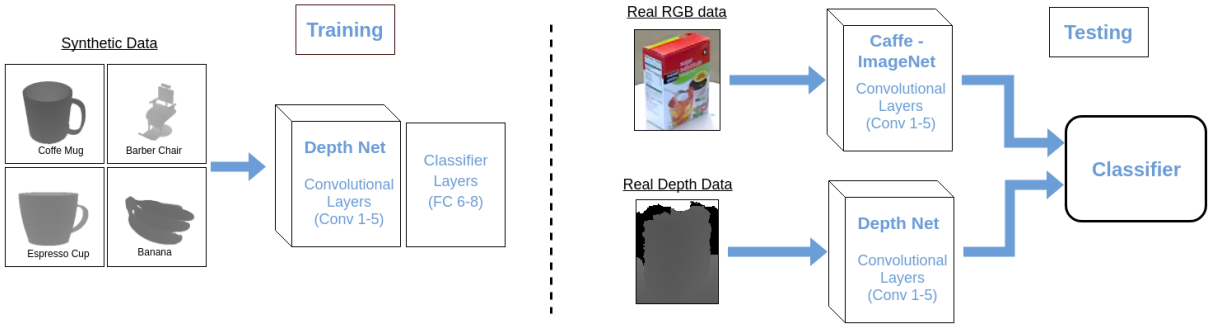


Figure 3.6. DepthNet and our associated RGB-D object classification framework. During training, we learn depth filters from the VANDAL synthetic data (left). During test (right), real RGB and depth data is processed by two distinct CNNs, each specialized over the corresponding modality. The features, derived from the activations of the fifth convolutional layer, are then fed into a cue integration classifier.

opted for CaffeNet, a slight variation of AlexNet [79] with slightly better performance. Although more modern networks have been proposed in the last years [143, 153, 62], it still represents the most popular choice among practitioners, and the most used in robot vision. Its well know architecture consists of 5 convolutional layers, interwoven with pooling, normalization and ReLU layers, plus three fully connected layers. [3]

Although the standard choice in robot vision is using the output of the seventh activation layer as feature descriptors, several studies in the vision community show that lower layers, like the sixth and the fifth, tend to have higher generalization properties [183]. We followed this trend, and opted for the fifth layer (by vectorization) as deep depth feature descriptor (an ablation study supporting this choice is reported in Section 3.1.3). We name in the following as **DepthNet** the CaffeNet architecture trained on VANDAL using as output feature the fifth layer, and **Caffe-ImageNet** the same architecture trained over ImageNet.

Once DepthNet has been trained, it can be used as any depth feature descriptor, alone or in conjunction with Caffe-ImageNet for classification of RGB images. We explore this last option, proposing a system for RGB-D object categorization that combines the two feature representations through a multi kernel learning classifier [118]. Figure 3.6 gives an overview of the overall RGB-D classification system. Note that DepthNet can be combined with any other RGB and/or 3D point cloud descriptor, and that the integration of the modal representations can be achieved through any other cue integration approach. This underlines the versatility of DepthNet, as opposed to recent work where the depth component was tightly integrated within the proposed overall framework, and as such unusable outside of it [43, 176, 28, 86].

3.1.3 Experiments

We assessed the DepthNet, as well as the associated RGB-D framework of Figure 3.6, on two publicly available databases. Section 3.2.3 describes our experimental setup and the databases used in our experiments.Section 3.2.4.1 reports a set of experiments assessing the performance of DepthNet on depth images, compared to Caffe-ImageNet, while in Section 3.2.4.3 we assess the performance of the whole RGB-D framework with respect to previous approaches. We conducted experiments on the Washington RGB-D [80] and the JHUIT-50 [86] object datasets, already described in Section 2.5.

All experiments, as well as the training of DepthNet, were done using the publicly available

[3] CaffeNet differs from AlexNet in the pooling, which is done there before normalization.

Caffe framework [72]. As described above, we obtained DepthNet by training a CaffeNet over the VANDAL database. The network was trained using Stochastic Gradient Descent for 50 epochs. Learning rate started at 0.01 and gamma at 0.5 (halving the learning rate at each step). We used a variable step down policy, where the first step took 25 epochs, the next 25/2, the third 25/4 epochs and so on. These parameters were chosen to make sure that the test loss on the VANDAL test data had stabilized at each learning rate. Weight decay and momentum were left at their standard values of 0.0005 and 0.9. Training and test data was centered by removing the mean pixel.

To assess the quality of the DepthNet features we performed three set of experiments:

1. *Object classification using depth only:* features were extracted with DepthNet and a linear SVM[4] was trained on it. We also examined how the performance varies when extracting from different layers of the network, comparing against a Caffe-ImageNet used for depth classification, as in [137].

2. *Object classification using RGB + Depth:* in this setting we combined our depth features with those extracted from the RGB images using Caffe-ImageNet. While [43] train a fusion network to do this, we simply use an off the shelf Multi Kernel Learning (MKL) classifier [118].

For all experiments we used the training/testing splits originally proposed for each given dataset. For linear SVM, we set C by cross validation. When using MKL, we left the default values of 100 iterations for online and 300 for batch and set p and C by cross validation.

Previous works using Caffe-ImageNet as feature extractor for depth, apply some kind of input preproccessing [137, 43, 176]. While we do compare against the published baselines, we also found that by simply normalizing each image (min to 0 and max to 255), one achieves very competitive results. Also, since our DepthNet is trained on depth data, it does not need any type of preprocessing over the depth images, obtaining strong results over raw data. Because of this, in all experiments reported in the following we only consider raw depth images and normalized depth images.

3.1.4 Results

In this section we present experiments on RGB-D data, from both the Washington and JHUIT databases, assessing the performance of our DepthNet-based framework of Figure 3.6 against previous approaches. Table 3.1 shows in the top row our results, followed by results obtained by Caffe-ImageNet using the pool5 activations as features, as well as results from the recent literature based on convolutional neural networks. First, we see that the results in the RGB column stresses once more the strength of the pool5 activations as features: they achieve the best performance without any form of fine tuning, spatial pooling or sophisticated non-linear learning, as done instead in other approaches [43, 176, 28]. Second, DepthNet on raw depth data achieves the best performance among CNN-based approaches with or without fine tuning like [137, 43], but it is surpassed by approaches encoding explicitly spatial information through pooling strategies, and/or by using a more advanced classifier than a linear SVM, as we did. We would like to stress that we did not incorporate any of those strategies in our framework on purpose, to better assess the sheer power of training a given convolutional architecture on perceptually different databases. Still, nothing prevents in future work the merging of DepthNet with the best practices in spatial pooling and non-linear classifiers, with a very probable further increase in performance. Lastly, we see that in spite of the lack of such powerful tools, our

[4]Liblinear: http://www.csie.ntu.edu.tw/ cjlin/liblinear/

framework achieves the best performance on RGB-D data. This clearly underlines that the representations learned by DepthNet are both powerful and able to extract different nuances from the data than Caffe-ImageNet. Rather than the actual overall accuracy reported here in the table, we believe this is the breakthrough result we offer to the community in this work.

Method:	RGB	Depth Mapping	RGB-D
DepthNet RGB-D Framework	$\mathbf{88.49 \pm 1.8}$	81.68 ± 2.2	$\mathbf{92.25 \pm 1.3}$
Caffe-ImageNet Pool5	$\mathbf{88.49 \pm 1.8}$	81.11 ± 2	90.79 ± 1.2
Caffe-ImageNet FC7 finetuning[43]	84.1 ± 2.7	83.8 ± 2.7	91.3 ± 1.4
Caffe-ImageNet FC7[137]	83.1 ± 2.0	–	89.4 ± 1.3
CNN only[28]	82.7 ± 1.2	78.1 ± 1.3	87.5 ± 1.1
CNN + FisherKernel + SPM[28]	86.8 ± 2.2	$\mathbf{85.8 \pm 2.3}$	91.2 ± 1.5
CNN + Hypercube Pyramid + EM[176]	87.6 ± 2.2	85.0 ± 2.1	91.4 ± 1.4
CNN-SPM-RNN+CT[29]	85.2 ± 1.2	83.6 ± 2.3	90.7 ± 1.1
CNN-RNN+CT[27]	81.8 ± 1.9	77.7 ± 1.4	87.2 ± 1.1
CNN-RNN[146]	80.8 ± 4.2	78.9 ± 3.8	86.8 ± 3.3

Table 3.1. Comparison of our DepthNet framework with previous work on the Washington database. With *depth mapping* we mean all types of depth preprocessing used in the literature.

Experiments over the JHUIT database confirms the findings obtained over the Washington collection (Table 3.2). Here our RGB-D framework obtains the second best result, with the state of the art achieved by the proposers of the database with a non CNN-based approach. Note that this database focuses over the fine-grained classification problem, as opposed to object categorization as explored in the experiments above. While the results reported in Table 3.2 on Caffe-ImageNet using FC7 seem to indicate that the choice of using pool5 remains valid, the explicit encoding of local information is very important for this kind of tasks [177, 7]. We are inclined to attribute to this the superior performance of [86]; future work incorporating spatial pooling in our framework, as well as further experiments on the object identification task in the Washington database and on other RGB-D data collections will explore this issue.

3.1.4.1 Ablation study

We present here an ablation study, aiming at understanding the impact of choosing features from the last fully convolutional layer as opposed to the more popular last fully connected layer, and of using normalized depth images instead of raw data. By comparing our results with those obtained by Caffe-ImageNet, we also aim at illustrating up to which point the features learned from VANDAL are different from those derived from ImageNet.

Figure 3.7 shows results obtained on the Washington database, with normalized and raw depth data, using as features the activations of the fifth pooling layer (pool5), of the sixth fully connected layer (FC6), and of the seventh fully connected layer (FC7). Note that this last set of activations is the standard choice in the literature. We see that for all settings, pool5 achieves the best performance, followed by FC6 and FC7. This seems to confirm recent findings on RGB data [183], indicating that pool5 activations offer stronger generalization capabilities when used as features, compared to the more popular FC7. The best performance is obtained by DepthNet, pool5 activations over raw depth data, with a 83.8% accuracy. DepthNet achieves also better results compared to Caffe-ImageNet over normalized data. To get a better feeling of how performance varies when using DepthNet or Caffe-ImageNet, we plotted the per-class accuracies obtained using pool5 and raw depth data. We sorted them in descending order according to the

Caffe-ImageNet scores (Figure 3.8).

While there seems to be a bulk of objects where both features perform well (left), DepthNet seems to have an advantage over challenging objects like apple, onion, ball, lime and orange (right), where the round shape tends to be more informative than the specific object texture. This trend is confirmed also when performing a t-SNE visualization [101] of all the Washington classes belonging to the high-level categories 'fruit' and 'device' (Figure 3.9). We see that in general the DepthNet features tend to cluster tightly the single categories while at the same time separating them very well. For some classes like dry battery and banana, the difference between the two representations is very marked. This does not imply that DepthNet features are always better than those computed by Caffe-ImageNet.Figure 3.8 shows that CaffeNet features obtain a significantly better performance compared to DepthNet over the classes binder and mushroom, to name a few. The features learned by the two networks seem to focus on different perceptual aspects of the images. This is likely due to the different set of samples used during training, and the consequent different learned filters (Figure 3.1).

From these figures we can draw the following conclusions: (a) DepthNet provides the overall stronger descriptor for depth images, regardless of the activation layer chosen and the presence or not of preprocessing on the input depth data; (b) the features derived by the two networks

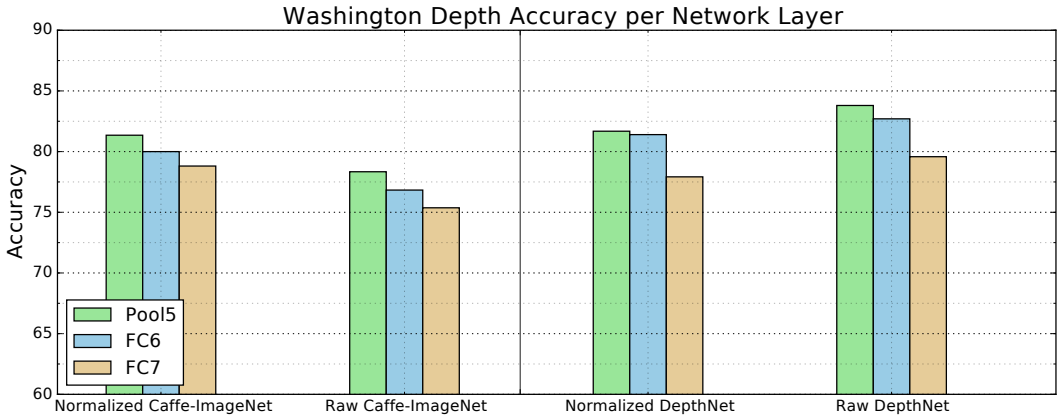


Figure 3.7. Accuracy obtained by DepthNet and Caffe-ImageNet over the Washington database, using as features pool5, FC6 and FC7. Results are reported for raw and normalized depth images.

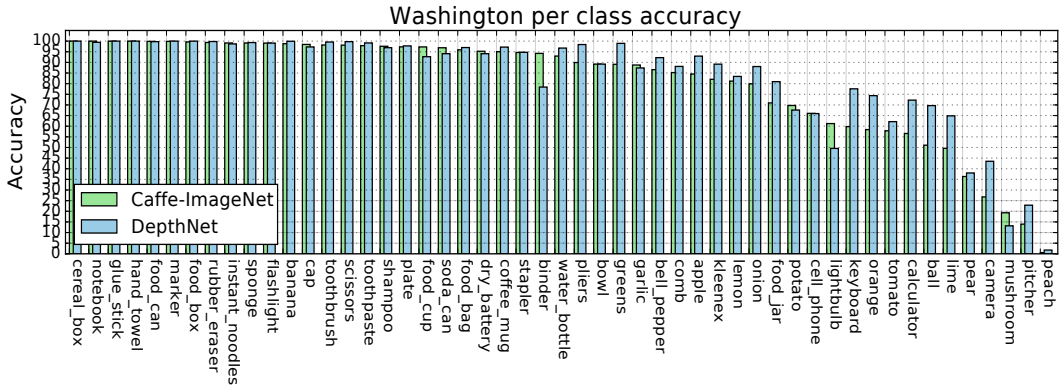
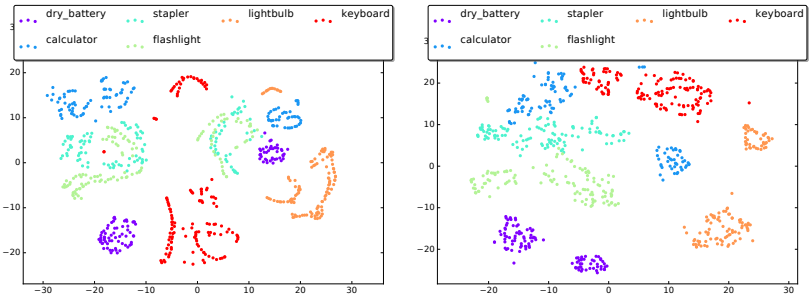


Figure 3.8. Accuracy per class on the Washington dataset, depth images. Classes sorted by the Caffe-ImageNet accuracies.

(a) Device classes as seen by Caffe-ImageNet (left) and DepthNet (right)



(b) Fruit classes as seen by Caffe-ImageNet (left) and DepthNet (right)

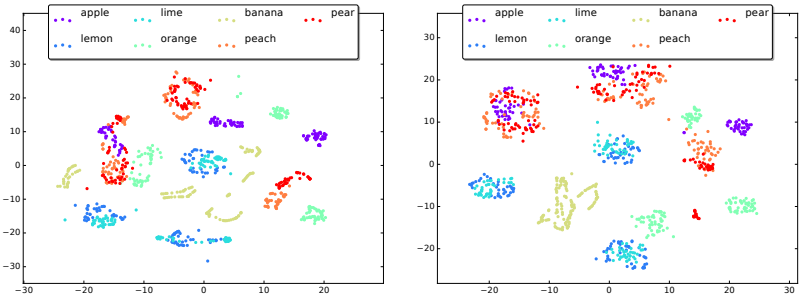


Figure 3.9. t-SNE visualizations for the categories device (top) and fruit (bottom).

tend to capture different features of the data, and as such are complementary.

3.1.4.2 Computational Analysis

The whole training of DepthNet on the VANDAL dataset took 4 days (for 50 epochs) with a single Nvidia 1080 GPU. The training phase have been slowed by the intensive use of online data augmentation (performed by the CPU), which helped to increase the features robustness and the final performance. Training for longer times, increasing the total number of epochs to 100, did not show any increase of accuracy, both on the VANDAL training set and on the Washington test set. However, during testing the DepthNet model behave pratically as a regular Alexnet/Caffenet from a computational point of view, with an inference time of ~10ms for a single image batch which makes DepthNet suitable for any real time use.

3.1.5 Conclusions

In this work we focused on object classification from depth images using convolutional neural networks. We argued that, as effective as the filters learned from ImageNet are, the perceptual features of 2.5D images are different, introducing a sensible domain shift which can be reduced by training on images with the same modality. To this purpose, we created VANDAL, the first depth image database synthetically generated, and DepthNet, the first deep convolutional neural network trained on synthetic depth data from scratch. We showed experimentally that the features derived from such data, using the very same CaffeNet architecture widely used over ImageNet, are stronger while at the same time complementary to them.

We see this work as the very beginning of a long research thread. By its very nature, DepthNet could be plugged into all previous work using CNNs pre-trained over ImageNet for extracting depth features. It might substitute that module, or it might complement it; the open issue is when this will prove beneficial in terms of spatial pooling approaches, learning methods

Method:	RGB	Depth Mapp.	Depth Raw	RGB-D
DepthNet Pool5	–	**54.37**	**55.0**	**90.3**
Caffe-ImageNet Pool5	88.05	53.6	38.9	89.6
Caffe-ImageNet FC7[137]	82.08	47.87	26.11	83.6
CSHOT + Color pooling + MultiScale Filters[86]	–	–	–	**91.2**
HMP[86]	81.4	41.1	–	74.6

Table 3.2. Comparison of our DepthNet framework with previous work on the JHUIT database. As only one split is defined, we do not report std.

and classification problems. A second issue we plan to investigate is the impact of the deep architecture over the filters learned from VANDAL. While in this work we chose on purpose to not deviate from CaffeNet, it is not clear that this architecture, which was heavily optimized over ImageNet, is able to exploit at best our synthetic depth database. While preliminary investigations with existing architectures have not been satisfactory, we believe that architecture surgery might lead to better results. Furthermore, including the 3D models from [170] might greatly enhance the generality of the DepthNet. Finally, we believe that the possibility to use synthetic data as a proxy for real images opens up a wide array of possibilities: for instance, given prior knowledge about the classification task of interest, would it be possible to generate on the fly a task specific synthetic database, containing the object categories of interest under very similar imaging conditions, and train and end-to-end deep network on it? How would performance change compared to the use of network activations as done today? Future work will focus on these issues.

3.2 DECO

3.2.1 Introduction

As we have seen in the previous section, robots need to recognize what they see around them to be able to act and interact with it. Recognition must be carried out in the RGB domain, capturing mostly the visual appearance of things related to their reflectance properties, as well as in the depth domain, providing information about the shape and silhouette of objects and supporting both recognition and interaction with items. The current mainstream state of the art approaches for object recognition are based on Convolutional Neural Networks (CNNs, [82]), which use end-to-end architectures achieving feature learning and classification at the same time. Some notable advantages of these networks are their ability to reach much higher accuracies on basically any visual recognition problem, compared to what would be achievable with heuristic methods; their being domain-independent, and their conceptual simplicity. Despite these advantages, they also present some limitations, such as high computational cost, long training time and the demand for large datasets, among others.

This last issue has so far proved crucial in the attempts to leverage over the spectacular success of CNNs over RGB-based object categorization [152, 79] in the depth domain. Being CNNs data-hungry algorithms, the availability of very large scale annotated data collections is crucial for their success, and architectures trained over ImageNet [37] are the cornerstone of the vast majority of CNN-based recognition methods. Besides the notable exception of [20], the mainstream approach for using CNNs on depth-based object classification has been through transfer learning, in the form of a mapping able to make the depth input channel compatible with the data distribution expected by RGB architectures.

Looking from another point of view, the task of depth→RGB mapping can be related to the colorization of grayscale images using deep nets: Cheng et al [30] proposed a colorization pipeline based on three different hand-designed feature extractors to determine the features from different levels of an input image. Larsson et al [81] used an architecture consisting of two parts. The first part is a fully convolutional version of VGG-16 used as feature extractor, and the second part is a fully-connected layer with 1024 channels predicting the distributions of hue and the chroma for each pixel given its feature descriptors from the previous level. Iizuka et al [70] proposed an end-to-end network able to learn global and local features, exploiting the classification labels for better image colorization. Their architecture consists of several networks followed by fusion layer for the colorization task.

In the field of depth colorization, the majority of works exploit an ad-hoc mapping [56, 136, 43, 176], but we argue that this strategy is sub-optimal. By hand-crafting the mapping for the depth data colorization, one has to make strong assumptions on what information, and up to which extent, should be preserved in the transfer learning towards the RGB modality. While some choices might be valid for some classes of problems and settings, it is questionable whether the family of algorithms based on this approach can provide results combining high recognition accuracies with robustness across different settings and databases. Inspired by recent works on colorization of gray-scale photographs [70, 81, 30], we tackle the problem by exploiting the power of end-to-end convolutional networks, proposing a deep depth colorization architecture able to learn the optimal transfer learning from depth to RGB for any given pre-trained convnet. Our deep colorization network takes advantage of the residual approach [62], learning how to map between the two modalities by leveraging over a reference database (Figure 3.10, top), for any given architecture. After this training stage, the colorization network can be added on top of its reference pre-trained architecture, for any object classification task (Figure 3.10, bottom). We call our network (DE)^{2}CO: DEep DEpth COlorization.

(DE)^{2}CO performance can be assessed in several ways. A first qualitative analysis, comparing the colorized depth images obtained by (DE)^{2}CO and by other state of the art hand-crafted approaches, gives intuitive insights on the advantages brought by learning the mapping as opposed to choosing it, over several databases. We further deepen this analysis with an experimental evaluation of our and other existing transfer learning methods on the depth channel only, using four different deep architectures and three different public databases, with and without fine-tuning. Finally, we tackle the RGB-D object recognition problem, combining (DE)^{2}CO with off-the shelf state of the art RGB deep networks, benchmarking it against the current state of the art in the field. For all these experiments, results clearly support the value of DECO algorithm.

3.2.2 Colorization of depth images

Although depth and RGB are modalities with significant differences, they also share enough similarities (edges, gradients, shapes) to make it plausible that convolutional filters learned from RGB data could be re-used effectively for representing colorized depth images. The approach currently adopted in the literature consists of designing ad-hoc colorization algorithms, as revised in the previous section. We refer to these kind of approaches as *hand-crafted depth colorization*. Specifically, we choose ColorJet [43], *SurfaceNormals* [13] and *SurfaceNormals++* [3] as baselines against which we will assess our data driven approach because of their popularity and effectiveness.

In the rest of the section we first briefly summarize ColorJet (Section 3.2.2.1), SurfaceNormals and SurfaceNormals++ (Section 3.2.2.2). We then describe our deep approach to depth colorization (Section 3.2.2.3). To the best of our knowledge, (DE)^{2}CO is the first deep colorization architecture applied successfully to depth images.

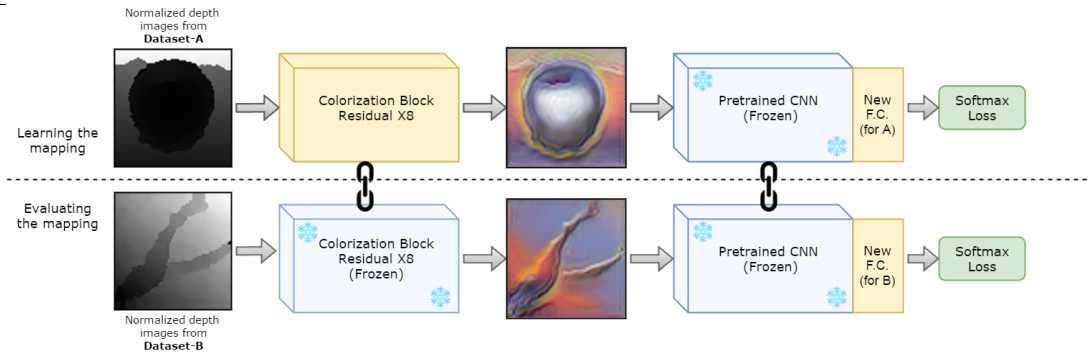


Figure 3.10. The (DE)^{2}CO pipeline consists of two phases. First, we learn the mapping, from depth to color, maximizing the discrimination capabilities of a network pre trained on ImageNet. In this step the network is frozen and we are only learning the mapping and the final layer. We then evaluate the colorization on a *different* depth dataset: here we also freeze the colorization network and only train a new final layer for the testbed dataset.

3.2.2.1 Hand-Crafted Depth Colorization: ColorJet

ColorJet works by assigning different colors to different depth values. The original depth map is firstly normalized between 0-255 values. Then the colorization works by mapping the lowest value to the blue channel and the highest value to the red channel. The value in the middle is mapped to green and the intermediate values are arranged accordingly [43]. The resulting image exploits the full RGB spectrum, with the intent of leveraging at best the filters learned by deep networks trained on very large scale RGB datasets like ImageNet. Although simple, the approach gave very strong results when tested on the Washington database, and when deployed on a robot platform. Still, ColorJet was not designed to create realistic looking RGB images for the objects depicted in the original depth data (Figure 3.12, bottom row). This raises the question whether this mapping, although more effective than other methods presented in the literature, might be sub-optimal. In Section 3.2.2.3 we will show that by fully embracing the end-to-end philosophy at the core of deep learning, it is indeed possible to achieve significantly higher recognition performances while at the same time producing more realistic colorized images.

3.2.2.2 Hand-Crafted Depth Colorization: SurfaceNormals

The Surface normals mapping has been often used to convert depth images to RGB [13, 169, 43]. The process is straightforward: for each pixel in the original image the corresponding surface normal is computed as a normalized $3D$ vector, which is then treated as an RGB color. Due to the inherent noisiness of the depth channel, such a direct conversion results in noisy images in the color space. To address this issue, the mapping we call *SurfaceNormals++* was introduced by Aakerberg [3]: first, a recursive median filter is used to reconstruct missing depth values, subsequently a bilateral filter smooths the image to reduce noise, while preserving edges. Next, surface normals are computed for each pixel in the depth image. Finally the image is sharpened using the unsharp mask filter, to increase contrast around edges and other high-frequency components.

3.2.2.3 Deep Depth Colorization: (DE)^{2}CO

(DE)^{2}CO consists of feeding the depth maps, normalized into grayscale images, to a *colorization network* linked to a standard CNN architecture, pre-trained on ImageNet.

Given the success of deep colorization networks from grayscale images, we first tested existing architectures in this context [178]. Extensive experiments showed that while the visual appearance of the colorized images was very good, the recognition performances obtained when combining such network with pre-trained RGB architectures was not competitive. Inspired by the generator network in [14], we propose here a *residual* convolutional architecture (Figure 3.11). By design [62], this architecture is robust and allows for deeper training. This is helpful here, as $(DE)^2CO$ requires stacking together two networks, which for not very deep architectures might lead to vanishing gradient issues. Furthermore, residual blocks works at pixel level [14] helping to preserve locality.

Our architecture works as follows: the 1x228x228 input depth map, reduced to 64x57x57 size by a conv&pool layer, passes through a sequence of 8 residual blocks, composed by 2 small convolutions with batch normalization layers and leakyRelu as non linearities. The last residual block output is convolved by a three features convolution to form the 3 channels image output. Its resolution is brought back to 228x228 by a *deconvolution* (upsampling) layer.

Our whole system for object recognition in the depth domain using deep networks pre-trained over RGB images can be summarized as follows: the entire network, composed by $(DE)^2CO$ and the classification network of choice, is trained on an annotated reference depth image dataset. The weights of the chosen classification network are kept frozen in their pre-trained state, as the only layer that needs to be retrained is the last fully connected layer connected to the softmax layer. Meanwhile, the weights of $(DE)^2CO$ are updated until convergence.

After this step, the depth colorization network has learned the mapping that maximizes the classification accuracy on the reference training dataset. It can now be used to colorize *any* depth image, from any data collection. Figure 3.12, top rows, shows exemplar images colorized with $(DE)^2CO$ trained over different reference databases, in combination with two different architectures (CaffeNet, an implementation variant of AlexNet, and VGG-16 [143]). We see that, compared to the images obtained with ColorJet and SurfaceNormal++, our colorization technique emphasizes the objects contours and their salient features while flatting the object background, while the other methods introduce either high frequency noise (SurfaceNormals++) or emphasize background gradient instead of focusing mainly on the objects (ColorJet). In the next subsection we will show how this qualitative advantage translates also into a numerical advantage, i.e. how learning $(DE)^2CO$ on one dataset and performing depth-based object recognition on another leads to a very significant increase in performance on several settings, compared to hand-crafted color mappings.

3.2.3 Experiments

We evaluated our colorization scheme on three main settings: an ablation study of how different depth mappings perform when the network weights are kept frozen (Section 3.2.4.1), a comparison of depth performance with network finetuning (Section 3.2.4.2) and finally an assessment of $(DE)^2CO$ when used in RGB-D object recognition tasks (Section 3.2.4.3).

Databases We considered three datasets: the Washington RGB-D [80], the JHUIT-50 [86] and the BigBIRD [144] object datasets, which are the main public datasets for RGB-D object recognition. Dataset details can be found in Section 2.5.

Hand-crafted Mappings According to previous works [43, 3], the two most effective mappings are *ColorJet* [43] and *SurfaceNormals* [13, 3]. For ColorJet we normalized the data between 0 and 255 and then applied the mapping using the OpenCV libraries[5]. For the SurfaceNormals mapping we considered two versions: the straightforward conversion of the depthmap to surface normals [13] and the enhanced version *SurfaceNormals++*[3] which uses

[5] "COLORMAP_JET" from *http://opencv.org/*

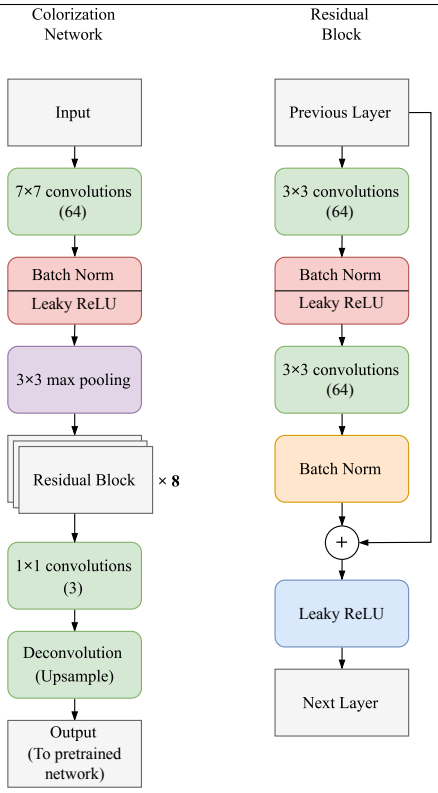


Figure 3.11. Overview of the (DE)^{2}CO colorization network. On the left, we show the overall architecture; on the right, we show details of the residual block.

extensive pre-processing and generally performs better[6].

3.2.4 Results

3.2.4.1 Ablation Study

In the ablation study we compared our (DE)^{2}CO method against hand crafted mappings, using pre-trained networks as feature extractors and only retraining the last classification layer. We did this on the three datasets described above, over four architectures: CaffeNet (a slight variant of the AlexNet [78]), VGG16 [143] and GoogleNet [152] were chosen because of their popularity within the robot vision community. We also tested the recent ResNet50 [62], which although not currently very used in the robotics domain, has some promising properties. In all cases we considered models pretrained on ImageNet [37], which we retrieved from Caffe's *Model Zoo*[7].

Training (DE)^{2}CO means minimizing the multinomial logistic loss of a network trained on RGB images. This means that our network is attached between the depth images and the pre-trained network, of which we freeze the weights of all but the last layer, which are relearned from scratch (see Figure 3.10). We trained each network-dataset combination for 50 epochs using the Nesterov solver [115] and 0.007 starting learning rate (which is stepped down after 45%). During this phase, we used the whole *source* datasets, leaving aside only 10% of the samples for validation purposes.

When the dataset on which we train the colorizer is different from the test one, we simply retrain the new final layer (freezing all the rest) for the new classes.

[6]The authors graciously gave us their code for our experiments.
[7]*https://github.com/BVLC/caffe/wiki/Model-Zoo*

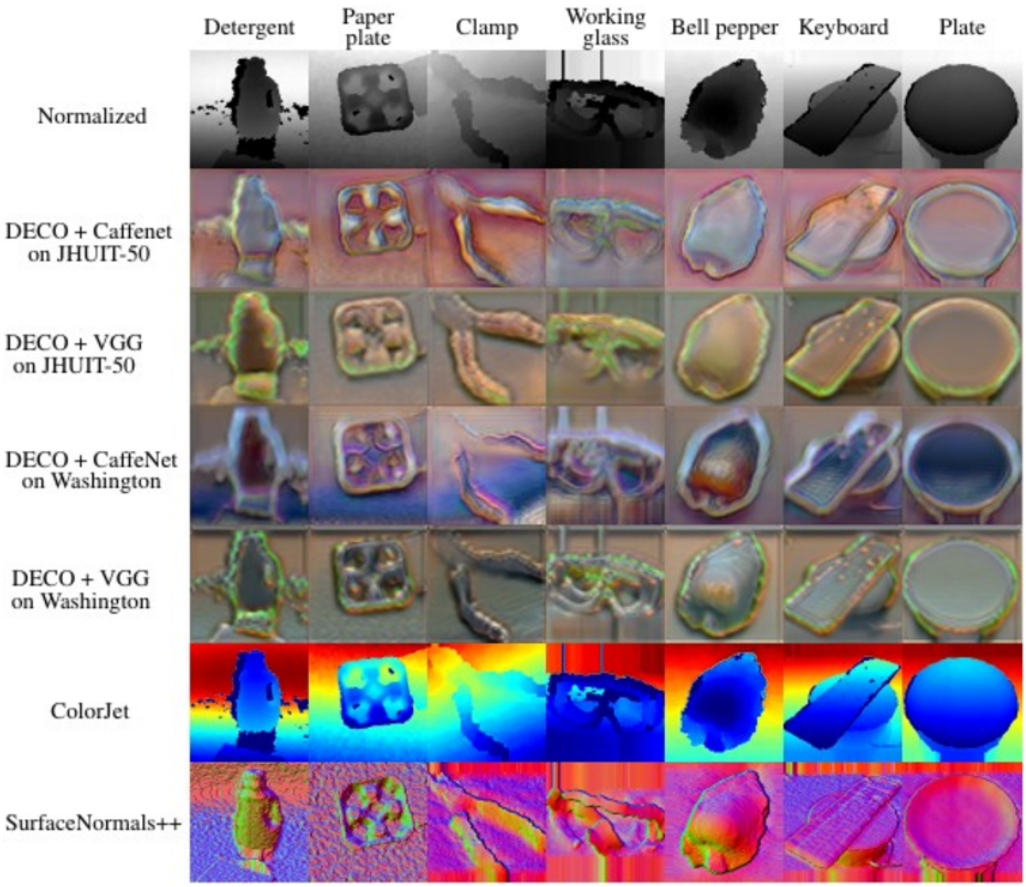


Figure 3.12. $(DE)^2CO$ colorizations applied on different objects, taken from[86, 80, 144]. Top row shows the depth maps mapped to grayscale. From the second to the fourth row, we show the corresponding $(DE)^2CO$ colorizations learned on different settings. Fifth row shows *ColorJet* views [43], while the last row shows the surface normals mapping [3] *SurfaceNormals++*. These images showcase $(DE)^2CO$'s ability to emphasize the object's shape and to capture its salient features.

Effectively we are using the pre-trained networks as feature extractors, as done in [136, 43, 176] and many others; for a performance analysis in the case of network finetuning we refer to Section 3.2.4.2. In this setting we used the Nesterov (for Washington and JHUIT-50) and ADAM (for BigBIRD) solvers. As we were only training the last fully connected layer, we learned a small handful of parameters with a very low risk of overfitting.

Table 3.4 reports the results from the ablation while Figure 3.13 focuses on the class recall for a specific experiment. For every architecture, we report the results obtained using ColorJet, SurfaceNormals (plain and enhanced) and (DE)^{2}CO learned on a reference database between Washington or JHUIT-50, and (DE)^{2}CO learned on the combination of Washington and JHUIT-50. For the CaffeNet and VGG networks we also present results on simple grayscale images. We attempted also to learn (DE)^{2}CO from BigBIRD alone, and in combination with one (or both) of the other two databases. Results on BigBIRD only were disappointing, and results with/without adding it to the other two databases did not change the overall performance. We interpret this result as caused by the relatively small variability of objects in BigBIRD with respect to depth, and for sake of readability we decided to omit them in this work.

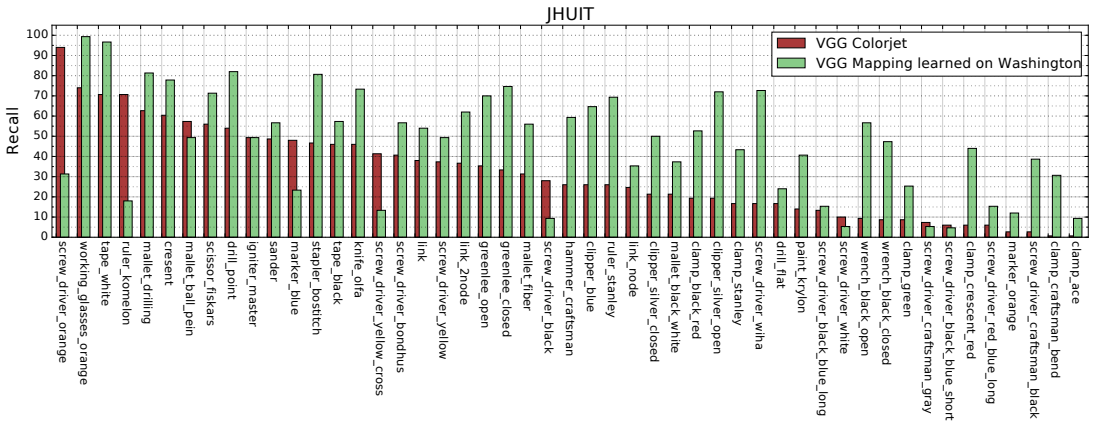


Figure 3.13. Per class recall on JHUIT-50, using VGG, with (DE)^{2}CO learned from Washington. Recalls per class are sorted in decreasing order, according to ColorJet performance. In this setting, (DE)^{2}CO, while generally performing better, seems to focus on different perceptual properties and is thus, compared with the baseline, better at some classes rather than others.

We see that, for all architectures and for all reference databases, (DE)^{2}CO achieves higher results. The difference goes from +1.7%, obtained with CaffeNet on the Washington database, to the best of +16.8% for VGG16 on JHUIT-50. JHUIT-50 is the testbed database where, regardless of the chosen architecture, (DE)^{2}CO achieves the strongest gains in performance compared to hand crafted mappings. Washington is, for all architectures, the database where hand crafted mappings perform best, with the combination Washington to CaffeNet being the most favorable to the shallow mapping. On average it appears the CaffeNet is the architecture that performs best on this datasets; still, it should be noted that we are using here all architectures as feature extractors rather than as classifiers. On this type of tasks, both ResNet and GoogLeNet-like networks are known to perform worse than CaffeNet [11], hence our results are consistent with what reported in the literature. In Table 3.5 we report a second ablation study performed on the width and depth of (DE)^{2}CO architecture. Starting from the standard (DE)^{2}CO made of 8 residual blocks with 64 filters for each convolutional layer (which we found to be the best all-around architecture), we perform additional experiments by doubling and halving the number of residual blocks and the number of filters in each convolutional layer. As it can be seen, the

Network	Time (ms)
CaffeNet	695
VGG	1335
GoogleNet	1610
ResNet-50	1078
$(DE)^2CO$ colorizer	400

Network	Time (s)
CaffeNet	1.87
$(DE)^2CO$ + CaffeNet	1.23
VGG	2.91
$(DE)^2CO$ + VGG	2.16

Table 3.3. Left: forward-backward time for 50 iterations, as by *caffe time*. Right: feature extraction times for 100 images; note that using $(DE)^2CO$ actually speeds up the procedure. We explain this by noting that $(DE)^2CO$ uses single channel images and thus needs to transfer only $\frac{1}{3}$ of the data from memory to the GPU - clearly the bottleneck here.

$(DE)^2CO$ architecture is quite robust but can be potentially finetuned to each target dataset to further increase performance.

Method:	Washington[80]	JHUIT-50[86]	BigBIRD[144]
VGG16 on Grayscale	74.9	33.7	22
VGG16 on ColorJet	75.2	35.3	19.9
VGG16 on SurfaceNormals	75.3	30.8	16.8
VGG16 on SurfaceNormals++	77.3	35.8	11.5
VGG16 $(DE)^2CO$ learned on Washington	**79.6**	**52.7**	22.8
VGG16 $(DE)^2CO$ learned on JHUIT-50	78.1	51.2	23.7
CaffeNet on Grayscale	76.6	44.6	22.9
CaffeNet on ColorJet	78.8	45.0	22.7
CaffeNet on SurfaceNormals	79.3	38.3	18.9
CaffeNet on SurfaceNormals++	81.4	44.8	14.0
CaffeNet $(DE)^2CO$ learned on Washington	**83.1**	53.1	**28.6**
CaffeNet $(DE)^2CO$ learned on JHUIT-50	79.1	**57.5**	25.2
GoogleNet on ColorJet	73.5	40.0	21.8
GoogleNet on SurfaceNormals	72.9	36.5	18.4
GoogleNet on SurfaceNormals++	**76.7**	41.5	13.9
GoogleNet $(DE)^2CO$ learned on Washington	–	**51.9**	**25.2**
GoogleNet $(DE)^2CO$ learned on JHUIT-50	76.6	–	24.4
ResNet50 on ColorJet	75.1	38.9	18.7
ResNet50 on SurfaceNormals	77.4	33.2	16.5
ResNet50 on SurfaceNormals++	**79.6**	45.4	13.8
ResNet50 $(DE)^2CO$ learned on Washington	–	**45.5**	23.9
ResNet50 $(DE)^2CO$ learned on JHUIT-50	76.4	–	**24.7**

Table 3.4. Object classification experiments in the depth domain, comparing $(DE)^2CO$ and hand crafted mappings, using 5 pre-trained networks as feature extractors. Best results for each network-dataset combination are in **bold**, overall best in **red bold**. Extensive experiments were performed on VGG and Caffenet, while GoogleNet and ResNet act as reference.

filters/blocks	4 blocks	8 blocks	16 blocks
32 filters	56.5	52.8	57.1
64 filters	56.8	53.1	53.6
128 filters	53.1	53.9	53.3

Table 3.5. $(DE)^2CO$ ablation study, learned on Washington, tested on JHUIT-50. Grid search optimization over width and depth of generator architecture shows improved results.

3.2.4.2 Finetuning

In our finetuning experiments we focused on the best performing network from the ablation, the CaffeNet (which is also used by current competitors [43, 3]), to see up to which degree the network could exploit a given mapping. The protocol was quite simple: all layers were free to move equally, the starting learning rate was 0.001 (with step down after 45%) and the solver was *SGD*. Training went on for 90 epochs for the Washington and JHUIT-50 datasets and 30 eps. for BigBIRD (a longer training was detrimental for all settings). To ensure a fair comparison with the static mapping methods, the $(DE)^2CO$ part of the network was kept frozen during finetuning.

Results are reported in Table 3.6. We see that here the gap between hand-crafted and learned colorization methods is reduced (very likely the network is compensating existing weaknesses). *SurfaceNormals++* performs pretty well on Washington, but less so on the other two datasets (it's actually the worse on BigBIRD). Surprisingly, the simple grayscale conversion is the one that performs best on BigBIRD, but lags clearly behind on all other settings. $(DE)^2CO$ on the other hand, performs comparably to the best mapping on every single setting **and** has a 5.9% lead on JHUIT-50; we argue that it is always possible to find a shallow mapping that performs very well on a specific dataset, but there are no guarantees it can generalize.

3.2.4.3 RGB-D classification

While this work focuses on how to best perform recognition in the depth modality using convnets, we wanted to provide a reference value for RGB-D object classification using $(DE)^2CO$ on the depth channel. To classify RGB images we follow [3] and use a pretrained VGG16 which we finetuned on the target dataset (using the previously defined protocol). RGB-D classification is then performed, without further learning, by computing the weighted average (weight factor α was cross-validated) of the *fc8* layers from the RGB and Depth networks and simply selecting the most likely class (the one with the highest activations). This cue integration scheme can be seen as one of the simplest, off-the-shelf algorithm for doing classifications using two different modalities [159]. We excluded BigBIRD from this setting, due to lack of competing works to compare with.

Results are reported in Table 3.7 and Table 3.8. We see that $(DE)^2CO$ produces results on par or slightly superior to the current state of the art, even while using an extremely simple feature fusion method. This is remarkable, as competitors like [3, 43] use instead sophisticated, deep learning based cue integration methods. Hence, our ability to compete in this setting is all due to the $(DE)^2CO$ colorization mapping, clearly more powerful than the other baselines. It is worth stressing that, in spite of the different cue integration and depth mapping approaches compared in Table 3.7 and Table 3.8, convnet results on RGB are already very high, hence in this setting the advantage brought by a superior performance on the depth channel tends to be covered. Still, on Washington we achieve the second best result, and on JHUIT-50 we get the new state of the art.

Method:	Washington[80]	JHUIT-50[86]	BigBIRD[144]
Grayscale colorization	82.7 ± 2.1	53.7	**29.6**
ColorJet colorization	83.8 ± 2.7	54.1	25.4
SurfaceNormals++ colorization	**84.5** ± 2.9	55.9	17.0
$(DE)^2CO$ learned on Washington	84.0 ± 2.0	60.0	–
$(DE)^2CO$ learned on JHUIT-50	82.3 ± 2.3	**62.0**	–
$(DE)^2CO$ learned on Washington + JHUIT-50	84.0 ± 2.3	61.8	28.0

Table 3.6. CaffeNet finetuning using different colorization techniques.

Method:	RGB	Depth	RGB-D
FusionNet[43]	84.1 ± 2.7	83.8 ± 2.7	91.3 ± 1.4
CNN + Fisher [90]	**90.8** ± 1.6	81.8 ± 2.4	**93.8** ± 0.9
DepthNet [20]	88.4 ± 1.8	83.8 ± 2.0	92.2 ± 1.3
CIMDL [169]	87.3 ± 1.6	84.2 ± 1.7	92.4 ± 1.8
FusionNet enhanced[3]	89.5 ± 1.9	**84.5** ± 2.9	93.5 ± 1.1
$(DE)^2CO$	89.5 ± 1.6	84.0 ± 2.3	93.6 ± 0.9

Table 3.7. Selected results on Washington RGB-D.

Method:	RGB	Depth	RGB-D
DepthNet [20]	88.0	55.0	90.3
Beyond Pooling [86]	–	–	91.2
FusionNet enhanced[3]	**94.7**	56.0	95.3
$(DE)^2CO$	**94.7**	**61.8**	**95.7**

Table 3.8. Selected results on JHUIT-50.

3.2.4.4 Computational Analysis

While the training of the $(DE)^2CO$ module took only few hours, One of the concerns for using our method as a replacement to the standard finetuning approach was the computational burden that the $(DE)^2CO$ modules could add at test time. However, we got surprising inference performance which can be seen in Table 3.3, where we report runtimes for the considered networks. As the results show, while $(DE)^2CO$ requires some extra computation time, this is actually offset by the fact that only $\frac{1}{3}$ of the data is being moved to the GPU w.r.t. a depth image mapped into a 3 channel RGB image, thus *decreasing* the total inference time; this makes $(DE)^2CO$ the fastest solution for real time 2.5D recognition.

3.2.5 Conclusions

This section introduced a network for learning deep colorization mappings. Our $(DE)^2CO$ architecture follows the residual philosophy, learning how to map depth data to RGB images for a given pre-trained convolutional neural network. By using $(DE)^2CO$ as opposed to the hand-crafted colorization mappings commonly used in the literature, we obtained a significant jump in performance over three different benchmark databases, using four different popular deep networks pre trained over ImageNet. The visualization of the obtained colorized images further confirms how our algorithm is able to capture the rich informative content and the different facets of depth data.

Future work will further investigate the effectiveness and generality of our approach, testing

it on other RGB-D classification and detection problems, with various fine-tuning strategies and on several deep networks, pre-trained over different RGB databases, and in combination with RGB convnet with more advanced multimodal fusion approaches.

3.3 LODEAR

3.3.1 Introduction

Within the current fast-growing range of successful applications of computer vision, those related to action and activity recognition from videos are among the most popular. This in turns drives research towards algorithms able to perform these tasks in various scenarios, from video summarization to action understanding, to automatic video indexing and retrieval, human robot interaction and many others. Although most of the work done so far has focused on third-person action and activity recognition, the recent diffusion of wearable devices able to acquire seamlessly videos has elicited attention towards first-person videos. First person activity recognition, i.e. the recognition of fine-grained actions performed by the camera's wearer such as pouring, opening, lifting etc, is more challenging than third person action recognition due to the lack of information about the actor's pose. Moreover, sharp movements by the wearer make it unfeasible the use of tracking algorithms, while at the same time it causes big shakes in videos. To cope with these challenges, the community has focused on modeling and recognizing hands and the handled objects from the videos, proposing several shallow and deep approaches [106, 110, 156, 124, 167].

Within this context, several authors pointed out the importance of using 3D information, going beyond the use of RGB data only. This has been done using 3D point clouds [47] or depth images [109, 155] in various forms, combined with standard images. The work presented in this section follows this trend, proposing a deep architecture that uses depth information to prime RGB-based first-person activity recognition. Specifically, we propose to use depth information as an attention mechanism to guide the network towards learning selective features, able to recognize robustly the hand motion patterns as well as the object being manipulated. We do so by blending depth and RGB images together so to weight the RGB channels according to the depth information, and thus guide the deep learning of activity recognition according to this knowledge. We called our model *LODEAR*. After this first step, we implement a CNN-RNN network trained in a weakly supervised fashion, but without the optical flow branch, as the initial depth blending already enables a Conv-LSTM block (as proposed in [148]) to capture effectively important knowledge about the hand-object interactions. We instead further embrace this weakly supervised scenario, and we introduce a Mean Teacher model [157] that allows to leverage over multiple iterations and makes ultimately the overall architecture more robust. To the best of our knowledge, this is the first work in first-person activity recognition that (a) uses depth as an attention mechanism to guide the learning process, and (b) makes use of self labeling with a Mean Teacher algorithm. Experiments on two commonly used benchmark databases, accompanied by an ablation study, confirm the value of our approach, achieving the new state of the art.

3.3.2 Method

The proposed method builds on the architecture presented in [148], with two important novelties: (1) a simple scheme for leveraging over depth information by using it as an attention map over RGB pixel space; (2) the use of a Mean Teacher approach that is able to further increase the overall architecture performance. To the best of our knowledge, this is the first use of Mean teacher models on first-person action recognition tasks.

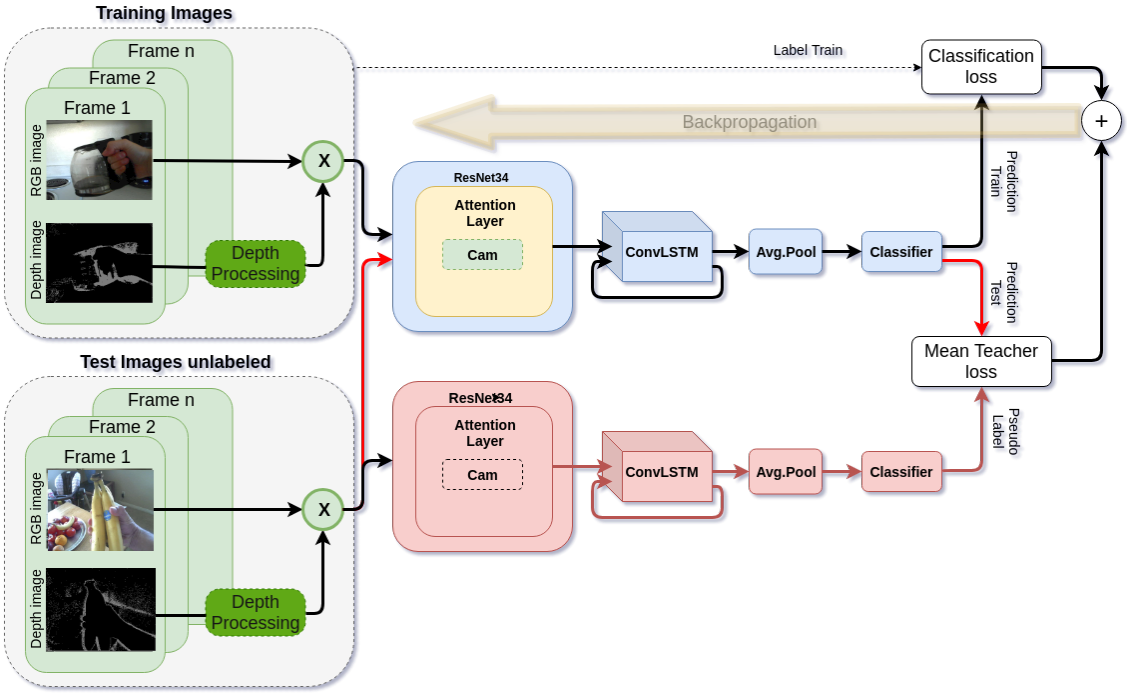


Figure 3.14. Overview of the proposed architecture. The blue blocks represent the full deep model, made by backbone, Conv-LSTM layer and classifier. The red blocks show the Mean Teacher model, built from the main model through temporal mean average, which provides pseduo-labels on the test set. Green blocks show depth-RGB integration for each sequence frame (light green). Finally, the white blocks represent the two losses used during training.

3.3.3 Depth and RGB integration schema

A simple blending schema is applied to the RGB and depth input data, resulting into an RGB-enriched image. This is in contrast with widely used methods that leverage RGB-D data by using a multiple network approach ([21], [43]). This blending step allows us to exploit depth information in order to weight the information contained in the RGB image according to each pixel depth, while keeping a single architecture approach and thus avoiding the burden of a multi branch architecture.

Specifically, the depth data is first normalized into the range $[0, 1]$ (respectively from the furthest to the nearest). The depth pixel values are then clamped between $[min, max]$ values to avoid too much extreme values, which could result into loss of background information or into a focus on the part extremely close to the camera stronger than intended. This process gives us the DepthMap (Figure 3.15, center).

After this pre-processing of the depth, we multiply each DepthMap with its corresponding RGB image. This corresponds to exploiting each depth image as an attention map at the pixel level, pushing the focus on the most central part of the frame (central part in term of depth). The result of this operation is an enhanced RGB image that can be normally processed by an RGB backbone architecture (RGB*DepthMap). Figure 3.15 shows examples of the original RGB image, the corresponding depth data, and the enhanced RGB*DepthMap image obtained with our procedure. The overall blending process can be summarized by the following equation:

$$depth_{i,[1,0]} = 1 - depth_{i,[0,1]} \quad \text{with} \quad depth_{i,[0,1]} \in [0,1], \quad \forall i \mid i = 1,.,.,nFrames \tag{3.1}$$

$$DepthMap_i = \begin{cases} min & depth_{i,[1,0]} < min \\ depth_{i,[1,0]} & min < depth_{i,[1,0]} < max \\ max & depth_{i,[1,0]} > max \end{cases} \tag{3.2}$$

$$InputNetwork_i = RGB_i * DepthMap_i \tag{3.3}$$

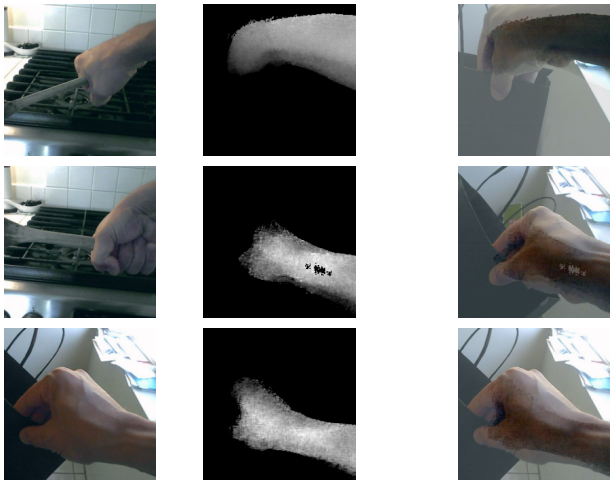
3.3.4 Feature processing

The RGB-enriched frames are processed by a backbone architecture. We chose a ResNet-34 network, pre-trained on ImageNet, as it demonstrated to be a powerful and stable backbone network [148] while at the same time being not computationally intensive. For each RGB frame, we compute the Class Activation Maps (CAM) [185]. The CAM guides the network in determining which part of the frame is supporting classification of a given class action. For each class c, the corresponding CAM can be represented as:

$$M_c(i) = \sum_l w_l^c f_l(i) \tag{3.4}$$

where $f_l(i)$ is the activation of a unit l in the final convolutional layer at spatial location i, while w_l^c represents the weight corresponding to class c for unit l. The CAM is then converted into a probability map, and then applied at feature level as a spatial attention map convolved with the output of the backbone network [148].

After feeding each sequence frame to the above steps, a *Conv-LSTM* layer [171] is responsible of processing the sequence of features obtained from the last layer of the backbone, with a final fully connected layer as classifier.



(a) RGB (b) DepthMap (c) RGB*DepthMap

Figure 3.15. Comparison of RGB, DepthMap and RGB*DepthMap images. The enriched RGB*DepthMap images clearly show how the Depth integration works as an attention map at pixel level, by focusing more on the image zones where the action takes place.

3.3.5 Self labeling through Mean Teacher

The overall method can be further enhanced by using a self labeling technique that leverages on a Mean Teacher model [157]. Starting from a fully trained architecture, we increase the final accuracy of the model by building a Mean Teacher consisting of model weights averaged over training iterations, and by using this Mean Teacher to provide label predictions of the test set. These predictions can be effectively used as pseudo labels [128] to train the main student model on the test set. With this approach both the student and the Mean Teacher model keep improving their accuracy.

3.3.6 Experiments

This subsection reports experiments validating our architecture, describing our experimental setup, i.e. the databases chosen, the algorithms we compared ourselves against and the training protocol we followed in our experiments. Section 3.3.7 reports our results, while Section 3.3.7.1 details an ablation study illustrating the workings of our approach.

3.3.6.1 Databases

All the experiments reported in this section are performed on two RGB-D first person datasets: the Grasp Understanding dataset (GUN-71, [124]) and the Wearable Computer Vision Systems dataset (WCVS, [110]).

- GUN-71 contains 12,000 RGB-D images divided into 71 categories, representing everyday grasps in natural interactions. It is a challenging dataset, where 28 objects per grasp have been used, for a total of 28x71 = 1988 different hand-object configurations. The grasp actions are performed by 8 different subjects (4 males and 4 females) and are made in 5 different houses. To perform the experiments we followed the leave-one-out cross-validation protocol, training our model on the first seven subjects and testing on the eighth. We repeated the procedure for all subjects and reported the average accuracy as a result.

Granularity	Action Labels	
Level 1	**Manipulation**	**Non-Manipulation**
Level 2	One hand	Walk Talk
	Two hands	Stairs Seating
	Pick-up	Stand
	Others	Screen

Figure 3.16. Left: Hierarchy of Action Labels in the WCVS database, Right: Examples of images corresponding to the granularity used in our experiments.

- WCVS contains data collected by 4 different users in 2 different indoor scenarios. The action sequences are manually labeled with 3 different granularity level for activity labels. Figure 3.16, table on the left shows the first two levels of granularity of the labels. At the third level, the manipulation actions of the second level, are divided in finer categories. The low frequency of these labels at this level, makes it impossible to train a classifier. For this reason we evaluate our method on the manipulation action annotated at the level two, as done in [156], following the same leave-one-subject-out cross validation scheme.

The two datasets are very different, presenting different challenges, hence they allow us to show the robustness of our method. Indeed, the first dataset (GUN-71) has many classes and the actions are labeled with respect to the grasp made by each subject, independently of the object used. As opposed to this, the structure of the second dataset (WVCS) represents a great challenge due to the large intra-class variations caused by multiple users and scenarios, although in the setting we use it consists of only 4 classes.

3.3.6.2 Baselines

We compare our method with other existing approaches for first person action recognition and more generic algorithms. Starting from the work of [148], we evaluated their method using only the RGB information on both datasets. Later we used the same network for Depth images by using two identical networks for both modalities and concatenating the final features for the prediction.
Differently, the method proposed by [156] presents a structure already suitable to exploit different input modalities. In their work the authors proposed a network that aims to preserve the distinctive parts of each individual input modality and simultaneously exploit their shared information. The influence on the final prediction of each individual mode and of the shared information is weighed through some manually set hyper-parameters. In the case of the WVCS, optical flow is also used as input modality. Note this information cannot be extracted from the GUN-71 dataset. To improve performance, a hand detection module is added to the network.
Other simpler networks such as VGG [142] and Resnet [63] have been used to evaluate their performance and show how these types of networks are not sufficient to analyze video sequences. In addition, different fusion methods have been tested, such as the Score Fusion [155] method, where roughly averages the classification probabilities of the two modalities are made, and the Features Fusion or DCCA [5] method where the correlation between the two modalities is maximized.

3.3.6.3 Training details

We trained the whole architecture with a three-stage procedure: during the first stage only the Conv-LSTM and the final classification layer are trained, while the ResNet-34 pretrained on ImageNet weights are kept frozen, effectively using this network just as a feature extractor, while in a second stage the last residual block convolutional filters are trained jointly with the Conv-LSTM and the final classifier layer. Finally, after the model has reached full convergence, the Mean Teacher model is activated to further enhance the accuracy (third stage).

It is worth underlying that the Mean Teacher model is never trained by a backpropagation signal, hence it requires only a small computation overhead and it does not increase the complexity of the architecture (which still remains a single-network one), while constantly increasing the accuracy performance.

We found that an alignment error between RGB and Depth images is often present in both datasets. Moreover, we don't want to produce an alignment too tight between RGB and Depth images, because this could produce a focus too strong on a very narrow zone of the image. For these reasons, we applied a random zoom and shift on the depth images during the training phase, and a zoom with a constant factor during the test phase, which effectively helped to reduce information loss related to those issues. A standard data augumentation protocol of horizontal flipping and random cropping to the whole RGB*DepthMap image has also been used.

The optimizer chosen to train our model for all stages is ADAM, with batch size equal to 32. The learning rate used during the three stages are the following: 0.01 (0.001 for WCVS), $0.0001, 0.0001$; with two step decays of 0.1 for each stage.

In order to standardize the sequences length over the dataset we have chosen to use 6 frames per sequence for both datasets. For the first one, in situations where there were fewer frames than necessary, we repeated them by cycling until a 6 frame sequence is obtained, while for the second dataset, which often show big sequences of very similar frames, we have selected the 6 frames per video in a uniform way over time.

3.3.7 Results

In this section the results obtained by testing our architecture on the GUN-71 and WCVS datasets are presented, while comparing them with the most effective methods present in the literature. Looking at the results in Table 3.9, we see how the accuracy of the Depth-only methods is significantly lower then the RGB-only ones, where the differences between the two modalities can reach up to 10%. Moreover, RGB + Depth integration must be carefully planned, as it can produce worse performance w.r.t RGB only. This is for instance the case for Appearance Stream, that in the RGB-only case achieves a better performance than when applied on RGB+Depth (Feature Fusion). Nevertheless, in other cases it can produce consistent improvements.

Our method provides the new state of the art on both datasets, ranging from +3% on WCVS to a +5% on GUN-71 w.r.t the current state of the art algorithms. The Mean Teacher loss is able to produce a boost of 2.5 ~3.5% of accuracy depending on the dataset. A detailed analysis of the Depth integration is presented in the following paragraph.

We would like to finally note that in the MDNN+TSN case [156],the authors combined two different methods, obtaining a result equal to 71.83%. It is worth stressing that the use of an ensemble method reliably increases the performance compared to the use a single network [55]. In spite of this, the result achieved by MDNN+TSN is still lower than the accuracy of our method, by about a 2%.

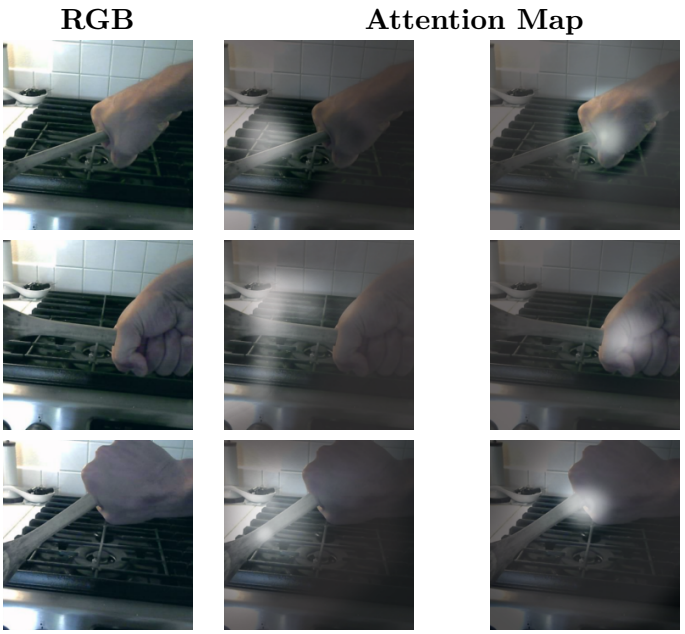


Figure 3.17. Comparison between the attention maps of baseline method [148] (second column) w.r.t. attention map from our model (third column) and the plain RGB image. The attention map coming from the proposed method are clearly better centered on the zone of the image most informative with respect to the action being performed.

3.3.7.1 Ablation study

In this ablation study we analize the effects of each implementation choice w.r.t. the final accuracy. We compared our method, with and without Mean Teacher, to the baseline method [148] for both datasets, experimenting with individual RGB and Depth modality, as well as the two modalities exploited by two separated network with a final features concatenation.

From Table 3.10 we can see that the Depth-only performances are poor (very evident on GUN-71), while using two networks for the two modalities does not give an evident contribution (on WCVS) or even worsen the performance (on GUN-71). This detrimental effect probably happens because this dataset has several classes very similar to each other, and thus it is very difficult to recognize the type of grasp only from depth images, which might present coarse and poorly defined data. At the same time, even RGB performances are not very high, thus preventing a noticeable boost while combining these two modalities. Finally, the Mean Teacher loss is able to produce a significant boost which helps to establish a new state of the art performance on both datasets.

3.3.7.2 Computational Analysis

The whole model is computational heavy to train, being made by a deep and wide backbone network(ResNet101), a Conv-LSTM module, and some online operations to blend depth with RGB. Moreover, the method needed several training steps in order to train properly, reaching 7 days of training on the bigger GUN-71 dataset and around 4 days of training on the smaller WCVS. At inference time, the whole method required $150ms$ to perform a single forward pass, thus requiring modifications for using on real time data. For example, some computational time

Method	Accuracy GUN-71	Accuracy WCVS
Deep-RGB [124]	11.31	-
Best From [124]	17.97	-
CNN-RGB [110]	-	52.0
CNN(MultiWindow)-RGB [110]	-	57.00
Appearance Stream [63],[142]	26.00	60.36
Depth Stream [63],[142]	15.60	58.47
Motion Stream [142]	-	37.34
Score Fusion [155]	26.24	59.32
Feature Fusion	29.36	62.16
DCCA [5]	32.56	60.45
TSN-RGB [167]	-	66.02
TSN-Flow[167]	-	59.48
TSN-Depth[167]	-	59.32
TSN-FLow+RGB(Score Fusion)[167]	-	67.05
TSN-FLow+RGB+Depth(Score Fusion)[167]	-	70.09
MDNN [156]	33.89	65.67
MDNN + hand [156]	34.04	67.04
Proposed architecture w/o Mean Teacher	**36.46**	**70.62**
Proposed architecture with Mean Teacher	**39.07**	**73.22**

Table 3.9. Results on GUN-71 and WCVS datasets.

Method	**Accuracy GUN-71**	**Accuracy WCVS**
RGB, network BMVC [148]	23.33	68,48
Depth, network BMVC [148]	8.71	60,92
RGB + D, Feature Fusion	20.90	68,56
Proposed architecture w/o Mean Teacher	36.46	70,62
Proposed architecture with Mean Teacher	39.07	73.22

Table 3.10. Ablation on GUN-71 and WCVS datasets.

can be regained by performing RGB*DepthMaps calculations directly on the GPU, with the use of specific GPU-only libraries as Nvidia *DALI* [8]. Finally, smaller backbone and/or smaller input resolutions could be used, lightening the computational cost at the expense of final accuracy.

3.3.8 Conclusions

This section has presented LODEAR, a deep architecture for first person activity recognition based on the use of RGB and depth information. As opposed to most of previous works in visual recognition where these two sensor channels are treated with separated architectures and then merged at different stages of the overall network, we use the depth as a sort of attention map, and blend it with the corresponding RGB image. This helps the backbone network to learn to recognize activities giving more or less weight to different parts of each frame. Temporal

[8] https://github.com/NVIDIA/DALI

information is exploited through the use of a Conv-LSLTM, with an architecture similar to [148]. We further strengthen the approach by adding a self labeling block over the sequence classification. Results confirm the value of our architecture. Future work will investigate the impact of using depth data as proposed in architectures maintaining the optical flow branch. Furthermore, we will explore the possibility to hallucinate depth maps (with approaches inspired by ADAGE) for the cases where such information should not be available, and test our findings on large scale RGB databases such as [34].

Chapter 4

RGB modality

This chapter presents different strategies to deal with the domain adaptation and transfer learning problem in the RGB domain. In Section 4.1 SBADAGAN is illustrated: a novel adversarial method for reducing domain shift on a pixel level, for digits and objects classification. Section 4.2 shows ADAGE, an hybrid pixel/feature adaptation method for multi source domain adaptation and generalization on digits data. Finally, Section 4.3 reveals a preliminary work on multi-source domain adaptation for semantic segmentation on urban scenes, which produced promising results.

4.1 SBADA-GAN

4.1.1 Introduction

The ability to generalize across domains is challenging when there is ample labeled data on which to train a deep network (source domain), but no annotated data for the target domain. To attack this issue, a wide array of methods have been proposed, most of them aiming at reducing the shift between the source and target distributions (see Section 2.4 for a review of previous work). An alternative is mapping the source data into the target domain, either by modifying the image representation [46] or by directly generating a new version of the source images [16]. Several authors proposed approaches that follow both these strategies by building over Generative Adversarial Networks (GANs) [52]. A similar but inverse method maps the target data into the source domain, where there is already an abundance of labeled images [162].

We argue that these two mapping directions should not be alternative, but complementary. Indeed, the main ingredient for adaptation is the ability of transferring successfully the style of one domain to the images of the other. This, given a fixed generative architecture, will depend on the application: there may be cases where mapping from the source to the target is easier, and cases where it is true otherwise. By pursuing both directions in a unified architecture, we can obtain a system more robust and more general than previous adaptation algorithms.

With this idea in mind, we designed SBADA-GAN: Symmetric Bi-directional ADAptive Generative Adversarial Network. Its features are (see Figure 4.1):

- it exploits two generative adversarial losses that encourage the network to produce target-like images from the source samples and source-like images from the target samples. Moreover, it jointly minimizes two classification losses, one on the original source images and the other on the transformed target-like source images;
- it uses the source classifier to annotate the source-like transformed target images. Such pseudo-labels help regularizing the same classifier while improving the target-to-source generator model by backpropagation;

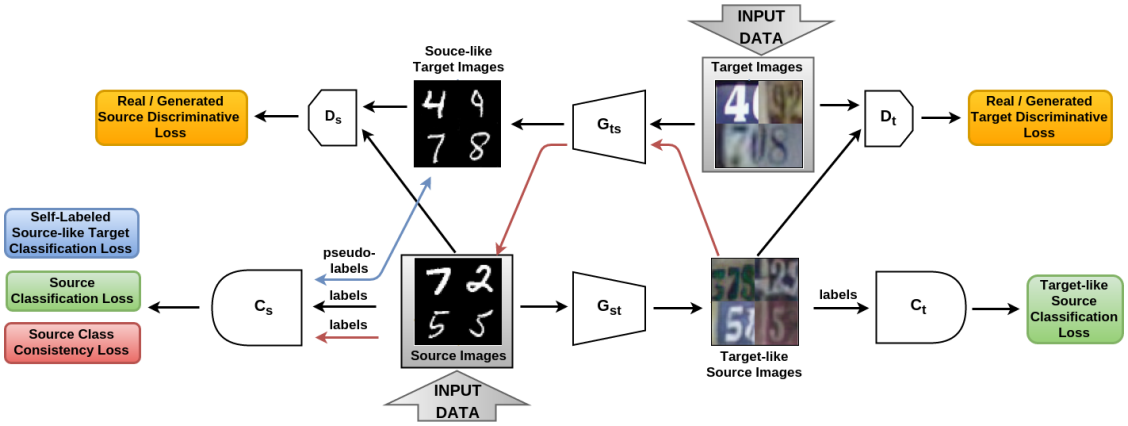


Figure 4.1. SBADA-GAN, training: the data flow starts from the Input Data arrows. The bottom and top row show respectively the source-to-target and target-to-source symmetric directions. The generative models G_{st} and G_{ts} transform the source images to the target domain and vice versa. D_s and D_t discriminate real from generated images of source and target. Finally the classifiers C_s and C_t are trained to recognize respectively the original source images and their target-like transformed versions. The bi-directional blue arrow indicates that the source-like target images are automatically annotated and the assigned pseudo-labels are re-used by the classifier C_s. The red arrows describe the class consistency condition by which source images transformed to the target domain through G_{st} and back to the source domain through G_{ts} should maintain their ground truth label.

- it introduces a new semantic constraint on the source images: the *class consistency loss*. It imposes that by mapping source images towards the target domain and then again towards the source domain they should get back to their ground truth class. This last condition is less restrictive than a standard reconstruction loss [186, 76], as it deals only with the image annotation and not with the image appearance. Still, our experiments show that it is highly effective in aligning the domain mappings in the two directions;
- at test time the two trained classifiers are used respectively on the original target images and on their source-like transformed version. The two predictions are integrated to produce the final annotation.

Our architecture yields realistic image reconstructions while competing against previous state-of-the-art classifiers and exceeding them on four out of six different unsupervised adaptation settings. An ablation study showcasing the importance of each component in the architecture, and investigating the robustness with respect to its hyperparameters, sheds light on the inner workings of the approach, while providing further evidence of its value.

4.1.2 Method

4.1.2.1 Model

SBADA-GAN focuses on unsupervised cross domain classification. Let us start from a dataset $\boldsymbol{X_s} = \{\boldsymbol{x}_s^i, y_s^i\}_{i=0}^{N_s}$ drawn from a labeled source domain $\mathcal{S}$, and a dataset $\boldsymbol{X_t} = \{\boldsymbol{x}_t^j\}_{j=0}^{N_t}$ from a different unlabeled target domain $\mathcal{T}$, sharing the same set of categories. The task is to maximize the classification accuracy on $\boldsymbol{X_t}$ while training on $\boldsymbol{X_s}$. To reduce the domain gap, we propose to adapt the source images such that they appear as sampled from the target domain by training a generator model G_{st} that maps any source samples $\boldsymbol{x}_s^i$ to its target-like version $\boldsymbol{x}_{st}^i = G_{st}(\boldsymbol{x}_s^i)$ defining the set $\boldsymbol{X}_{st} = \{\boldsymbol{x}_{st}^i, y_s^i\}_{i=0}^{N_s}$ (see Figure 4.1, bottom row). The model is also augmented

with a discriminator D_t and a classifier C_t. The former takes as input the target images $\boldsymbol{X}_t$ and target-like source transformed images $\boldsymbol{X}_{st}$, learning to recognize them as two different sets. The latter takes as input each of the transformed images $\boldsymbol{x}_{st}^i$ and learns to assign its task-specific label y_s^i. During the training procedure for this model, information about the domain recognition likelihood produced by D_t is used adversarially to guide and optimize the performance of the generator G_{st}. Similarly, the generator also benefits from backpropagation in the classifier training procedure.

Besides the source-to-target transformation, we also consider the inverse target-to-source direction by using a symmetric architecture (see Figure 4.1, top row). Here any target image $\boldsymbol{x}_t^j$ is given as input to a generator model G_{ts} transforming it to its source-like version $\boldsymbol{x}_{ts}^j = G_{ts}(\boldsymbol{x}_t^j)$, defining the set $\boldsymbol{X}_{ts} = \{\boldsymbol{x}_{ts}^j\}_{j=0}^{N_t}$. As before, the model is augmented with a discriminator D_s which takes as input both $\boldsymbol{X}_{ts}$ and $\boldsymbol{X}_s$ and learns to recognize them as two different sets, adversarially helping the generator.

Since the target images are unlabeled, no classifier can be trained in the target-to-source direction as a further support for the generator model. We overcome this issue by *self-labeling* (see Figure 4.1, blue arrow). The original source data $\boldsymbol{X}_s$ is used to train a classifier C_s. Once it has reached convergence, we apply the learned model to annotate each of the source-like transformed target images $\boldsymbol{x}_{ts}^j$. These samples, with the assigned pseudo-labels $y_{t_{self}}^j = \text{argmax}_y(C_s(G_{ts}(\boldsymbol{x}_t^j))$, are then used transductively as input to C_s while information about the performance of the model on them is backpropagated to guide and improve the generator G_{ts}. Self-labeling has a long track record of success for domain adaptation: it proved to be effective both with shallow models [17, 58, 113], as well as with the most recent deep architectures [138, 161, 132]. In our case the classification loss on pseudo-labeled samples is combined with our other losses, which helps making sure we move towards the optimal solution: in case of a moderate domain shift, the correct pseudo-labels help to regularize the learning process, while in case of large domain shift, the possible mislabeled samples do not hinder the performance (see Section 4.1.4.3 for a detailed discussion on the experimental results).

Finally, the symmetry in the source-to-target and target-to-source transformations is enhanced by aligning the two generator models such that, when used in sequence, they bring a sample back to its starting point. Since our main focus is classification, we are interested in preserving the class identity of each sample rather than its overall appearance. Thus, instead of a standard image-based reconstruction condition we introduce a *class consistency* condition (see Figure 4.1, red arrows). Specifically, we impose that any source image $\boldsymbol{x}_s^i$ adapted to the target domain through $G_{st}(\boldsymbol{x}_s^i)$ and transformed back towards the source domain through $G_{ts}(G_{st}(\boldsymbol{x}_s^i))$ is correctly classified by C_s. This condition helps by imposing a further joint optimization of the two generators.

4.1.2.2 Learning

Here we formalize the description above. To begin with, we specify that the generators take as input a noise vector $\boldsymbol{z} \in \mathcal{N}(0,1)$ besides the images, this allows some extra degree of freedom to model external variations. We also better define the discriminators as $D_s(\boldsymbol{x})$, $D_t(\boldsymbol{x})$ and the classifiers as $C_s(\boldsymbol{x})$, $C_t(\boldsymbol{x})$. Of course each of these models depends from its parameters but we do not explicitly indicate them to simplify the notation. For the same reason we also drop the superscripts i, j.

The source-to-target part of the network optimizes the following objective function:

$$\min_{G_{st},C_t} \max_{D_t} \quad \alpha\mathcal{L}_{D_t}(D_t, G_{st}) + \beta\mathcal{L}_{C_t}(G_{st}, C_t) \ , \tag{4.1}$$

where the classification loss $\mathcal{L}_{C_t}$ is a standard *softmax cross-entropy*

$$\mathcal{L}_{C_t}(G_{st}, C_t) = \mathbb{E}_{\substack{\{\boldsymbol{x}_s, \boldsymbol{y}_s\}\sim\mathcal{S} \\ \boldsymbol{z}_s\sim noise}}[-\boldsymbol{y}_s \cdot \log(\hat{\boldsymbol{y}}_s)] \ , \tag{4.2}$$

evaluated on the source samples transformed by the generator G_{st}, so that $\hat{\boldsymbol{y}}_s = C_t(G_{st}(\boldsymbol{x}_s, \boldsymbol{z}_s))$ and $\boldsymbol{y}_s$ is the one-hot encoding of the class label y_s. For the discriminator, instead of the less robust binary cross-entropy, we followed [104] and chose a *least square* loss:

$$\begin{aligned}\mathcal{L}_{D_t}(D_t, G_{st}) =& \mathbb{E}_{\boldsymbol{x}_t\sim T}[(D_t(\boldsymbol{x}_t) - 1)^2] + \\ & \mathbb{E}_{\substack{\boldsymbol{x}_s\sim S \\ \boldsymbol{z}_s\sim noise}}[(D_t(G_{st}(\boldsymbol{x}_s, \boldsymbol{z}_s)))^2] \ .\end{aligned} \tag{4.3}$$

The objective function for the target-to-source part of the network is:

$$\begin{aligned}\min_{G_{ts}, C_s} \max_{D_s} \ & \gamma\mathcal{L}_{D_s}(D_s, G_{ts}) + \\ & \mu\mathcal{L}_{C_s}(C_s) + \eta\mathcal{L}_{self}(G_{ts}, C_s) \ ,\end{aligned} \tag{4.4}$$

where the discriminative loss is analogous to Equation (4.3), while the classification loss is analogous to Equation (4.2) but evaluated on the original source samples with $\hat{\boldsymbol{y}}_s = C_s(\boldsymbol{x}_s)$, thus it neither has any dependence on the generator that transforms the target samples G_{ts}, nor it provides feedback to it. The *self* loss is again a classification softmax cross-entropy:

$$\mathcal{L}_{self}(G_{ts}, C_s) = \mathbb{E}_{\substack{\{\boldsymbol{x}_t, \boldsymbol{y}_{t_{self}}\}\sim\mathcal{T} \\ \boldsymbol{z}_t\sim noise}}[-\boldsymbol{y}_{t_{self}} \cdot \log(\hat{\boldsymbol{y}}_{t_{self}})] \ . \tag{4.5}$$

where $\hat{\boldsymbol{y}}_{t_{self}} = C_s(G_{ts}(\boldsymbol{x}_t, \boldsymbol{z}_t))$ and $\boldsymbol{y}_{\boldsymbol{t_{self}}}$ is the one-hot vector encoding of the assigned label $y_{t_{self}}$. This loss back-propagates to the generator G_{ts} which is encouraged to preserve the annotated category within the transformation.

Finally, we developed a novel *class consistency* loss by minimizing the error of the classifier C_s when applied on the concatenated transformation of G_{ts} and G_{st} to produce $\hat{\boldsymbol{y}}_{cons} = (C_s(G_{ts}(G_{st}(\boldsymbol{x}_s, \boldsymbol{z}_s), \boldsymbol{z}_t)))$:

$$\mathcal{L}_{cons}(G_{ts}, G_{st}, C_s) = \mathbb{E}_{\substack{\{\boldsymbol{x}_s, \boldsymbol{y}_s\}\sim S \\ \boldsymbol{z}_s, \boldsymbol{z}_t\sim noise}}[-\boldsymbol{y}_s \cdot \log(\hat{\boldsymbol{y}}_{cons})] \ . \tag{4.6}$$

This loss has the important role of aligning the generators in the two directions and strongly connecting the two main parts of our architecture.

By collecting all the presented parts, we conclude with the complete SBADA-GAN loss:

$$\begin{aligned}&\mathcal{L}_{SBADA-GAN}(G_{st}, G_{ts}, C_s, C_t, D_s, D_t) = \\ &\quad \alpha\mathcal{L}_{D_t} + \beta\mathcal{L}_{C_t} + \gamma\mathcal{L}_{D_s} + \mu\mathcal{L}_{C_s} + \eta\mathcal{L}_{self} + \nu\mathcal{L}_{cons} \ .\end{aligned} \tag{4.7}$$

Here $(\alpha, \beta, \gamma, \mu, \eta, \nu) \geq 0$ are weights that control the interaction of the loss terms. While the combination of six different losses might appear daunting, it is not unusual [15]. Here, it stems from the symmetric bi-directional nature of the overall architecture. Indeed each directional branch has three losses as it is custom practice in the GAN-based domain adaptation literature [162, 16]. Moreover, the ablation study reported in Section 4.1.4.3 indicates that the system is remarkably robust to changes in the hyperparameter values.

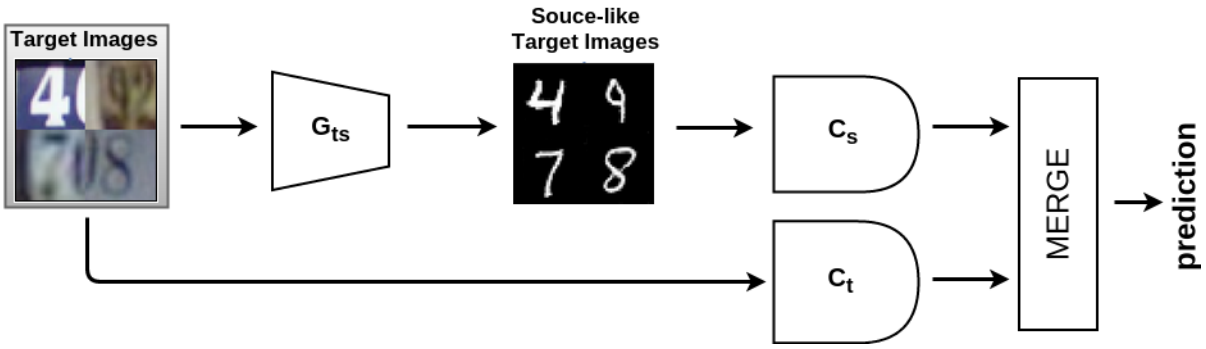


Figure 4.2. SBADA-GAN, test: the two pre-trained classifiers are applied respectively on the target images and on the transformed source-like target images. Their outputs are linearly combined for the final prediction.

4.1.2.3 Testing

The classifier C_t is trained on $\boldsymbol{X}_{st}$ generated images that mimic the target domain style, and is then tested on the original target samples $\boldsymbol{X}_t$. The classifier C_s is trained on $\boldsymbol{X}_s$ source data, and then tested on $\boldsymbol{X}_{ts}$ samples, that are the target images modified to mimic the source domain style. These classifiers make mistakes of different type assigning also a different confidence rank to each of the possible labels. Overall the two classification models complement each other. We take advantage of this with a simple ensemble method $\sigma C_s(G_{ts}(\boldsymbol{x}_t, \boldsymbol{z}_t)) + \tau C_t(\boldsymbol{x}_t)$ which linearly combines their probability output, providing a further gain in performance. A schematic illustration of the testing procedure is shown in Figure 4.2. We set the combination weights σ, τ through cross validation (see Section 4.1.3.2 for further details).

4.1.3 Experiments

4.1.3.1 Datasets and Adaptation Scenarios

We evaluate SBADA-GAN on several unsupervised adaptation scenarios:

MNIST→MNIST-M, **MNIST↔USPS**, **SVHN↔MNIST**, **Synth Signs→GTSRB**. All the datasets have been scaled to 32×32 pixels to align their resolution, and we used the corresponding evaluations protocols of [15, 16, 46, 50, 59].

4.1.3.2 Implementation details

We composed SBADA-GAN starting from two symmetric GANs, each with an architecture[1] analogous to that used in [16]. The model is coded[2] in Python and we ran all our experiments in the Keras framework [31]. We use the ADAM [77] optimizer with learning rates for the discriminator and the generator both set to 10^{-4}. The batch size is set to 32 and we train the model for 500 epochs not noticing any overfitting, which suggests that further epochs might be beneficial. The α and γ loss weights (discriminator losses) are set to 1, β and μ (classifier losses) are set to 10, to prevent that generator from indirectly switching labels (for instance, transform 7's into 1's). The class consistency loss weight ν is set to 1. All training procedures start with the self-labeling loss weight, η, set to zero, as this loss hinders convergence until the classifier is fully trained. After the model converges (losses stop oscillating, usually after 250 epochs) η is set to 1 to further increase performance. Finally the parameters to combine the classifiers at

[1] See all the model details in the supplementary material.
[2] Code available at https://github.com/engharat/SBADAGAN

	MNIST→USPS	USPS→MNIST	MNIST→MNIST-M	SVHN→MNIST	MNIST→SVHN	Synth S.→GTSRB
Source Only	78.9	57.1 ± 1.7	63.6	60.1 ± 1.1	26.0 ± 1.2	79.0
CORAL [149]	81.7	-	57.7	63.1	-	86.9
MMD [161]	81.1	-	76.9	71.1	-	91.1
DANN [46]	85.1	73.0 ± 2.0	77.4	73.9	35.7	88.7
DSN [15]	91.3	-	83.2	82.7	-	93.1
CoGAN [93]	91.2	89.1 ± 0.8	62.0	not conv.	-	-
ADDA [162]	89.4 ± 0.2	90.1 ± 0.8	-	76.0 ± 1.8	-	-
DRCN [50]	91.8 ± 0.1	73.7 ± 0.1	-	82.0 ± 0.2	40.1 ± 0.1	-
PixelDA [16]	95.9	-	98.2	-	-	-
DTN [154]	-	-	-	84.4	-	-
TRUDA [138]	-	-	86.7	78.8	40.3	-
ATT [132]	-	-	94.2	86.2	52.8	96.2
UNIT [95]	95.9	93.5	-	90.5	-	-
DA_{ass} fix. par. [59]	-	-	89.5	95.7	-	82.8
DA_{ass} [59]	-	-	89.5	**97.6**	-	**97.7**
Target Only	96.5	99.2 ± 0.1	96.4	99.5	96.7	98.2
SBADA-GAN C_t	96.7	94.4	99.1	72.2	59.2	95.9
SBADA-GAN C_s	97.1	87.5	98.4	74.2	50.9	95.7
SBADA-GAN	**97.6**	**95.0**	**99.4**	76.1	**61.1**	96.7
GenToAdapt [135]	92.5 ± 0.7	90.8 ± 1.3	-	84.7 ± 0.9	36.4 ± 1.2	-
CyCADA [1]	94.8 ± 0.2	95.7 ± 0.2	-	88.3 ± 0.2	-	-
Self-Ensembling [2]	98.3 ± 0.1	99.5 ± 0.4	-	99.2 ± 0.3	42.0 ± 5.7	98.3 ± 0.3

Table 4.1. Comparison against previous work. SBADA-GAN C_t, C_s reports respectively the accuracies produced by the classifier trained in the target domain space, and the results produced by training in the source domain space and testing on the target images mapped to this space. SBADA-GAN reports the results obtained by a weighted combination of the softmax outputs of these two classifiers. Note that all competitors convert SVHN to grayscale, while we deal with the more complex original RGB version. The last three rows report results from online available pre-print papers.

test time are $\sigma \in [0, 0.1, 0.2, \ldots, 1]$ and $\tau = (1 - \sigma)$ chosen on a validation set of 1000 random samples from the target in each different setting.

4.1.4 Results

4.1.4.1 Quantitative Results

Table 4.1 shows results on our evaluation settings. The top of the table reports results by thirteen competing baselines published over the last two years. The Source-Only and Target-Only rows contain reference results corresponding to the no-adaptation case and to the target fully supervised case. For SBADA-GAN, besides the full method, we also report the accuracy obtained by the separate classifiers (C_s,C_t) before the linear combination.

SBADA-GAN improves over the state of the art in four out of six settings. In these cases the advantage with respect to its competitors is already visible in the separate C_s and C_t results and it increases when considering the full combination procedure. Moreover, the gain in performance of SBADA-GAN reaches up to +8 percentage points in the MNIST→SVHN experiment. This setting was disregarded in many previous works: differently from its inverse SVHN→MNIST, it requires a difficult adaptation from the grayscale handwritten digits domain to the widely variable and colorful street view house number domain. Thanks to its bi-directionality, SBADA-GAN leverages on the inverse target to source mapping to produce highly accuracy results.

Conversely, in the SVHN→MNIST case SBADA-GAN ranks eighth out of the thirteen baselines in terms of performance. Our accuracy is on par with ADDA's [162]: the two approaches share the same classifier architecture, although the number of fully-connected neurons of SBADA-GAN is five time lower. Moreover, compared to DRCN [50], the classifiers

Setting	S	T map to S	S map to T	T
MNIST → USPS	0.206	0.219	0.106	0.102
MNIST → MNIST-M	0.206	0.207	0.035	0.032
MNIST → SVHN	0.206	0.292	0.027	0.012
Synth S. → GTSRB	0.105	0.136	0.128	0.154

Table 4.2. Dataset mean SSIM: this measure of data variability suggests that our method successfully generates images with not only the same style of a chosen domain, but also similar perceptual variability.

of SBADA-GAN are shallower with a reduced number of convolutional layers. Overall here SBADA-GAN suffers of some typical drawbacks of GAN-based domain adaptation methods: although the style of a domain can be easily transferred in the raw pixel space, the generative process does not have any explicit constraint on reducing the overall data distribution shift as instead done by the alternative feature-based domain adaptation approaches. Thus, methods like DA_{ass} [59], DTN [154] and DSN [15] deal better with the large domain gap of the SVHN→MNIST setting.

Finally, in the Synth Signs → GTSRB experiment, SBADA-GAN is just slightly worse than DA_{ass}, but outperforms all the other competing methods. The comparison remains in favor of SBADA-GAN when considering that its performance is robust to hyperparameter variations (see Section 4.1.4.3 for more details).

4.1.4.2 Qualitative Results

To complement the quantitative evaluation, we look at the quality of the images generated by SBADA-GAN. First, we see from Figure 4.3 how the generated images mimic the style of the chosen domain, even when going from the simple MNIST digits to the SVHN colorful house numbers.

Visually inspecting the data distribution before and after domain mapping defines a second qualitative evaluation metric. We use t-SNE [101] to project the data from their raw pixel space to a simplified 2D embedding. Figure 4.4 shows that the transformed dataset tends to replicate faithfully the distribution of the chosen final domain.

A further measure of the quality of the SBADA-GAN generators comes from the diversity of the produced images. Indeed, GAN's generators may collapse and output a single prototype that maximally fools the discriminators. To evaluate the diversity of samples generated by SBADA-GAN we choose the Structural Similarity (SSIM, [168]) that correlates well with the human perceptual similarity judgments. Its values range between 0 and 1 with higher values corresponding to more similar images. We follow the same procedure used in [117] by randomly choosing 1000 pairs of generated images within a given class. We also repeat the evaluation over all the classes and calculate the average results. Table 4.2 shows the results of the mean SSIM metric and indicates that the SBADA-GAN generated images not only mimic the same style, but also successfully reproduce the variability of a chosen domain.

4.1.4.3 Ablation Study

Starting from the core source-to-target GAN module we analyze the effect of adding all the other model parts. At first we add the symmetric target-to-source GAN model. These two parts are then combined and the domain transformation loop is closed by adding the class consistency condition. Finally the model is completed by introducing the target self-labeling procedure. We empirically test each of these model steps on the MNIST→USPS setting and report the results

(a) MNIST to USPS **(b)** USPS to MNIST

(c) MNIST to MNIST-M **(d)** MNIST-M to MNIST

(e) MNIST to SVHN **(f)** SVHN to MNIST

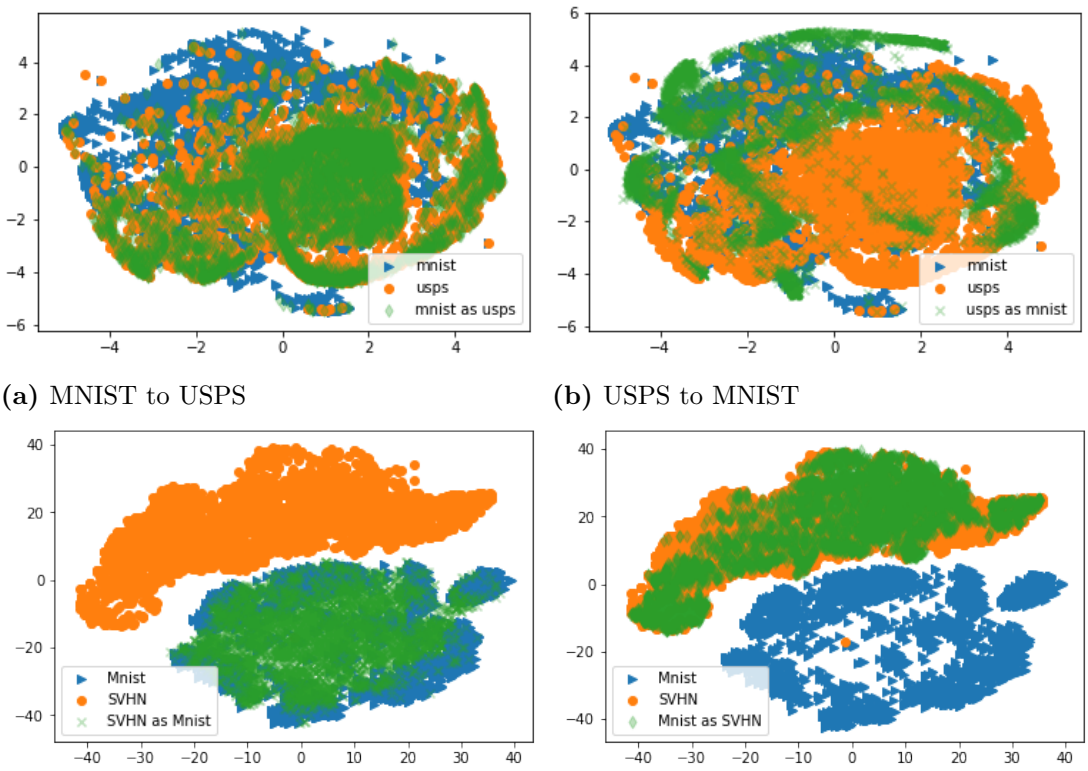
(g) Synth S. to GTSRB **(h)** GTSRB to Synth S.

Figure 4.3. Examples of generated digits: we show the image transformation from the original domain to the paired one as indicated under every sub-figure. For each of the (a)-(h) cases, the original/generated images are in the top/bottom row.

in Table 4.3. We see the gain achieved by progressively adding the different components, with the largest advantage obtained by the introduction of self-labeling.

An analogous boost due to self-labeling is also visible in all the other experimental settings with the exception of MNIST↔SVHN, where the accuracy remains unchanged if η is equal or larger than zero. A further analysis reveals that here the recognition accuracy of the source classifier applied to the source-like transformed target images is quite low (about 65%, while in all the other settings reaches $80-90\%$), thus the pseudo-labels cannot be considered reliable. Still, using them does not hinder the overall performance.

The crucial effect of the class consistency loss can be better observed by looking at the generated images and comparing them with those obtained in two alternative cases: setting $\nu = 0$, i.e.,not using any consistency condition between the two generators G_{st} and G_{ts}, or substituting our class consistency loss with the standard cycle consistency loss [186, 76] based



(a) MNIST to USPS **(b)** USPS to MNIST

(c) SVHN to MNIST **(d)** MNIST to SVHN

Figure 4.4. t-SNE visualization of source, target and source mapped to target images. Note how the mapped source covers faithfully the target space both in the (a),(b) case with moderated domain shift and in the more challenging (c),(d) setting.

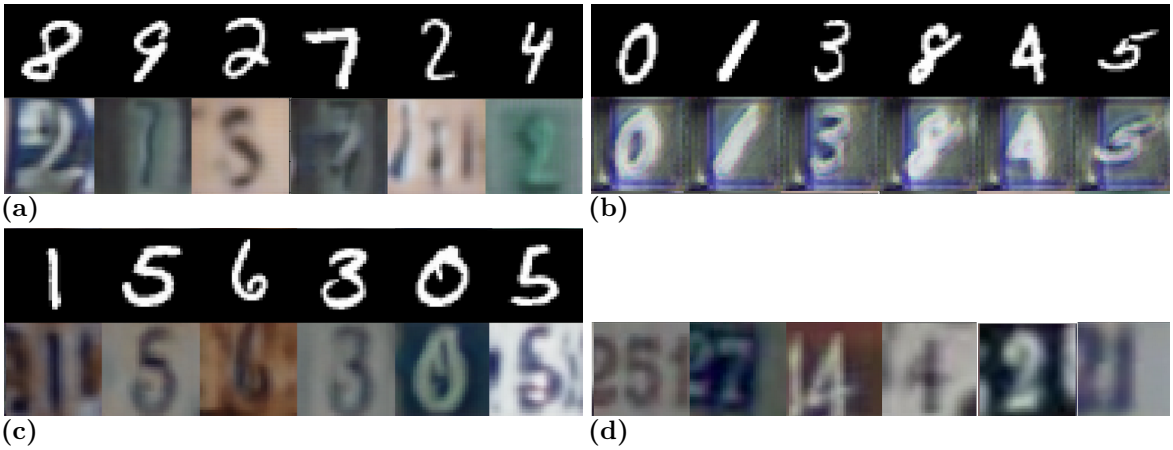
on image reconstruction. For this evaluation we choose the MNIST→SVHN case which has the strongest domain shift and we show the generated images in Figure 4.5. When the consistency loss is not activated, the G_{ts} output images are realistic, but fail at reproducing the correct input digit and provide misleading information to the classifier. On the other hand, using the cycle-consistency loss preserves the input digit but fails in rendering a realistic sample in the correct domain style. Finally, our class consistency loss allows to preserve the distinct features belonging to a category while still leaving enough freedom to the generation process, thus it succeeds in both preserving the digits and rendering realistic samples.

4.1.4.4 CycleGAN vs SBADA-GAN

To further clarify the difference between the two methods, we remind that CycleGAN is unsupervised and works only when transferring style across similarly shaped categories (e.g.,horses→zebras), not across domains. SBADA-GAN instead deals with domains containing multiple categories. The images samples in Figure 4.5(b) are indeed obtained with CycleGAN: training on them produces an accuracy of 25.5%, much lower than the corresponding 61.1% of SBAD-GAN. Moreover, CycleGAN has a single transformed image as output, while SBADA-GAN exploits a noise vector as input producing multiple outputs for each input image: this is critical for classification as it provides variability through data augmentation, it avoids overfitting and eases generalization. For completeness we also ran an experiment on the challenging Office Dataset [131]: here both the images produced by CycleGAN and SBADA-GAN (see Figure 4.6) are given as input to a pre-trained AlexNet and the classification accuracy is respectively 52.0% and

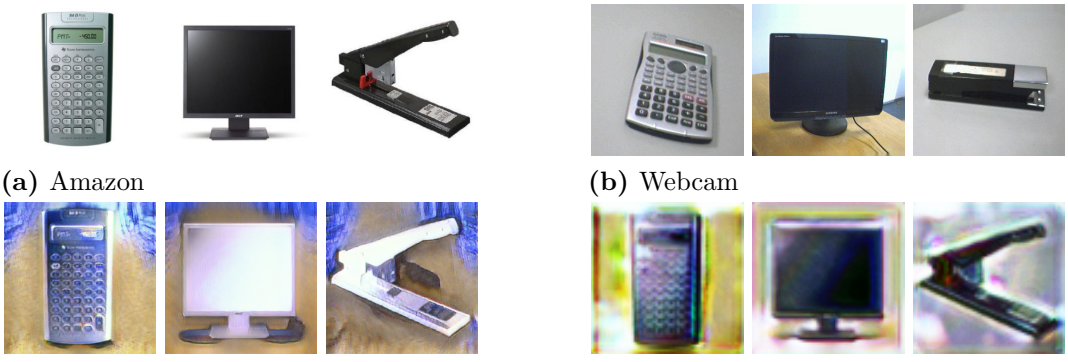
S→T GAN		T→S GAN		Class Consist.	Self Label.	Accuracy
$\mathcal{L}_{D_t}$	$\mathcal{L}_{C_t}$	$\mathcal{L}_{D_s}$	$\mathcal{L}_{C_s}$	$\mathcal{L}_{cons}$	$\mathcal{L}_{self}$	MNIST→USPS
✓	✓					94.23
		✓	✓			91.55
✓	✓	✓	✓			94.90
✓	✓	✓	✓	✓		95.45
✓	✓	✓	✓	✓	✓	97.60

Table 4.3. Analysis of the role of each SBADA-GAN component. We ran experiments by turning on the different losses of the model as indicated by the checkmarks.



(a) **(b)** **(c)** **(d)**

Figure 4.5. G_{ts} outputs (lower line) and their respective inputs (upper line) obtained with: (a) no consistency loss, (b) image-based cycle consistency loss [186, 76], (c) our class consistency loss. In (d) we show some real SVHN samples as a reference.



(a) Amazon **(b)** Webcam

(c) CycleGAN **(d)** SBADA-GAN

Figure 4.6. CycleGAN [186] vs SBADA-GAN on the Amazon-Webcam experiment of the Office Dataset [131].

50.7%, both lower than the reference 61.6% result produced by the baseline without adaptation. These results confirm the known difficulty of GAN-based method to deal with domain shifts due to poses and shapes.

4.1.4.5 Robustness Study

SBADA-GAN is robust to the specific choice of the consistency loss weight ν, given that it is different from zero. Changing it in $[0.1, 1, 10]$ induces a maximum variation of 0.6 percentage points in accuracy over the different settings. An analogous evaluation performed on the classification loss weights (β,μ) reveals that changing them in the same range used for ν causes a maximum overall performance variation of 0.2 percentage points. Furthermore SBADA-GAN is minimally sensitive to the batch size used: halving it from 32 to 16 samples while keeping the same number of learning epochs reduces the performance only of about 0.2 percentage points. Such robustness is particularly relevant when compared to competing methods. For instance the most recent DA_{ass} [59] needs a perfectly balanced source and target distribution of classes in each batch, a condition difficult to satisfy in real world scenarios, and halving the originally large batch size reduces by 3.5 percentage points the final accuracy. Moreover, changing the weights of the losses that enforce associations across domains with a range analogous to that used for the SBADA-GAN parameters induces a drop in performance up to 16 accuracy percentage points.

4.1.4.6 Computational Analysis

SBADA-GAN, as most of the available adversarial methods, still keeps some instabilities which often require to restart the training from scratch. For these reasons, the whole method has been trained several times before reaching the final accuracy, requiring more than 15 days of total training. Once trained, the inference time are pretty low, thanks to the use of small generative networks and classifiers, with a total number of model parameters involved for the testing lower than 2 millions; with its $12ms$ of generators+classifiers forward pass time, it can be used for any real time task, at 30 or even 60fps. Neverhless, one of the reasons that contributed to the test speed of SBADA-GAN is the low input resolution (32x32). Preliminary tests on higher resolutions showed slower training and test times, as well as increased instabilities problems, thus requiring probably some changes to the proposed model.

4.1.5 Conclusions

This section presented SBADA-GAN, an adaptive adversarial domain adaptation architecture that maps simultaneously source samples into the target domain and vice versa with the aim to learn and use both classifiers at test time. To achieve this, self-labeling is exploited to regularize the classifier trained on the source, and we impose a class consistency loss that improves greatly the stability of the architecture, as well as the quality of the reconstructed images in both domains.

We explain the success of SBADA-GAN in several ways. To begin with, thanks to the the bi-directional mapping we avoid deciding a priori which is the best strategy for a specific task. Also, the combination of the two network directions improves performance providing empirical evidence that they are learning different, complementary features. Our class consistency loss aligns the image generators, allowing both domain transfers to influence each other. Finally the self-labeling procedure boost the performance in case of moderate domain shift without hindering it in case of large domain gaps.

4.2 ADAGE

4.2.1 Introduction

Domain Adaptation (DA) is at its core the quest for principled algorithms enabling the generalization of visual recognition methods. Given at least a source domain for training, the goal is to obtain recognition results as good as those achievable on source test data on *any* other target domain, in principle belonging to a different probability distribution. While originally defined assuming to have access to annotated data from a single source domain, and to unlabeled data from a different target domain [131], there is growing interest on how to leverage over multiple sources, and for domain generalization (DG), i.e. the case when it is not possible to access target data of any sort a priori. Algorithm-wise, three strategies have been proposed, i.e. dealing with model [38, 88], feature [98, 126], or image adaptation [128, 1]. A basic assumption for both feature and image adaptation approaches is the existence of a shared space among domains, however only feature-based methods attempt to explicitly identify it [65, 71, 15]. In the image-based approaches, the domain generic component is always silently recombined with the specific domain style to obtain images that show the same content of the target, but with source-like appearance or vice-versa [1, 128, 95]. Moreover, although these methods have shown to be effective in the single source scenario , it is questionable whether they could be extended to multi-source DA, or to DG.

With this work we make two contributions: (1) we introduce image adaptation for DG, (2) we propose an architecture that exploits the power of layer aggregation to hallucinate samples of the latent pixel space shared among domains. We call our method Agnostic DomAin GEneralization (ADAGE). To our knowledge it is the first solution to introduce an image-level component in an end-to-end deep learning architecture for DG and that can work seamlessly also in the multi-source unsupervised DA setting.

We start by acknowledging that the notion of visual cross-domain generic information is intuitive yet ambiguous, as ground truth examples of pure semantic images without a characteristic style do not exist. Thus, while it is possible to interpret the produced samples as capturing domain agnostic knowledge, it should be clear that they are built for the network's benefit only and we do not expect them to be pleasant to the human eye. Practically, we let the network learn what this generic information is through a mapping guided by adversarial adaptive constraints. These constraints are applied directly on the agnostic space, rather than on standard images that always contain domain-specific information.

To realize the mapping we define a dedicated convolutional structure loosely related to a previous image colorization network [21]. The new architecture has a low number of parameters which prevents overfitting and at the same time allows to comfortably accommodate two gradient reversal layers that adversarially exploit both image and feature classification across domains. As the image domain discriminator maintains the ability to evaluate the similarity of a target image to the different source domains, it is straightforward to extend the method to multi-source DA, and learn how to bias the classification loss towards the sources that are more similar to the target. Differently from GAN-based methods that need a typical alternating training between image adaptation and classification, we train the whole model of ADAGE with a single optimizer while performing adversarial training by inverting the gradient originating from two domain discriminators at image and feature level.

We test ADAGE in the DG and multi-source DA scenarios, comparing against recent approaches [88, 182, 172]. In all experiments, for both settings, ADAGE significantly outperforms the state of the art. An ablation study and visualizations of the agnostic domain images complete our experimental study.

4.2.2 Agnostic Domain Generalization

We assume to observe $i = 1 \dots S$ source domains with the ith domain containing N_i labeled instances $\{x_j^i, y_j^i\}_{j=1}^{N_i}$, where x_j^i is the jth input image and $y_j^i \in \{1 \dots M\}$ is the class label. In addition we also have an unlabeled target domain whose data $\{x_j^t\}_{j=1}^{N_t}$ might (DA) or might not (DG) be provided at training time. All the source and target domains share the same label space, but their marginal distribution is different thus inducing a domain shift. The goal of ADAGE is to achieve domain generalization by hallucinating images stripped down of domain specific information, that thus can be seen as samples of a machine-created agnostic domain. We obtain this by learning to modify the images such that it becomes impossible to identify their original source domain both from their pixels and from the extracted features, while maintaining their relevant semantic information. Figure 4.7 shows our architecture, consisting of two main components: (1) the *Hallucinator* block, in charge of generating the agnostic images from the input samples, and (2) the *Domain Generalizer*, that performs adaptation from the new domain. The architecture is end-to-end, meaning that the two components are interconnected and trained jointly.

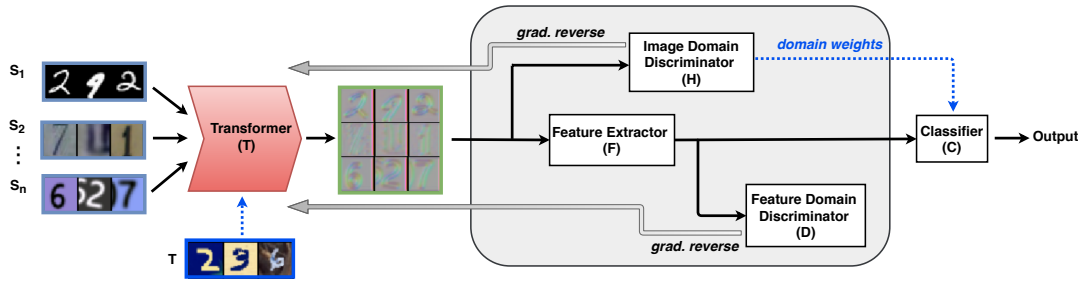


Figure 4.7. Main structure of ADAGE. All samples (including target ones, in the DA setting) follow the same path in the network. The inverted gradient from I flows through H driving image modifications towards domain confusion. Similarly, the gradient from D also inverted, is back propagated through F and H so that both the feature and the image dedicated blocks benefit from a further push towards the domain agnostic space. The classification gradient travels through the whole network, excluding I and D.

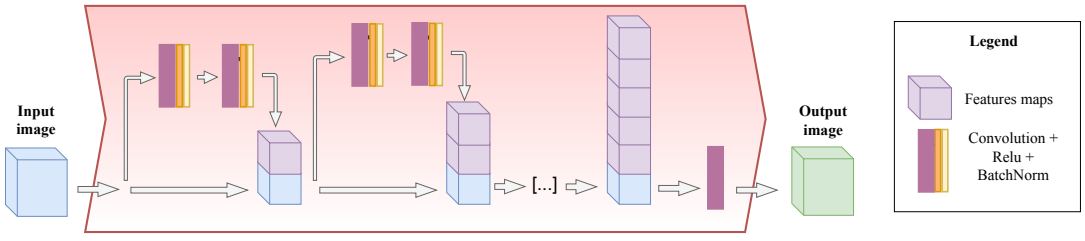


Figure 4.8. The Hallucinator. The output of the two multicolor blocks (Convolutional + Relu + Batch Normalization) are concatenated with the previous inputs, forming a group of images and features maps that grow along the depth of the network. The number of features increases from 3 (input data) to 256 (final aggregation step), while a last Convolutional layer squeezes the features back into 3 channels, interpretable as an RGB image.

The **Hallucinator (H)** Figure 4.8 modifies the input images to remove their domain-specific style. To achieve this, we got inspiration from the colorization literature and define a new structure exploiting the power of layer aggregation [175]: the output of two 3×3 convolutional layers, each followed by Relu and Batch Normalization are stacked up with the input and

propagated to every subsequent layer (see **??**). Specifically, the produced feature build up in size resulting in a growing sequence of $\{3, 8, 16, 32, 64, 128\}$ maps, after which a convolution layer brings them down to 3 channels, interpretable as RGB images. With respect to previous mapping architectures proposed within the context of depth colorization [21], our hallucinator has a significantly lower number of parameters thanks to its incrementally aggregative structure. This is crucial for generalization both because it reduces the risk of overfitting to the available sources and because leaves space for building a multi-branch following network able to impose constraints that in turn will lead to learning a stronger and more stable hallucinator in our end-to-end framework.

The **Domain Generalizer** is composed by the *Image Domain Discriminator* I, the *Feature Domain Discriminator* D and the *Feature Extractor* F. The first two impose respectively an adversarial generalization condition on the pixels and on the feature extracted from the images produced by H, while the third defines an intermediate step between the first two. Moreover, thanks to its direct connection with the *Classifier* C, it allows to maintain the basic semantic knowledge in the hallucinated images, so that, despite they lack domain style, their label can still be recognized.

The Image Domain Discriminator I receives as input the images produced by H and predicts their domain label. More in detail, this module is a multi-class classifier that learns to distinguish among the S source domains in DG, and $S+1$ in DA (including the target), by minimizing a simple cross-entropy loss $\mathcal{L}_I$. The information provided by this module is used in two ways: to adversarially guide the hallucinator H to produce images with confused domain identity, and to estimate a similarity measure between the source and the target data when available. The first task is executed through a gradient reversal layer as in [46]. The second is obtained as a byproduct of the domain classifier I by collecting the probability of every source sample in each batch to be recognized as belonging to the target.

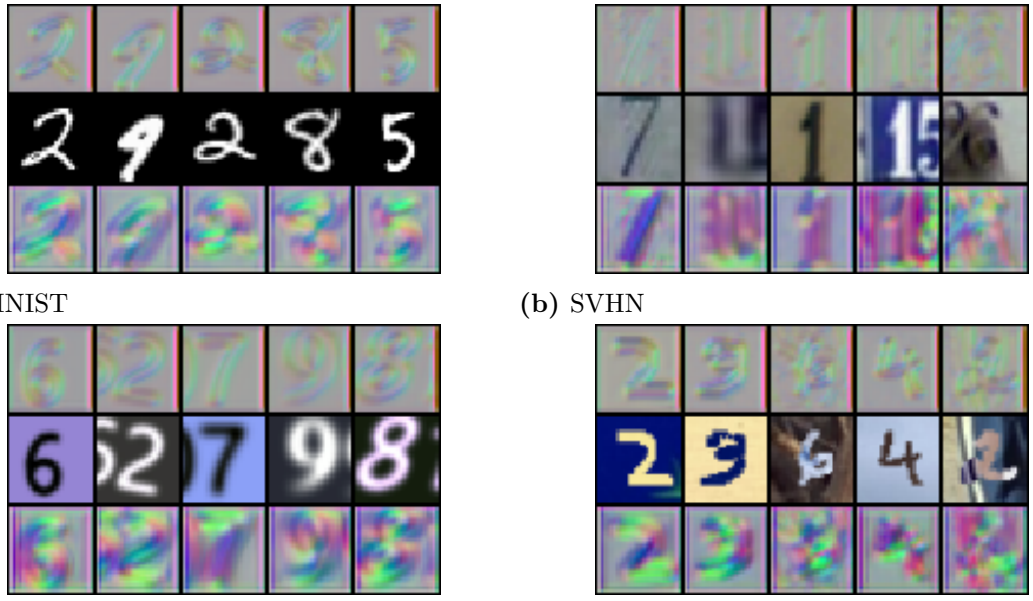
The Feature Domain Discriminator D is analogous to I but, instead of images, it takes as input their features, performing domain classification by minimizing the cross-entropy loss $\mathcal{L}_D$. During backpropagation, the inverted gradient regulates the feature extraction process to confuse the domains. Finally, the Feature Extractor F, as well as the Classifier C, is a standard deep learning module. We built both of them with the same network structure used in [182] to put them on equal footing. In particular, in the DG setting the classifier learns to distinguish among the M categories of the sources by minimizing the cross-entropy loss $\mathcal{L}_C$, while for the DA setting it can also provide the classification probability on the target samples $p(x^t) = C(F(H(x_t)))$ that is used to minimize the related entropy loss $\mathcal{L}_E = p(x^t)log(p(x^t))$.

If we indicate with θ the network parameters and we use subscripts to identify the different network modules, we can write the overall loss function optimized by ADAGE as:

$$\begin{aligned}&\mathcal{L}(\theta_H, \theta_F, \theta_D, \theta_I, \theta_C) = \\ &\sum_{i=1}^{S,S+1} \sum_{j=1}^{N^i} \mathcal{L}_C^{j,i}(\theta_H, \theta_F, \theta_C) + \eta \mathcal{L}_E^{j,i=S+1}(\theta_H, \theta_F, \theta_C) \\ &\qquad - \lambda \mathcal{L}_D^{j,i}(\theta_H, \theta_F, \theta_D) - \gamma \mathcal{L}_I^{j,i}(\theta_H, \theta_I) \ . \end{aligned} \tag{4.8}$$

We remark that, as specified by its superscripts, $\mathcal{L}_E^{j,i=S+1}$ is only active in the DA setting, while $\mathcal{L}_D$ and $\mathcal{L}_I$ in the DA case deal with an $\{S+1\}$-multiclass task involving also the target together with the source domains.

As can be noted from Equation (4.8), the number of meta-parameters of our approach is very limited. For λ we use the same rule introduced by [46] that grows the importance of the feature domain discriminator with the training epochs: $\lambda_k = \frac{2}{1+exp(-10k)} - 1$, where $k = \frac{current\_epoch}{total\_epochs}$. We set $\gamma_k = 0.1\lambda_k$ so that only a small portion of the full gradient of the image



(a) MNIST **(b)** SVHN

(c) SYNTH **(d)** MNIST-M

Figure 4.9. Examples of domain-agnostic digits generated by Hallucinator H in the three source experiments with MNIST-M as target. The top row show images produced in the DG setting by H. The central line shows the original images and in the bottom row we display images produced by H in the DA setting. **Reminder**: although we can always visualize the *domain agnostic images* to better understand the inner functioning of the network, they are not trained to be be pleasant to the human eye.

domain discriminator is backpropagated: in this way we can still get useful similarity measures among the domains while progressively guiding the hallucinator to make them alike. When the image adaptation part is enough to close the domain gap, the feature discriminator loss might be abnormally high causing divergence. We easily obviate such extreme cases by maintaining a record on the initial feature discriminative loss and avoiding the loss backpropagation if it is higher than twice its initial value. Finally, the experimental evaluation indicates that ADAGE is robust to the exact choice of η, thus we keep it always fixed to 0.5 just for simplicity.

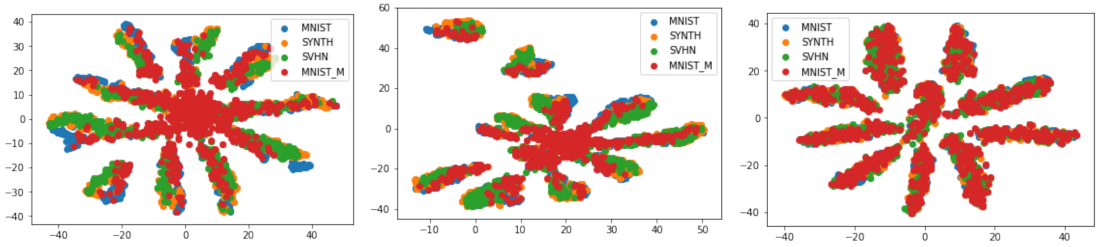
4.2.3 Experiments

	Sources	SVHN MNIST-M SYNTH	SVHN MNIST SYNTH	MNIST-M SYNTH MNIST	Avg.
	Target	MNIST	MNIST-M	SVHN	
DG	combine sources	98.7	62.6	69.5	76.9
DG	MLDG [88]	99.1	61.2	69.7	76.7
DG	ADAGE	99.1	**66.3**	**76.4**	**80.3**
DA	combine sources	98.7	62.6	69.5	76.9
DA	combine DANN [182]	92.5	65.1	77.6	78.4
DA	MDAN [182]	97.9	68.7	81.6	82.7
DA	ADAGE	**99.3**	**88.5**	**86.0**	**91.3**

	Sources	SYNTH MNIST MNIST-M USPS	SYNTH MNIST SVHN USPS	Avg.
	Target	SVHN	MNIST-M	
DG	combine sources	73.2	61.9	67.5
DG	MLDG [88]	68.0	65.6	66.8
DG	ADAGE	**75.8**	**67.0**	**71.4**
DA	combine sources	73.2	61.9	67.5
DA	combine DANN [172]	68.9	71.6	70.3
DA	DCTN [172]	77.5	70.9	74.2
DA	ADAGE	**85.3**	**85.3**	**85.3**

Table 4.4. Classification accuracy results on the digits images. *Left*: experiments with three sources. *Right*: experiments with four sources.

We tested on the DG and multi-source DA scenarios. Our framework can easily switch



(a) combine sources (b) DG ADAGE (c) DA ADAGE

Figure 4.10. TSNE plots of features from the three source experiment with MNIST-M as target

between the two cases with a few key differences. For DG the image I and the feature D domain discriminators deal with S domains, while for DA they need to distinguish among $S+1$ domains including the target. Moreover, in DA, the unlabeled target data trigger the classification block C to activate the entropy loss and to use the source domain weights provided by the image domain discriminator I. Specifically these weights make sure that our classifier is biased towards the sources more similar to the target.

4.2.4 Domain Generalization

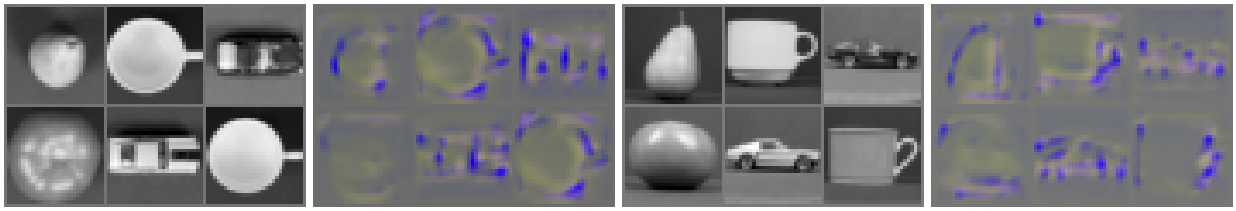
Datasets We focus on five digits datasets and one object classification dataset. *MNIST* [83] contains 70k centered, 28×28 pixel, grayscale images of single digit numbers on a black background. *MNIST-M* [46] is a variant where the background is substituted by a randomly extracted patch obtained from color photos of BSDS500 [8]. *USPS* [45] is a digit dataset automatically scanned from envelopes by the U.S. Postal Service containing a total of 9,298 16×16 pixel grayscale samples; the images are centered, normalized and show a broad range of font styles. *SVHN* [116] is the challenging real-world Street View House Number dataset. It contains over 600k 32×32 pixel color samples, while we focused on the smaller version of almost 100k cropped digits. Besides presenting a great variety of shapes and textures, images from this dataset often contain extraneous numbers in addition to the labeled, centered one. The Synthetic Digits (*SYNTH*) collection [46] consists of 500k images generated from Windows^{TM} fonts by varying the text (that includes different one-, two-, and three-digit numbers), positioning, orientation, background and stroke colors, as well as the amount of blur. Finally, the *ETH80 object dataset* consists of 8 object classes with 10 instances for each class and 41 different views of each instance with respect to pose angles. All the images are subsampled to 28×28 and greyscaled.

Scenarios We consider three experimental scenarios on digits images already presented in previous work. A first case from [182] involves **three sources** chosen in {MNIST, MNIST-M,

	Target	M_0	M_{15}	M_{30}	M_{45}	M_{60}	M_{75}	Avg.
	D-MTAE [49]	82.5	96.3	93.4	78.6	94.2	80.5	87.6
	CCSA [114]	84.6	95.6	94.6	82.9	94.8	82.1	89.1
DG	MMD-AAE [89]	83.7	96.9	95.7	85.2	95.9	81.2	89.8
	CROSS-GRAD [139]	88.3	**98.6**	**98.0**	97.7	**97.7**	91.4	**95.3**
	ADAGE	**88.8**	97.6	97.5	**97.8**	97.6	**91.9**	95.2

Table 4.5. Domain Generalization accuracy results on experiments with five MNIST-rotated sources. For compactness we only indicate the considered target.

	Target	ETH_{00}	ETH_{22}	ETH_{45}	ETH_{68}	ETH_{90}	Avg.
DG	combine sources	70.0	93.8	96.2	98.8	81.2	88.0
	D-MTAE [49]	-	-	-	-	-	87.9
	MLDG [88]	**70.0**	85.0	95.0	97.5	73.7	84.2
	ADAGE	67.5	**95.0**	**100.0**	**100.0**	**88.8**	**90.2**



ETH_{00} hall. ETH_{00} ETH_{90} hall. ETH_{90}

Table 4.6. Top: DG accuracy results on experiments with ETH-80 rotated sources. Bottom: real and hallucinated image examples.

SYNTH, SVHN}. Each dataset, with the exception of SYNTH, is used in turn as target. All the images are resized to 28×28 pixels and subsets of 20k and 9k samples are chosen respectively from each source and from the target. A second case from [172] involves **four sources** by adding USPS to the previous dataset group, and focuses on two possible targets, SVHN and MNIST-M. Even in this case the images are resized to 28×28 pixels, and 25/9k samples are drawn from each dataset to define the source/target sets. A third case from [49] involves **five sources** and exploits rotated variants of MNIST. Specifically we started by randomly choosing 100 images for each of the 10 classes and indicating this basic view with M_0. The versions $\{M_{15}, M_{30}, M_{45}, M_{60}, M_{75}\}$ are obtained by rotating the images of 15 degrees in counterclock-wise direction. Note that the authors of [172] kindly shared the exact splits used for their paper, while for all the other experiments we considered multiple random selections of the samples from the datasets. For the **object classification experiment**, we followed [49] focusing on the ETH80-p setting that covers 5 domains built from equally spaced pitch-rotated views of the 8 objects. Each domain is considered in turn as the target, while the remaining ones are the sources.

Implementation Details We have developed ADAGE with the Pytorch framework. For our experiments all the datasets were normalized and zero-centered. The mean and standard deviation of the target for data normalization are calculated batch-by-batch during the testing process. A standard random crop of $90 - 100\%$ of the total image size was applied as data augmentation. The training procedure runs for 600 epochs with Adam optimizer [77] . The initial learning rate is set to $1e^{-3}$ and step down after 80% of the training. All the experiments are repeated tree times and we report the average on the obtained classification accuracy.

Results in Table 4.4 (top part) As a main baseline for the three and four sources settings we use the naïve *combine sources* strategy that consists in learning a classifier on all the source data combined together. For a fair comparison we produced these results by keeping on only the feature extractor F and the classifier C, while turning off all the adaptive blocks in the domain generalizer. We benchmark against the meta-learning method MLDG [88] using the code provided by the authors and running the experiments on our settings. The obtained results indicate that ADAGE outperforms all the reference sota baselines in DG both using three and four sources with an advantage up to 3 percentage points. Interestingly, using four sources slightly worsens the performances when SVHN is the target: our interpretation is that adding the USPS dataset increases the domain shift between the training and test domains, making the adaptation somehow more difficult.

Results in Table 4.5 For the five sources experiments on rotated digit images we benchmark against two autoencoder-based DG methods D-MTAE and MMD-AAE respectively presented

in [49] and [89], as well as against the metric-learning CCSA method [114] and the very recent CROSS-GRAD [139]. The results indicate that ADAGE outperforms three of the four competitors and has results similar to CROSS-GRAD which proposes an adaptive solution based on data augmentation that could potentially be combined with ADAGE.

Results in Table 4.6 For the object classification experiments on ETH80-p, ADAGE obtains an average accuracy of 90.2% , outperforming D-MTAE [49] and MLDG [88].

4.2.5 Domain Adaptation

We extend our analysis to the multi-source DA setting considering the same three and four scenarios on digits images described in the previous section. In terms of implementation details, the only difference with respect to what already discussed above is that we now have all the unlabeled target samples at training time, so their mean and standard deviation can be calculated at once. Moreover, for the training process we used the RmsProp optimizer [158], running for 200 epochs with initial learning rate of $5e^{-4}$.

Results in Table 4.4 (bottom part) We benchmark ADAGE against reference results from previous DA works. In particular for the three sources experiments the comparison is with the Multisource Domain Adversarial Network MDAN [182]. Since this method builds over the DANN algorithm [46] the result obtained with DANN applied on the combination of all the sources (combine DANN) is also reported. For the four sources experiments the main comparison is instead with the Deep Cocktail Network (DCN) [172], a recent method able to work even with partial class overlap among the sources. The results indicate that ADAGE outperforms the competing methods also in this setting with an average advantage up to 11 percentage points. As a further test we verified the obtained weights assigned by the I network component in the three source setting: when using MNIST-M as target they converge to $\{0.5, 0.3, 0.2\}$ respectively for MNIST, SVHN, SYNTH, which sounds reasonable given the visual similarity among the domains.

While ADAGE is specifically tailored for the multi-source settings, we checked its behaviour also in the case of single source DA with access to unlabeled target data. As a proof of concept experiment, we tested ADAGE using SVHN as source and MNIST as target. With the same protocol used in our DA experiments, we achieve 95.7% accuracy, which is on par with the very recent [59] and better than several others competitive methods [1, 128, 95, 132].

4.2.6 Ablation Study and Qualitative Results

Our ablation study analyzes the effect of progressively enabling the key components of the domain generalizer alone, and in combination with the hallucinator.

Results in Table 4.7 we start by evaluating the performance obtained when we do not generate the domain agnostic samples. In this case the hallucinator H is removed from the network and the original images of all sources are fed directly to the domain generalizer. In this case, since we cannot modify the original images, the only active adaptive component is D that operates on the features. Moreover the classifier can also take advantage of the entropy loss (that we indicate with E) in the DA setting. The results indicate that feature alignment is very helpful for DA but can induce confusion in DG with results lower than those of the combine sources baseline. Another important result is obtained when only H is enabled and the features are extracted directly from the generated images with the components I and D off. In this case the network is not performing any effort to align the domains and the final accuracy is just slightly better than the combine sources baseline. This shows that the advantage of ADAGE is clearly not just due to the use of a deeper architecture. Keeping the hallucinator H active together with the D component produces a good advantage in accuracy but only in the DA

combine sources		D	D+E	H	H+E	H+D	H+I	H+D+I	H+E+I	H+D+E	H+D+E+I	H_{res}+D+E+I
62.6	DG	53.0	53.0	63.2	63.2	62.2	61.4	66.3	61.4	62.2	66.3	65.8
	DA	65.9	75.1		63.9	69.9	60.8	68.8	63.9	82.4	88.5	87.6

Table 4.7. Ablation analysis on the experiment with three sources and target MNIST-M. We turn on and off the different parts of the model: **H**= Hallucinator, **E**= Entropy, **D**= Feature Domain Discriminator, **I**= Image Domain Discriminator. Note that H+D+E+I corresponds to our whole method ADAGE.

setting ($H + D = 69.9$). Here adding E provides a further advantage ($H + D + E = 82.4$). Overall the entropy loss appears quite effective in the considered scenario: our intuition is that the presence of multiple sources helps reducing the risk that the entropy loss might mislead the classifier. The contribution of the image domain discriminator I is negligible by itself and this behavior can be explained considering that we backpropagate only a small part of the I gradient ($\gamma = 0.1\lambda$, see Section 4.2.2). However its beneficial effect becomes evident in collaboration with the other network modules: passing from $H + D + E$ to $H + D + E + I$ implies an improvement in accuracy of at least 4 percentage points in the difficult DG setting, which shows that the adversarial guidance provided by I on H allows for an image adaptation process complementary to the feature adaptation one. Note that, since the image domain discriminator backpropagates only on the hallucinator, it is not possible to test any combination containing I but not H.

Finally we propose a benchmark against an existing residual structure previously used to transform pixels in depth image colorization [21]. When plugging in this residual version of the hallucinator (H_{res}) we observe that the overall classification performance is slightly lower than what obtained with our original aggregative H. Besides this small variation, the most important difference is that our hallucinator has only 1/3 of the parameters of [21], thus it is faster in training and allows to avoid overfitting while mapping the source domain images into a compact agnostic space.

4.2.6.1 Qualitative Analysis

Figure 4.9 shows the agnostic images generated by the hallucinator, in the three source experiment with target MNIST-M, while the bottom part of Table 4.6 shows examples of ETH-80 original and hallucinated images. The main effect of H is that of removing the backgrounds and enhancing the edges: this is quite clear in the DG setting for both digits and objects, while in the DA case the produced digits images appear slightly more confused. Figure 4.10 shows the TSNE embedding of features extracted immediately before the final classifier. In the DA setting we completely align the feature spaces of the domains, resulting in a clear per class clustering. In the DG setting the results are less clean, but the clusters are still tighter than those obtained by the combine source baseline.

4.2.6.2 Computational Analysis

ADAGE shows a computational cost similar to SBADA-GAN when used with a single source, while being more stable during the training phase, thus requiring fewer runs to reach its best performance. However, having several sources as inputs simply linearly increase the size of the training set and the corresponding training time, ending on few days of total train time. During the test phase, ADAGE shows a similar speed w.r.t. SBADA-GAN, with $14\tilde{ms}$ of inference time for hallucinator, feature extractor and classifier used in sequence. For ADAGE are worth the same considerations about input image sizes done for SBADA-GAN in Section 4.1.4.6

4.2.7 Conclusions

This section shows the first end-to-end joint image- and feature-level adaptive solution for Domain Generalization. We define a new network, named ADAGE, able to hallucinate domain agnostic images guided by two adversarial adaptive conditions at pixel and feature level. ADAGE can be seamlessly used both for Domain Generalization and multi-source unsupervised Domain Adaptation: it achieves impressive results on several benchmarks, outperforming the current state of the art by a significant margin.

4.3 Towards Multi-Source Adaptive Semantic Segmentation

4.3.1 Introduction

Semantic segmentation has recently become one of the most prominent task in computer vision. Indeed the ability to assign a label to each pixel of an input image is crucial whenever a very detailed description of the observed scene is needed, as in fine-grained object categorization [180] and autonomous driving [179, 160]. However, due to the complexity of manual labeling each image pixel, this task is plagued by the scarcity of large annotated datasets, which are instead essential to leverage the power of deep learning algorithms. Synthetic images appear a useful alternative, but they reduce only in part the described issue. In the case of urban scene scenarios for autonomous driving, computer games can be used to generate automatically images with their ground truth labels, but their level of realism is still low which induces the further need of domain adaptation methods. Thus, while solving the lack of data problem, other challenges come from the development of methods able to reduce the domain gap. Up today, those two aspects of the same problem has always been tackled separately. On one side several research groups have focused on developing different simulators with an increasing set of visual details like urban layouts, buildings, vehicles and several weather conditions, with the aim of augmenting the realism of the produced images [40, 125]. On the other side, many recent works focus on integrating techniques to align the domains either at feature, pixel or output label space level, even considering combination of those levels with different adversarial losses [160, 26, 1]. Each of the proposed synthetic domains is generally used to train a model and test it on real images, but the different synthetic sources are always kept separated even if this choice limits again the amount and variance of annotated samples usable as source. The domain adaptation literature for object classification has shown that integrating multiple sources helps generalization [41, 182, 42]. With our work we import this strategy for the first time in the semantic segmentation framework, studying how the positive trend can be maintained by practically merging the two solutions described above. The path to this goal is not trivial due to the practical differences in class statistics across domains, as well as in texture, resolution and aspect ratio for which we propose best practice rules. Moreover, we go over the simple source sample combination, exploiting a multi-level strategy that adapts each single source to the target while cooperating with the adaptation of the joint data source. Besides the standard synthetic to real direction, we extend our analysis to the case of a synthetic dataset used as target when the source combines real images and a different synthetic collection. This setting allows to better understand the difference across various synthetic sources and paves the way to the simultaneous exploitation of both the synthetic-to-real and real-to-synthetic adaptive directions [128].

4.3.2 Method

An overall view of the proposed architecture can be seen in Figure 4.11. Our domain adaptation method starts with a segmentation network $\mathbf{G}$ which takes the sources annotated images (I^s, Y^s) and the unlabeled target images (I^t) as input. The network ends with an *Adaptive Classification Module* that contain separate classification branches for each source as well as a domain discriminator $\mathbf{D}$. Each source classification branch produces a segmentation softmax output $P^s = \mathbf{G}(I^s) \in \mathbb{R}^{H,W,C}$, where (H, W) are the height and width image dimensions and C is the number of categories. The used semantic segmentation loss is

$$\mathcal{L}^s_{seg}(I^s) = -\sum_{h,w} \sum_{c=1,ldots,C} Y^s_{h,w,c} \log(P^s_{h,w,c}), \tag{4.9}$$

where $s = 1, 2$ for the two sources.

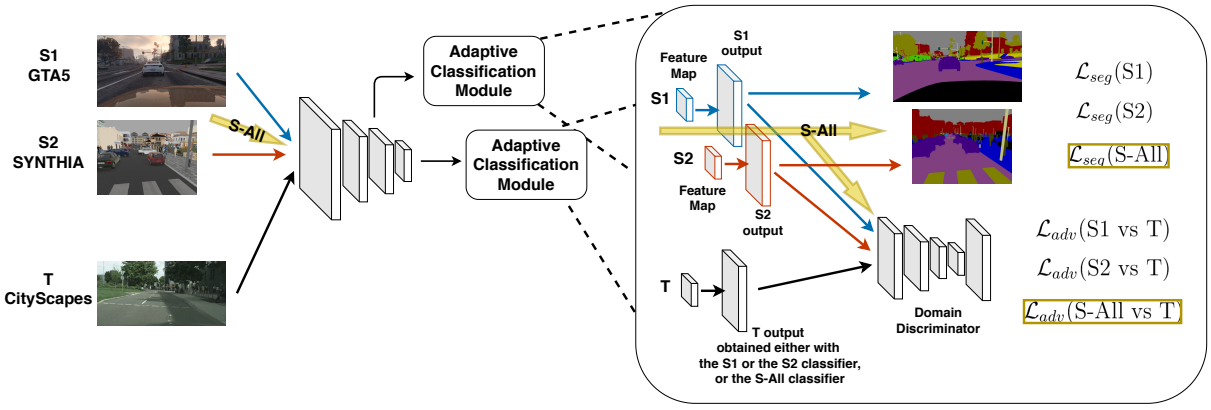


Figure 4.11. Training Phase: our network has two *Adaptive Classification Modules* at different levels. In each module the source segmentation is predicted either with two separate source-specific branches or just using one overall *S-All* branch (we did not explicitly draw the *S-All* branch to avoid cluttering the image). The segmentation loss is computed based on the sources ground truth. Moreover a domain discriminator is used adversarially to reduce the domain shift comparing the target T either with each source-specific output, or with the output obtained by *S-All*.

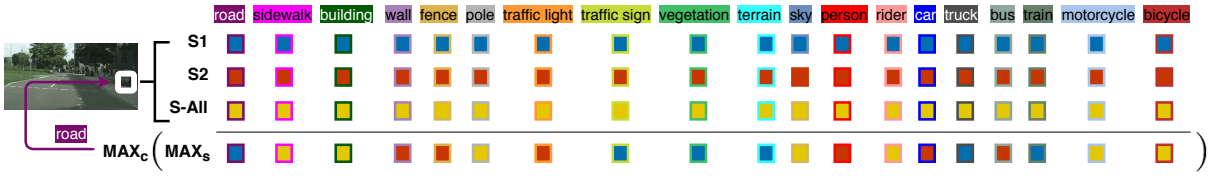


Figure 4.12. Test Phase: each classifier produces a semantic segmentation output (*S1*:blue, *S2*:red, *S-All*:yellow). For every pixel we apply a max-pooling operator over the three outputs. Finally the class assigned to the pixel is the one with the highest score over the C classes ($C = 19$ when testing on Cityscapes).

The domain discriminator **D** takes as input the segmentation output of both the source and target data and is optimized through the binary loss

$$\mathcal{L}_d(P) = -\sum_{h,w}(1-z)\log(\mathbf{D}(P)_{h,w,0}) + (z)\log(\mathbf{D}(P)_{h,w,1}), \tag{4.10}$$

with z = 0 if the sample is drawn from the target domain, and z = 1 for the sample from the source domains. Finally the adversarial loss whose gradients backpropagates on the segmentation network to maximize the confusion between P^s and P^t is

$$\mathcal{L}_{adv}(I^t) = -\sum_{h,w}\log(\mathbf{D}(P^t)_{h,w,1}). \tag{4.11}$$

To further improve the adaptation effect involving inner-features, another adaptive classification module is also applied to a lower network level. Thus the overall loss is

$$\mathcal{L}(I_s, I_t) = \sum_{k=feature,output}\left\{\sum_{s=1,2}\lambda^s_{seg}\mathcal{L}^s_{seg}(I^s) + \lambda^s_{adv}\mathcal{L}^s_{adv}(I^t)\right\}_k \tag{4.12}$$

and the network is optimized on the basis of the following criterion

$$\max_{\mathbf{D}}\min_{\mathbf{G}}\mathcal{L}(I_s, I_t). \tag{4.13}$$

We also repeated the whole training considering a single source branch that sees all the images together regardless of the domain identity: we indicate it as *S-All*, with its own $\mathcal{L}_{seg}^{S-All}$ loss. From the predictions of each available source and from *S-All*, we finally need a single segmentation target output. For this purpose we apply a max-pooling operator that runs on the prediction logits $\hat{Y}$ and selects the highest score per class, then followed by a second max-pooling over the classes:

$$\text{Assigned Label}(h,w) = \max_{c=1...,C} \max_{s=\{1,2,S-All\}} (\hat{Y}^s_{h,w,c}). \tag{4.14}$$

Note that, by keeping only *S-All* we fall back to the single source original method in [160].

4.3.3 Adding Pixel-level Adaptation

As explained above the proposed adaptation process is applied both at the output and at the feature level. Inspired by the extensive GAN-based literature on style-transfer, we integrated in our method also a pixel-level adaptation process, directly modifying the input images. Specifically we used the Unsupervised Image-to-Image Translation (UNIT, [95]) method. It assumes that a pair of corresponding images in two different domains can be mapped to the same latent code in a shared space. By using a Coupled GANs [94] and imposing weight sharing constraints on the mapping functions, the method is able to change the style of an image so that it looks like coming from a different domain. We applied UNIT to produce target-like copies of the source images. After this (totally unsupervised) pre-processing step, the proposed architecture is used on the new stylized sources.

4.3.4 Experiments

We used three publicly available datasets in our experiments: Cityscapes [32], GTA5 [123] and SYNTHIA [125]. We ran each experiment by choosing two datasets as sources domains, and the third as target (unsupervised) domain. In previous works on those datasets, the standard setting consisted in evaluating the recognition performance only of the shared classes across domains, thus operating a subselection on Cityscapes when used against Synthia. We find it natural that different data collections may have only partially overlapping class sets and it should not be necessary to proceed every time to an ad-hoc class choice [172]. Thus, we decided to keep all the datasets with their own original categories. Furthermore we investigate the effect of the resolution on the final segmentation accuracy considering a high and a low resolution case. In the first, all the images keep their own original size, while in the second they are all downscaled by halving the native image dimensions. Finally we remark that the three analyzed domains present remarkable differences on mean values. Since the adversarial approaches are very sensitive to non-zero mean data, we have chosen to work by removing from each dataset its own calculated image mean.

4.3.4.1 Implementation details

The main backbone of our segmentation network is the DeepLabv2 [24], which uses a ResNet-101 pretrained on ImageNet and COCO [92]. This architecture incorporates atrous convolution, which effectively enlarge the field of view of filters without increasing the number of parameters. Within the *Adaptive Classification Module* we have two separate network branches, one for each source, producing a 2D predictions followed by an interpolation function that rises the resolution to that of the original ground truth label (during training). At test time the same interpolation function was used to calculate accuracy using the target ground truth as reference. Following [160, 188], the module contains also a discriminator that classify the images on the basis of their

Res	Sources	Target	No Adapt	*S1*	*S2*	*S-All*	Max Merge
Orig.	GTA5, Synthia	Cityscapes	39.98	39.55	34.51	41.81	**42.76**
	Cityscapes, GTA5	Synthia	35.55	35.25	34.07	36.37	**37.52**
	Cityscapes, Synthia	GTA5	37.97	41.17	23.60	**40.57**	39.49
Redu.	GTA5, Synthia	Cityscapes	-	39.44	33.36	40.89	**41.32**
	Cityscapes, GTA5	Synthia	-	30.52	30.02	32.87	**33.11**
	Cityscapes, Synthia	GTA5	-	44.93	23.28	41.87	**42.78**

Table 4.8. Performance values on the chosen experiments expressed with mIoU. The proposed method outperform the no adaptation results as well as single branches and *S-All* method on all the experiments but the one with GTA5 as target at high resolution, where it lags behind *S-All* result due to the poor performance of S2 branch.

Setting	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motocycle	Bycycle
T: Cityscapes, *S-All*	85.0	36.3	79.9	21.5	18.0	29.5	25.5	19.3	81.4	23.5	78.0	57.6	23.9	75.4	35.0	40.7	2.6	31.7	29.9
T: Cityscapes, Max Merge	87.8	37.1	80.2	20.3	14.9	29.8	26.0	20.5	82.0	31.4	78.0	57.6	25.3	80.5	31.6	43.5	0.0	29.8	36.1
T: GTA5, Max Merge	82.4	29.4	56.5	41.6	6.7	31.1	26.4	19.3	64.4	7.4	88.1	42.8	50.3	74.2	36.8	31.4	0.0	32.5	29.3
T: Synthia, Max Merge	68.3	66.8	86.6	1.7	1.7	39.5	27.2	10.9	73.7	0.0	90.6	55.5	33.7	55.7	0.0	48.5	0.0	23.0	29.3

Table 4.9. Intersection over Union for each experiment category. The experiments are performed on full resolution. Some particular categories (road, terrain, cars) seems to better exploit the power of the proposed method w.r.t the *S-All* one, and they contribute to the final accuracy increase due to their frequent presence on the scene.

source or target domain label. The discriminators model is the same of DCGAN [121], with convolutional layers interspersed by Leaky Relu non-linearities. Note that although there are two adaptive classification modules in the network, the classification output produced by the inner module has shown to be less reliable than the ending one which is actually the only used at test time.

The network is trained with the Adam solver and learning rate 0.0001, while for the architecture hyperparameters we kept the same values of [160]. The number of iterations was set to 50k, but we observed convergence already after 20k iterations.

4.3.5 Results

The main experiment results are reported in Table 4.8. The values reported are the mean Intersection Over Union (mIoU) which is the standard accuracy measurement used on semantic segmentation tasks.

The proposed method is able to improve the *S-All* results on almost all the performed experiments, even while the single source branch prove to reach lower accuracy w.r.t. the *S-All* result, getting a boost ranging from 0.4% to 1.2%, while w.r.t. the results without any adaptation at all (No Adapt column) the difference of performance are from 1.5% to 2.7%. Looking more into detail, the most difficult setting is the one with GTA5 as target domain, as the Synthia source domain fails to properly reach an acceptable accuracy, and this worsen the final performance in the full resolution case. The input data resolution has an impact on final accuracy ranging from 1.44% in the case of Cityscapes as target, to 4.41% in the case of Synthia target, showing that in order to obtain the best possible accuracy is preferable to keep resolution as high as possible, while at the same time demonstrates that in some cases a lower resolution can dramatically speed up the training phase (around 3x faster in our case) while losing a small amount of accuracy (target Cityscapes experiment).

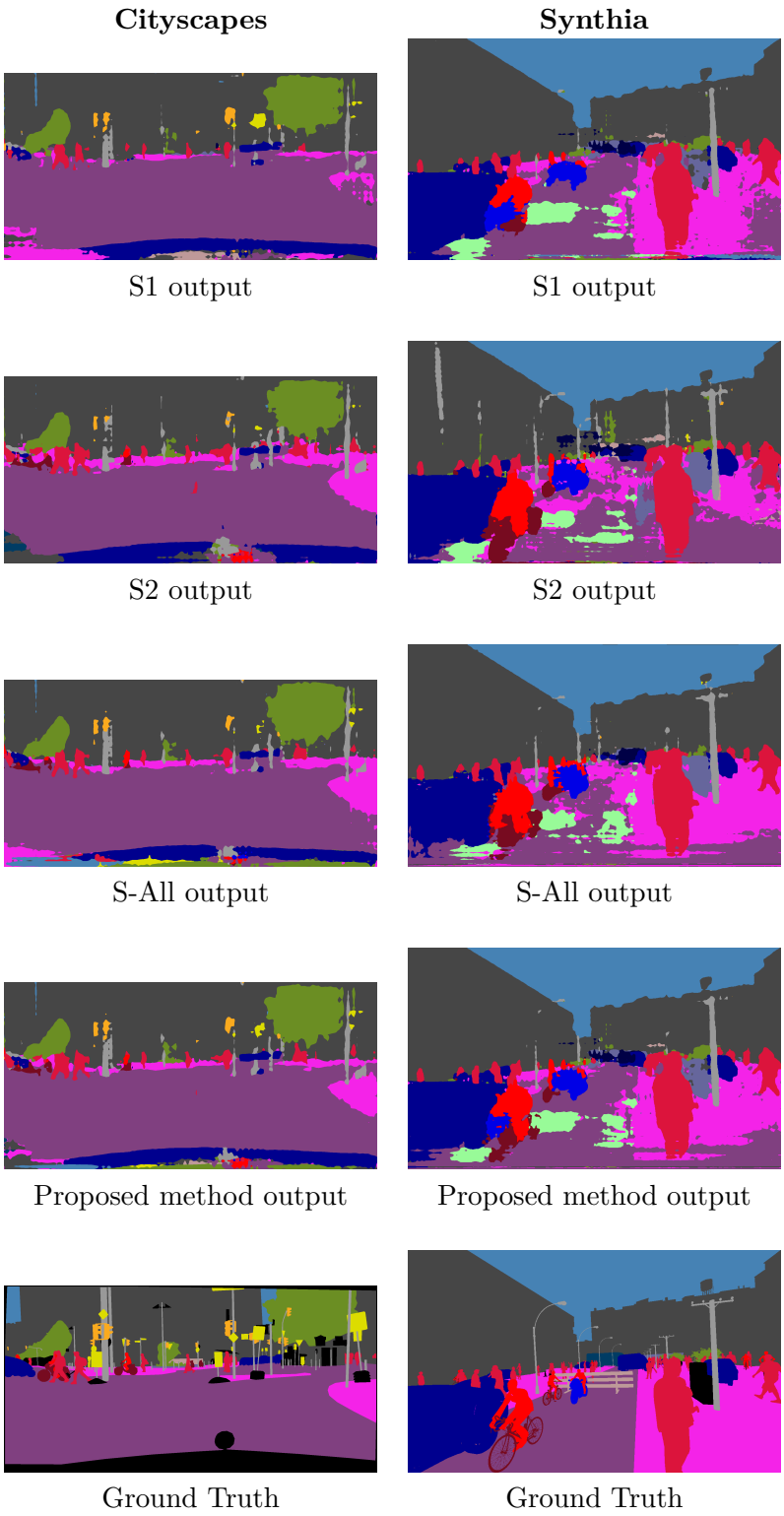


Figure 4.13. Predicted labels in the case of Cityscapes and Synthia target datasets. The proposed method is able to better recognize some parts of the images like road pieces (dark violet) w.r.t. single branches or *S-All* approach.

Looking at per-class IoU measurements in Table 4.9, we noticed how the overall increase of performance can be attributed to some specific classes IoU improvement; terrain, road, vegetation and car seem to be the classes which better take advantage of the proposed method. This effect can be noticed also in the produced images in Figure 4.13, where some parts of the road are better reproduced in our method w.r.t *S-All* output.

A final additional experiment have been performed by applying UNIT method to the GTA5 and Synthia datasets in order to convert their style to the Cityscapes one, after which the proposed architecture have been trained regularly with two stylized GTA5 and Synthia datasets as sources and Cityscapes as target. The measured accuracy obtained by merging the two branches *S1* and *S2* is 44.5%, which is very promising result, taking also into account that it can be further improved by exploiting *S-All* branch too. The UNIT architecture and our method have been trained separately because of the huge amount of GPU memory required in order to train them jointly.

4.3.5.1 Computational analysis

The proposed method is very heavy from a computational point of view at training time. Even without UNIT, the combination of high resolution input, an high-capacity backbone, four output branches and two discriminators requires to use an high-end 16GB GPU for one week just to perform one training. Unfortunately, we failed to run the method on usual 12GB GPU without using some Pytorch hacks which slowed down even more the training phase. However, it should be noted that similar issues are shared with all networks which works on semantic segmentation for urban scenes. At test time the model requires $170ms$ to load one input image and to give the predicted semantic output, requiring extra care for real time uses.

4.3.6 Conclusions

In this section we have presented a study on multi-source domain adaptation on semantic segmentation tasks. The study revealed how simply putting all the sources together is a sub-optimal approach, and we proposed a simple method to leverage on individual sources as well as *S-All* method. The experiment performed show promising results, with a small but steady improvement on the majority of settings. Further investigation is required in order to better understand the effect of some parameters like the chosen data resolution and the datasets means, and the possibility of applying a style transfer method like UNIT jointly with the domain adaptation method into a fully integrated architecture.

Chapter 5

Final considerations and Future Work

This dissertation started by studying the problems of Domain Adaptation and Transfer Learning on Computer Vision. We restricted our attention to RGB, depth and RGB-D modalities, for which we presented some novel methods and approaches with the aim to reduce the statistical distance between source and target domains, and to properly transfer knowledge from one domain to another.

Regarding Domain Adaptation, we have seen how several methods have already been developed by the research community, with algorithms that try to reduce the domain shift on a pixel level or on a feature level. With the rise of generative adversarial models, a plethora of techniques has been developed to get an image conversion from a source domain to a target one, by preserving some characteristics of the former and by inheriting other properties, as the image style, from the latter. These approaches, combined with more traditional feature-based adaptation procedures, led to a partial reduction of performance loss on several domains. Starting from this spot, we developed SBADA-GAN, a novel generative network which reached state of the art accuracy on most of the evaluated digits datasets by converting source and target images back and forth at a pixel level. Then, we developed ADAGE, a new approach for hybrid pixel/feature adaptation, which gives strong performance on multi-source domain adaptation and generalization settings.

Moving on a different job, we have seen how semantic segmentation quickly has quickly become one of the most important tasks in Computer Vision, with a spin-off for the whole development of important fields like autonomous driving, medical images analysis, and so on. With the purpose of performing semantic segmentation on urban scenes, a preliminary set of experiments led to the development of a deep neural network capable of reducing the domain shift directly on the output level in a multi-source approach.

Focusing on the depth domain, we highlighted the critical issues in the object recognition tasks through the use of depth sensors, which enables the use of robots in several difficult conditions (like low brightness) but at the same time places new challenges for researchers. Until few years ago a general dataset for depth object classification was not available, forcing to fine-tuning on ImageNet pretrained models though the use of some mapping functions. To overcome this problem, we developed the VANDAL dataset and the associated DepthNet model, capable of good accuracy on standard depth benchmarks by exploiting synthetically generated depth maps. At the same time, we developed $(DE)^2CO$, a new alternative approach to standard fine-tuning for mapping depth images to the RGB domain, which shows state of the art performance on the majority of evaluated depth and RGB-D settings. Last work on the RGB-D domain is an integrated system for egocentric action recognition, which exploit a simple depth-RGB fusion

method in order to apply an attention mechanism both on a pixel level and on a feature level, as well as a Mean Teacher model to further enhance performance. This approach shows promising results, with state of the art accuracy on the difficult GUN-71 dataset.

In conclusion, our work demonstrates that sensible gain of performance can be obtained by either exploiting pre-existing knowledge, or by properly adapt data or model between domains. We are confident that the ideas, models and approaches presented in this thesis can contribute to the advances in the field of Domain Adaptation and Transfer Learning. At the same time, we believe that these works can represent a starting point for further improvements.

A more detailed study on how to properly exploit synthetically generated images, with the development of some guiding principles, could led to an improved version of VANDAL dataset and to a set of even more robust visual features, which could be used for different depth recognition tasks. At the same time, it is not clear if the fine-tuning step could be completely removed when using an additional module as $(DE)^2CO$ on any depth→RGB mapping task, requiring further studies to understand this phenomenon. The newborn attention mechanism perfectly paired with the task of action recognition from videos, but much more work is needed for understanding how to properly exploit features activation and depth data at the same time; for example, an integrated model which applies depth maps as an additional source of attention directly on the high level features could further increase the overall accuracy. Another interesting research topic could be how to benefit from depth maps, optical flow and RGB data on a single architecture while limiting the number of additional network branches, thus providing a fast, lightweight integrated model which could run on mobile devices like smartphones.

Going back to the RGB domain, the recent development of new GAN architectures and training principles led to the use of high resolution data, with impressive results for tasks like super resolution and realistic images generation. These innovations could be brought into the SBADA-GAN framework for source↔target mapping at higher resolutions. At the same time, the inability of GAN models to coherently map objects with different shapes is preventing their use on settings with huge domain shifts, as in the Office dataset. In order to solve this issue, new generative techniques are probably needed, which could implement a trade-off mechanism to choose which and how many properties should be preserved from the source domain, as is performed by some neural style transfer algorithms. A very challenging setting is the Domain Generalization, where ADAGE performed strongly on the limited setting of digits recognition. In many practical cases we do not have any clue on the target data where the visual classifier will operate; that requires the development of new solutions to reduce overfitting on the training dataset and to increase generalization abilities of the trained visual features. Semantic segmentation field is still at the beginning: deeper knowledge is required to maximize the methods's accuracy, especially on difficult settings like urban scenarios.

We have just scraped the surface of the big and challenging world of Deep Learning.

Bibliography

[1] CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In *Blind Submission, International Conference on Learning Representations (ICLR)*, 2018. 8, 10, 47, 53, 59, 62

[2] Self-ensembling for visual domain adaptation. In *Blind Submission, International Conference on Learning Representations (ICLR)*, 2018. 47

[3] A. Aakerberg and K. Nasrollahi. Depth value pre-processing for accurate transfer learning based rgb-d object recognition. In *IJCAI*, 2017. 9, 24, 25, 26, 28, 31, 32

[4] C. H. Anderson, D. C. Van Essen, and B. A. Olshausen. Directed visual attention and the dynamic control of information flow. In *Neurobiology of attention*, pages 11–17. Elsevier, 2005. 1

[5] G. Andrew, R. Arora, J. Bilmes, and K. Livescu. Deep canonical correlation analysis. In *International conference on machine learning*, pages 1247–1255, 2013. 37, 40

[6] G. Angeletti, B. Caputo, and T. Tommasi. Adaptive deep learning through visual domain localization. In *International Conference on Robotic Automation (ICRA)*, 2018. 7

[7] A. Angelova and P. M. Long. Benchmarking large-scale fine-grained categorization. In *IEEE Winter Conference on Applications of Computer Vision*, pages 532–539. IEEE, 2014. 20

[8] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, 2011. 10, 57

[9] U. Asif, M. Bennamoun, and F. Sohel. Efficient rgb-d object categorization using cascaded ensembles of randomized decision trees. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1295–1302. IEEE, 2015. 13

[10] U. Asif, M. Bennamoun, and F. A. Sohel. Rgb-d object recognition and grasp detection using hierarchical cascaded forests. *IEEE Transactions on Robotics*, 2017. 9

[11] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. Factors of transferability for a generic convnet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9):1790–1802, 2016. 29

[12] M. Blum, J. T. Springenberg, J. Wülfing, and M. Riedmiller. A learned feature descriptor for object recognition in rgb-d data. In *2012 IEEE International Conference on Robotics and Automation*, pages 1298–1303. IEEE, 2012. 13

[13] L. Bo, X. Ren, and D. Fox. Unsupervised feature learning for RGB-D based object recognition. In *13th International Symposium on Experimental Robotics, ISER*, 2012. 9, 24, 25, 26

[14] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *arXiv preprint arXiv:1612.05424*, 2016. 26

[15] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain Separation Networks. In *Neural Information Processing Systems (NIPS)*, 2016. 45, 46, 47, 48, 53

[16] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with gans. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 8, 42, 45, 46, 47

[17] L. Bruzzone and M. Marconcini. Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32 (5):770–787, 2010. 44

[18] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. Rota Bulò. Autodial: Automatic domain alignment layers. In *International Conference on Computer Vision (ICCV)*, 2017. 7

[19] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. Rota Bulò. Just dial: domain alignment layers for unsupervised domain adaptation. In *International Conference on Image Analysis and Processing (ICIAP)*, 2017. 7

[20] F. M. Carlucci, P. Russo, and B. Caputo. A deep representation for depth images from synthetic data. In *ICRA*, 2017. 2, 23, 32

[21] F. M. Carlucci, P. Russo, and B. Caputo. (de)2co: Deep depth colorization. *IEEE Robotics and Automation Letters*, 2018. 2, 35, 53, 55, 60

[22] R. P. T. T. C. B. Carlucci, F. M. Hallucinating agnostic images to generalize across domains. 2019. 2

[23] A. J. Champandard. Semantic style transfer and turning two-bit doodles into fine artworks. *arXiv preprint arXiv:1603.01768*, 2016. 9

[24] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018. 10, 64

[25] T. Q. Chen and M. Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016. 9

[26] Y. Chen, W. Chen, Y. Chen, B. Tsai, Y. F. Wang, and M. Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *ICCV*, 2017. 10, 62

[27] Y. Cheng, X. Zhao, K. Huang, and T. Tan. Semi-supervised learning for rgb-d object recognition. In *2014 22nd International Conference on Pattern Recognition*, pages 2377–2382. IEEE, 2014. 13, 20

[28] Y. Cheng, R. Cai, X. Zhao, and K. Huang. Convolutional fisher kernels for rgb-d object recognition. In *2015 International Conference on 3D Vision*, pages 135–143. IEEE, 2015. 9, 13, 18, 19, 20

[29] Y. Cheng, X. Zhao, K. Huang, and T. Tan. Semi-supervised learning and feature evaluation for rgb-d object recognition. *Computer Vision and Image Understanding*, 139:149–160, 2015. 13, 20

[30] Z. Cheng, Q. Yang, and B. Sheng. Deep colorization. In *ICCV*, 2015. 24

[31] F. Chollet. keras. https://github.com/fchollet/keras, 2017. 46

[32] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 12, 64

[33] K. Crammer, M. Kearns, and J. Wortman. Learning from multiple sources. *J. Mach. Learn. Res.*, 9:1757–1774, June 2008. ISSN 1532-4435. 8

[34] D. Damen, H. Doughty, G. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The epic-kitchens dataset. In *Proc ECCV*, 2018. 41

[35] H. Daume III. Frustratingly easy domain adaptation. In *Annual Meeting of the Association of Computational Linguistics (ACL)*, 2007. 8

[36] M. F. Deering. The limits of human vision. In *2nd International Immersive Projection Technology Workshop*, volume 2, 1998. 1

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 23, 27

[38] A. D'Innocente and B. Caputo. Domain generalization with domain-specific aggregation modules. In *German Conference on Pattern Recognition (GCPR)*, 2018. 8, 53

[39] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014. 9

[40] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Conference on Robot Learning (CoRL)*, 2017. 10, 62

[41] L. Duan, I. W. Tsang, D. Xu, and T.-S. Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *International Conference on Machine Learning (ICML)*, 2009. 8, 62

[42] L. Duan, D. Xu, and I. W. Tsang. Domain adaptation from multiple sources: A domain-dependent regularization approach. *IEEE Transactions on Neural Networks and Learning Systems*, 23(3):504–518, March 2012. 62

[43] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard. Multimodal deep learning for robust rgb-d object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 681–687. IEEE, 2015. 9, 13, 18, 19, 20, 24, 25, 26, 28, 29, 31, 32, 35

[44] B. Foundation. Blender. URL https://www.blender.org/. 2, 16

[45] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001. 10, 57

[46] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030, 2016. 7, 8, 10, 42, 46, 47, 55, 57, 59

[47] G. Garcia-Hernando, S. Yuan, S. Baek, and T. Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *Proc. CVPR*, 2018. 33

[48] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 9

[49] M. Ghifary, W. B. Kleijn, M. Zhang, and D. Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *International Conference on Computer Vision, (ICCV)*, 2015. 9, 57, 58, 59

[50] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision (ECCV)*, 2016. 7, 46, 47

[51] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 9

[52] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Neural Information Processing Systems (NIPS)*. 2014. 7, 42

[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 2

[54] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *International Conference on Computer Vision (ICCV)*, 2011. 8

[55] J. Guo and S. Gould. Deep cnn ensemble with data augmentation for object detection. *arXiv preprint arXiv:1506.07224*, 2015. 38

[56] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European conference on computer vision*, pages 345–360. Springer, 2014. 9, 13, 24

[57] S. Gupta, J. Hoffman, and J. Malik. Cross modal distillation for supervision transfer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 9

[58] A. Habrard, J.-P. Peyrache, and M. Sebban. Iterative self-labeling domain adaptation for linear structured image classification. *International Journal on Artificial Intelligence Tools*, 22(5), 2013. 44

[59] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *International Conference on Computer Vision (ICCV)*, 2017. 7, 46, 47, 48, 52, 59

[60] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *2014 IEEE international conference on Robotics and automation (ICRA)*, pages 1524–1531. IEEE, 2014. 14

[61] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2000. 1

[62] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 18, 24, 26, 27

[63] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 37, 40

[64] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 9

[65] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *European Conference on Computer Vision (ECCV)*, 2012. 8, 53

[66] J. Hoffman, S. Gupta, and T. Darrell. Learning with side information through modality hallucination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 826–834, 2016. 9

[67] J. Hoffman, S. Gupta, J. Leong, S. Guadarrama, and T. Darrell. Cross-modal adaptation for rgb-d detection. In *International Conference in Robotics and Automation (ICRA)*, 2016. 9

[68] J. Hoffman, M. Mohri, and N. Zhang. Algorithms and theory for multiple-source adaptation. In *Neural Information Processing Systems (NIPS)*, 2018. 8

[69] C. J. Holder, T. P. Breckon, and X. Wei. Depth not needed-an evaluation of rgb-d feature encodings for off-road scene understanding by convolutional neural network. *arXiv preprint arXiv:1801.01235*, 2018. 9

[70] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Let there be Color!: Joint End-to-end Learning of Global and Local Image Priors for Automatic Image Colorization with Simultaneous Classification. *ACM Transactions on Graphics (Proc. of SIGGRAPH 2016)*, 35(4), 2016. 24

[71] I.-H. Jhuo, D. Liu, D. T. Lee, and S.-F. Chang. Robust visual domain adaptation with low-rank reconstruction. In *Computer Vision and Pattern Recognition (CVPR)*, 2012. 8, 53

[72] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093, 2014. URL http://arxiv.org/abs/1408.5093. 19

[73] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, 2016. 9

[74] F. P. Jorge Sánchez and T. Mensink. Improved fisher vector for large scale image classification, 2010. URL http://www.image-net.org/challenges/LSVRC/2010/ILSVRC2010_XRCE.pdf. 1

[75] A. Khoreva, R. Benenson, J. Hosang, M. Hein, and B. Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*, 2017. 10

[76] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. In *International Conference on Machine Learning, ICML*, pages 1857–1865, 2017. 8, 43, 49, 51

[77] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 46, 58

[78] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *NIPS*. 2012. 13, 27

[79] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 13, 18, 23

[80] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011. 9, 11, 13, 14, 16, 18, 26, 28, 30, 32

[81] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. In *ECCV*, 2016. 24

[82] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *NIPS*. 1990. 23

[83] 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. 10, 57

[84] C. Li and M. Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2479–2486, 2016. 9

[85] C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision*, pages 702–716. Springer, 2016. 9

[86] C. Li, A. Reiter, and G. D. Hager. Beyond spatial pooling: Fine-grained representation learning in multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4913–4922, 2015. 11, 13, 14, 16, 18, 20, 23, 26, 28, 30, 32

[87] D. Li, Y. Yang, Y. Z. Song, and T. M. Hospedales. Deeper, broader and artier domain generalization. In *International Conference on Computer Vision (ICCV)*, 2017. 8

[88] D. Li, Y. Yang, Y. Song, and T. M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Conference of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2018. 8, 53, 56, 58, 59

[89] H. Li, S. Jialin Pan, S. Wang, and A. C. Kot. Domain generalization with adversarial feature learning. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 9, 57, 59

[90] W. Li, Z. Cao, Y. Xiao, and Z. Fang. Hybrid rgb-d object recognition using convolutional neural network and fisher vector. In *Chinese Automation Congress (CAC), 2015*, pages 506–511. IEEE, 2015. 32

[91] Y. Li, Z. Ye, and J. Rehg. Delving into egocentric actions. In *Proc. CVPR*, 2015. 9

[92] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 64

[93] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Neural Information Processing Systems (NIPS)*. 2016. 8, 47

[94] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016. 64

[95] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Neural Information Processing Systems (NIPS)*, 2017. 8, 47, 53, 59, 64

[96] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. 2016. 10

[97] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 10

[98] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *International Conference on Machine Learning (ICML)*, 2017. 7, 53

[99] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 13

[100] M. Ma, H. Fan, and K. Kitani. Going deeper into first-person activity recognition. In *Proc. CVPR*, 2016. 10

[101] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 21, 48

[102] M. Mancini, S. R. Bulo, B. Caputo, and E. Ricci. Robust place categorization with deep domain generalization. *IEEE Robotics and Automation Letters*, 2018. 8

[103] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *Neural Information Processing Systems (NIPS)*, 2009. 8

[104] X. Mao, Q. Li, H. Xie, R. Y. Lau, and Z. Wang. Multi-class generative adversarial networks with the l2 loss function. *arXiv preprint arXiv:1611.04076*, 2016. 45

[105] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 14

[106] T. McCandless and K. Grauman. Object-centric spatio-temporal pyramids for egocentric activity recognition. In *Proc. BMVC*, 2013. 33

[107] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*. 8

[108] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 1

[109] M. Moghimi, P. Azagra, L. Montesano, A. C. Murillo, and S. Belongie. Experiments on an rgb-d wearable vision system for egocentric activity recognition. In *Proc. CVPR Workshops*, 2014. 10, 33

[110] M. Moghimi, P. Azagra, L. Montesano, A. C. Murillo, and S. Belongie. Experiments on an rgb-d wearable vision system for egocentric activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 597–603, 2014. 11, 33, 36, 40

[111] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin. Evaluation of traffic sign recognition methods trained on synthetically generated data. In *International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 2013. 11

[112] A. Mordvintsev, C. Olah, and M. Tyka. Inceptionism: Going deeper into neural networks. 2015. 9

[113] E. Morvant. Domain adaptation of weighted majority votes via perturbed variation-based self-labeling. *Pattern Recognition Letters*, 51:37–43, 2015. 44

[114] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Unified deep supervised domain adaptation and generalization. In *International Conference on Computer Vision (ICCV)*, 2017. 9, 57, 59

[115] Y. Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983. 27

[116] 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*, volume 2011, page 5, 2011. 11, 57

[117] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 70, 2017. 48

[118] F. Orabona, L. Jie, and B. Caputo. Online-batch strongly convex multi kernel learning. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 787–794. IEEE, 2010. 18, 19

[119] J. Papon and M. Schoeler. Semantic pose using deep networks trained on synthetic rgb-d. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 774–782, 2015. 14

[120] D. Pathak, P. Krahenbuhl, and T. Darrell. Constrained convolutional neural networks for weakly supervised segmentation. In *ICCV*, 2015. 10

[121] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 65

[122] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019. 1

[123] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *European Conference on Computer Vision (ECCV)*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing, 2016. 10, 12, 64

[124] G. Rogez, J. S. Supancic, and D. Ramanan. Understanding everyday hands in action from rgb-d images. In *Proceedings of the IEEE international conference on computer vision*, pages 3889–3897, 2015. 11, 33, 36, 40

[125] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 10, 12, 62, 64

[126] A. Rozantsev, M. Salzmann, and P. Fua. Beyond sharing weights for deep domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, 2018. 7, 53

[127] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 6, 13, 14

[128] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 8, 36, 53, 59, 62

[129] P. Russo, T. Tommasi, and B. Caputo. Towards multi-source adaptive semantic segmentation. In *International Conference on Image Analysis and Processing*, pages 292–301. Springer, 2019. 3

[130] M. Ryoo, B. Rothrock, and L. Matthies. Pooled motion features for first-person videos. In *Proc. CVPR*, 2015. 10

[131] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision, (ECCV)*, 2010. 50, 51, 53

[132] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *International Conference on Machine Learning, (ICML)*, 2017. 7, 44, 47, 59

[133] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Adversarial dropout regularization. In *ICLR*, 2018. 10

[134] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Neural Information Processing Systems (NIPS)*, 2016. 8

[135] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 7, 47

[136] M. Schwarz, H. Schulz, and S. Behnke. Rgb-d object recognition and pose estimation based on pre-trained convolutional neural network features. In *(ICRA)*, 2015. 9, 24, 29

[137] M. Schwarz, H. Schulz, and S. Behnke. Rgb-d object recognition and pose estimation based on pre-trained convolutional neural network features. In *ICRA*, 2015. 9, 13, 19, 20, 23

[138] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2110–2118. 2016. 7, 44, 47

[139] S. Shankar, V. Piratla, S. Chakrabarti, S. Chaudhuri, P. Jyothi, and S. Sarawagi. Generalizing across domains via cross-gradient training. In *International Conference on Learning Representations (ICLR)*, 2018. 9, 57, 59

[140] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014. 13

[141] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 8

[142] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 37, 40

[143] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 18, 26, 27

[144] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel. Bigbird: A large-scale 3d database of object instances. In *ICRA*, 2014. 11, 26, 28, 30, 32

[145] S. Singh, C. Arora, and C. Jawahar. First person action recognition using deep learned descriptors. In *Proc. CVPR*, 2016. 10

[146] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng. Convolutional-recursive deep learning for 3d object classification. In *Advances in neural information processing systems*, pages 656–664, 2012. 9, 13, 20

[147] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. *The German traffic sign recognition benchmark: a multi-class classification competition.* IEEE, 2011. 11

[148] S. Sudhakaran and O. Lanz. Attention is all we need: Nailing down object-centric attention for egocentric activity recognition. In *British Machine Vision Conference*, 2018. 10, 33, 35, 37, 39, 40, 41

[149] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *Conference of the Association for the Advancement of Artificial Intelligence (AAAI)*, 2016. 7, 47

[150] L. Sun, C. Zhao, and R. Stolkin. Weakly-supervised dcnn for rgb-d object recognition in real-world applications which lack large-scale annotated training data. *arXiv preprint arXiv:1703.06370*, 2017. 9

[151] Q. Sun, R. Chattopadhyay, S. Panchanathan, and J. Ye. A two-stage weighting framework for multi-source domain adaptation. In *Advances in Neural Information Processing Systems (NIPS)*. 2011. 8

[152] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 23, 27

[153] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 1, 18

[154] Y. Taigman, A. Polyak, and L. Wolf. Unsupervised cross-domain image generation. In *International Conference on Learning Representations (ICLR)*, 2017. 47, 48

[155] Y. Tang, Y. Tian, J. Lu, J. Feng, and J. Zhou. Action recognition in rgb-d egocentric videos. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3410–3414. IEEE, 2017. 10, 33, 37, 40

[156] Y. Tang, Z. Wang, J. Lu, J. Feng, and J. Zhou. Multi-stream deep neural networks for rgb-d egocentric action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018. 33, 37, 38, 40

[157] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017. 33, 36

[158] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. In *COURSERA: Neural Networks for Machine Learning*, 2012. 59

[159] T. Tommasi, F. Orabona, and B. Caputo. Discriminative cue integration for medical image annotation. *Pattern Recognition Letters*, 29(15):1996–2002, 2008. 31

[160] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 10, 62, 64, 65

[161] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *International Conference in Computer Vision (ICCV)*, 2015. 7, 44, 47

[162] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 7, 42, 45, 47

[163] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, volume 1, page 4, 2016. 9

[164] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6924–6932, 2017. 9

[165] V. Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013. 5

[166] R. Volpi, H. Namkoong, O. Sener, J. C. Duchi, V. Murino, and S. Savarese. Generalizing to unseen domains via adversarial data augmentation. In *Neural Information Processing Systems (NIPS)*, 2018. 9

[167] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016. 33, 40

[168] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *Trans. Img. Proc.*, 13(4):600–612, Apr. 2004. 48

[169] Z. Wang, R. Lin, J. Lu, J. Feng, et al. Correlated and individual multi-modal deep learning for rgb-d object recognition. *arXiv preprint arXiv:1604.01655*, 2016. 25, 32

[170] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 14, 23

[171] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015. 10, 35

[172] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 8, 53, 56, 58, 59, 64

[173] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014. 9

[174] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 10

[175] F. Yu, D. Wang, E. Shelhamer, and T. Darrell. Deep layer aggregation. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 54

[176] H. F. Zaki, F. Shafait, and A. Mian. Convolutional hypercube pyramid for accurate rgb-d object category and instance recognition. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1685–1692. IEEE, 2016. 9, 18, 19, 20, 24, 29

[177] N. Zhang, R. Farrell, and T. Darrell. Pose pooling kernels for sub-category recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3665–3672. IEEE, 2012. 20

[178] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. *ECCV*, 2016. 26

[179] Y. Zhang, P. David, and B. Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2020–2030, 2017. 10, 62

[180] B. Zhao, J. Feng, X. Wu, and S. Yan. A survey on deep learning-based fine-grained object classification and semantic segmentation. *International Journal of Automation and Computing*, 14(2):119–135, Apr 2017. 62

[181] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 10

[182] H. Zhao, S. Zhang, G. Wu, J. ao P. Costeira, J. M. F. Moura, and G. J. Gordon. Multiple source domain adaptation with adversarial learning. In *Workshop of the International Conference on Learning Representations (ICLR-W)*, 2018. 8, 53, 55, 56, 57, 59, 62

[183] L. Zheng, Y. Zhao, S. Wang, J. Wang, and Q. Tian. Good practice in cnn feature transfer. *arXiv preprint arXiv:1604.00133*, 2016. 18, 20

[184] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014. 13

[185] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 35

[186] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 8, 43, 49, 51

[187] X. Zhu, H. Zhou, C. Yang, J. Shi, and D. Lin. Penalizing top performers: Conservative loss for semantic segmentation adaptation. In *ECCV*, 2018. 10

[188] Y. Zou, Z. Yu, B. Vijaya Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 289–305, 2018. 64