DISSERTATION

Object Classification for Robot Vision through RGB-D Recognition and Domain Adaptation

conducted in partial fulfillment of the requirements for the degree of a

Doktor der technischen Wissenschaften (Dr. techn.)

supervised by

Ao.Univ.-Prof. Dipl.-Ing. Dr. techn. M. Vincze
E376 Automation and Control Institute

submitted at the

TU Wien
Faculty of Electrical Engineering and Information Technology

by

Mohammad Reza Loghmani, MSc
DOB 08.06.1992
Matr. Nr.: 01652646

Vienna, June 2020 Mohammad Reza Loghmani

Acknowledgment

Throughout the development of this thesis I have been priviledged to meet and work with many inspiring people and without whom this work would not have been possible.

First and foremost, I would like to thank my supervisor Prof. Markus Vincze and my mentor Dr. Timothy Patten for their continuous support. They have always granted me the freedom to pursue my own ideas, but also provided valuable guidance and new impulses when needed. Furthermore, I would like to thank Prof. Barbara Caputo and Prof. Tatiana Tommasi for their support and mentorship. I am also indepted to the general public, who has funded this work through the European Union´s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 676157, project ACROSSING.

A big thank you goes to all my exceptional colleagues at the Vision for Robotics lab, who make the group the stimulating and fun working environment it is. Thanks to all of you for the insightful scientific discussions, the absurd pseudo-legal conversations over lunch, and the bittersweet nerfgun shootouts. Especially, I would like to thank my friends Jean-Baptiste, Simon, Matthias and Stefan for the moments shared inside and outside of the office. Last but not least, I would like to express my sincere gratitude to my parents Farahnaz and Shahram and my brother Hamid, who enabled and always encouraged me to follow my own path. To my friends Teresa, Britta, David, Aline, Miguel, Jürgen and Mirosław for making Vienna my home. To my friends Tomaso, Francesco, Anastasios, Giorgio, Giulia, Stefano, Maria, Fabio, Fabrizio, Riccardo, Luca, Daniele and Annapaola for showing me that friendship can reach across any distance. Finally, a warm thank to my partner Caroline for sharing much-needed laughter with me and supporting me through thick and thin.

Abstract

Object recognition, or object classification, is an essential skill for robot visual perception systems since it constitutes the foundation for higher-level tasks like object detection, pose estimation and manipulation. Nonetheless, recognizing objects in unconstrained environments remains arduous with robots facing challenges such as intra-class variation, occlusion, clutter, viewpoint variation, and changes in light and scale.

Deep *convolutional neural networks* (CNNs) have revolutionized object classification and computer vision as a whole. However, standard computer vision benchmarks often fail to address all the challenges of robot vision. This results in the development of classification models that perform poorly when deployed on a robot in-the-wild.

In this thesis, we perform a systematic study of object recognition for robot vision and propose algorithmic innovations that tackle different aspects of this multifaceted problem. We first collect a robot-centric dataset called *autonomous robot indoor dataset* and test the performance of well-known CNN architectures on it. This evaluation indicates two main lines of research for more reliable and robust object recognition: (i) the integration of geometric information as depth data with the standard RGB data, and (ii) the use of domain adaptation to bridge the gap between the training (source) data and the real (target) data the robot encounters. To combine RGB and depth data, we propose *recurrent convolutional fusion*: a novel architecture that extracts features from different layers of a two-stream CNN and combines them using a recurrent neural network. To perform domain adaptation on RGB-D data, we propose a multi-task learning method that, in addition to the standard recognition task, learns to predict the relative rotation between the RGB and depth image of a sample. We go one step further and consider the more realistic problem of *open set domain adaptation* (OSDA), that requires to adapt two domains when the target contains not only the known classes of the source, but also unknown classes. We propose *positive-unlabeled reconstruction encoding*, an algorithm that uses the theoretical framework of *positive-unlabeled learning* and a novel loss based on sample reconstruction to recognize the unknown classes of the target. We further improve upon this algorithm by proposing *rotation-based open set* that performs both the adaptation and the known/unknown recognition using the self-supervised task of relative rotation.

Extensive quantitative and qualitative experiments on standard benchmarks and newly collected datasets empirically validate our algorithmic contributions. These methods push the state of the art in RGB-D object recognition and domain adaptation and brings us closer to build robotic systems with human-like recognition performance.

Kurzfassung

Die Objekterkennung oder Objektklassifizierung ist eine wesentliche Fähigkeit der visuellen Wahrnehmung von Robotern, da sie die Grundlage für übergeordnete Aufgaben wie die Erkennung von Objekten, die Schätzung von Position und Orientierung (Pose) sowie die Objektmanipulation bildet. Das Erkennen von Objekten in offenen Umgebungen gestaltet sich nach wie vor jedoch als schwierig, da Roboter mit Herausforderungen wie klasseninterner Variation, Verdeckung, Unordnung sowie wechselnden Blickwinkeln, Lichtverhältnissen und Maßstäben konfrontiert sind.

Tiefe *Convolutional Neural Networks* (CNNs) haben die Objektklassifizierung und Computer Vision als Ganzes revolutioniert. Standard-Benchmarks in Computer Vision thematisieren jedoch häufig nicht alle Herausforderungen, die auch für Sehen am Roboter (Robot Vision) gelten. Dies führt zur Entwicklung von Klassifikationsmodellen, die bei der Anwendung auf Robotern in natürlicher Umgebung schlecht abschneiden.

In dieser Arbeit führen wir eine systematische Untersuchung zur Objekterkennung für Robot-Vision-Systeme durch und schlagen algorithmische Innovationen zu verschiedenen Aspekten dieser facettenreichen Herausforderung vor. Wir sammeln zunächst einen roboterzentrierten Datensatz, das sogenannte *Autonomous Robot Indoor Dataset*, und testen damit die Leistung bekannter CNN-Architekturen. Diese Bewertung zeigt zwei grundlegende Forschungswege für eine zuverlässigere und stabilere Objekterkennung auf: (i) die Integration geometrischer Informationen als Tiefendaten in die Standard-RGB-Daten und (ii) die Verwendung der Domänenanpassung zur Überbrückung der Lücke zwischen den (Quell-)Daten aus dem Training und den realen (Ziel-)Daten, auf die der Roboter trifft. Um RGB- und Tiefendaten zu kombinieren, schlagen wir *Recurrent Convolutional Fusion*, eine neuartige Architektur, vor. Sie extrahiert Merkmale aus verschiedenen Schichten eines Zwei-Pfad-CNN und kombiniert sie unter Verwendung eines wiederkehrenden (recurrent) neuronalen Netzwerks. Um eine Domänenanpassung an RGB-D-Daten durchzuführen, schlagen wir eine Multitasking-Lernmethode vor, die zusätzlich zu standardmäßigen Erkennungsaufgaben lernt, die relative Rotation zwischen dem RGB Bild und dem entsprechenden (respektiven) Tiefenbild vorherzusagen. Wir gehen noch einen Schritt weiter und betrachten das realistischere Problem von *Open Set Domain Adaptation* (OSDA), bei dem zwei Domänen angepasst werden müssen, wenn das Ziel nicht nur die bekannten Klassen der Quelle, sondern auch unbekannte Objektklassen enthält. Wir schlagen *Positive-Unlabeled Reconstruction Encoding* vor, einen Algorithmus, der den theoretischen Rahmen von *Positive-Unlabeled Learning* und eine neuartige Kostenfunktion basierend auf der Probenrekonstruktion verwendet, um die unbekannten Klassen des Ziels zu erkennen. Wir verbessern diesen Algorithmus

weiter, indem wir ein *Rotation-Based Open Set* vorschlagen, das sowohl die Anpassung als auch die bekannte/unbekannte Erkennung unter Verwendung der selbstüberwachten relativen Rotation durchführt.

Umfangreiche quantitative und qualitative Experimente zu Standard-Benchmarks und zu neu gesammelten Datensätzen bestätigen die algorithmischen Beiträge empirisch. Die hier vorgestellten Methoden treiben den Stand der Technik bei der RGB-D-Objekterkennung und Domänenanpassung voran und bringen uns dem Aufbau von Robotersystemen mit menschenähnlicher visueller Erkennungsleistung näher.

Contents

List of Figures

List of Tables

List of Abbreviations

AE Autoencoder.

AoD Attract or Distract.

ARID Autonomous Robot Indoor Dataset.

CNN Convolutional Neural Network.

CSDA Closed Set Domain Adaptation.

DA Domain Adaptation.

GRU Gated Recurrent Unit.

LSTM Long-Short Term Memory.

nnPU Non-Negative Positive-Unlabeled.

OSBP Open Set Back-Propagation.

OSDA Open Set Domain Adaptation.

PN Learning Positive-Negative Learning.

PU Learning Positive-Unlabeled Learning.

PURE Positive-Unlabeled Reconstruction Encoding.

RCFusion Recurrent Convolutional Fusion.

RNN Recurrent Neural Network.

ROD RGB-D Object Dataset.

SCAR Selected Completely At Random.

SR Service Robot.

STA Separate To Adapt.

UAN Universal Adaptation Network.

WOD Web Object Dataset.

Chapter 1

Introduction

"Not only did Dr. P fail to see faces, but he saw faces when there were no faces to see. In the street he might pat the heads of water hydrants and parking meters, taking these to be the heads of children; he would amiably address carved knobs on the furniture and be astounded when they did not reply. Such incidents multiplied, causing embarrassment, perplexity and fear."

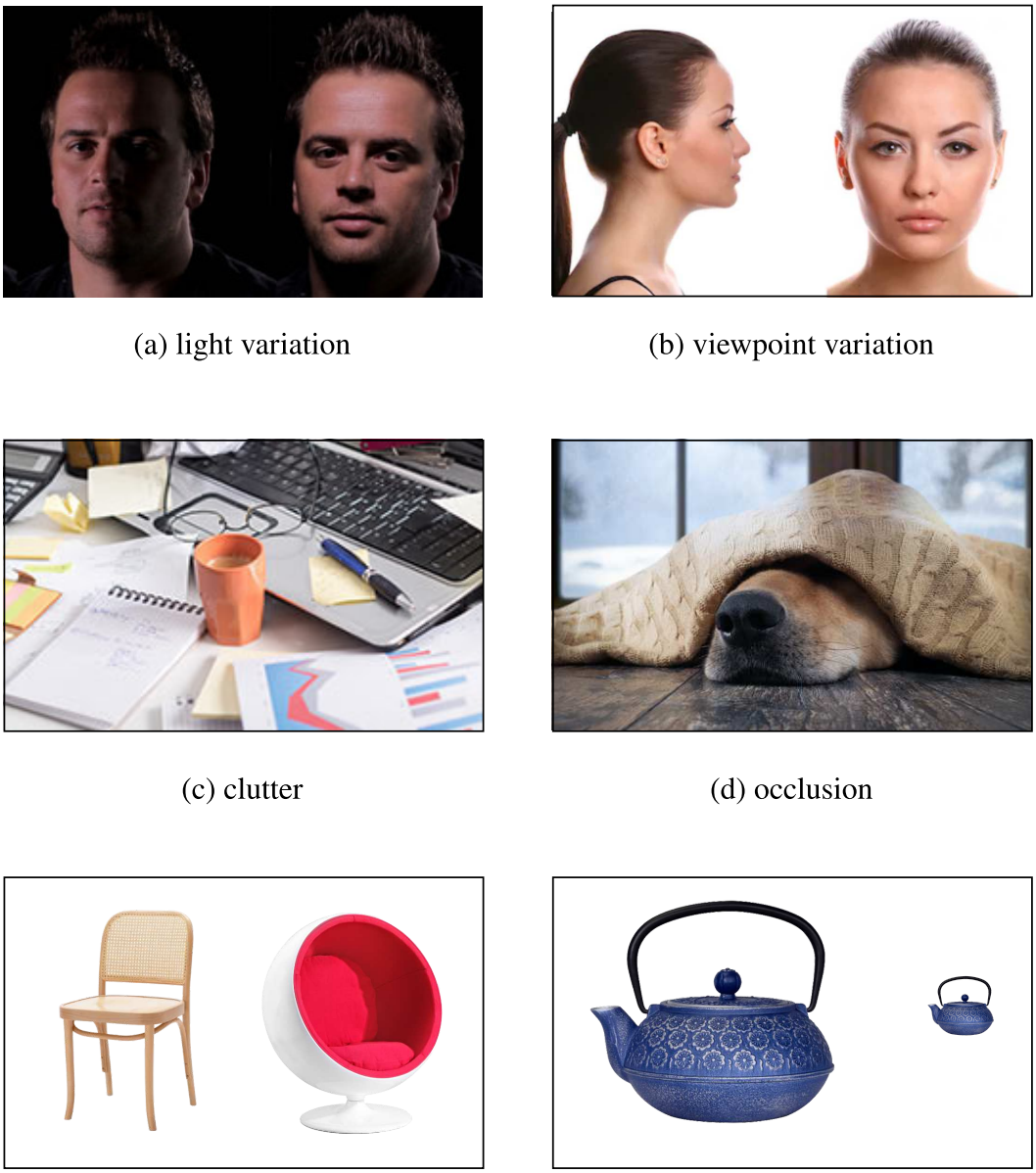
From "The man who mistook his wife for a hat"
by Dr. Oliver Sacks

Let us start with a thought experiment. Imagine to suddenly lose the ability to correctly recognize objects, while all the other cognitive and perceptual abilities remain unaltered. Would you be able to prepare your favourite dish, go to work or brush your teeth? Even the most mundane task requires the ability to correctly recognize objects and the lack of such ability would create the same "embarrassment, perplexity and fear" described by Dr Oliver Sacks.

Humans start to develop the ability to recognize objects from birth. Within the first four months of age, infants showcase the ability to recognize three-dimensional shapes and a rudimental understanding of their parts [1]. Despite this early development, recognition skills do not reach adult-like performance until adolescence. In particular, what appears to develop slowly are high-level abstraction abilities such as differentiating exemplars within a particular object category (for example, two different cups) or recognizing the same exemplar from multiple viewpoints (for example, a teapot seen from above looks quite different from the canonical side view) [2]. Figure 1.1 illustrates these and other challenges of object recognition, such as handling occlusion, clutter, and change in light and scale. How the human brain eventually develops the complex set of visual skills to rapidly and accurately recognize objects still remains largely unknown.

In recent years, robotic systems transitioned from the isolated industrial environments to the controlled chaos that are our streets and houses. Similarly to humans, robots need the ability to recognize objects to understand and operate in these environments. For example, a self-driving car needs to be constantly aware of its ever-changing surroundings, distinguishing between drivable roads, cars, sidewalks, buildings and people. Or a *service robot* (SR) at home that helps to tidy up the rooms needs to recognize the objects that are misplaced in order to bring them back to where they belong. However, robots are required to possess human-like recognition performance at

deployment, without the luxury of spending years in refining these skills. In addition, the cameras that enable robot perception do not reach the level of sophistication of human eyes. It should, therefore, come as no surprise that enabling robots to recognize objects is still an open problem in the research community.



(a) light variation (b) viewpoint variation

(c) clutter (d) occlusion

(e) intra-class variation (f) scale variation

Figure 1.1: Examples of challenges in object recognition.

1.1 Problem statement

In computer vision, object recognition is defined as a classification problem that consists in predicting one out of K possible semantic labels given an image of an object. The term "object" is used loosely to refer to any entity that has a well-defined semantic meaning for humans. Examples of object recognition can be recognizing hand-written digits, distinguishing between images of cats and dogs, or discriminating several office items. Since the groundbreaking work of Krizhevsky *et al.* [3], deep *convolutional neural networks* (CNNs) have become the standard models to perform object recognition. However, training a CNN to learn discriminative features requires a large amount of data with task-specific annotation.

Exactly the generation of large-scale labeled datasets is the number one enabler of the deep learning revolution. The petabytes worth of images on the Internet, available through Web search engines, have allowed the generation of datasets with millions of samples, such as the iconic ImageNet [4]. However, CNNs trained with such datasets learn a representation of the visual world that is biased by (i) the algorithm of the Web search engine that provides the images, and (ii) the human photographers that have taken the pictures in the first place. It is then natural to wonder whether the features learned from Web-based datasets can generalize well to data observed by a robot, despite these biases.

As discussed in the previous section, humans heavily rely on shape information to recognize objects. Such geometric cues are ambiguous in standard RGB images due to the intrinsic information loss occurring during the image formation process when projecting the three-dimensional world into a two-dimensional image plane. A potential solution has emerged from the game industry in the form of an RGB-D (Kinect-style) camera. These sensors use range imaging technologies to capture the information about the camera-scene distance as a depth image. However, most large-scale datasets only provide RGB data and this has slowed down the development of CNN-based solution for RGB-D object recognition.

Collecting and annotating a large amount of RGB-D data to train CNNs can be very costly. Deploying a robot to collect a large amount of data for each new application is simply deleterious. An alternative is to use simulation: generate a large synthetic training set with computer graphics software by rendering 3D object models. However, CNNs trained with the synthetically generated (source) data can perform poorly on real (target) data due to the differences between synthetic and real images. Unsupervised *domain adaptation* (DA) is a field of research that accounts for the difference between source and target data by considering them as drawn from two different marginal distributions. DA approaches provide predictions on a set of target samples using only annotated source samples, with the unlabeled target samples available transductively. However, existing DA strategies are sub-optimal for tackling RGB-D data since they implicitly assume that the data come from a single modality. This assumption effectively ignores the inter-modal relations between RGB and depth, discarding important information to reduce the domain gap.

Most existing DA approaches have another important limitation: they focus on the so-called *closed set scenario* where source and target cover the same known set of categories.

These approaches fall under the umbrella of *closed set domain adaptation* (CSDA). However, in real-world robotic applications, the robot encounters objects that do not necessarily belong to the set of known categories. The *open set scenario* aligns with this realistic setting by considering a situation where the target data contain, in addition to the known source categories, additional unknown categories. *Open set domain adaptation* (OSDA) therefore extends DA to either assign each target samples to one of the known classes or reject it as unknown. This field of study is still in its infancy and requires more research before being deployed in real-world robotic applications.

1.2 Contributions and outline

In this thesis, we aim at advancing the field of object recognition for real-world robotic applications. To this end, we first identify the specific challenges faced in robot vision that are currently unsolved. Guided by our findings, we propose several algorithmic innovations in RGB-D object recognition and DA that follow the high-level pipeline in figure 1.2. We describe the details of our contribution in the following.

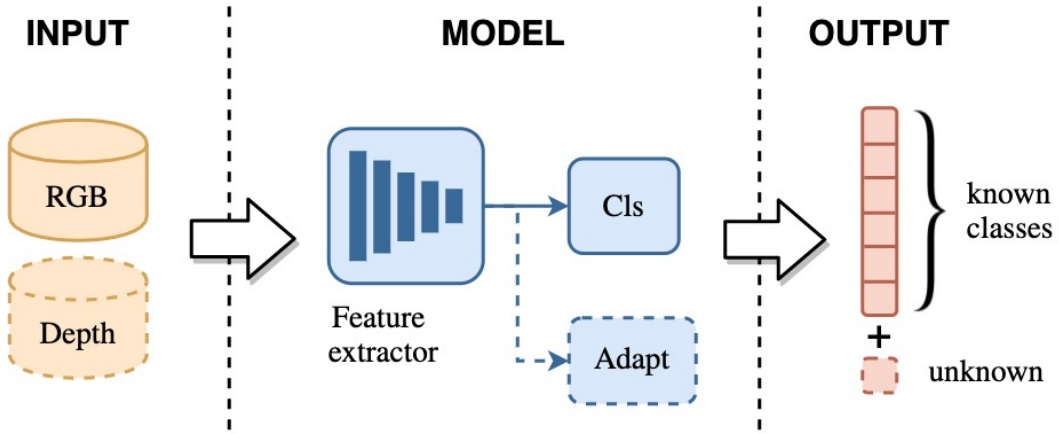


Figure 1.2: Generalized diagram of the proposed recognition pipelines. The *input* is RGB images with the potential addition of depth images (see chapters 3 and 4). The *model* is a deep convolutional neural network that can be roughly divided into three modules: feature extractor, classifier (Cls), and adaptation module (Adapt). The feature extractor generates the discriminative features from the input data that are then used by the classifier to generate the class prediction. The adaptation module is added when there is a domain shift, like in chapters 4, 5, and 6. The *output* is the prediction of one out of K known classes, with the potential addition of the unknown class, like in chapters 5 and 6.

1.2.1 Recognizing objects in-the-wild: where do we stand?

The progress in designing effective object recognizers is slowed down by the lack of a testbed able to accurately represent the world perceived by the robot in-the-wild. In

order to fill this gap, we introduce a large-scale, multi-view object dataset collected with an RGB-D camera mounted on a mobile robot, called *autonomous robot indoor dataset* (ARID). The dataset embeds the challenges faced by a robot in a real-life application and provides a useful tool for validating object recognition algorithms. We evaluate the performance of a collection of well-established deep convolutional networks on the new dataset and analyze the transferability of deep representations from Web images to robotic data. Despite the promising results obtained with such representations, the experiments demonstrate that object classification with real-life robotic data is far from being solved. With a comparative study on subsets of ARID, we identify small images and occlusions, as well as an overall domain gap between the Web and the robotic domain, to be the main factors contributing to the underwhelming classification accuracy.
The aforementioned contributions are discussed in details in chapter 2 and have been published in the scientific paper [Loghmani, ICRA 2018].

1.2.2 Recurrent convolutional fusion for RGB-D object recognition

The addition of depth data has the potential to tackle the above-mentioned open challenges, i.e. to disambiguate scale information and deal with occlusions. However, the robot vision community still lacks an effective method to synergically use both RGB and depth modalities to improve object recognition. In order to take a step in this direction, we introduce a novel end-to-end CNN architecture for RGB-D object recognition called *recurrent convolutional fusion* (RCFusion). Our method generates compact and highly discriminative multi-modal features by combining RGB and depth information representing different levels of abstraction. Extensive experiments on two RGB-D benchmark datasets, *RGB-D object dataset* (ROD) and *JHUIT-50*, show that RCFusion significantly outperforms state-of-the-art approaches in both the object categorization and instance recognition tasks. In addition, experiments on the more challenging *object clutter indoor dataset* (OCID) confirm the validity of our method in the presence of clutter and occlusion. Unfortunately, preliminary experiments on ARID show that current RGB-D cameras lack the technological maturity to reliably capture depth information in an unconstrained setup. In summary, the experiments with RCFusion show that complementing the RGB data with reliable depth information can be greatly beneficial for handling occlusions and object recognition in general.
The aforementioned contributions are discussed in details in chapter 3 and have been published in the scientific paper [Loghmani, RA-L 2019].

1.2.3 Unsupervised domain adaptation through inter-modal rotation for RGB-D object recognition

Collecting and annotating application-specific RGB-D data to exploit the benefits of depth information is very costly. DA can be used to compensate for the lack of an adequate RGB-D dataset by taking advantage of automatically generated synthetic data, that come with "free" annotation, to make effective predictions on real data.

However, existing DA methods are not designed to cope with the multi-modal nature of RGB-D data, which are widely used in robotic vision. We propose a novel RGB-D DA method that reduces the synthetic-to-real domain shift by exploiting the inter-modal relation between the RGB and depth image. Our method consists of training a CNN to solve, in addition to the main recognition task, the pretext task of predicting the relative rotation between the RGB and depth image. Due to the lack of adequate benchmarks for synthetic-to-real RGB-D recognition, we define two benchmark datasets for object categorization and instance recognition. With extensive experiments, we show the benefits of leveraging the inter-modal relations for RGB-D DA.
The aforementioned contributions are discussed in details in chapter 4 and in the submitted scientific paper [Loghmani, RA-L 2020].

1.2.4 Positive-unlabeled learning for open set domain adaptation

In real-world applications, robots encounter objects that do not necessarily belong to the set of known categories. OSDA focuses on bridging the domain gap between a labeled source domain and an unlabeled target domain, while also rejecting target classes that are not present in the source as unknown. The challenges of this task are closely related to those of *positive-unlabelled* (PU) learning where it is essential to discriminate between positive (known) and negative (unknown) class samples in the unlabeled target data. With this newly discovered connection, we leverage the theoretical framework of PU learning for OSDA and, at the same time, we extend PU learning to tackle uneven data distributions. Our method called *positive-unlabeled reconstruction encoding* (PURE) combines domain adversarial learning with a new non-negative risk estimator for PU learning based on self-supervised sample reconstruction. To evaluate the method, we define a new open set metric that balances better the contribution of recognizing the known classes and rejecting the unknown samples compared to previous metrics. With experiments on standard benchmark dataset for digit recognition (MNIST, MNIST-M, USPS, SVHN) and object classification (CIFAR, STL, office-31), we validate our risk estimator and demonstrate that our approach allows reducing the domain gap without suffering from negative transfer. However, the performance of our method can deteriorate when dealing with the high variability of real-world images, such as the images in office-31, due to the difficulty of self-supervised sample reconstruction.
The aforementioned contributions are discussed in details in chapter 5 and in the submitted scientific paper [Loghmani, PRL 2020].

1.2.5 On the effectiveness of image rotation for open set domain adaptation

We propose a novel OSDA method that replaces sample reconstruction, used in PURE, with a simpler yet familiar self-supervised task: image rotation. To avoid negative transfer, OSDA can be tackled by first separating the known and unknown target samples and then aligning known target samples with the source data. We show that training a deep network to predict image rotation can be used both for known/unknown

separation and for domain alignment. In order to show that our new approach can deal with real-world images, we perform extensive experiments on the standard OSDA benchmarks *office-31* and *office-home*. A comparative evaluation with existing OSDA methods on these two datasets shows that our method outperforms the state of the art using the power of image rotation.
The aforementioned contributions are discussed in details in chapter 6 and in the submitted scientific paper [Bucci & Loghmani, ECCV 2020].

1.3 Related work

The contributions of this thesis stand on the literature of several sub-fields of computer vision including RGB-D object recognition, DA, self-supervised learning, open set recognition, and anomaly detection.

1.3.1 RGB-D object recognition

The diffusion of RGB-D cameras fueled an increasing effort in designing visual algorithms able to exploit the additional depth information provided by these sensors. Classical approaches for RGB-D object recognition [5], [6] used a combination of different hand-crafted feature descriptors, such as SIFT, textons, and depth edges, on the two modalities (RGB and depth) to perform object matching. More recently, several methods have exploited shallow learning techniques to generate features from RGB-D data in an unsupervised learning framework [7]–[9]. Since the ground-breaking work of Krizhevsky *et al.* [3], data-hungry CNNs have been the go-to solution for feature extraction. While large-scale datasets of RGB images, such as ImageNet [4], allowed the generation of powerful CNN-based models for RGB feature extraction, the lack of a depth counterpart posed the problem of how to extract features from depth images. An effective and convenient strategy to circumvent the problem is to colorize the depth images to exploit CNNs pre-trained on RGB data. Several hand-crafted colorization approaches have been proposed to map the raw depth value of each pixel [10] or derived physical quantities, such as position and orientation [11] or local surface normals [12], to colors. Carlucci *et al.* [13] proposed instead a leaning-based approach to colorize the depth images by training a colorization network. Other methods use alternatives to RGB-trained CNNs for extracting features from depth data. Li *et al.* [14] generate the depth features using a modified version of HONV [15] encoded with fisher vector [16]. Carlucci *et al.* [17] generate artificial depth data using 3D CAD models to train a CNN that extracts features directly from raw depth images.

The aforementioned methods focus on effectively extracting features from the depth data and use trivial strategies to combine the two modalities for the final prediction. For example, Carlucci *et al.* [13] simply select the class with the highest activation among the RGB and depth predictions, while Eitel *et al.* [10] and Aakerberg *et al.* [18] use a fully connected layer to learn to fuse the predictions from the two modalities. Alternatively, a few works prioritize the development of an effective modality fusion. Wang *et al.* [19] alternate between maximizing the discriminative characteristics of each

modality and minimizing the inter-modality distance in feature space. Wang *et al.* [20] obtain the multi-modal feature by using a custom layer to separate the individual and correlated information of the extracted RGB and depth features. Both methods combine the two modalities by processing features extracted from one layer of the CNNs and rely on cumbersome multi-stage optimization processes.

Recent works from related areas, such as object detection and segmentation from color images, show the benefits of using features extracted from multiple layers of a CNN. Hariharan *et al.* [21], increase the resolution of higher-level features by combining information from lower layers at a pixel level for segmentation purposes. Bell *et al.* [22] perform object detection at different scales using features extracted from different layers of a pre-trained network. These methods mostly take advantage of the difference in receptive fields in various layers of the neural network and use a simple combination of pooling and linear transformations to process the extracted features.

1.3.2 Self-supervised visual tasks

Self-supervised learning is used to compensate for the lack of annotated data by training the network on a pretext task for which the supervision (or ground truth) can be defined from the data themselves. Several self-supervised tasks have been defined to tackle computer vision tasks. Some examples include predicting the location of a patch [23], solving a jigsaw puzzle [24], colorizing a gray-scale image [25], and inpainting a removed patch [26]. Arguably, one of the most effective self-supervised tasks consists of rotating the input images by multiples of 90° and training the network to predict the rotation of each image [27]. This pretext task has been successfully used in a variety of applications such as network pre-training [27], anomaly detection [28], and DA [29].

1.3.3 Anomaly detection

Also known as outlier or novelty detection, this field aims at learning a model from a set of *normal* samples to be able to detect out-of-distribution (*anomalous*) instances. The research literature in this area is wide with three main kinds of approaches. *Distribution-based* methods [30]–[33] focus on modeling the distribution of the available normal data so that the anomalous samples can be recognized as those with a low likelihood under the learned probability function. *Reconstruction-based* methods [34]–[40] focus on learning to reconstruct the normal samples from an embedding or a set of basis functions. Anomalous data are then recognized by having a larger reconstruction error with respect to normal samples. *Discriminative* methods [41]–[45] focus on training a classifier on the normal data and use its predictions to distinguish between normal and anomalous samples. Note that outlier detectors generally deal with binary known/unknown problem and do not further discriminate semantic categories within the known class.

1.3.4 Open set Recognition

Open set recognition can be considered the extension of anomaly detection to a multi-class classification problem. In fact, this field shares the basic anomaly detection

objective, but with the extra challenge of discriminating also among multiple known categories. Beside several *kernel* [46], *nearest neighbor* [47] and *sparse-representation* [48] based shallow models, Bendale and Boult [49] present a deep learning solution based on the openmax layer, designed to model and exploit the likelihood of known per-class failures through a weibull distribution. Ge *et al.* [50] propose a generative variant that exploits this layer also on synthesized unknown data. Yoshihasi *et al.* [51] improve over openmax by exploiting the combination of classification and reconstruction of input data.

1.3.5 Closed set domain adaptation

DA accounts for the difference between source and target data by considering them as drawn from two different marginal distributions. In its naive closed set formulation, the source and target domain contain the same semantic categories. The literature of CSDA can be divided into three groups based on the strategy used to reduce the domain shift. *Discrepancy-based* methods [52]–[54] define a metric to measure the distance between source and target data in feature space. The defined metric is then minimized during the training of the network to reduce the domain shift. *Adversarial* methods [55]–[57] aim at training a domain discriminator and a generator network in an adversarial fashion so that the generator converges to a solution that makes the source and target data indistinguishable for the domain discriminator. *Self-supervised* methods [58]–[60] train a network to solve an auxiliary self-supervised task on the target (and source) data, in addition to the main task, to learn robust cross-domain representations.

The methods discussed so far are based on the implicit assumption that the data come from a single modality. While no approach specifically aims at adapting multi-modal data when both source and target domain contain RGB-D information[1], there are a few related cases that are worth mentioning. Spinello and Arras [62], and Hoffman *et al.* [63] adapt RGB data to depth data by considering them as source and target domain, respectively. Li *et al.* [64] tackle the case where the source dataset is composed of RGB-D images, while the target dataset only contains RGB images. The focus is therefore on how to combine the RGB and depth data of the source domain rather than on the adaptation itself. Finally, Wang and Zhang [65] consider the case where both source and target data are RGB-D images, but ignore the multi-modal nature of the data and apply a standard domain adversarial method to reduce the domain shift.

1.3.6 Open set domain adaptation

OSDA is a more realistic version of CSDA, where the source and target distribution do not contain the same categories. The term "OSDA" was first introduced by Busto and Gall [66] that considered the setting where each domain contains, in addition to the shared categories, a set of private categories. The currently accepted definition

[1] To our knowledge, Qi *et al.* [61] also propose a method to tackle the problem of multi-modal domain adaptation. However, this work does not deal with RGB-D data, but rather focuses on the combination of video and audio and applying it to the RGB-D scenario would require non-trivial adjustments to their method.

of OSDA was introduced by Saito *et al.* [67] that considered the target as containing all the source categories plus an additional set of private categories that should be considered *unknown*. To date, only a handful of papers tackled this problem. *Open set back-propagation* (OSBP) [67] is an adversarial method that consists in training a classifier to obtain a large boundary between source and target samples whereas the feature generator is trained to make the target samples far from the boundary. *Separate to adapt* (STA) [68] is an approach based on two stages: first, a multi-binary classifier trained on the source is used to estimate the similarity of target samples to the source and then, target data with extreme high and low similarity are re-used to separate known and unknown classes while the features across domains are aligned through adversarial adaptation. *Attract or distract* (AoD) [69] starts with a mild alignment with a procedure similar to [67] and refines the decision by using metric learning to make each known class more concentrate and push unknown class away from the centroids of the known classes. *Universal adaptation network* (UAN)[2] [70] uses a pair of domain discriminators to both generate a sample-level transferability weight and to promote the adaptation in the automatically discovered common label set.

1.4 Scientific papers

The work of this thesis has led to several scientific papers that are published or currently under submission in peer reviewed conferences and journals.

[Merdivan, NeurIPS 2017] E. Merdivan, M. Loghmani, and M. Geist. **Reconstruct & Crush Network**, in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4548-4556, 2017.

[Loghmani, ICRA 2018] M. Loghmani, B. Caputo, and M. Vincze, **Recognizing Objects in-the-Wild: Where do we Stand?**, in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2170-2177, 2018.

[Loghmani, PerCom 2018] M. Loghmani, T. Patten, and M. Vincze, **Towards Socially Assistive Robots for Elderly: An End-to-end Object Search Framework**, in *Proc. of IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 890-895, 2018.

[Loghmani, RA-L 2019] M. Loghmani, M. Planamente, B. Caputo, and M. Vincze, **Recurrent Convolutional Fusion for RGB-D Object Recognition**, in *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 3, pp. 2878-2885, 2019.

[Liappas, SWC 2019] N. Liappas, J. Terius-Padron, E. Machado, M. Loghmani, R. Garcia-Betances, M. Vincze, I. Quero, and M. Cabrera-Umpierrez, **Best Practices on Personalization and Adaptive Interaction Techniques in**

[2]UAN is originally proposed for the universal domain adaptation setting that can be considered as a superset of OSDA, so it can also be this setting.

the Scope of Smart Homes and Active Assisted Living, in *Proc. of IEEE Smart World Congress Workshops (SWC Workshops)*, 2019.

[Loghmani, SWC 2019] M. Loghmani, C. Haider, Y. Chebotarev, C. Tsiourti, and M. Vincze, **Effects of Task-dependent Robot Errors on Trust in Human-Robot Interaction: A Pilot Study**, in *Proc. of IEEE Smart World Congress Workshops (SWC Workshops)*, 2019.

[Loghmani, RA-L 2020] M. Loghmani, L. Robbiano, M. Planamente, K. Park, B. Caputo, and M. Vincze, **Unsupervised Domain Adaptation through Inter-modal Rotation for RGB-D Object Recognition**, Under submission at *Robotics and Automation Letters (RA-L)*.

[Loghmani, PRL 2020] M. Loghmani, M. Vincze, and T. Tommasi, **Positive-Unlabeled Learning for Open Set Domain Adaptation**, Accepted for publication in *Pattern Recognition Letters (PRL)*.

[Bucci & Loghmani, ECCV 2020]S. Bucci[3], M. Loghmani[3], and T. Tommasi, **On the Effectiveness of Image Rotation for Open Set Domain Adaptation**, Under submission at *European Conference on Computer Vision (ECCV)*.

[3]equal contributions

Chapter 2

Recognizing objects in-the-wild: where do we stand?

The advent of deep learning has had a huge impact on the object recognition task after decades of plateaued results. Arguably the primary driving force of the deep learning revolution is the availability of large scale datasets that can be used as standard references and benchmarks from the computer vision community. The majority of these datasets, such as the popular ImageNet [4], Pascal VOC [71], and Caltech-256 [72], are composed of images collected through Web search engines. However, the representation of the visual world provided by these datasets implies a bias from the observer (a human photographer) and the Web search engines [73] that are incoherent with the representation perceived by, for example, a service robot (SR). It is then legitimate to ask whether the features learned from Web-based datasets can generalize well to robotic data, despite the aforementioned bias. In the past years, the RGB-D Object Dataset (ROD) [6] has become "de facto" standard in the robotic community for the object classification task [74] [9] [17]. Despite its well-deserved fame, this dataset has been acquired in a very constrained setting and does not present all the challenges that a robot faces in a real-life deployment.

In order to fill the existing gap in the robot vision community between research benchmark and real-life application, we introduce a large-scale, multi-view object dataset collected with an RGB-D camera mounted on a mobile robot (see figure 2.1), called *autonomous robot indoor dataset*. The data are autonomously acquired by a robot patrolling a defined human environment. The dataset presents 6,000+ RGB-D scene images and 120,000+ 2D bounding boxes for 153 common everyday objects appearing in the scenes. Analogously to ROD, the object instances are organized into 51 categories, each containing three different object instances. In contrast, our dataset is designed to include real-world characteristics such as variation in lighting conditions, object scale and background as well as occlusion and clutter. To our knowledge, no other robotic dataset embeds all the challenges of real-life data. All the collected data, together with the information needed to replicate the experiments, are publicly available at `https://www.acin.tuwien.ac.at/en/vision-for-robotics/software-tools/autonomous-robot-indoor-dataset/`.

In addition to introducing a new dataset, we inspect the effectiveness of features learned from the Web domain on robotic data and compare them with the features learned from the lab-collected data (ROD). This comparison is made possible by

Figure 2.1: Glimpse at the data collection process with the robotic platform (left) acquiring data of a cluttered scene populated with everyday objects.

collecting a second dataset, called Web Object Dataset (WOD), containing the images downloaded from the Web representing the same categories as ARID. The acquisition of WOD is performed by using query expansion strategies from [75] on different search engines followed by a manual cleaning to remove noisy images. Exhaustive experiments with different CNN architectures demonstrate that, despite the greater similarity between ROD and ARID, models learned from Web images are more effective. Finally, we study the classification results on subsets of ARID representing three problematic characteristics of robotic data: small images, occlusion and clutter. The experiments identify small images and occlusions as the main challenges of robotic data, indicating a path to follow for the resolution of the object classification problem for robotics.

In summary, the contributions of this chapter are:

- ARID: a new RGB-D object dataset, collected in-the-wild with a mobile robot, that provides a "litmus test" for the validation of object recognition algorithms developed for robotic applications,
- WOD: a new Web-based object dataset with the same object categories present in ARID,
- a detailed analysis of available RGB-D datasets from a robotic perspective,
- comprehensive experiments with several well-established CNN architectures, comparing the effectiveness of data coming from the Web and laboratory domain in generating features for object classification in robotics, and

Figure 2.2: Sample of objects used in Autonomous Robot Indoor Dataset. Each object shown belongs to a different category.

- a study of the main factors responsible for the difficulties faced by classifiers on robotic data.

The content of this chapter is based on the published paper

M. Loghmani, B. Caputo, and M. Vincze, **Recognizing Objects in-the-Wild: Where do we Stand?**, in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2170-2177, 2018.

2.1 Autonomous robot indoor dataset

In the following, we describe the characteristics of ARID by highlighting its most significant peculiarities. We also unveil the protocol used for the autonomous data collection and the details of the provided annotation. Finally, we compare ARID with existing RGB-D object datasets.

2.1.1 Scope and Motivation

ARID contains RGB and depth images of daily life objects belonging to 51 categories. Each object category contains three instances, for a total of 153 physical objects, and it coincides with one of the 51 WordNet leaf nodes used to determine the categories ROD. In other words, there is a complete overlap between the categories represented in the two datasets, ARID and ROD. Figure 2.2 gives a concrete idea of the dataset's content by showing one sample object per category.

Since we are mostly interested in autonomous assistive robots operating in indoor environments, the object classes considered in ROD are a valid representative. These objects consist of a large variety of food items, such as fruit, vegetables and packed goods, and human-made objects common to homes and offices. Nevertheless, our goal is not to extend and contribute to ROD, but rather fill the gap between research-oriented datasets and real-life data by introducing a robotic dataset collected in-the-wild. While ROD contains images collected in a constrained setting (fixed camera-object distance, static background, invariant light conditions), our dataset includes all the nuisances of robotic data by acquiring it directly with a mobile robot navigating autonomously in an indoor environment. More precisely, the following challenges are taken into account:

- variation of lighting conditions,
- object scale variation,
- significant changes in the viewpoint,
- partial view and occlusion,
- clutter, and
- background variation.

We hope that this work provides the robot vision community with a tool to advance the visual capabilities of robots in order to accelerate their integration in our lives.

2.1.2 Data Acquisition Protocol

In order to avoid a human bias in data acquisition and to observe the objects from the robot's perspective, the data is collected by mobile robot with an RGB-D camera. In particular, the mobile robotic platform is powered by a Pioneer P3-DX with a customized structure that supports an Asus Xtion Pro camera mounted on a pan/tilt unit (see figure 2.1).

The data collection is performed in 10 different sessions conducted during different days and at different times of the day: this allows a natural variation of the lighting conditions among the data. At each run, 30-31 objects are spread in the environment where the mobile robot patrols predefined waypoints. When a waypoint is reached, the camera scans the scene with a horizontal movement of the pan/tilt unit and acquires RGB and depth data, both with a resolution of 640x480 pixels and a frame rate of 30 Hz. The RGB and the depth frames are later synchronized based on their acquisition time and unmatched frames are discarded. Each session lasts for approximately one hour in which the robot continuously loops over four distinct waypoints. In order to guarantee the appropriate variability in terms of camera-object distance and object view, the objects are randomly moved in between two patrolling loops.

2.1.3 Annotation

In order to discard similar frames, every fifth frame is chosen for annotation for a total of over 6,000 frames. For each frame, a bounding box annotation indicates the location and the label (at instance level) of every visible object for a total of over 120,000 2D bounding boxes for the whole dataset. A modified version of Sloth annotation

Figure 2.3: Example of an RGB-D frame from the Autonomous Robot Indoor Dataset with 2D bounding box annotation.

tool [76] is used for this purpose. In case of occlusion or partial view, if the object is still distinguishable, a bounding box is drawn around the visible part of the object. Figure 2.3 shows a sample frame, together with its bounding box annotation. Since the objects are captured in a realistic scenario rather than in isolation, the dataset is also suitable for object detection. In addition, the availability of object labels at instance level allows the dataset to be used for object categorization as well as instance re-identification (also referred to as instance recognition).

2.1.4 Other datasets

For indoor objects, the most relevant existing datasets are JHUIT-50, BigBIRD, iCub-World Transformation, ROD, and the Active Vision Dataset.

ROD [6] contains 300 objects from 51 categories spanning from fruit and vegetables to tools and containers. Despite the availability of multiple views, each object is presented in isolation and variation in lighting condition, object scale and background are missing. The corresponding scene dataset, the RGB-D Scene Dataset [77], presents multiple objects in the same scene, but considers only five object categories. BigBIRD [78] contains 125 common human-made objects, with particular focus on boxes and bottles. This dataset is specifically designed for instance recognition and the selected objects belong to very few categories. In addition, occlusion, clutter, scale and light variation are not captured. A more recent dataset, the Active Vision Dataset [79], uses a subset of 33 objects from BigBIRD in densly acquired scenes. The data is directly acquired by a robot and it embeds most of the nuisances typical of real-life data. Nevertheless, the limited number of considered objects makes this dataset unsuitable for object categorization. JHUIT-50 [80] contains 50 industrial objects and hand tools used in mechanical operations. The objects are captured in isolation and from multiple

Table 2.1: Summary of the characteristics of different RGB-D datasets with focus on variation in lighting condition, variation in scale, multiple views, occlusion, clutter, variation in background and whether or not the data are collected directly from a robot. *Not Available* (NA) indicates that the dataset focuses on object instances rather than categories and the number of categories is unknown.

Dataset	Characteristic						
Name	**# classes**	**light var.**	**scale var.**	**multiview**	**occlusion**	**clutter**	**bkg var.**	**robot**
RGB-D Object Dataset [6]	51			✓				
RGB-D Scene Dataset [77]	5		✓	✓	✓	✓	✓	
BigBIRD [78]	NA			✓				
Active Vision Dataset [79]	NA	✓	✓	✓	✓		✓	✓
JHUIT-50 [80]	NA			✓				
li2016hierarchical [81]	NA			✓	✓	✓		
iCubWorld Transf. [82]	15	✓	✓	✓			✓	✓
Autonomous Robot Indoor Dataset (ARID)	51	✓	✓	✓	✓	✓	✓	✓

viewpoints. Similarly to the Active Vision Dataset, JHUIT-50 is more suitable for instance recognition rather that object categorization. In addition, nuisances such as occlusion, clutter, scale and light variation are not captured. The corresponding scene dataset [81], includes occlusion and clutter, but limits the number of considered objects to 10. iCubWorld Transformation [82] contains 150 common indoor objects from 15 different categories. The data are collected directly with the iCub humanoid robot [83]. This dataset addresses specifically variance in the background as well as the variance in scale and rotation of the object. Nevertheless, each object is presented in isolation, avoiding problems caused by cluttered scenes.

Despite the high-quality that characterizes each of these datasets, their constrained setting makes them incoherent with real-life data. In addition, only the Active Vision Dataset and the iCubWorld Transformation present data collected directly from a robot. Table 2.1 presents a summary of the characteristics of the datasets discussed above and highlights that, differently from other datasets, ARID embeds all these characteristics.

2.2 Web object dataset

In order to evaluate the transferability of features learned from the Web to robotic data, we collect WOD. This dataset is composed of images downloaded from the Web representing objects from the same categories as ARID. The images are downloaded from multiple search engines (Google, Yahoo, Bing and Flickr) using the method proposed by Massouh et al. [75]. This method uses a concept expansion strategy by leveraging visual and natural language processing information to minimize the noise while maximizing

the visual variability. The remaining noise is then manually removed, leaving a total of 50,547 samples.

2.3 Experiments

In this section, we take advantage of the availability of ARID to perform experiments that disclose the characteristics of robotic data. In particular, we want to (i) analyze the transferability of features from the Web domain to the robotic domain and (ii) study the characteristics of robotic data to identify the main sources of complication for classifying objects and explore possible solutions. In the following, section 2.3.1 describes the evaluation protocol for the datasets and network architectures and section 2.3.2 shows quantitative results on the object categorization task.

2.3.1 Setup: datasets & network architectures

Datasets The limited availability of robotic data raises the question of whether data coming from a more accessible domain, the Web domain, can be effectively used instead of data collected in the lab to learn features that are transferable to the robotic data. In particular, we want to compare the performance of well-known CNN architectures on robotic data (ARID), when trained on Web data (WOD) and on lab-collected data (ROD). Figure 2.4 shows example crops from each dataset. In order to allow a fair evaluation, a subset of 40,000 samples from ARID dataset is selected, such that all the involved datasets are approximately the same size. It is worth noticing that, since WOD does not contain depth information, only RGB data are considered for all datasets. For each dataset, we consider multiple training/test splits and average the results to obtain the final classification accuracy. In particular, for ARID, each split uses one different object instance per class in the test set, for ROD, we use the first three splits indicated by the authors and, for WOD, each split uses 25% of the data in the test set.

Network architectures We employ some of the most utilized network architectures in the literature, CaffeNet[1], VGG-16, Inception v2, ResNet-18 and ResNet-50. All networks are pre-trained on ImageNet and then fine-tuned on the desired dataset, according to the guidelines provided in [84], [85].

2.3.2 Results

What are the baseline performances of the networks on the three considered datasets? In order to provide a reference for the upcoming evaluations, we assess the performances of all considered networks for each of the three datasets (ARID, ROD, WOD) when training and test set come from the same dataset. Figure 2.5 shows that the different networks consistently obtain a higher accuracy on WOD. Unsurprisingly, ARID appears to be the most challenging dataset and all the networks achieve an accuracy much lower (on average, ~ 0.4 lower) on ARID than on the other two datasets.

[1] A slightly modified version of AlexNet in which the normalization is performed after the pooling.

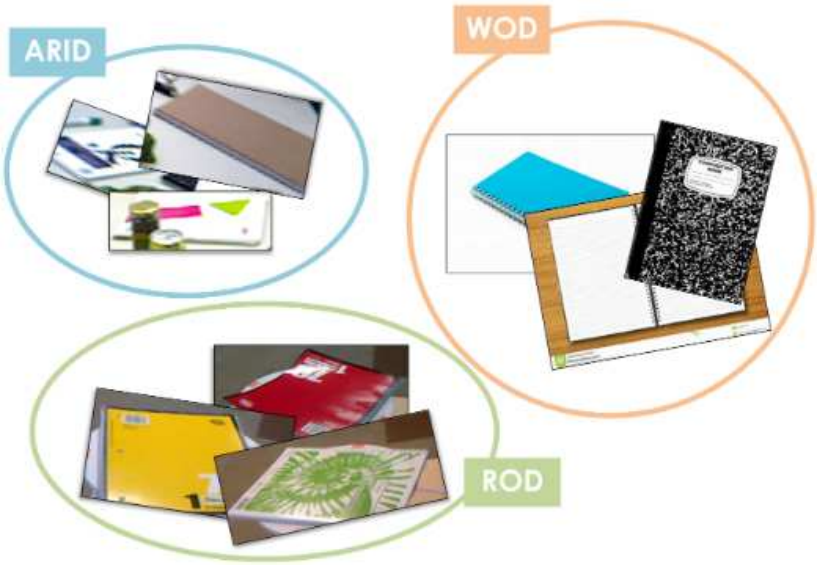


Figure 2.4: Example crops from three datasets: Autonomous Robot Indoor Dataset (ARID), Web Object Dataset (WOD), RGB-D Object Dataset (ROD). The figure shows three random samples from the *notebook* category for each dataset to showcase their visual differences.

How do features learned from the Web domain and features learned from the lab domain transfer to robotic data? The networks fine-tuned on ROD and WOD are then tested on ARID to evaluate the transferability of the learned features to the robotic data. The results in figure 2.6 show that all the networks undergo a performance drop when the training and test set belong to different datasets with respect to the case in which both sets belong to the same dataset (see figure 2.5). The domain shift responsible for this negative inflection of the classification results occurs because the data composing training and test set are drawn from different distributions [86]. However, features learned from Web data (WOD) consistently allow a higher classification accuracy (with improvements up to 0.05) on robotic data (ARID) than features learned from lab-collected data (ROD) on all networks, with the exception of CaffeNet. The key factor to interpret this phenomenon is the greater variability of Web images: while ROD contains a limited number of instances per class, with some classes containing only three instances, in WOD each sample potentially represents a different object instance. Deep networks, like ResNet-50, with high capacity and generalization power, take advantage of this richness in information to generate better models. This is further highlighted by the difference between the accuracy of ResNet-50 and the mean accuracy of all tested networks when training with WOD (see table 2.2). The results of this experiment have a twofold implication: (i) despite the greater visual affinity between the laboratory and the robotic domain, data from the Web domain generate more effective models for object classification in robotic applications, and (ii) CNNs, when used in their plain stand-alone form and without any prior, do not perform satisfactorily for object classification in robotics.

What makes robotic data so challenging? In order to better understand which

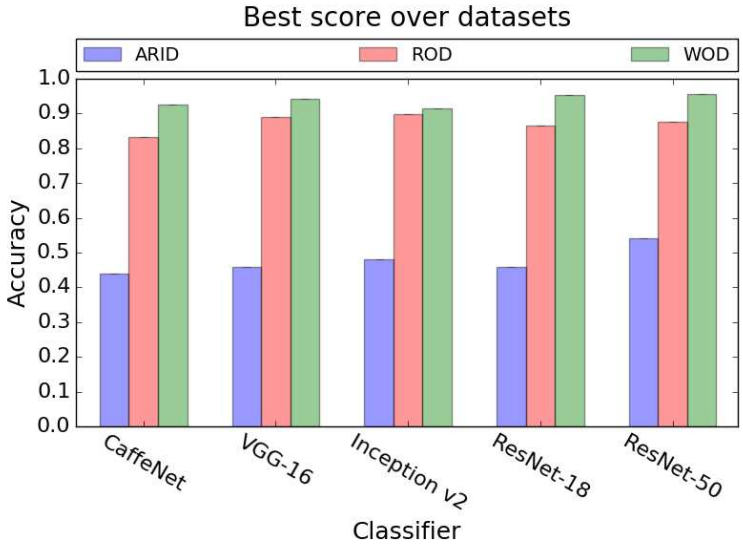


Figure 2.5: Accuracy of different deep convolutional networks on three datasets: Autonomous Robot Indoor Dataset (ARID), RGB-D Object Dataset (ROD) [6] and Web Object Dataset (WOD). The results are obtained by training and testing on different splits of the same dataset.

Table 2.2: Accuracy of multiple deep convolutional networks on different training/test combination of three datasets: Autonomous Robot Indoor Dataset (ARID), RGB-D Object Dataset (ROD) [6] and Web Object Dataset (WOD). For each training/test set combination, the mean and maximum accuracy among the considered networks is shown.

Dataset		**Network**					**Statistics**	
Train on	*Test on*	*CaffeNet*	*VGG-16*	*Inception-v2*	*ResNet-18*	*ResNet-50*	*Mean*	*Max*
ROD	ROD	0.832	0.889	0.897	0.864	0.876	0.872	0.897
ROD	ARID	0.291	0.270	0.266	0.243	0.337	0.281	0.337
WOD	WOD	0.924	0.942	0.914	0.953	0.956	0.938	0.956
WOD	ARID	0.268	0.297	0.282	0.282	0.388	0.303	0.388
ARID	ARID	0.441	0.458	0.481	0.458	0.540	0.476	0.540

characteristics of robotic data negatively influence the results of the object classification task, we independently analyze three key variables: image dimension, occlusion and

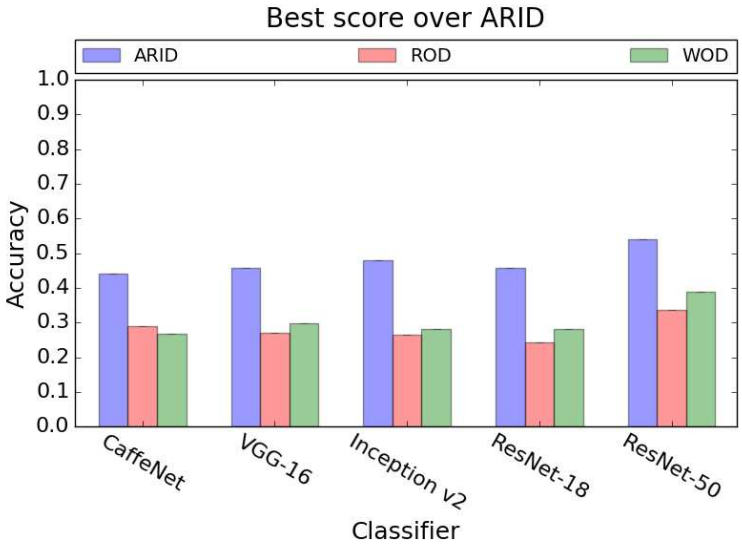


Figure 2.6: Accuracy of different deep convolutional networks on Autonomous Robot Indoor Dataset (ARID). The results are obtained by training independently on ARID, RGB-D Object Dataset (ROD) [6] and Web Object Dataset (WOD) and testing on ARID.

clutter[2]. Image dimension is a variable related to the camera-object distance: when the camera is not near enough to clearly capture the object, the object occupies only few pixels in the whole frame, making the classification task more challenging. For obvious reasons, this problem is emphasized when dealing with small and/or elongated objects, such as dry batteries or glue sticks. Occlusion occurs when a portion of an object is hidden by another object or when only part of the object enters the field of view. Since distinctive characteristics of the object might be hidden, occlusion makes the classification task considerably more challenging. Clutter refers to the presence of other objects in the vicinity of the considered object. The simultaneous presence of multiple objects may interfere with the classification task. Table 2.3 shows the classification results of the best performing model of table 2.2 (ResNet-50 trained on WOD) on three subsets of ARID, each containing samples with the characteristics discussed above. The set of small images is obtained by taking half of ARID containing images with the smallest area, while the occlusion and clutter set have been manually selected. It is worth noticing that the three sets are mutually exclusive in order to avoid interference between the analyzed variables. The occlusion and, especially, the small images set exhibit low accuracy, thus negatively affecting the classification score of the whole dataset. The difficulty of classifying small images is further confirmed by the results in figure 2.7, where classes representing small or elongated objects have the lowest accuracy.

How can we attenuate the domain gap between Web and robotic data? We explore

[2]Since ARID is collected in-the-wild, by definition, the data acquisition is performed in an unconstrained manner. For this reason, rigorously isolating other characteristics of the data, such as light variation, background variation and different object view is prohibitive.

Table 2.3: Accuracy of ResNet-50 trained on Web Object Dataset and tested on Autonomous Robot Indoor Dataset (ARID). "ARID[x]" indicates a sub-set x of ARID. "WOD aug. y" indicates WOD with a type y of data augmentation. "Adapt." indicates the use of a domain adaptation strategy during training.

Dataset		**Network**
Train on	**Test on**	**ResNet-50**
WOD	ARID[small image]	0.230
WOD	ARID[occlusion]	0.273
WOD	ARID[clutter]	0.558
WOD aug. small	ARID[small image]	0.240
WOD aug. occlusion	ARID[occlusion]	0.318
WOD aug. clutter	ARID[clutter]	0.543
WOD	ARID	0.388
WOD aug. small+occlusion	ARID	0.441
WOD adapt.	ARID	0.582

two possible training strategies to improve the classification performance on these challenging cases: problem-specific data augmentation and DA. Data augmentation is a very common practice [17], [87] that consists in artificially increasing the size of the training set by adding processed version of the original data. In particular, we tackle the small images by resizing the original WOD samples to different scales, and we simulate occlusion by randomly adding rectangular noise patches to the original images. Table 2.3 shows that this solution brings an improvement, especially on the occluded samples. Training with these two augmentation techniques and testing on the whole ARID dataset brings an overall improvement of +5.3% over the vanilla case. Finally, we apply a standard DA method, DANN [55], to explicitly reduce the domain shift between WOD and ARID. This method encourages the feature extractor to generate domain-invariant features using adversarial learning. A domain discriminator is trained to distinguish source (WOD) from target (ARID) samples, while the feature extractor is trained to fool the discriminator using a gradient reversal layer. Table 2.3 (WOD adapt.) shows that applying DANN brings a very significant improvement of +19.4% over the vanilla case. It is worth mentioning that, differently from the data augmentation case, applying DA requires to transductively use unlabeled test data during training. In many applications, collecting data without manual annotation is feasible and cheap and is therefore worth considering given the significant gain in performance when applying DA.

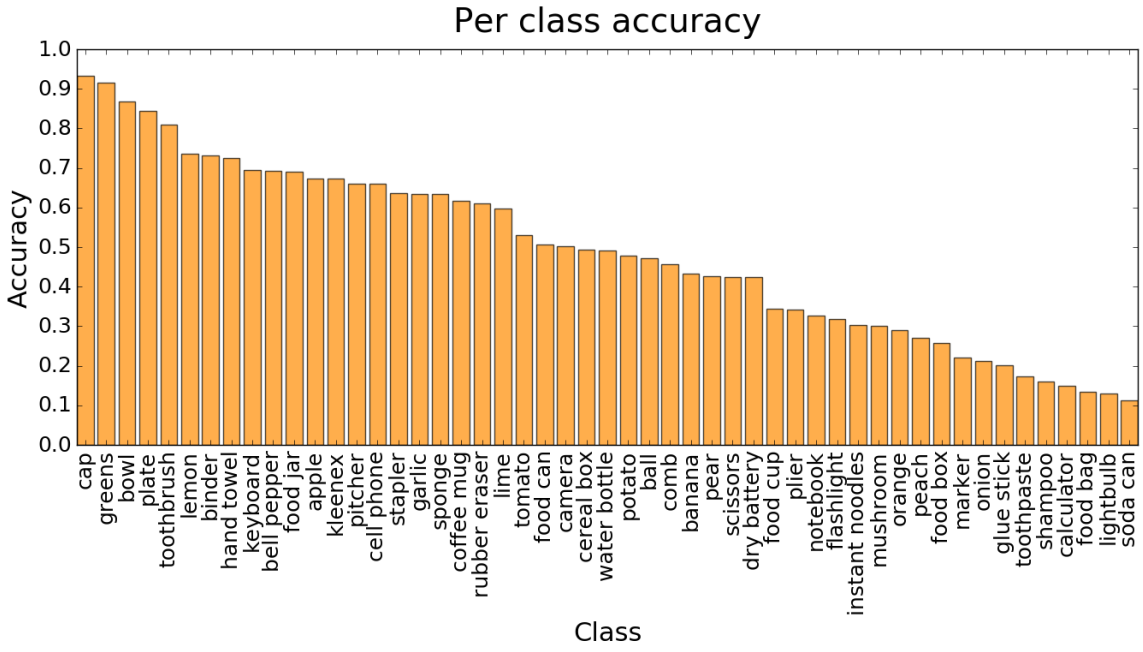


Figure 2.7: Accuracy of each of the 51 classes of the Autonomous Robot Indoor Dataset obtained with a ResNet-50 trained on the augmented Web Object Dataset.

2.4 Discussion

In this chapter, we have presented ARID: a large-scale, multi-view, RGB-D object dataset collected with a mobile robot in-the-wild. This dataset is designed to capture the challenges a robot faces when deployed in an indoor environment and fills the current gap in the robot vision community between research oriented datasets and real-life data. Furthermore, with an extensive comparative study, we have shown that it is possible to overcome the complication of collecting a large amount of robotic data for training data-craving deep convolutional networks by using images downloaded from the Web. We have found that, despite being relatively easy to obtain, Web-based data allow the generation of more effective deep models than the lab-collected counterpart for the classification of robotic images. Nevertheless, object classification remains a challenging task in robotics and current algorithms present results that are insufficient for a successful integration of robotic systems in our homes. In order to shed light on the difficulties of this task, we have analyzed the effects of specific factors, such as object dimension, occlusion and clutter, on the performance. Results indicate that clutter is rather a secondary problem: occlusions and small objects more seriously degrade the classification accuracy. Training the network with data augmentation and, especially, DA strategies can significantly improve the performance and partially compensate for the domain shift between Web and robotic data. However, these strategies have a limited impact in improving the classification of small images. When the object is captured in very few pixels in the frame, the information necessary to recognize the object is simply lacking. The robot must therefore get closer to the object to determine its identity. This type of solution, which is concerned with manipulating the viewpoint

of the camera in order to get better information, falls under the umbrella of *active vision* [79] and is out of the scope of this thesis. In the following, we will rather focus on how to design better classifiers to get the most out of a static frame.

In chapter 3, we investigate whether the integration of depth information with the standard RGB images can improve the classification performance on cluttered and occluded images. In chapter 4, we further explore the use of DA on RGB-D data in robotic applications.

Highlights:

- ➔ Deep neural networks trained on Web images are a good starting point for object recognition.
- ➔ Classification accuracy on robotic data is lower than on Web data due to challenges like small objects and occlusions.
- ➔ Data augmentation and especially domain adaptation can be used to improve classification accuracy on robotic data.

Chapter 3

Recurrent Convolutional Fusion for RGB-D Object Recognition

Despite the interesting results achieved for object recognition using standard RGB images, there are inherent limitations due to the loss of data caused by projecting the 3-dimensional world into a 2-dimensional image plane. The use of RGB-D (Kinect-style) cameras alleviates these shortcomings by using range imaging technologies to provide information about the camera-scene distance as a depth image. These sensors became ubiquitous in robotics due to their affordable price and the rich visual information they provide. In fact, while the RGB image contains color, texture and appearance information, the depth image contains additional geometric information and is more robust with respect to lighting and color variations. Since RGB-D cameras are already deployed in most SRs, improving the performance of robot perceptual systems through a better integration of RGB and depth information constitutes a "free lunch". In fact, depth images can help dealing with open robot vision challenges, such as occlusions, by disambiguating the spatial arrangement of the elements in the image.

In the last year, research in RGB-D object recognition followed the deep learning trend, with numerous algorithms [10], [17], [18] exploiting features learned from CNNs instead of the traditional hand-crafted features. The common pipeline involves two CNN streams, operating on RGB and depth images respectively, as feature extractors. However, the lack of a large-scale dataset of depth images to train the depth CNN forced the vision community to find practical workarounds. Much effort has been dedicated to develop methods that colorize the depth images to exploit CNNs pre-trained on RGB images. However, the actual strategies to extract and combine the features from the two modalities have been neglected. Several methods simply extract features from a specific layer of the two CNNs and combine them through a fully connected or a max pooling layer. We argue that these strategies are sub-optimal because (a) they assume that the selected layer always represents the best abstraction level to combine RGB and depth information and (b) they do not exploit the full range of information from the two modalities during the fusion process.

In this chapter, we propose a novel end-to-end architecture for RGB-D object recognition called *recurrent convolutional fusion* (RCFusion). Our method extracts features from multiple hidden layers of the CNNs for RGB and depth, respectively, and combines them through a recurrent neural network (RNN), as shown in figure 3.1. Our idea is that combining RGB and depth features from several levels of abstraction can provide

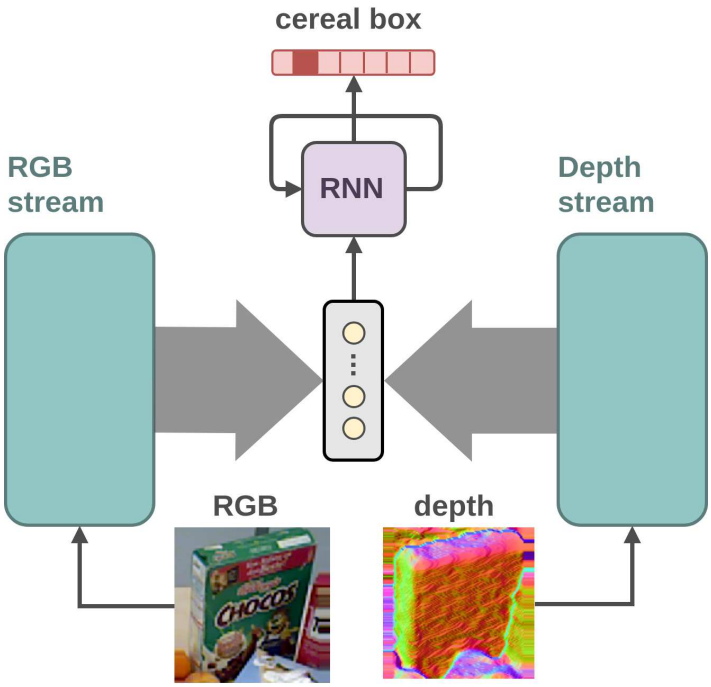


Figure 3.1: High-level scheme of recurrent convolutional fusion. The blue boxes are instantiated with convolutional neural networks (CNNs) and the thick arrows represent multiple feature vectors extracted from different layers of a CNN.

greater information to the classifier to make the final prediction. Although RNNs are typically used to process sequential data, this type of neural networks have been proven to be very effective information compressors [88] and scale well in the parameters with respect to the number of extracted features, as discussed in section 3.1.3. In addition, we provide experimental evidence that this solution is superior to simply fusing the concatenated features with a fully connected layer (see ablation study in section 3.2.4).

We evaluate our method on standard object recognition benchmarks, ROD and JHUIT-50 [81], and we compare the results with the best performing methods in the literature. The experimental results show that our method outperforms the existing approaches and establishes new state-of-the-art results for both datasets. In order to further consolidate the effectiveness of our method, we adapt an object segmentation dataset, called Object Clutter Indoor Dataset (OCID) [89], to the instance recognition task to further evaluate RCFusion. OCID has been recently released to provide object scenes with high level of clutter and occlusion, arguably two of the biggest challenges faced by robotic visual perception systems [90]. Our method confirms its efficacy also on this challenging dataset, despite the small amount of training data available. An implementation of the method, relying on tensorflow [91], is publicly available at: `https://github.com/MRLoghmani/rcfusion`.

In summary, the contributions of this chapter are:

- a novel architecture for RGB-D object recognition that sequentially combines RGB and depth features representing different levels of abstraction,

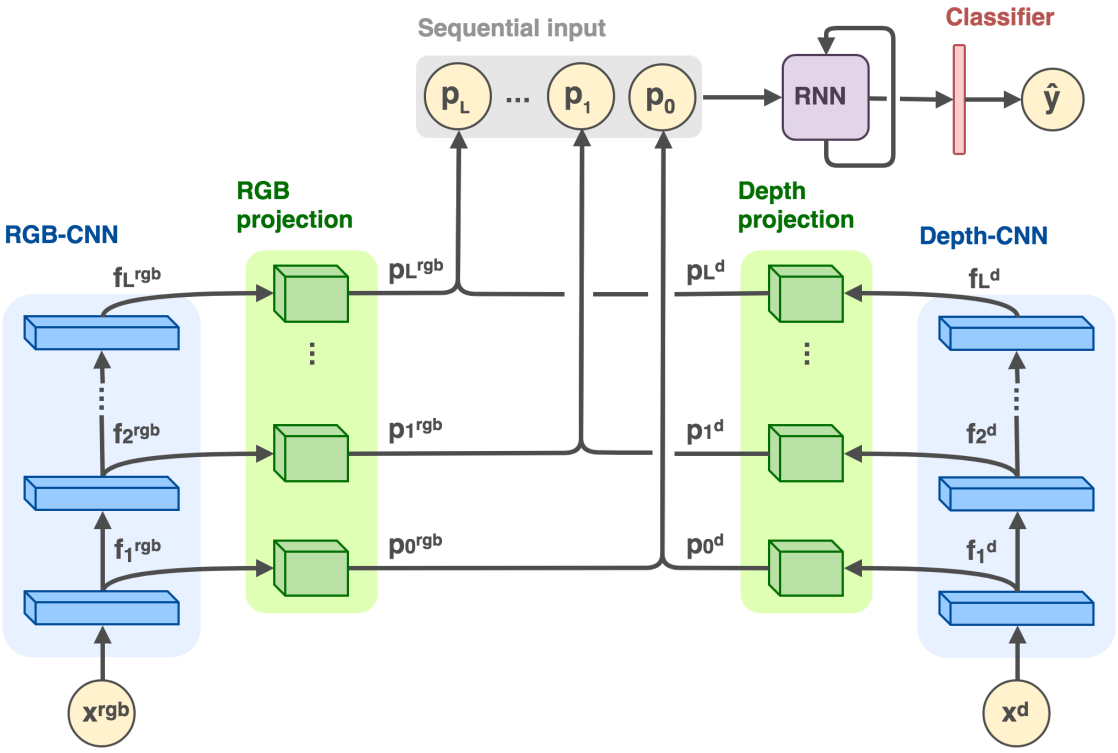


Figure 3.2: Architecture of recurrent convolutional fusion. It consists of two streams of convolutional neural networks (CNNs) that process RGB and depth images, respectively. The output of corresponding hidden layers from the two streams are projected into a common space, concatenated and sequentially fed into a recurrent neural network (RNN) that synthesizes the final multi-modal features. The output of the RNN is then used by a classifier to determine the label of the input data.

- state-of-the-art performance on the most popular RGB-D object recognition benchmark datasets,
- introduction of a new benchmark with robotic-oriented challenges, i.e. clutter, occlusion and little training data.

The content of this chapter is based on the published paper

M. Loghmani, M. Planamente, B. Caputo, and M. Vincze, **Recurrent Convolutional Fusion for RGB-D Object Recognition**, in *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 3, pp. 2878-2885, 2019.

3.1 Recurrent convolutional fusion

Our multi-modal deep neural network for RGB-D object recognition is illustrated in figure 3.2. The network's architecture has three main stages:

1. *multi-level feature extraction:* two streams of convolutional networks, with the same architecture, are used to process RGB and depth data (RGB-CNN and Depth-CNN), respectively, and extract features at different levels of the networks;

2. *feature projection and concatenation:* features extracted from each level of the RGB- and Depth-CNN are individually transformed through projection blocks and concatenated to create the corresponding RGB-D feature;

3. *recurrent multi-modal fusion:* RGB-D features extracted from different levels are sequentially fed to an RNN that produces a descriptive and compact multi-modal feature.

The output of the recurrent network is then used by a softmax classifier to infer the object label. The network can be trained end-to-end with a cross-entropy loss using standard backpropagation algorithms based on stochastic gradient descent. In the following, we describe in greater detail each of the aforestated characteristics of RCFusion.

3.1.1 Multi-level feature extraction

CNNs process the input with sets of filters learned from a large amount of data. These filters represent progressively higher levels of abstraction, going from the input to the output: edges, textures, patterns, parts, and objects [92]. Methods for RGB-D object recognition commonly combine the output of one of the last layers of the RGB- and Depth-CNN (typically the last layer before the classifier) and assume that the chosen layer represents the appropriate level of abstraction to combine the two modalities. We argue that it is possible to remove this assumption by combining RGB and depth information at multiple layers across the CNNs and use them all to generate a highly discriminative RGB-D feature. Let us denote with $x^{rgb} \in \mathcal{X}^{rgb}$ the RGB input images, with $x^d \in \mathcal{X}^d$ the depth input images and $y \in \mathcal{Y}$ the labels, where $\mathcal{X}^{rgb}$, $\mathcal{X}^d$ and $\mathcal{Y}$ are the RGB/depth input and label space. We further denote with f_i^{rgb} and f_i^d the output of layer i of RGB-CNN and Depth-CNN, respectively, with $i = 1,...,L$ and L the total number of layers of each CNN. Notably, visualizing the learned filters has shown [92] that, for a given task, a chosen CNN architecture consistently generates features with the same level of abstraction from a reference layer. For example, AlexNet [3] learns various types of Gabor filters in the first convolutional layer. So, the same architecture is chosen for RGB- and Depth-CNN to ensure the same abstraction level at corresponding layers.

3.1.2 Feature Projection and Concatenation

One of the main challenges in combining features obtained from different hidden layers of the same network is the lack of a one-to-one correspondence between elements of the different feature vectors. More formally, f_i^* and f_j^*, with $i \neq j$ and $*$ indicating any of the superscripts rgb or d, have (in general) different dimensions and thus belong

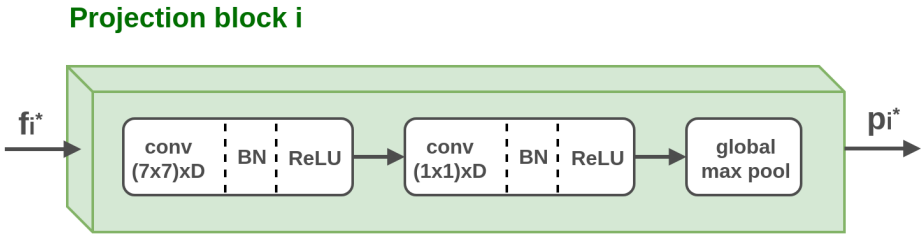


Figure 3.3: Implementation of the projection block that transforms the feature f_i^* into the projected feature p_i^*. $conv(k \times k) \times D$ indicates a convolutional layer with D filters of size $(k \times k)$, BN indicates a batch normalization layer and $ReLU$ indicates an activation layer with ReLU non-linearity.

to distinct feature spaces, $\mathcal{F}_i$ and $\mathcal{F}_j$. In order to make features coming from different levels of abstraction comparable, we project them into a common space $\bar{\mathcal{F}}$:

$$p_i^* = G_i^*(f_i^*) \quad \text{s.t.} \quad p_i^* \in \bar{\mathcal{F}} \tag{3.1}$$

The projection block $G_i(.)$ performs a set of non-linear operations to transform a volumetric input into a vector of dimensions (1×D). More specifically, $G_i(.)$ is defined by two convolutional layers (with batch normalization and ReLU non-linearity) and a global max pooling layer, as shown in figure 3.3. The projected RGB and depth features of each layer i are then concatenated to form $p_i = \left[p_i^{rgb}; p_i^d\right]$.

3.1.3 Recurrent Multi-modal Fusion

In order to create a compact multi-modal representation, the set $\left\{p_1, \ldots, p_L\right\}$ is sequentially fed to an RNN. Recurrent models align the positions of the elements in the sequence to steps in computation time and generate a sequence of hidden states h_i as a function of the previous hidden state h_{i-1} and the current input p_i. For our method, we use an instantiation of an RNN called *gated recurrent unit* (GRU) [93]. This network is considered to be a variation of *long-short term memory* (LSTM) [94] that requires 25% less parameters. GRU has been proven to be able to retain information even in extremely long sequences with thousands of elements [88].

GRU computes the n^{th} element of the hidden state at step i as an adaptive linear interpolation:

$$h_i^n = (1 - z_i^n)h_{i-1}^n + z_i^n \tilde{h}_i^n, \tag{3.2}$$

where z_i^n is called update gate and is computed as

$$z_i^n = sigmoid(\theta_z p_i + \gamma_z h_i)^n, \tag{3.3}$$

where $sigmoid(.)$ is the sigmoid function and θ_z and γ_z are the trainable parameters of the gate. Essentially, the update gate determines how much the unit updates its content. The candidate activation $\tilde{h}_i$ in equation 3.2 is computed as

$$\tilde{h}_i^n = tanh(\theta_h p_i + \gamma_h(r_i \odot h_{i-1}))^n, \tag{3.4}$$

where r_i is the reset gate, θ_h and γ_h are trainable parameters and $\odot$ is the element-wise multiplication operation. Similarly to z_i^n, the reset gate r_i^n is computed as

$$r_i^n = sigmoid(\theta_r p_i + \gamma_r h_i)^n, \tag{3.5}$$

where θ_r and γ_r are the trainable parameters of the gate. When r_i^n assumes values close to zero, it effectively resets the hidden state of the network to the current input p_i. This double-gate mechanism has the goal of ensuring that the hidden state progressively embeds the most relevant information of the input sequence $\{p_1, \ldots, p_L\}$.

The RNN, combined with a softmax classifier, models a probability distribution over a sequence by being trained to predict the category label given the sequence of projected RGB-D features. In particular, the prediction of the j^{th} class of the multinomial distribution of K object categories is obtained as

$$\hat{y}_j = Pr(y_j = 1|p_1, ..., p_1) = \frac{exp(h_L^T \theta_c^j)}{\sum_{k=1}^{K} exp(h_L^T \theta_c^k)}, \tag{3.6}$$

where θ is the matrix of trainable parameters of the classifier and $\theta_{j(/_k)}$ represents its $j^{th}(/k^{th})$ row, and the superscript T represents the transpose operation.

The choice of a recurrent network for this operation is twofold: (a) the hidden state of the network acts as a memory unit and embeds a summary of the most relevant information from the different levels of abstraction, and (b) the number of parameters of the network is independent of L, while for a more straightforward choice, such as a fully connected layer, it grows linearly with L. Although RNNs are typically used to process time series data, our atypical deployment is supported by previous works [95], [96] that have shown that these type of networks are also useful in compressing and combining information from different sources. We empirically demonstrate in the ablation study in section 3.2.4 that a recurrent network is more effective than a typical fully connected layer in aggregating the RGB and depth features from different levels of abstraction.

3.2 Experiments

In the following, we evaluate RCFusion on ROD, JHUIT-50, and OCID. After revealing the protocol used to set up the experiments, we discuss the setting used for training the network. Then, we show how the performances of our method compare to the existing literature. Finally, we perform an ablation study to identify the contribution of the different elements of our method.

3.2.1 Datasets

RGB-D Object Dataset Since ROD is used in the evaluation of chapter 2, the reader can refer to section 2.1.4 and 2.3.1 for a description of the dataset. Despite using the same evaluation protocol as chapter 2, we obviously use both the RGB and the depth data for this chapter's experiments.

JHUIT-50 It contains 14,698 RGB-D images capturing 50 common workshop tools, such as clamps and screw drivers. Since this dataset presents few objects, but very similar to each other, it can be used to assess the performance of RCFusion in the instance recognition task. For the evaluation, we follow the standard experimental protocol defined in [81], where training data are collected from fixed viewing angles between the camera and the object while the test data are collected by freely moving the camera around the object.

Object Clutter Indoor Dataset It includes 96 cluttered scenes representing common objects organized in three subsets: ARID20, ARID10, and YCB10. The ARID20 and ARID10 subsets contain scenes that include, respectively, up to 20 and 10 out of 59 objects from ARID [90]. Similarly, the YCB10 subset contains scenes with up to 10 objects from Yale-CMU-Berkeley object and model set [97]. Each scene is built incrementally by adding one object at a time and recording new frames at each step. Two ASUS-PRO cameras, positioned at different heights, are used to simultaneously record each scene. Further scene variation is introduced by changing the support plane (floor and table) and the background texture. Since OCID has been acquired to evaluate object segmentation methods in cluttered scenes, semantic labels are not provided by the authors. In order to adapt this dataset to a classification task, we crop out the objects from each frame and annotate them with semantic labels similar to ROD. To avoid redundancies, we go sequentially through the frames of each scene and save only the crops that have an overlap with the bounding box of a newly introduced object. We then filter out the classes with less than 20 images to ensure a minimum amout of training samples per class. We use the crops from the ARID20 subset to train the network for an instance recognition task and then use the crops from the ARID10 subset for testing. Overall, we obtain 3,939 RGB-D images capturing 49 distinct objects. The original datasets, as well as the crops and annotation used in this paper are available at `https://www.acin.tuwien.ac.at/en/vision-for-robotics/software-tools/object-clutter-indoor-dataset/`

Notes on the Autonomous Robot Indoor Dataset The reader might wonder why the evaluation does not include experiments with ARID. The answer lies in the quality of the raw depth images provided by the RGB-D camera in unconstrained setups. Popular RGB-D datasets ensure the depth quality by either collecting the data in constrained settings [6], [81] or by aggregating information from different acquisitions through temporal filtering [89] or scene reconstruction [79]. Since ARID aims at simulating the deployment of a robot in the wild, the data are collected as provided by the sensor without altering the behavior of the robot for data collection purposes. Figure 3.4 shows two examples of depth frames from ARID. The unreliability and high-level of noise of the sensor results in large areas with no depth information (black pixels) and objects indistinguishable from the supporting surface. This directly translates into a poor 14.9% classification accuracy when training a ResNet-18 to predict the 51 categories in ARID from the depth images. In these circumstances, the depth information is negligible and it is more sensible to simply use the RGB data for object recognition. It is worth noting that this discussion is not limited to the specific camera

used to collect ARID, but it exemplifies the entire range of RGB-D sensors on the market at the time of writing.

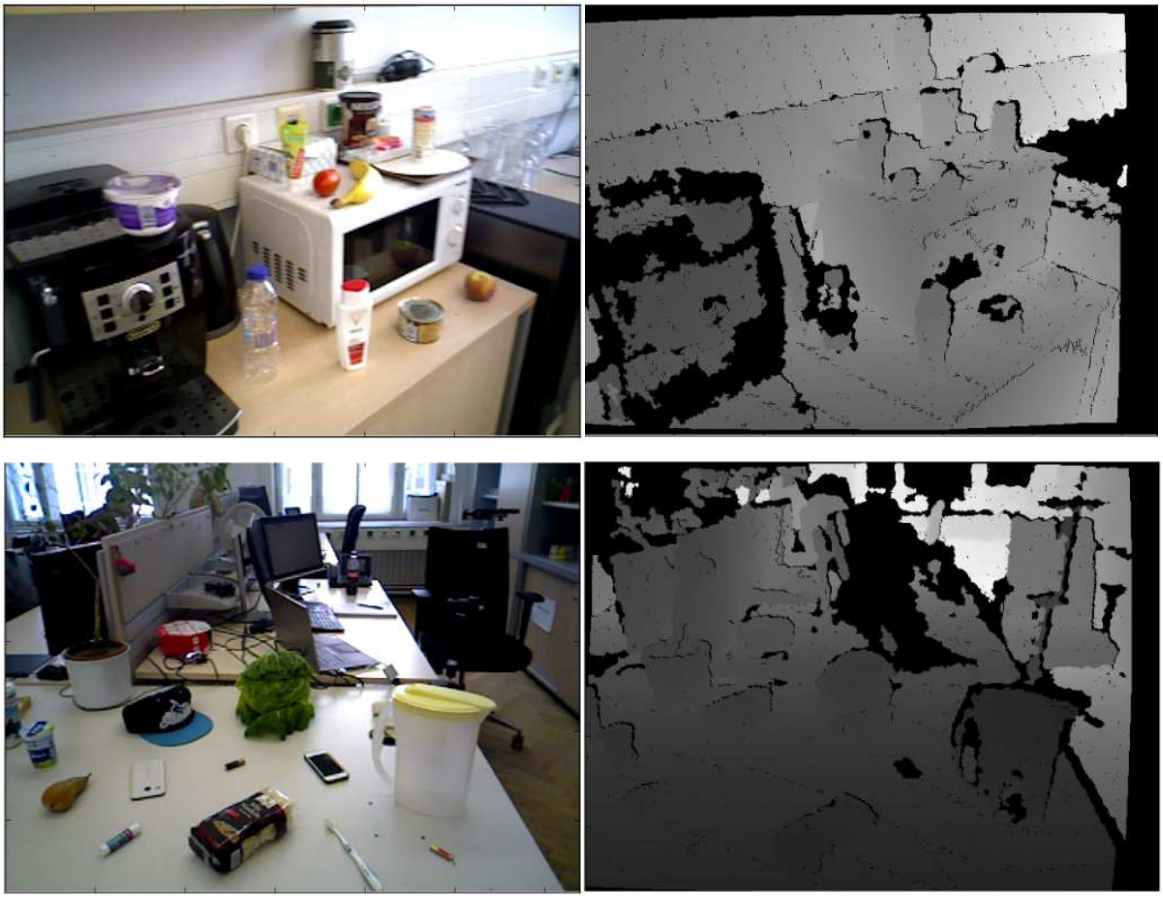


Figure 3.4: Two examples of RGB-D frames from the Autonomous Robot Indoor Dataset (ARID). Due to the limitations of current RGB-D sensors, the raw depth data are very noisy and unreliable when collected in an unconstrained setup. This results, for examples, in large areas with no depth information (black pixels) or objects indistinguishable from the supporting surface. Note that the black borders that appear mostly on the right of the depth images are simply the effect of registering the depth to the RGB image and not a shortcoming of the sensor.

3.2.2 Architecture

The network architecture of RCFusion passes through independent design choices of three main elements: RGB-/Depth-CNN, projection blocks and RNN.

RGB-/Depth-CNN With computational and memory efficiency in mind, we choose a CNN architecture with a relatively small number of parameters. Since residual networks have become a standard choice, we deploy ResNet-18, the most compact

representation proposed by He *et al.* [98]. ResNet-18 has 18 convolutional layers organized in five residual blocks ($\sim$ 40,000 parameters). We extract our features after each skip connection in the network. The network has two skip connections per residual blockand we start extracting from the second block: this results in $L = 8$ extracted features per network. An implementation of ResNet-18 pre-trained on ImageNet is available in [99].

Projection blocks The projection blocks, shown in figure 3.3, are designed in such a way that the first convolutional layer focuses on exploiting the spatial dimensions of the input, width and height, with $D = 512$ filters of size (7×7), while the second convolutional layer exploits its depth with $D = 512$ filters of size (1×1). Finally, the global max pooling computes the maximum of each depth slice. This instantiation of the projection blocks has provided the best performances among those that we tried.

RNN In a trade-off between network capacity and small number of parameters, we use the popular GRU [93]. In our experiments, we process the sequence of projected vectors with a single GRU layer with a number of memory neurons $M = 50$. An implementation of GRU can be found in all the most popular deep learning libraries, including tensorflow.

3.2.3 Training

We train our model using RMSprop optimizer with batch size 64, learning rate 0.001, momentum 0.9, weight decay 0.0002 and max norm 4. The architecture specific parameters have been fixed through a grid search to projection depth $D = 512$ and memory neurons $M = 50$. The weights of the two ResNet-18 used as the RGB- and Depth-CNN are initialized with values obtained by pre-training the networks on ImageNet. The rest of the network is initialized with Xavier initialization method in a multi-start fashion, where the network is initialized multiple times and, after one epoch, only the most promising model continues the training. All the parameters of the network, including those defining the RGB- and Depth-CNN, are updated during training. The input to the network is synchronized RGB and depth images pre-processed following the procedure in [18], where the depth information is encoded with surface normals, the best non-learned colorization method. For JHUIT-50 and OCID, we compensate for the small training set with simple data augmentation techniques: scaling, horizontal and vertical flip, and 90 degree rotation.

3.2.4 Results

In order to validate our method, we first compare the performance of RCFusion to existing methods on two benchmark datasets, ROD and JHUIT-50. We then test our method on a more challenging dataset, OCID, and perform an ablation study to showcase the contribution of each component of the method.

How does RCFusion perform on standard benchmark datasets? We benchmark RCFusion on ROD and JHUIT-50 against other methods in the literature. Table 3.1

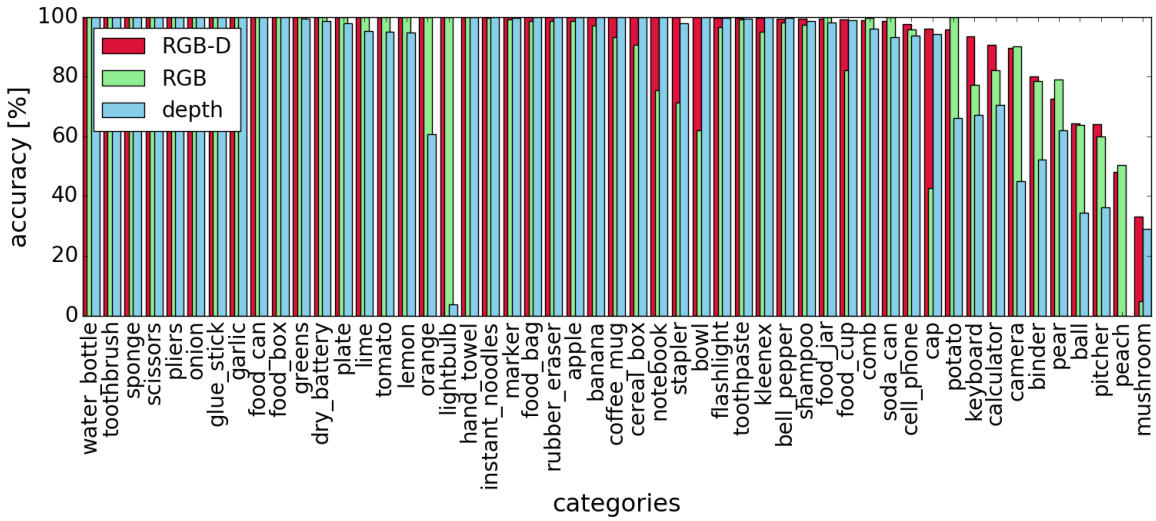


Figure 3.5: Per class accuracy (%) of recurrent convolutional fusion on RGB-D Object Dataset [6].

shows the results on ROD for the object categorization task. The reported results for the RGB and depth modality are obtained by training a classifier on the final features of the RGB- and Depth-CNN, respectively. The reported multi-modal RGB-D results show that our method outperforms all the competing approaches. In addition, the results of the single modality predictions demonstrate that ResNet-18 is a valid trade-off between small number of parameters and high accuracy. In fact, on the RGB modality, the accuracy is second only to [14], where they use a VGG network [100] that introduces considerably more parameters than ResNet-18. For the depth modality, ResNet-18 provides higher accuracy than all the competing methods.

In order to gain a better insight on the performance of RCFusion, we consider the accuracy on the individual categories of ROD. Figure 3.5 shows that the multi-modal approach either matches or improves over the results on the single modalities for almost all categories. For categories like "lightbulb", "orange" or "bowl", where the accuracy on one modality is very low, RCFusion learns to rely on the other modality. An interesting insight on the functioning of the method is given by comparing, for each category, which other categories generate the misclassification. Table 3.3 indicates, for few example classes, the most frequently misclassified class in the RGB, depth and RGB-D case. When an object class is confused with distinct classes in the individual modalities, like for "keyboard" and "calculator", the RGB-D modality can perform better. However, when an object class is confused with the same classes in both RGB and depth modalities, like for "pear" and "potato", the RGB-D modality can perform slightly worse than the single modalities. This highlights a weakness of the method that will be the subject of future investigations.

Table 3.2 shows the results on JHUIT-50 for the instance recognition task. For the individual modalities, ResNet-18 shows again a compelling performance. In the multi-modal RGB-D classification, our method clearly outperforms all the competing approaches with a margin of 2% on the best existing method, DECO [13]. In summary,

Table 3.1: Accuracy (%) of several methods for object recognition on RGB-D Object Dataset [6]. **Bold**: highest result; *italic:* other considerable results.

RGB-D Object Dataset			
Method	**RGB**	**Depth**	**RGB-D**
LMMMDL [19]	74.6±2.9	75.5.8±2.7	86.9±2.6
FusionNet [10]	84.1±2.7	83.8±2.7	91.3±1.4
CNN w/ FV [14]	**90.8**±1.6	81.8±2.4	*93.8*±0.9
DepthNet [17]	88.4±1.8	83.8±2.0	92.2±1.3
CIMDL [20]	87.3±1.6	*84.2*±1.7	92.4±1.8
FusionNet enhenced [18]	*89.5*±1.9	*84.5*±2.9	*93.5*±1.1
DECO [13]	*89.5*±1.6	84.0±2.3	*93.6*±0.9
RCFusion	*89.6*±2.2	**85.9**±2.7	**94.4**±1.4

Table 3.2: Accuracy (%) of several methods for object recognition on JHUIT-50 [81]. **Bold**: highest result; *italic*: other considerable results.

JHUIT-50			
Method	**RGB**	**Depth**	**RGB-D**
DepthNet [17]	88.0	55.0	90.3
FusionNet enhanced [18]	*94.7*	56.0	*95.3*
DECO [13]	*94.7*	**61.8**	*95.7*
RCFusion	**95.1**	*59.8*	**97.7**

RCFusion establishes new state-of-the-art results on the two most popular datasets for RGB-D object recognition, demonstrating its robustness against changes in the dataset and the task.

Is RCFusion able to cope with challenging cluttered and occluded scenes? To evaluate the performance of our method on more robotic-oriented data, we show experiments on OCID. This dataset has been recorded with the specific goal of creating highly cluttered and occluded object scenes (see figure 3.7). Since objects are presented in clutter rather than in isolation, using multiple modalities is useful to cope with ambiguous views, thus making OCID particularly relevant to evaluate algorithms for RGB-D object recognition. In addition, its small training set of 2,428 cropped images represents an additional challenge. Table 3.4 shows the results on OCID for the instance recognition task. As well as our method, we also report the results of DECO, that showed competitive performance on RBG-D Object Dataset and JHUIT-50. The results on the single modalities show that the depth data alone are not very informative for this task, with a gap of 50% with respect to the RGB modality. Nevertheless, our method leverages both modalities and obtains an improvement of 6.1% in accuracy with respect to the

Table 3.3: Most frequently misclassified classes in RGB, depth and RGB-D for selected reference classes.

	Misclassification cases		
Reference class	**RGB**	**Depth**	**RGB-D**
calculator	keyboard	hand towel	hand towel
keyboard	calculator	binder	calculator
pear	apple	apple	apple
potato	lime	lime	lime

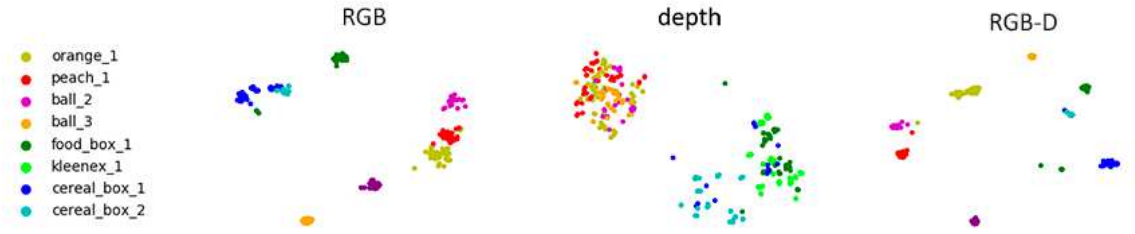


Figure 3.6: t-SNE visualization of the final features obtained for RGB, depth and RGB-D modalities.

RGB modality alone. On the contrary, DECO reveals its limits and maintains the same performance of the RGB modality even in the multi-modal case. This result is due to the simple strategy used in DECO for the multi-modal fusion: the final prediction is made by selecting the class with the maximum probability among the RGB and depth predictions. The more complex modality fusion of RCFusion thus translates into a non-trivial improvement of over 10% in accuracy with respect to DECO.

How does combining RGB and depth data reflect in the feature space? An interesting intuition of the effectiveness of RCFusion comes from the visualization of the features learned on the OCID dataset. Figure 3.6 represents the two dimensional t-SNE embedding of the final features of the different modalities. As expected, the t-SNE embedding of the depth features clusters together objects with similar shapes. For example, objects with near-spherical shapes like "orange_1", "pear_1" and "ball_2(/3)" are grouped together. The RGB modality provides more discriminative features, but similar pairs of objects, like ("orange_1"-"peach_1") and ("cereal_box_1"-"cereal_box_2") are very close to each other. Instead, the embedding of the RGB-D features neatly separates each object in discernible clusters.

How does each component of RCFusion contribute to the final performance? To observe the contribution of the two main elements of RCFusion, multi-level feature extraction and recurrent fusion, we alternatively remove these elements and compare the performance with the full version of the method. Table 3.4 presents the results of these variations on OCID. It can be noticed that using only the features from the last layer of the RGB-/Depth-CNN (RCFusion - res5) drops the performance by 2% in accuracy.

Figure 3.7: Examples of object crops from the Object Cluttered Indoor Dataset [89] with their instance label.

Table 3.4: Accuracy (%) of DECO [13] and variations of RCFusion on Object Clutter Indoor Dataset [89]. "RCFusion - res5" is the variation of RCFusion when only the features from the last residual layer (res5) are used for classification. "RCFusion - fc" is the variaton of RCFusion with a fully connected layer used instead of the recurrent neural network for combining the RGB and depth features. **Bold**: highest result; *italic*: other considerable results.

Object Clutter Indoor Dataset			
Method	**RGB**	**Depth**	**RGB-D**
DECO [13]	*80.7*	**36.8**	80.7
RCFusion	**85.5**	*35.0*	**91.6**
RCFusion - res5	-	-	*89.6*
RCFusion - fc	-	-	88.5

This confirms that explicitly using features from several levels of abstraction improves the multi-modal recognition compared to only using the final features of single modalities. Analogously, if instead of using the RNN we concatenate the multi-modal features from the projection blocks and fuse them with a fully connected layer, the performance drops by 3.1% in accuracy. This confirms that a more sophisticated fusion mechanism that effectively combines the modalities while retaining the crucial information from the different levels of abstraction is crucial for obtaining a final discriminative RGB-D feature.

3.3 Discussion

In this chapter, we have presented RCFusion: a multi-modal deep neural network for RGB-D object recognition. Our method uses two streams of convolutional networks to extract RGB and depth features from multiple levels of abstraction. These features are concatenated and sequentially fed to an RNN to obtain a compact RGB-D feature that is used by a softmax classifier for the final classification. We show the validity of our approach by outperforming the existing methods for RGB-D recognition on two standard benchmarks, RGB-D Object Dataset and JHUIT-50. We also stress

test RCFusion with some of the main challenges of robotic vision by evaluating it on OCID. In fact, not only does this dataset present highly cluttered and occluded scenes, but it also provides few training samples. Despite these challenges, RCFusion presents compelling results on OCID and marks the superiority of our multi-modal fusion mechanism. But it is not all sunshine and rainbows: the technology of current RGB-D cameras fail to provide reliable results when deployed in an unconstrained setup. This can undermine the use of RGB-D classifiers in real-world applications.

Overall, our results show that, when the depth information are reliable their addition to RGB data can be greatly beneficial for object recognition in robotics. In order to benefit from these advantage, one would need to collect and annotate RGB-D data for the specific application at hand. This can be very costly, especially when dealing with object categorization that needs a large amount of data to cover a variety of instances for each category.

In chapter 2, we have seen that DA can be a brilliant tool to exploit cheap data sources to obtain compelling results on real data, without the need for manual annotation. In the next chapter, we investigate how to effectively apply DA on RGB-D data.

Highlights:

- ➔ When deployed in an unconstrained setup, current RGB-D cameras can deliver unreliable and noisy depth data.
- ➔ When the data is reliable, integrating depth and RGB information can significantly boost recognition accuracy, especially in presence of occlusion and clutter.
- ➔ The strategy used to fuse the two modalities is crucial to improve over the single modalities.
- ➔ For classes where both modalities yield similar misclassification errors, the multi-modal prediction could reinforce this error instead of solving it.

Chapter 4

Unsupervised Domain Adaptation through Inter-modal rotation for RGB-D Object Recognition

The large amount of annotated data required to train CNNs can be very costly and represents one of the main bottlenecks for their deployments in robotics. An attractive workaround that requires no manual annotation consists in generating a large synthetic training set by rendering 3D object models with computer graphics software, such as Blender [101]. However, the difference between the synthetic (source) training data and the real (target) test data severely undermines the recognition performance of the network. Unsupervised Domain Adaptation (DA) is a field of research that accounts for the difference between source and target data by considering them as drawn from two different marginal distributions. DA approaches provide predictions on a set of target samples using only annotated source samples, with the unlabeled target samples available transductively. This field has flourished in the last decade and has produced numerous strategies to reduce the shift between the source and target distributions both at feature [52], [55] and at pixel level [57], [102]. However, existing DA strategies implicitly assume that the data come from a single modality. We claim that this assumption leads to sub-optimal results when dealing with multi-modal data since the natural inter-modal relations of the data are ignored.

In this chapter, we propose the first DA method tailored to RGB-D data. We define a multi-task learning problem that consists of training a CNN to solve a supervised main task and a self-supervised pretext (or auxiliary) task from pairs of RGB and depth images. The main task is the object recognition problem that we want to solve. The pretext task is an artificial problem created to encourage the network to generate domain-invariant features by learning geometric relations between the RGB and depth modalities: we rotate the RGB and depth image of a sample and ask the network to predict the relative rotation that re-aligns them (see figure 4.1). Due to its self-supervised nature, both source and target data can be used to train the model on the pretext task, while the supervision of the source data is used to train the model on the main task (see figure 4.2). To evaluate our method on object categorization and instance recognition, we define two benchmark datasets, each composed of a synthetic and a real part. For instance recognition, we render the HomeBrewedDB (HB) [103] models as source dataset and use the real RGB-D sequences of the same dataset as target dataset.

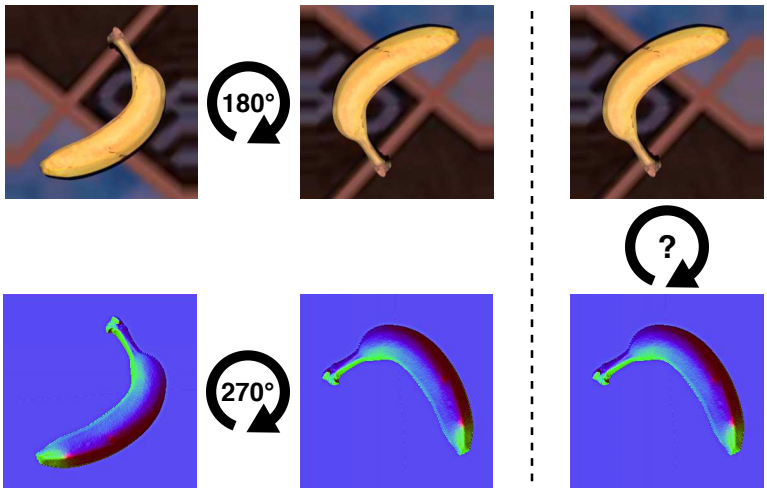


Figure 4.1: Q: "By how much should the RGB image (top) be rotated to align with the depth image (bottom)?" A: "90°". This question describes the self-supervised task of predicting the relative rotation between the RGB and depth image of a sample after they have been independently rotated. The depth is shown with surface normal colorization [18].

For object categorization, no existing dataset presents both synthetic and real data. We use the popular RGB-D Object Dataset (ROD) [77] for the real data and collect the synthetic counterpart ourselves. Therefore, we propose synROD: a dataset generated by collecting and rendering 3D object models from the same categories as ROD using publicly available Web resources. Extensive experiments on these datasets show that our newly defined pretext task effectively reduces the synthetic-to-real domain gap and outperforms existing DA approaches that do not leverage the inter-modal relations of RGB-D data.

In summary, the contributions of this chapter are:

- a novel multi-modal DA algorithm for RGB-D object recognition that reduces the domain gap by leveraging the relation between RGB and depth data,
- two benchmark datasets to evaluate RGB-D DA methods on object categorization and instance recognition, including the newly collected synROD, and
- quantitative and qualitative experiments that showcase the superior performance of our method compared to existing DA approaches.

The content of this chapter is based on the submitted paper

M. Loghmani, L. Robbiano, M. Planamente, K. Park, B. Caputo, and M. Vincze, **Unsupervised Domain Adaptation through Inter-modal Rotation for RGB-D Object Recognition**, Under submission at *Robotics and Automation Letters (RA-L)*.

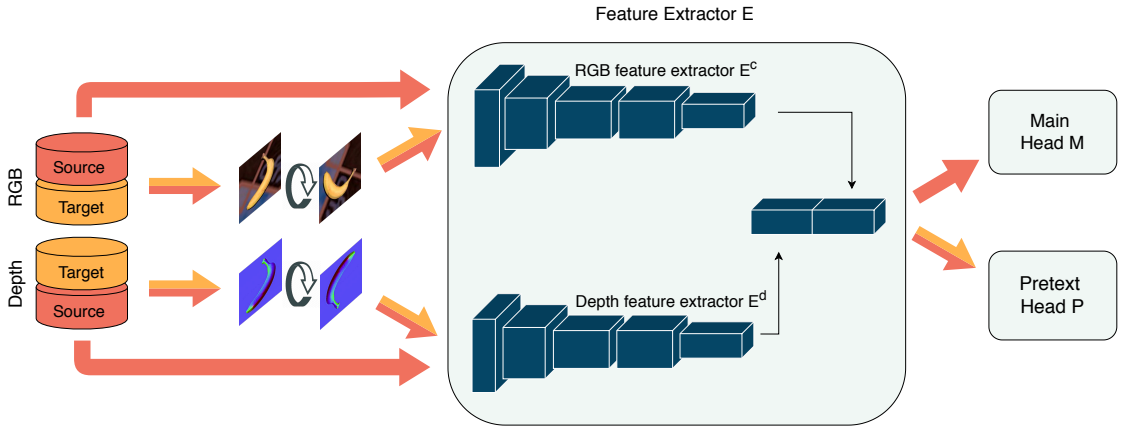


Figure 4.2: Overview of our method for RGB-D domain adaptation. We use a convolutional neural network (blue squares) that consists of a two-stream feature extractor E that flows into two network heads, the main head M and the pretext head P. M is trained for object recognition using the labeled source data (red arrow); P is trained with both source and target samples where the RGB and depth image are independently rotated before being fed to the network (orange+red arrow).

4.1 Dataset

In this section, we present synROD and the protocol followed for its creation. More specifically, section 4.1.1 describes the criteria used to define the scope of the dataset and collect the 3D object models from Web resources; section 4.1.2 illustrates the procedure used to render 2.5D scenes from the 3D object models. The dataset is publicly available at `https://www.acin.tuwien.ac.at/en/vision-for-robotics/software-tools/synthetic-to-real-rgbd-datasets/`.

4.1.1 Selecting 3D Object Models

RGB-D DA has not been explored in the literature yet, so there are no standard benchmark datasets to evaluate methods developed for this purpose. The main challenge of defining a dataset to evaluate DA methods is to identify two distinct sets of data that exhibit the same annotated classes but have been collected in different conditions. In particular, we are interested in the synthetic-to-real domain shift, where the source domain presents RGB-D synthetic data, while the target domain presents RGB-D real data. Existing 3D object datasets, such as ModelNet [104] and ShapeNet [105], do not have a corresponding real dataset that shares the same classes. In addition, the lack of texture for some models makes them unusable for our purpose where we are interested in both the shape (depth) and the texture (color) of the object. To overcome this problem, we collect a new synthetic dataset called synROD. We selected the object models for synROD in such a way that each one belongs to one of the 51 categories defined by ROD, arguably the most used RGB-D dataset in robotics for object categorization [10], [13], [18], [106]. We query the objects from the free catalogs

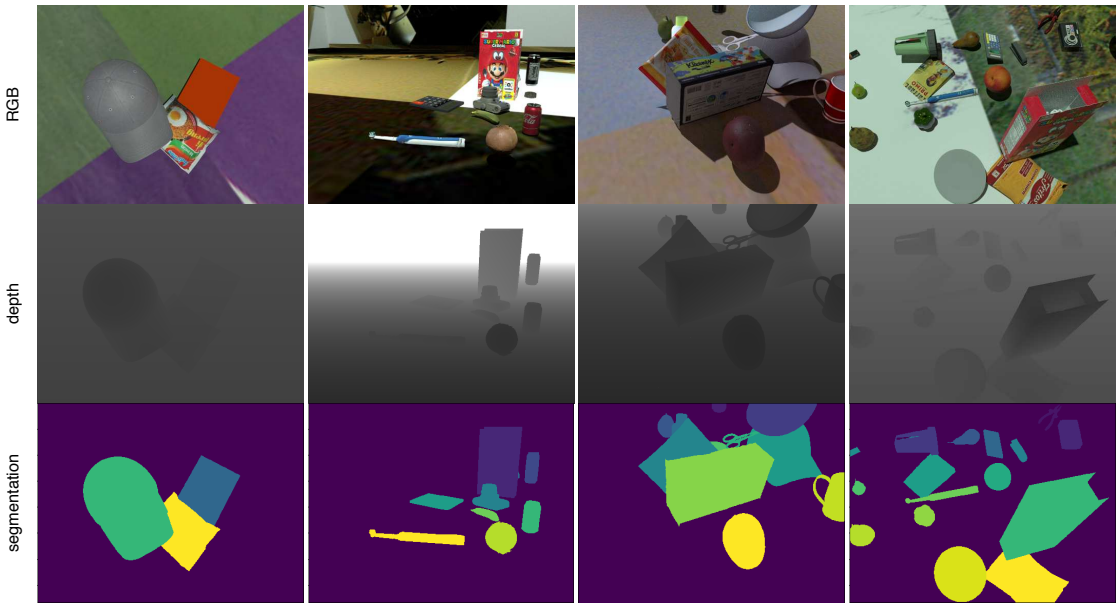


Figure 4.3: Examples of rendered scenes from synROD with increasing level of clutter from left to right. For each, we showcase the RGB, raw depth and segmentation mask image.

of public 3D model repositories, such as 3D Warehouse and Sketchfab, and only keep models that present texture information to be able to render the RGB modality in addition to the depth. All models are processed to harmonize the scale and canonical pose prior to the rendering stage. The final result of the selection stage is a set of 303 textured 3D models from the 51 object categories of ROD, for an average of about 6 models per category.

4.1.2 Rendering 2.5D scenes

We render 2.5D scenes using a ray-tracing engine in Blender to simulate photorealistic lighting. Each scene consists of a rendered view of a randomly selected subset of the models placed on a 1.2×1.2 meter virtual plane. The poses of the camera and the light source are sampled from an upper hemisphere of the plane with varying radius. To obtain natural and realistic object poses, each model is dropped on the virtual plane using a physics simulator. The number of objects in each scene varies from five to 20 to create different levels of clutter. To ensure a balanced dataset, we condition the selection of the models to insert in every scene to the number of past appearances. The background of the virtual space containing the objects is randomized by using images from the MS-COCO dataset [107]. We rendered approximately 30,000 RGB-D scenes with semantic annotation at pixel level (see figure 4.3).

4.2 Method

In this section, we present our method for RGB-D DA. More specifically, section 4.2.1 provides a high-level overview of the method, section 4.2.2 describes the details of the relative rotation task, section 4.2.3 and 4.2.4 specify the architecture and training/test protocol of the CNN.

4.2.1 Overview

Our goal is to train a neural network to predict the object class of the target data, using only labeled source data and unlabelled target data. We formulate our problem as a multi-task classification by training the network to solve a main supervised task and a pretext self-supervised task. The main task consists of using the supervision of the source data to learn to predict object labels. The pretext task consists of predicting the relative rotation between a pair of RGB and depth images that have been independently rotated. Since the ground truth for this simple pretext task can be generated automatically from the data, we can train the network to predict the relative rotation using both source and target data in a self-supervised fashion. Learning this inter-modal relation yields domain-invariant features and consequently improves the object class prediction on the target data without the need for direct supervision.

4.2.2 Pretext Task

Predicting image rotation is a simple yet effective pretext task to learn robust visual representations [27]–[29]. This self-supervised task consists in rotating a given image by a multiple of 90° and training a CNN to predict the rotation that has been applied. However, predicting the rotation of an individual image is only possible with datasets such as PACS [108] where the pose of the subject is coherent throughout the samples. For example, the giraffe images in PACS always represent the animal in an upright position. For datasets where the object appears in a variety of poses, predicting the image rotation is an ill-posed problem (see figure 4.4). To overcome this issue and adapt the task to RGB-D data, we define the task of predicting the relative rotation between the RGB image x^c and depth image x^d of an RGB-D sample. Let us denote with $rot90(x,i), i \in [0,3]$ the function that rotates clockwise a 2D image x by $i * 90°$. Given an RGB-D sample (x^c, x^d), we select $j, k \in [0,3]$ at random to compute $\tilde{x}^c = rot90(x^c,j)$ and $\tilde{x}^d = rot90(x^d,k)$, and indicate with z the one-hot encoded label indicating the relative rotation between them. More precisely, the relative rotation label is computed as $z = one\_hot((k - j) \ mod \ 4)$, where $one\_hot(.)$ is the function that generates the one-hot encoding and mod is the modulo operator. The pretext task consists of predicting z given $(\tilde{x}^c, \tilde{x}^d)$, or in other words: "how many times should the RGB image be rotated by 90° clockwise to align with the depth image?". Figure 4.5 shows all the possible combinations for which a pair of RGB and depth images can be rotated and their corresponding relative rotation.

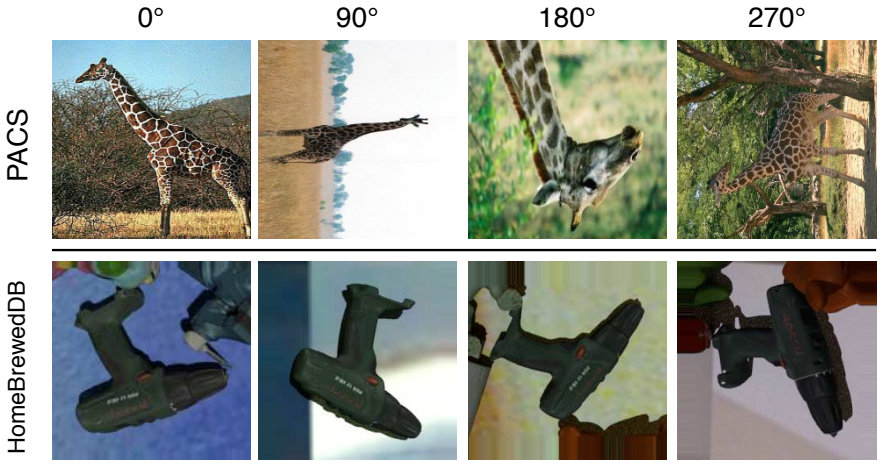


Figure 4.4: Examples images from PACS [108] (top row) and HomebrewedDB [103] (bottom row) that are rotated by 0°, 90°, 180°, and 270°. It is easy to guess the rotation of the PACS samples based on the background and our prior knowledge of the subject, while the same does not hold for the HomebrewedDB samples. This illustrates why predicting the image rotation by looking at each image individually, as in [27], is an ill-posed task.

4.2.3 Network architecture

Figure 4.2 shows the structure of the CNN we use for our method. A feature extractor E generates RGB-D features that are provided as input to both the main head M and the pretext head P. Each of these modules is a neural network defined with differentiable operation, so the whole network can be trained end-to-end using standard backpropagation.

Feature extractor Following the literature of RGB-D object recognition [13], [18], [109], we use a two-stream CNN with a late fusion approach to generate RGB-D features. In other words, two identical CNNs, E^c and E^d, are used to process the RGB and depth image, respectively. The outputs of these two networks are then concatenated along the channel dimension to compose the final RGB-D feature. In chapter 3, we have shown that this naive approach to extract RGB-D features can be improved using RCFusion. However, since the focus of this chapter is on the domain adaptation strategy, we evaluate all the methods (including ours) using this simple feature extractor. For our experiments, we define E^c and E^d as the ResNet-18 [98] architecture without the final fully connected and global average pooling layers.

Main head The network M solves a $\mathcal{C}$-way classification problem, where $\mathcal{C}$ indicates the number of object classes we want to predict. It is defined as *[gap, fc(1000), fc($\mathcal{C}$)]*, where *gap* indicates a global average pooling operation, *fc(n)* indicates a fully connected layer with n neurons. *fc(1000)* uses batch normalization and ReLU activation function, while *fc($\mathcal{C}$)* uses softmax activation function.

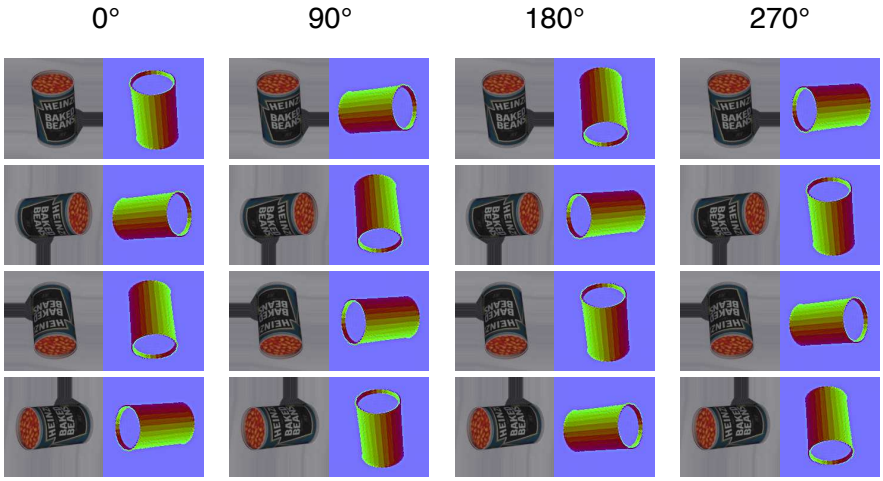


Figure 4.5: All the possible combinations of RGB and depth rotation for a given relative rotation $\{0°, 90°, 180°, 270°\}$.

Pretext head The network P solves the 4-way classification problem of predicting the rotation between the RGB and depth image. It is defined as *[conv(1 × 1,100), conv(3 × 3, 100), fc(100), fc(4)]*, where *conv(k × k, n)* indicates a 2D convolutional layer with kernel size $k \times k$ and n neurons. All convolutional and fully connected layers use batch normalization and ReLU activation function, except for *fc(4)* that uses softmax activation function. It is worth mentioning that, differently from M, we use convolutional layers in P to better preserve the spatial information. In section 4.3.4 we show that this leads to superior performance compared to adopting the architecture of M for both heads.

4.2.4 Optimization

Let us denote with $S = \{((x_i^{sc},x_i^{sd}),y_i^s)\}_{i=1}^{N_s}$ the set of labeled source data and $T = \{((x_i^{tc},x_i^{td})\}_{i=1}^{N_t}$ the set of unlabeled target data, where (x^{*c},x^{*d}) denotes the pair of RGB and depth images of a sample and y^s denotes the one-hot encoded object class label. From S and T, we can generate a transformed set of source and target data, $\widetilde{S} = \{((\tilde{x}_i^{sc},\tilde{x}_i^{sd}),z_i^s)\}_{i=1}^{\widetilde{N}_s}$ and $\widetilde{T} = \{((\tilde{x}_i^{tc},\tilde{x}_i^{td}),z_i^t)\}_{i=1}^{\widetilde{N}_t}$, that is used to define the relative rotation task. We train the CNN to minimize the objective function $\mathcal{L} = \mathcal{L}_m(y^s,\hat{y}^s) + \lambda_p\mathcal{L}_p(z^s,\hat{z}^s,z^t,\hat{z}^t)$, where $\mathcal{L}_m$ and $\mathcal{L}_p$ are respectively the cross-entropy loss of the main and pretext task, and λ_p is a weight to regulate the contribution of the corresponding pretext loss term. More precisely

$$\mathcal{L}_m = -\frac{1}{N_s}\sum_{i=1}^{N_s} y_i^s \cdot \log(\hat{y}_i^s), \tag{4.1}$$

$$\mathcal{L}_p = -\frac{1}{\widetilde{N}_s}\sum_{i=1}^{\widetilde{N}_s} z_i^s \cdot \log(\hat{z}_i^s) - \frac{1}{\widetilde{N}_t}\sum_{j=1}^{\widetilde{N}_t} z_j^t \cdot \log(\hat{z}_j^t), \tag{4.2}$$

where $\hat{y}^s = M(E(x^{sc},x^{sd}))$ and $\hat{z}^* = P(E(\tilde{x}^{*c},\tilde{x}^{*d}))$. At test time, the pretext head P is discarded and the predictions of the target data are computed as $\hat{y}^t = M(E(x^{ct},x^{dt}))$.

Algorithm 1 RGB-D Domain Adaptation

Require:
Labeled source dataset $S = \{((x_i^{sc},x_i^{sd}),y_i^s)\}_{i=1}^{N_s}$
Unlabeled target dataset $T = \{((x_i^{tc},x_i^{td})\}_{i=1}^{N_t}$
Ensure:
Object class prediction for the target data $\{\hat{y}_i^t\}_{i=1}^{N_t}$

procedure TRAINING(S,T)
Get transformed set $\widetilde{S} = \{((\tilde{x}_i^{sc},\tilde{x}_i^{sd}),z_i^s)\}_{i=1}^{\widetilde{N}_s}$
Get transformed set $\widetilde{T} = \{((\tilde{x}_i^{tc},\tilde{x}_i^{td}),z_i^t)\}_{i=1}^{\widetilde{N}_t}$
for each iteration **do**
Load mini-batch from S
Compute main loss $\mathcal{L}_m$
Load mini-batches from $\widetilde{S}$ and $\widetilde{T}$
Compute pretext loss $\mathcal{L}_p$
Update weights of M from $\nabla\mathcal{L}_m$
Update weights of P from $\nabla\mathcal{L}_p$
Update weights of E from $\nabla\mathcal{L}_m$ and $\nabla\mathcal{L}_p$
procedure TEST(T)
for each (x_i^{tc},x_i^{td}) in T **do**
Compute $\hat{y}_i^t = M(E(x_i^{ct},x_i^{dt}))$

The pseudo-code is presented in algorithm 1.

4.3 Experiments

In this section, we present the experimental protocol and the evaluation results of our method. More precisely, section 4.3.1 describes the adopted datasets, section 4.3.2 presents the baseline methods we compare against our method, section 4.3.3 presents the implementation details for training the CNN, and section 4.3.4 show quantitative and qualitative results on RGB-D DA.

4.3.1 Datasets

ROD & synROD: Since its release in 2011, ROD has become the main reference dataset for RGB-D object recognition in the robotics community. It contains 41,877 RGB-D images of 300 objects commonly found in house and office environments grouped in 51 categories. Each object is recorded on a turn-table with the RGB-D camera placed at approximately one meter distance at 30°, 45° and 60° angle above the horizon. As mentioned in section 4.1, synROD is a synthetic dataset created using object models from the same categories as ROD. To make the two datasets comparable, we randomly select and extract approximately 40,000 objects crops from synROD to match the dimensions

of ROD. In our experiments, we evaluate RGB-D DA methods by considering synROD as the synthetic source dataset and ROD as the real target dataset.

HomebrewedDB: It is a more recent dataset used for 6D pose estimation that features 17 toy, 8 household and 8 industry-relevant objects, for a total of 33 instances. HB provides high-quality object models reconstructed using a 3D scanner and 13 validation sequences. Each sequence contains three to eight objects on a large turntable and is recorded using two RGB-D cameras at 30° and 45° angle above the horizon. To re-purpose this dataset for the instance recognition problem, we extract the object crops from all the validation sequences, for a total of 22,935 RGB-D samples, and we refer to it as realHB. We create a synthetic version of this dataset by rendering the reconstructed object models using the same procedure used for synROD (see section 4.1), and we refer to it as synHB. In order to make the two datasets comparable, we randomly select and extract about 25,000 objects crops from synHB to match the dimensions of realHB. In our experiments, we evaluate RGB-D DA methods by considering synHB as the synthetic source dataset and realHB as the real target dataset.

4.3.2 Baseline methods

For our baselines, we consider four different DA methods: *MMD* [52], *DANN* [55], *Rotation* [29] and *AFN* [54]. The first two are arguably the most used and well-established DA methods; *AFN* is chosen as the current state of the art, while *Rotation* is the most relevant to our method.

MMD Long *et al.* [52] encourage the final layers of a neural network to generate domain-invariant features by minimizing the empirical maximum mean discrepancy, a metric that measures the discrepancy between two domain distributions.

DANN Ganin *et al.* [55] encourage the feature extractor to generate domain-invariant features using adversarial learning. A domain discriminator is trained to distinguish source from target samples, while the feature extractor is trained to fool the discriminator using a gradient reversal layer.

AFN Xu *et al.* [54] observe that, in the absence of an explicit adaptation, the target data present a significantly lower average feature norm that the source data. Therefore, the feature norm of the one-to-last layer of the network is iteratively increased for both domains to achieve adaptation.

Rotation Xu *et al.* [29] encourage the feature extractor to generate domain-invariant features by predicting the absolute rotation [27] of an RGB image as a pretext task.

Since the aforementioned methods are not originally designed for multi-modal data, we use two strategies to evaluate their performance on RGB-D DA. First, we adapt each modality separately until convergence and then we freeze the feature extractors and train a fully connected layer on the concatenation of the adapted features (RGB-D).

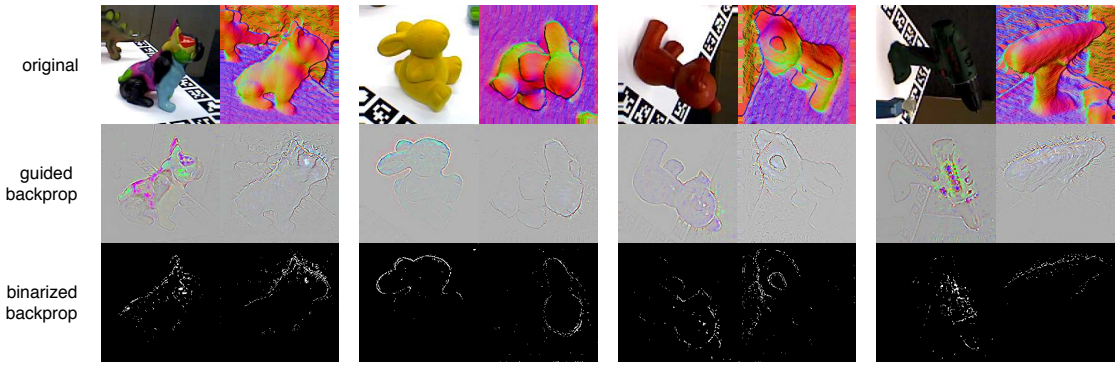


Figure 4.6: Visualization of the important pixels to predict the relative rotation. "original" indicates the RGB-D input of the network; "guided backprop" [111] indicates the saliency map of the input based on the last layer of the feature extractors E^c and E^d; "binary backprop" is the binarized version of "guided backprop" that highlights the peak values in white to facilitate visualization. The depth image is used with surfare normal colorization [18].

Second, similarly to our method, we apply them to the concatenation of the RGB and depth features generated by the feature extractor E, and train the network in an end-to-end fashion (RGB-D e2e). Finally, we also report the results on the single modalities to verify if it is beneficial to use multi-modal data.

4.3.3 Implementation details

The CNN is trained using SGD optimizer with momentum 0.9, learning rate 3×10^{-4}, batch size 64. Following [29], [54], [110], we include entropy-minimization with weight 0.1 as a DA-specific regularization, in addition to the more general weight decay 0.05 and dropout 0.5. The weights of the two ResNet-18, E^c and E^d, are initialized with values obtained by pre-training the networks on ImageNet [4], while the rest of the network is initialized with Xavier initialization. All the parameters of the network, including the pre-trained parameters, are updated during training. The input to the network is synchronized RGB and depth images pre-processed following the procedure in [18], where the depth information is colorized with surface normal encoding. This technique prevails as the best non-learned depth colorization method to effectively exploit networks pre-trained on ImageNet [13] and is widely adopted by state-of-the-art methods for RGB-D object recognition [106].

4.3.4 Results

Table 4.1 and 4.2 present the quantitative results of RGB-D DA on the two benchmark datasets, synROD→ROD and synHB→realHB, while figure 4.6 provides qualitative insights into the functioning of our method. This empirical evaluation allows us to answer important research questions.

Are standard DA methods effective on multi-modal data? Table 4.1 shows that applying a standard DA method on RGB-D data is not always effective. For example,

Table 4.1: Accuracy (%) of several methods for RGB-D domain adaptation on two synthetic-to-real shifts, synROD→ROD and synHB→realHB. Bold: highest result.

RGB-D Domain Adaptation			
Method		**synROD→ROD**	**synHB→realHB**
Source only	RGB	52.13	51.17
	depth	7.56	15.50
	RGB-D	50.57	49.71
	RGB-D e2e	47.70	49.45
DANN [55]	RGB	57.12	74.74
	depth	26.11	29.52
	RGB-D	59.09	75.23
	RGB-D e2e	59.51	74.95
MMD [52]	RGB	63.68	74.95
	depth	29.34	28.24
	RGB-D	62.10	77.96
	RGB-D e2e	62.57	77.26
Rotation [29]	RGB	63.21	84.46
	depth	6.70	5.62
	RGB-D	63.33	83.99
	RGB-D e2e	57.89	84.15
AFN [54]	RGB	64.63	84.04
	depth	30.72	31.67
	RGB-D	61.19	83.06
	RGB-D e2e	62.40	86.49
Ours		**66.68**	**87.28**

MMD and AFN perform worse when applied on the concatenation of RGB and depth features (RGB-D, RGB-D e2e) than when applied on the RGB features alone on synROD→ROD. These results are due to the fact that the depth modality is far less informative than the RGB for object recognition when compared in isolation. Therefore, in the absence of an effective strategy to exploit both modalities, the RGB-D case can provide lower accuracy than the RGB alone. Comparing the two strategies to apply the baseline methods on multi-modal data ("RGB-D" and "RGB-D e2e"), we noticed that no strategy clearly outperforms the other and the results are different depending on the method and the dataset used. It is also interesting to notice that AFN is not the best performing baseline on RGB-D data, despite being the considered the current state-of-the-art in DA.

Is the relative rotation an effective pretext task to perform RGB-D DA? Table 4.1

Table 4.2: Accuracy (%) of variations of our method for RGB-D domain adaptation on two synthetic-to-real shifts, synROD→ROD and synHB→realHB. Bold: highest result.

	Ablation Study		
Method	**synROD→ROD**	**synHB→realHB**	**avg. drop**
Target rotation	63.60	86.32	2.03
FC classifier	64.20	86.49	1.64
Ours	**66.68**	**87.28**	-

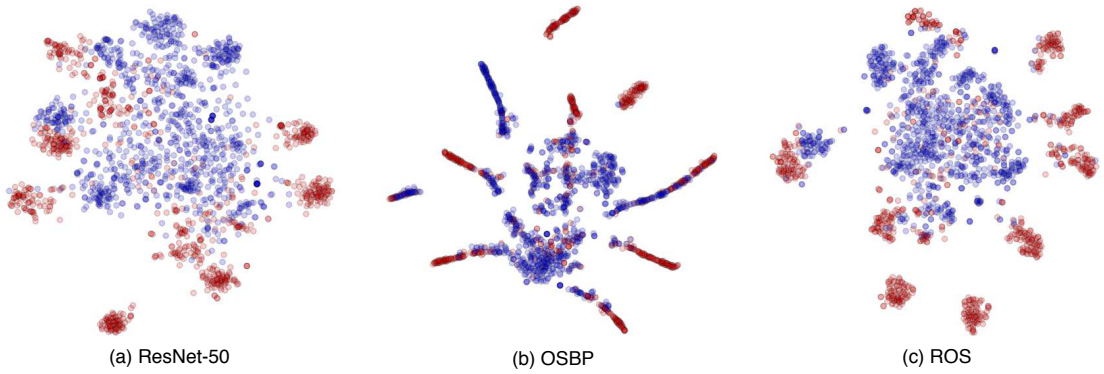


Figure 4.7: t-SNE [112] visualization of the HomebrewedDB [103] features extracted from the last hidden layer of the main head M. Red dots: source samples; blue dots: target samples. When adapting the two domains with our method (right), the two distributions align much better compared to the non-adapted case (left).

shows that predicting the relative rotation between the RGB and depth image is indeed an effective DA strategy, significantly improving over “Source only". This is also confirmed in figure 4.7 where the t-SNE [112] visualization of the features of the main head M show that our method effectively aligns the target and source distributions. More importantly, our method outperforms all considered baselines on both datasets. Compared to *Rotation*, that is the most related to our method, we have +3.35% improvement on synROD→ROD and +2.82% improvement on synHB→realHB.

How are the different components of our method affecting the final performance? We perform an ablation study to understand the impact of different components of our method on the overall performance. Following the example of [29], we investigate what happens when we only use the target domain to solve the pretext task, instead of using both domains. Table 4.2 shows that predicting the relative rotation of target samples is already sufficient to provide a significant improvement over “Source only", but it is not as effective as using both domains. The network learns more informative features when solving the pretext task with both domains due to the higher diversity in the

data. Finally, we evaluate the performance of our method when defining the pretext head P with the same architecture as M. Table 4.2 shows that this configuration leads to an average drop of -1.64% in accuracy. The results confirm that using convolutional layers instead of a pooling layer helps to better retain the spatial information necessary to predict the relative rotation.

What does the network learn to solve the relative rotation task? Figure 4.6 shows the most relevant pixels to predict the relative rotation for a few example samples in realHB. More precisely, we use guided backpropagation [111] to visualize which pixels of the RGB and depth input image maximally activate the last layer of the E^c and E^d to produce the correct prediction for the pretext task. First, we can notice that the most relevant pixels belong to the object, not the background or other elements in the image. This confirms that the network relies on the appearance of the object to make the prediction rather than learning "trivial" shortcuts [23]. Second, we can see that the network focuses on the same part of the object (e.g. the head of the bunny) in the RGB and depth image. This confirms that the prediction on the relative rotation is made by matching corresponding parts of the object in the two modalities.

4.4 Discussion

In this work, we propose the first method tailored to tackle the challenging problem of RGB-D DA. Our approach consists of training a network to solve the self-supervised task of predicting the relative rotation between the RGB and depth image, in addition to the main object recognition task. To evaluate the performance of our method, we define two synthetic-to-real benchmarks for instance recognition and object categorization, using the existing HB and a newly collected dataset called synROD. We empirically demonstrate that our self-supervised task successfully reduces the domain shift and outperforms all considered baselines, indicating that exploiting the inter-modal relations is key to perform DA on RGB-D data.

Being able to easily synthesize application-specific training data and make prediction on real data brings us a step closer to enable robot vision in the real world. However, when deployed in-the-wild, it is unrealistic to assume that the robot will only encounter objects from the training categories.

In chapter 5 and 6, we relax this closed set assumption and propose new methods to perform DA in the more realistic open set scenario.

Highlights:

- ➔ Using standard DA methods on RGB-D data can produce lower results than adapting the RGB modality alone.
- ➔ Solving a self-supervised task on both source and target domain effectively reduces the domain shift.
- ➔ Predicting the relative rotation between RGB and depth image helps the network learn a relationship between the two modalities.

Chapter 5

Positive-Unlabeled Learning for Open Set Domain Adaptation

In the standard framework of DA, labeled source and unlabeled target data are drawn from two different marginal distributions that cover the same set of categories. The setting is transductive, so the target is available at training time and is used for both adaptation and evaluation. Learning solutions for this task have flourished in the last decade but with numerous strategies proposed to align the distributions both at feature and at pixel level to close their domain shift. Nevertheless, they remain ineffective in the more realistic OSDA scenario [66], [67] where the source and target data contain both shared (known) and private (unknown) classes. Here, forcing adaptation without recognizing the outlier samples leads to negative transfer [113], adding confusion in the final known class recognition task. *Open set recognition* and *outlier detection* focus on cases in which the source classifier also needs to detect samples that belong to none of the training classes. A related line of research is that of *PU learning* [114] that deals with binary classification when the training data consists only of *positive* (P) and *unlabeled* (U) samples, where each unlabeled sample could be either positive or negative. Most PU formulations deal with labeled and the unlabeled data drawn randomly from the same marginal distribution.

In this chapter, we highlight for the first time the relation and complementarity of DA and PU learning for OSDA, showing how it is possible to get the best of both worlds. We cast OSDA in the theoretical framework of PU learning by considering the source samples as P and the target samples as U. Our DA solution exploits PU learning to detect unknown target samples and avoid negative transfer, while extending PU learning to the case of uneven data distributions. It is worth noting that, since OSDA is still in its infancy, we only consider standard RGB data rather than the RGB-D data used in chapter 3 and 4. This gives us the possibility to use standard benchmark datasets and avoid to further complicate an already arduous problem.

More precisely, the contributions of this chapter are:

- the *positive-unlabeled reconstruction encoding* (PURE) algorithm that trains an *autoencoder* (AE) to reconstruct the known samples and map the unknown samples to a semantically void vector,
- the integration of PURE with domain adversarial learning (OSDA-PURE), that extend PURE to OSDA,

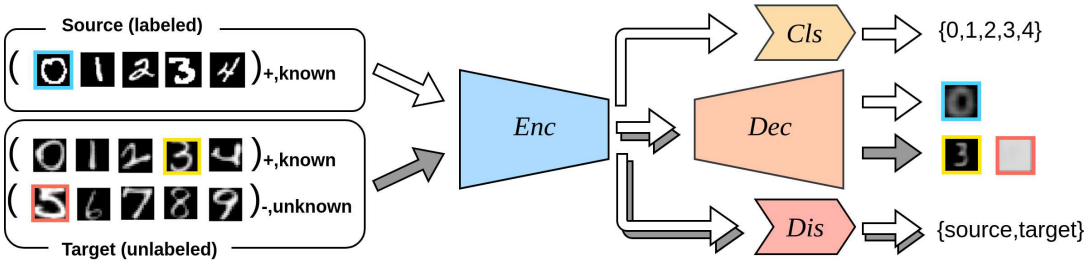


Figure 5.1: Schematic overview of our OSDA-PURE composed of an encoder *Enc*, a decoder *Dec*, a classifier *Cls* and a domain discriminator *Dis*. Note that the decoder reconstruction has a different effect on known and unknown target samples.

- a new evaluation metric for OSDA that penalizes large gaps in the recognition performance of known and unknown classes, and
- an extensive experimental analysis on the basis of our metric that shows the effectiveness of OSDA-PURE with respect to its competitors.

The content of this paper is based on the submitted paper

M. Loghmani, M. Vincze, and T. Tommasi, **Positive-Unlabeled Learning for Open Set Domain Adaptation**, Accepted for publication in *Pattern Recognition Letters (PRL)*.

5.1 Background

Problem Setting Let us consider a binary classification problem where our sample $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^d$ belongs to one of the two classes with labels $y \in \{-1, +1\}$. We define as $p(\boldsymbol{x},y)$ the underlying joint probability distribution and we indicate with $p(\boldsymbol{x})$ the marginal density. The respective class conditionals are $p_p(\boldsymbol{x}) = p(\boldsymbol{x}|y = +1)$ and $p_n(\boldsymbol{x}) = p(\boldsymbol{x}|y = -1)$, while $\pi_p = p(y = +1)$ and $\pi_n = p(y = -1) = 1 - \pi_p$ are the positive and negative class-prior probabilities.

In the standard setting for *positive-negative* (PN) learning, the data of the two classes are sampled independently from the respective marginals as $\mathcal{X}_p = \{\boldsymbol{x}_i^p\}_{i=1}^{N_p} \sim p_p(\boldsymbol{x})$ and $\mathcal{X}_n = \{\boldsymbol{x}_j^n\}_{j=1}^{N_n} \sim p_n(\boldsymbol{x})$ and the goal is to search for the optimal decision function f through *empirical risk minimization*. More precisely, if $f : \mathbb{R}^d \to \mathbb{R}$ is our decision function and we indicate with $l : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}$ the *loss function* that measures with $l(t,y)$ the error incurred by predicting the output t when the ground truth is y, then we can define the *risk* of f as

$$R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}[l(f(\boldsymbol{x}),y)] = \pi_p R_p^+(f) + \pi_n R_n^-(f) \ , \tag{5.1}$$

where $\mathbb{E}[\cdot]$ denotes the expectation operator, $R_p^+(f) = \mathbb{E}_{\boldsymbol{x}\sim p_p}[l(f(\boldsymbol{x}),+1)]$, and $R_n^-(f) = \mathbb{E}_{\boldsymbol{x}\sim p_n}[l(f(\boldsymbol{x}),-1)]$. The risk is empirically approximated by

$$\mathcal{L}(f) = \pi_p \widehat{R}_p^+(f) + \pi_n \widehat{R}_n^-(f) \ , \tag{5.2}$$

where $\widehat{R}_p^+(f) = (1/N_p)\sum_{i=1}^{N_p} l(f(\boldsymbol{x}_i^p), +1)$ and $\widehat{R}_n^-(f) = (1/N_n)\sum_{j=1}^{N_n} l(f(\boldsymbol{x}_j^n), -1)$. Finally, using θ to parametrize the function f, the final classifier is obtained by solving $\min_\theta \mathcal{L}(f_\theta)$.

PU Learning In the PU learning setting, the goal is to learn the binary classifier f only from positive and unlabeled data, where each unlabeled sample could be either positive or negative. Specifically, we consider the *case-control* scenario [115] where two sets of data are sampled independently as $\mathcal{X}_p = \{\boldsymbol{x}_i^p\}_{i=1}^{N_p} \sim p_p(\boldsymbol{x})$ and $\mathcal{X}_u = \{\boldsymbol{x}_j^u\}_{j=1}^{N_u} \sim p(\boldsymbol{x})$. Since $\mathcal{X}_n$ is unavailable, we need a new way to approximate $R_n^-(f)$ and estimate equation (5.1). [116] showed that starting from $\pi_n p_n(\boldsymbol{x}) = p(\boldsymbol{x}) - \pi_p p_p(\boldsymbol{x})$, we obtain $\pi_n R_n^-(f) = R_u^-(f) - \pi_p R_p^-(f)$ where $R_u^-(f) = \mathbb{E}_{\boldsymbol{x}\sim p}[l(f(\boldsymbol{x}), -1)]$, and $R_p^-(f) = \mathbb{E}_{\boldsymbol{x}\sim p_p}[l(f(\boldsymbol{x}), -1)]$. $R(f)$ can be approximated by

$$\mathcal{L}_{PU}(f) = \pi_p \widehat{R}_p^+(f) + \widehat{R}_u^-(f) - \pi_p \widehat{R}_p^-(f)\ , \tag{5.3}$$

where $\widehat{R}_p^-(f) = (1/N_p)\sum_{i=1}^{N_p} l(f(\boldsymbol{x}_i^p), -1)$ and $\widehat{R}_u^-(f) = (1/N_u)\sum_{j=1}^{N_u} l(f(\boldsymbol{x}_j^u), -1)$.

Non-negative PU Learning (nnPU) Since by definition $R(f) \geq 0 \ \ \forall f$, it should also hold that $\pi_n R_n^-(f) = R_u^-(f) - \pi_p R_p^-(f) \geq 0$. However, for the empirical estimate it might not be true that $\widehat{R}_u^-(f) - \pi_p \widehat{R}_p^-(f) \geq 0$, which can cause major overfitting problems when using flexible models, such as deep neural networks, to define f. A solution to this issue is presented by [117] through the introduction of a *non-negative risk estimator* for PU learning:

$$\mathcal{L}_{nnPU}(f) = \pi_p \widehat{R}_p^+(f) + \max\left\{0, \widehat{R}_u^-(f) - \pi_p \widehat{R}_p^-(f)\right\}\ . \tag{5.4}$$

It is worth noting that both equation (5.3) and (5.4) assume that the positive class-prior π_p is known. For the case-control scenario, strategies have been proposed to estimate π_p (e. g. [118]). In this chapter, we do not aim at obtaining a precise estimate of the class prior and simply set $\pi_p = 0.5$ throughout the experiments. The only exceptions are the experiments with selective bias (see figure 5.2), where we report results with different P/N ratio in the unlabeled data and set π_p to match this ratio.

5.2 Autoencoder-based classification loss

When the decision function f is modeled by a deep neural network, equation (5.4) is often instantiated with logarithmic loss as

$$\begin{aligned}\mathcal{L}_{LOGnnPU}(f) =& -\frac{\pi_p}{N_p}\sum_{i=1}^{N_p}\log(f(\boldsymbol{x}_i^p)) + \\ &+\max\left\{0, -\frac{1}{N_u}\sum_{j=1}^{N_u}\log(1-f(\boldsymbol{x}_j^u)) + \frac{\pi_p}{N_p}\sum_{i=1}^{N_p}\log(1-f(\boldsymbol{x}_i^p))\right\}.\end{aligned} \tag{5.5}$$

However, this choice produces unreliable predictions when the positive and the unlabeled samples belong to different domains. To alleviate this drawback, we need an alternative

discriminative loss that is also domain agnostic. With this aim, we propose to instantiate f as an AE, a neural network architecture composed of two parts: an encoder $Enc : \mathbb{R}^d \to \mathbb{R}^e$ that projects the input into the encoding space and a decoder $Dec : \mathbb{R}^e \to \mathbb{R}^d$ that re-projects the encoded data into the input space. The energy-based learning literature indicates that the AE can be used for discriminative purposes [119], [120]. In fact, an energy-based discriminator attributes low energy (low reconstruction error) to the regions near the data manifold and high energy (high reconstruction error) to other regions. This makes an AE particularly suitable for the PU setting where there is no direct supervision on the negative data, as evidenced by [121]. In addition, the DA literature shows how the self-supervised nature of AEs makes the learned representations resilient to the difference in data domains [59], [81].

We train the AE to correctly reconstruct the positive samples while mapping the negative samples to a semantically void vector. Formally, we define the loss of $\boldsymbol{x}$ belonging to the positive and negative class as

$$l(f(\boldsymbol{x}), +1) = |\boldsymbol{x} - Dec(Enc(\boldsymbol{x}))| \,, \tag{5.6}$$

$$l(f(\boldsymbol{x}), -1) = |\bar{\boldsymbol{x}} - Dec(Enc(\boldsymbol{x}))| \,, \tag{5.7}$$

where $|.|$ denotes the absolute value function and $\bar{\boldsymbol{x}} = \kappa \mathbf{1}$ is a uniform reference vector obtained by the product of a constant κ and a d-dimensional vector of ones $\mathbf{1}$. In practice, we found a good choice to set κ to the maximum value that the input can assume. For instance, in an image classification task, $\bar{\boldsymbol{x}}$ is defined as a white image. We refer to equation (5.6) and (5.7) respectively as positive and negative reconstruction losses and we use them to instantiate equation (5.4) as

$$\begin{aligned}\mathcal{L}_{AEnnPU}(f) &= \pi_p \widehat{R}_p^+(f) + \max\left\{0, \widehat{R}_u^-(f) - \pi_p \widehat{R}_p^-(f)\right\} \\ &= \frac{\pi_p}{N_p} \sum_{i=1}^{N_p} |\boldsymbol{x}_i^p - Dec(Enc(\boldsymbol{x}_i^p))| + \\ &+ \max\left\{0, \frac{1}{N_u} \sum_{j=1}^{N_u} |\bar{\boldsymbol{x}} - Dec(Enc(\boldsymbol{x}_j^u))| + - \frac{\pi_p}{N_p} \sum_{i=1}^{N_p} |\bar{\boldsymbol{x}} - Dec(Enc(\boldsymbol{x}_i^p))|\right\} .\end{aligned} \tag{5.8}$$

In the following we refer to the method minimizing this risk as **Positive and Unlabeled Reconstruction Encoding (PURE)**. At inference time, the classification output is determined with $y = sign(\tau - |\boldsymbol{x} - Dec(Enc(\boldsymbol{x}))|)$, where τ is a threshold we set. Since the goal of minimizing the loss function in equation (5.8) is to reconstruct the unlabeled positive samples and map the unlabeled negative samples to void vectors, the absolute error $|\boldsymbol{x} - Dec(Enc(\boldsymbol{x}))|$ should be lower for positive samples and higher for negative samples. Therefore, similarly to [122], we choose τ such that the top-π_p test samples with lower absolute error are classified as positive and vice versa. In section 5.4, we provide experimental evidence to validate PURE.

5.3 Open set domain adaptation as a PU problem

In the OSDA setting we have annotated samples $\{(\boldsymbol{x}_i^s, c_i^s)\}_{i=1}^{N_s}$ drawn from the source domain with marginal density $p_s(\boldsymbol{x})$ and unlabeled samples $\{\boldsymbol{x}_j^t\}_{j=1}^{N_t}$ from the target

domain with marginal density $p_t(\boldsymbol{x})$. The source domain is associated with a set of *known* classes $c^s \in \{1, \ldots, |\mathcal{C}_s|\}$ that are shared with the target domain $\mathcal{C}_s \subset \mathcal{C}_t$, but the target covers also a set $\mathcal{C}_{t\setminus s}$ of additional classes, which are considered *unknown*. As in closed set domain adaptation, it holds that $p_s \neq p_t$ and we further have that $p_s \neq p_t^{\mathcal{C}_s}$ where $p_t^{\mathcal{C}_s}$ denotes the distribution of the target domain belonging to the shared label space $\mathcal{C}_s$. Ultimately, the goal of OSDA algorithms is to learn a model using the annotation of the source data to assign the target samples to either one of the $|\mathcal{C}_s|$ shared classes or to the *unknown* class.

If we consider all the known classes as positive and the unknown classes as negative, we end up in a PU learning setting where the source data are the P set and the target data are the U set. However, since source and target belong to different domains, the *selected completely at random* (SCAR) assumption is not valid here. In addition, we are interested in further differentiating between the $|\mathcal{C}_s|$ known classes. In order to tackle these problems, we equip PURE with a multi-class classifier and a domain discriminator (see Fig. 5.1). While PURE provides a suitable starting point for learning domain-invariant features, the domain adversarial discriminator allows the explicit minimization of the distance between the source and target domain. More formally, we extend the architecture of PURE with two new branches starting from the encoder output: a discriminator Dis that is trained to solve the binary source vs target problem, whose gradient backpropagates with flipped sign as in [55] to encourage domain alignment, and a classifier Cls that is trained on the source samples to recognize the $|\mathcal{C}_s|$ known classes. The final objective function of OSDA-PURE is

$$\mathcal{L} = \alpha \mathcal{L}_{AEnnPU} - \beta \mathcal{L}_{Dis} + \gamma \mathcal{L}_{Cls} \ , \tag{5.9}$$

where α, β, and γ are hyper-parameters that weigh each loss term in the overall objective, and the losses are

$$\begin{aligned}
\mathcal{L}_{AEnnPU} = & \frac{\pi_s}{N_s} \sum_{i=1}^{N_s} |\boldsymbol{x}_i^s - Dec(Enc(\boldsymbol{x}_i^s))| + \\
& + \max \left\{ 0, \frac{1}{N_t} \sum_{j=1}^{N_t} |\bar{\boldsymbol{x}} - Dec(Enc(\boldsymbol{x}_j^t))| - \frac{\pi_s}{N_s} \sum_{i=1}^{N_s} |\bar{\boldsymbol{x}} - Dec(Enc(\boldsymbol{x}_i^s))| \right\} \ , \\
\mathcal{L}_{Dis} = & -\frac{1}{N_s} \sum_{i=1}^{N_s} \log(Dis(Enc(\boldsymbol{x}_i^s))) - \frac{1}{N_t} \sum_{j=1}^{N_t} \log(1 - Dis(Enc(\boldsymbol{x}_j^t))) \ , \\
\mathcal{L}_{Cls} = & -\frac{1}{N_s} \sum_{i=1}^{N_s} c_i^s \log(Cls(Enc(\boldsymbol{x}_i^s))) \ .
\end{aligned}$$

The network is trained end-to-end in a minimax optimization scheme to converge to a saddle point of the functional of equation (5.9), using stochastic gradient descent. We use θ to indicate the network parameters and subscripts to identify the different network components, thus formally we have:

$$\begin{aligned}
(\hat{\theta}_{Enc}, \hat{\theta}_{Dec}, \hat{\theta}_{Cls}) = & \underset{\theta_{Enc}, \theta_{Dec}, \theta_{Cls}}{\arg\min} \ \mathcal{L}(\theta_{Enc}, \theta_{Dec}, \theta_{Cls}, \hat{\theta}_{Dis}) \\
\hat{\theta}_{Dis} = & \underset{\theta_{Dis}}{\arg\max} \ \mathcal{L}(\hat{\theta}_{Enc}, \hat{\theta}_{Dec}, \theta_{Dis}, \hat{\theta}_{Cls}) \ .
\end{aligned}$$

5.4 Experiments

5.4.1 Datasets

Digits Several datasets of digit images are commonly used to study DA, namely MNIST ($70k$ images of white digit on a black background, [123]), MNIST-M (variant of MNIST with background substituted with color photos, [55]), USPS ($7k$ images of white digit on a black background, [124]) and SVHN ($600k$ color images of real-world street view house numbers, [125]) with each dataset considered as a different domain. For our experiments, we always map all the samples to the highest resolution in the considered domain pair. The first five digits (0-4) define the positive/known/source set, and the remaining five digits (5-9) are unknown samples, with the unlabeled/target set covering all the 10 classes.

CIFAR-STL Both CIFAR-10 ([126]) and STL-10 ([127]) are standard object classification datasets with 10 classes. CIFAR-10 contains $50k$ training and $10k$ test samples, while STL-10 has $5k$ training and $8k$ test data. For our experiments, all the images were converted to 32×32 resolution. In the open-set scenario, we define the classes *airplane, automobile, bird, cat, deer* as known and *dog, frog/monkey, horse, ship, truck* as unknown.

Office-31 This dataset ([86]) provides three domains, namely Amazon (A), DSLR (D) and Webcam (W), containing images of objects from 31 categories. Amazon contains 2820 product images from the vendor website. DSLR (534 images) and Webcam (795 images) contain similar pictures of objects taken in an office environment, with Webcam having lower quality images than DSLR. We adopt the standard open set protocol ([66], [67]) where, in alphabetical order, the first 10 classes (1-10) are shared classes, and the last 10 classes (21-31) are unknowns in the target domain.

5.4.2 Implementation details

The hyper-parameter described in the paragraphs below are selected by following [67] and focusing on learning rate and batch size in the intervals $\{10^{-2}, 10^{-4}\}$ and $\{8, 128\}$, respectively.

Digits For the network architecture used in the digits recognition experiments, we follow [55] to model *Enc*, while *Dec* is its mirrored copy. *Cls* is composed of two fully connected layers (fc) with (100, C_s) units, while *Dis* is composed of two fc with (100, 1) units. The final layer of both *Cls* and *Dis* is always followed by a *softmax* activation function while all the remaining inner layers of the network use *ReLU*. As it is standard practice, we use a higher capacity architecture for the experiments on SVHN to deal with the larger variability of real-world data [55]. We model *Enc* on the architecture proposed in [67] and use a mirrored architecture for *Dec*. Here *Cls* is composed of three fc (100, 100, C_s), while *Dis* is composed of two fc (1024, 1) and the inner layers use relu and batch normalization. For the experiments in Sec.5, Fig. 2 and 3, nnPU is

implemented by substituting *Dec* with a fc layer for binary classification. We train all our models using Adam optimizer with batch size 16, learning rate 0.0001, 50 epochs and Xavier initialization. The weights of the loss terms in the final objective function are set to $\alpha = 1.0$, $\beta = 1.0$, and $\gamma = 2.0$ for all the experiments. To compensate for the small training set of USPS, we augment the data five times with standard techniques such as scaling, translation and rotation.

CIFAR-STL We define a simple *Enc* with two convolutional blocks followed by two fully connected layers. More specifically, the two convolutional blocks contain three layers of 3×3 convolutions each with 32 and 64 filters each. The *Dec* is a mirrored version of *Enc*. *Cls* is composed of two fully connected layers (fc) with (100,C_s) units, while *Dis* is composed of two fc with (100,1) units. We use group normalization after each convolutional layer and leaky relu after all internal layers. We train our models using Adam optimizer with batch size 128, learning rate 0.001, 50 epochs and Xavier initialization. The weights of the loss terms in the final objective function are set to $\alpha = 1.0$, $\beta = 1.0$, and $\gamma = 2.0$ for all the experiments. At training time, we take a random crop of size 28×28 for each input image, while at test time we evaluate on the central crop of the same size.

Office-31 As already mentioned in the main paper, for the experiments on this dataset we defined a *truncated autoencoder* where the *Dec* is composed by one single fc with 4096 units. *Dis* is composed of three fc (1024, 1024, 1 unit), while *Cls* is composed of a single fc with C_s units. All the inner layers of the network use relu and batch normalization. We train our models with batch size 64, learning rate 0.003, 200 epochs and Xavier initialization for the parameters that are not pre-trained. The pre-trained weight of all the encoder layers between the input and the reconstructed hidden layer is not updated and, for the remaining part of the encoder, we use a learning rate ten times smaller than for the rest of the network. The loss weights of the objective function are set to $\alpha = 1.0$, $\beta = 0.1$, and $\gamma = 2.0$.

5.4.3 Open set metrics

The usual metrics adopted to evaluate OSDA are the average class accuracy over the known classes OS^*, and the accuracy of the unknown class *UNK*. They are generally combined to define $OS = \frac{|\mathcal{C}_s|}{|\mathcal{C}_s|+1} \times OS^* + \frac{1}{|\mathcal{C}_s|+1} \times UNK$ as a measure of the overall performance. However, we argue that treating the unknown as an additional class does not provide an appropriate metric. As an example, let us consider an algorithm that is not designed to deal with unknown classes (UNK=0.0%) but has perfect accuracy over 10 known classes (OS^*=100.0%). Although this algorithm is not suitable for open set scenarios, it presents a high score of OS=90.9%. With increasing number of known classes, this effect becomes even more acute, making the role of *UNK* negligible. For this reason, we propose a new metric defined as the harmonic mean of OS^* and *UNK*, $HOS = 2\frac{OS^* \times UNK}{OS^* + UNK}$. Differently from *OS*, *HOS* provides a high score only if the algorithm performs well both on known and on unknown samples, independently of $|\mathcal{C}_s|$. Using a harmonic mean instead of a simple average penalizes large gaps between OS^* and *UNK*.

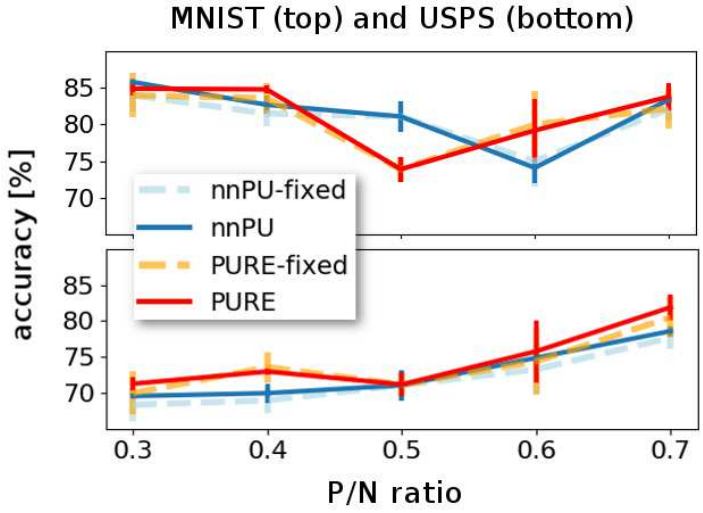


Figure 5.2: Mean accuracy and standard deviation over three runs in the PU setting with selection bias. The results show the performance of PURE and nnPU ([122]) at different P/N ratios with (-fixed) and without fixing the prior π_p.

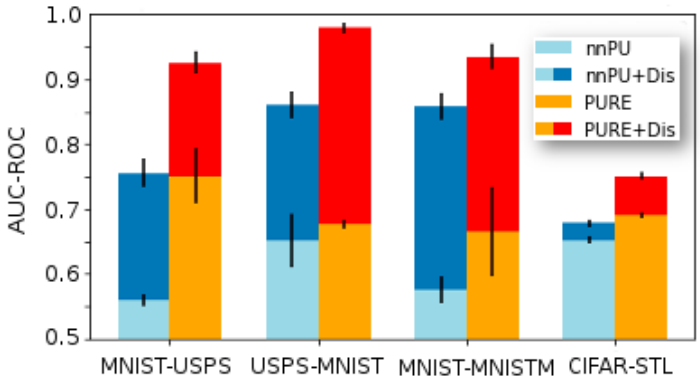


Figure 5.3: Mean AUC-ROC with standard deviation over three runs in the PU framework with domain shift. Results of PURE and nnPU [122] both without and with (+ *Dis*) the domain discriminator.

5.4.4 Results

In the first part of our experimental analysis we consider only the binary PU setting. We analyze the reconstruction loss of PURE (Eq. (5.8)) against the standard instantiation of nnPU with logarithmic loss (Eq. (5.5)), simply indicated in the following as nnPU. Moreover, we challenge nnPU and PURE with different domain shifts between the positive and the unlabeled data. In the second part, we focus on the multi-class OSDA scenario and evaluate the performance of OSDA-PURE. All the experiments are performed in a transductive setting and the implementation details are described in the supplementary materials.

Evaluating PURE *Is PURE a valid solution for the standard PU setting?* We start by comparing PURE with nnPU on MNIST. We evaluate their respective Receiver Operating Characteristic (ROC) curve when varying the sensitivity of the unknown detector by modifying the threshold τ to go from zero to complete recall. The area under the ROC curve (AUC-ROC) is similar for nnPU (0.995±0.001) and PURE

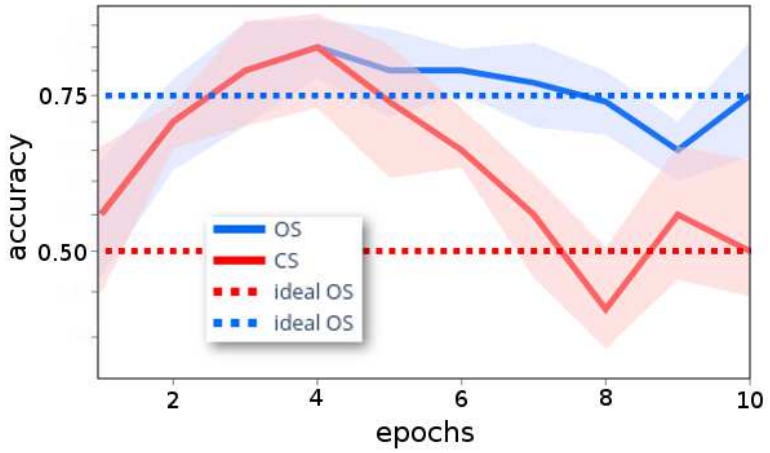


Figure 5.4: Accuracy of *Dis* at different learning epochs for the MNISTM-MNIST experiment in the closed set (CS) and open set (OS) scenario.

(0.996±0.001) showing that the reconstruction-based non-negative risk estimator of PURE is meaningful and reliable.

Are nnPU and PURE resilient to selection bias? A mild domain shift between positive labeled and unlabeled data can be due to selection bias with the P set containing *easier* positive samples than the U set. We reproduce the setting recently studied in [122] and we compare the accuracy of nnPU with that of PURE when considering different P/N ratios in the U data. For both nnPU and PURE, we choose τ such that the top-π_p test samples are classified as positive, as discussed in Sec. 5.2. Fig. 5.2 shows that, in presence of selection bias, both nnPU and PURE provide results firmly above chance with PURE slightly outperforming nnPU in the USPS case. It is noteworthy that both methods perform well even when the P/N ratio is skewed (e. g., P/N=0.3 and P/N=0.7). To test the robustness to the estimation of π_p, we set $\pi_p = 0.5$ and we maintain the same value of τ for all P/N ratios. The dashed lines in Fig.2 show that both nnPU and PURE maintain comparable accuracies to the case where the true prior is used.

Are nnPU and PURE resilient to cross-dataset domain shift? We investigate the challenging case where P and U belong to different domains by testing for classification across datasets (e. g. MNIST-USPS means that P is from MNIST and U is from USPS). Fig. 5.3 shows the AUC-ROC of both methods on four different domain shifts. In this setting, where the distance between the P and U distributions is larger than in the selection bias case, nnPU shows all its limits, with a performance close to chance (AUC-ROC=0.5) for MNIST-USPS and MNISTM-MNIST. PURE outperforms nnPU in all cases, with an advantage of up to +0.19 in the MNIST-USPS case. We also investigate the effects of adding a domain discriminator *Dis* to nnPU and PURE. Fig. 5.3 shows that both methods greatly benefit from *Dis* Still, PURE+*Dis* steadily outperforms nnPU+*Dis* in all considered cases. These results clearly indicate that both the AE and the adversarial domain discriminator independently contribute to the domain-invariance of the learned features.

Evaluating OSDA-PURE We compare our method against three baselines, all using Open Set SVM ([128]) as the final classifier: (OSVM) trains the classification model on

Table 5.1: Average class accuracy for known classes (OS^*), unknown classes (UNK), and both known and unknown classes measured with the OS and HOS metrics in the open set domain adaptation scenario for digits classification. **Bold**: highest result; *italic*: other considerable results.

Digits												
Method	**MNISTM-MNIST**				**SVHN-MNIST**				**USPS-MNIST**			
	OS	*OS**	*UNK*	***HOS***	*OS*	*OS**	*UNK*	***HOS***	*OS*	*OS**	*UNK*	***HOS***
OSVM	60.8	62.3	61.0	61.5	54.3	63.1	10.5	18.0	43.1	32.3	*97.5*	48.5
MMD+O	46.5	47.1	46.6	46.8	55.9	64.7	12.2	20.5	62.8	58.9	82.1	68.6
DANN+O	56.8	58.4	57.0	57.6	62.9	*75.3*	0.70	1.4	84.4	*92.4*	0.9	1.8
OSBP	*91.5*	**94.7**	*75.5*	*84.0*	*63.0*	59.1	*82.5*	*68.9*	*92.3*	91.2	**97.8**	*94.4*
STA	72.3	85.8	5.5	10.3	**76.9**	**75.4**	**84.4**	**79.6**	92.2	91.3	96.7	93.9
OSDA-PURE	**92.5**	*93.9*	**85.5**	**89.5**	61.9	59.8	72.4	65.5	**97.2**	**97.2**	97.2	**97.2**

Method	**MNIST-USPS**				**avg.**			
	OS	*OS**	*UNK*	***HOS***	*OS*	*OS**	*UNK*	***HOS***
OSVM	79.8	77.9	*89.0*	83.1	59.1	57.7	65.7	61.4
MMD+O	80.0	79.8	81.0	80.4	68.0	68.8	58.4	63.2
DANN+O	33.8	40.5	44.3	42.3	60.4	69.4	15.3	25.1
OSBP	*92.1*	**94.9**	78.1	85.7	*84.7*	85.0	*83.5*	*83.2*
STA	**93.0**	**94.9**	83.5	*88.8*	83.6	**86.9**	67.5	68.2
OSDA-PURE	91.6	*92.0*	**89.3**	**90.6**	**85.8**	*85.7*	**86.1**	**85.7**

CNN-generated features, the other two use features already adapted across domains by CNN models including Maximum Mean Discrepancy minimization (MMD+O) and adversarial domain discrimination (DANN+O). We also benchmark against the state-of-the-art methods[1].

How does OSDA-PURE perform on standard benchmark datasets? Following [67], we test our OSDA-PURE both on digits recognition and on object classification. For all experiments, we report OS, OS^*, UNK, and HOS, focusing on this last metric as a measure of overall performance. The top part of Table 5.1 presents the results on the digits datasets, also including the MNISTM-MNIST case that was not considered in [67] and for which we ran both OSBP and STA by using the code provided by the authors. OSDA-PURE outperforms the competing methods in three out of four tasks and presents the highest average HOS. In the SVHN-MNIST case, STA firmly outperforms all other methods. However, STA achieves poor results on MNISTM-MNIST, highlighting an instability in the performance that is present also in the object classification tasks. The bottom part of Table 5.1 presents the results of object classification on Office-31 and the (CIFAR, STL) pair. Office-31 has some well-known issue: unbalanced class statistics, noisy labels and very few samples per domain ([130]). Since it is a landmark dataset for DA, we still provide experiments on this dataset, but focusing only on the A-D and A-W pairs because A in the only domain with at least $1k$ samples. All the reported

[1] [129] report the OS metric, not OS^*, and no public code is available. This prevents a fair comparison since it is not possible to disentangle the contribution of the known and unknown class on the results.

Table 5.2: Average class accuracy for known classes (OS^*), unknown classes (UNK), and both known and unknown classes measured with the OS and HOS metrics in the open set domain adaptation scenario for object classification. **Bold**: highest result; *italic*: other considerable results.

Object Classification												
Method	**Office-31 A-D**				**Office-31 A-W**				**CIFAR-STL**			
	OS	*OS**	*UNK*	***HOS***	*OS*	*OS**	*UNK*	***HOS***	*OS*	*OS**	*UNK*	***HOS***
OSVM	59.6	59.1	64.6	61.7	57.1	55.0	78.1	64.5	46.7	45.3	53.5	49.1
MMD+O	47.8	44.3	*82.8*	57.7	41.5	36.2	*94.5*	52.3	45.2	43.5	53.6	48.0
DANN+O	40.8	35.6	**92.8**	51.5	31.0	24.3	**98.0**	38.9	44.5	43.2	50.7	46.7
OSBP	76.6	*76.4*	78.6	**77.5**	74.9	74.3	80.9	*77.5*	36.1	27.6	**78.8**	40.9
STA	**76.7**	**81.3**	30.7	44.6	**80.7**	**87.4**	13.7	23.3	*66.2*	*63.8*	*78.1*	**70.2**
OSDA-PURE	68.9	70.0	57.9	63.4	*80.3*	*80.8*	75.3	**78.0**	**69.9**	**68.6**	72.4	**70.4**
OSDA-PURE + init	*74.0*	75.0	64.0	*69.1*	79.7	80.4	72.7	76.4	-	-	-	-

Method	**STL-CIFAR**				**avg.**			
	OS	*OS**	*UNK*	***HOS***	*OS*	*OS**	*UNK*	***HOS***
OSVM	24.1	19.3	48.4	27.6	46.9	44.7	61.2	50.7
MMD+O	24.9	20.5	46.5	28.5	39.9	36.1	69.4	47.3
DANN+O	30.3	26.4	49.4	34.5	36.7	32.5	*72.7*	42.9
OSBP	21.2	6.2	**96.1**	11.6	52.2	46.1	**83.6**	*51.9*
STA	**55.0**	**52.7**	*66.3*	**58.7**	**69.7**	**71.3**	47.2	49.3
OSDA-PURE	*52.4*	*51.9*	54.8	*53.3*	*67.9*	*67.8*	65.1	**66.8**
OSDA-PURE + init	-	-	-	-	-	-	-	-

results on Office-31 are obtained using as backbone network AlexNet pre-trained on ImageNet ([3]). For OSDA-PURE, training from scratch a decoder that is a mirrored version of AlexNet (~ 60 million parameters) would not be feasible from the limited amount of samples of this dataset. Thus we defined a *truncated autoencoder* with *Dec* composed of a single fully connected (fc) layer that is trained to reconstruct the input to the last fc layer of *Enc*. By following [58], we also re-ran the OSDA-PURE experiments initializing the decoder with the transpose weights of the corresponding encoder layers (OSDA-PURE + init). The results on the object classification cases show that OSDA-PURE is the only method that maintains a good performance across all tasks. In fact, OSBP performs well on A-D and A-W and poorly on CIFAR-STL and STL-CIFAR, while STA presents the opposite behavior. It is worth having a closer look at the performance of STA in the Office-31 cases. If we focus only on *OS*, STA shows the best results. However, this metric masks its poor performance on unknown samples that is well represented by *HOS*.

Is there an internal equilibrium between PURE and the domain discriminator? During the learning process, our PU reconstruction loss moves apart target features from source features, while the domain discriminator aligns source and target distributions. These two antagonistic forces actually collaborate to isolate the unknown target samples while reducing the domain shift among the shared classes. Fig. 5.4 shows the accuracy of *Dis* on the target samples for MNISTM-MNIST. The desired condition in the closed set (CS) scenario is complete confusion across domains with accuracy 0.5, while in the OS scenario half of the classes (shared by source and target) should be confused (acc.

0.5) and half should be perfectly recognized as belonging to the target (accuracy 1.0) with an expected overall accuracy of $(0.5 \times 0.5 + 0.5 \times 1) = 0.75$. After an initial phase needed by *Enc* to learn and produce domain invariant features, the performance of *Dis* converges to the expected values. Note that *Dis* does not have any explicit information on the label difference between the two domains and reaches this performance through the adversarial game with the PURE loss. This confirms that the training objective of equation (5.9) is meaningful.

Is it possible to explain/interpret the result of the method? AEs have the side effect of visual transparency, which allows us to explore the inner working of OSDA-PURE also through a qualitative data visualization. Figure 5.5 shows the reconstruction effect on the target samples belonging to known and unknown classes in the digits experiments. We can see how the network is actually able to distinguish between the two cases, mapping the samples either to a meaningful digit or to an almost uniform image. Recall that we are not interested in a high-quality reconstruction and the reconstruction error is only used as a proxy for the known/unknown classification. As an example, the MNISTM-MNIST shift in Figure 5.5 shows that the reconstruction of the known sample is very blurry and mostly black, but is clearly different from the reconstruction of the unknown sample, thus allowing an accurate known/unknown prediction. Another interesting insight comes from the known examples of the MNIST-USPS and the SVHN-MNIST shift in the figure: instead of replicating the input, the AE encodes the input into prototypical examples in the feature space which is then decoded to an image with a recognizable digit. Besides the details of this qualitative evaluation, we can state that by resorting to the use of AE for the presented OSDA approach we can exploit the adaptive power of self-supervised reconstruction and further benefit of the transparency of the learning system leading to an easier explanation of the produced results.

5.5 Discussion

In this chapter, we propose a novel method to tackle the challenging problem of open set domain adaptation by casting it into the theoretical framework of PU learning. Our OSDA-PURE gets the best of both worlds: (a) it removes the SCAR assumption in PU learning by exploiting the self-supervised power of AEs and domain adversarial training, and (b) it isolates the unknown target samples reducing the effect of negative transfer through a novel reconstruction-based PU risk estimator. Experiments in the PU learning setting show that our AE-based risk estimator is clearly superior to the standard logarithmic instantiation when the P and U sets belong to different domains. Experiments in the OSDA setting show that OSDA-PURE: (i) is the only method that consistently improves over the OSVM-based baselines; (ii) outperforms all competitors in six out of nine cases; (iii) has the highest average performance in both the digit recognition cases and the object classification cases, by a large margin in the latter.

The results have also shown that there is space for improvement. Training a network on the sample reconstruction task may be difficult in the case of data scarcity due to the large amount of trainable parameters introduced by an auto-encoder architecture. In fact, an auto-encoder architecture has roughly double the number of parameters of

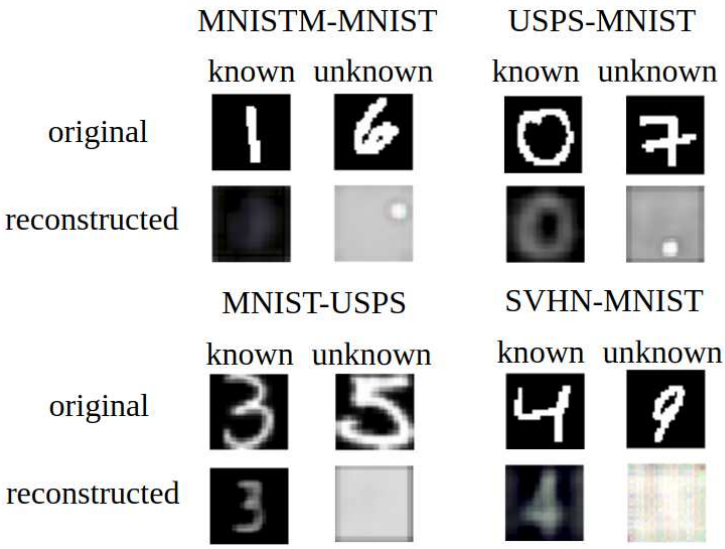


Figure 5.5: Qualitative analysis on the reconstructed target images from the PURE-OSDA digits experiments. Samples from known classes tend to be reconstructed to keep the original label, while samples from the unknown classes map to a mostly uniform image.

the backbone (encoder) that is based on. Training such a large count of parameters to deal with the nuances of real images without a sufficient amount of data leads to wild over-fitting. However, recent work have shown that other self-supervised tasks are suitable for cross-domain generalization [27], [60].

In the next chapter, we propose a novel OSDA method that relies on the self-supervised task of image rotation.

Highlights:

- ➔ Our autoencoder-based PU risk estimator is more resilient to domain shifts than its standard counterparts.
- ➔ The combination of adversarial domain discriminator and PURE can be used tackle OSDA without the risk of negative transfer.
- ➔ HOS is a much needed metric for OSDA that represents an appropriate balance between the performance on known and on unknown classes.

Chapter 6

On the Effectiveness of Image Rotation for Open Set Domain Adaptation

Recent DA literature has shown that training a CNN to jointly solve an auxiliary self-supervised task together with the main supervised problem helps the network learn more robust cross-domain features [29], [60]. Similarly, the anomaly detection literature has shown that the output of a CNN trained with self-supervision can be used to discriminate between normal and anomalous data [28], [131].

In this chapter, we propose for the first time to fully use the inherent properties of self-supervision both for cross-domain robustness and for anomaly detection to solve OSDA. Encouraged by the results discussed in chapter 4, we formulate the relative rotation task for single-modality RGB images. This is obtained by predicting the rotation between the original image (used as an anchor) and the rotated image. This task is the heart and soul of our novel OSDA method called Rotation-based Open Set (ROS). The schematic in figure 6.1 shows that ROS consists of two stages: (i) we train a network to predict the relative rotation of the source classes and use the output of the network to label each target samples as either known or unknown; (ii) we train a network to classify each sample as either one of the known classes or unknown while we reduce the domain gap by learning to predict the relative rotation on both source and target samples. To show that we can overcome the drawbacks of sample reconstruction from the previous chapter, we present an extensive evaluation based on two popular benchmark datasets, *office-31* and *office-home*. In fact, these datasets contain real images, but have only a few samples per class. To promote reproducibility in the field, we re-run the code of the existing OSDA methods and compare it to ROS using our HOS metric (presented in section 5.4.3). The results of the evaluation highlight that (a) there is a gap between the reproduced and reported performance of the existing methods, and (b) ROS defines the new state of the art on the considered benchmark datasets.

In summary, the contributions of this chapter are:

- a novel OSDA method that exploits rotation classification to tackle both known/unknown target separation and domain alignment,
- a reproducibility study on existing OSDA methods that highlights the urgent need for more attention on this subject, and

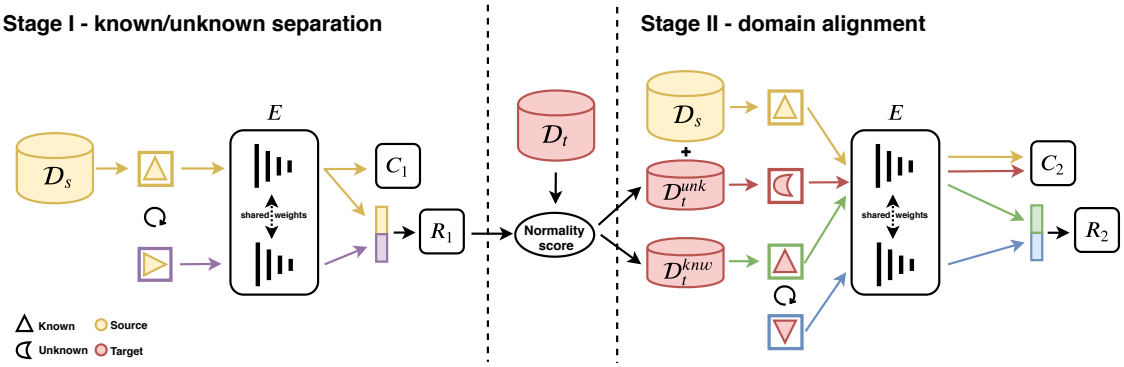


Figure 6.1: Schematic illustration of the proposed method *rotation-based open set* (ROS). Stage I: the source dataset $\mathcal{D}_s$ is used to train the encoder E, the semantic classifier C_1, and the multi-rotation classifier R_1 to perform known/unknown separation. C_1 is trained using the features of the original image, while R_1 is trained using the stacked features of the original and rotated image. After convergence, the prediction of R_1 on the target dataset $\mathcal{D}_t$ is used to generate a normality score that defines how the target samples are split into a known target dataset $\mathcal{D}_t^{knw}$ and an unknown target dataset $\mathcal{D}_t^{unk}$. Stage II: E, the semantic+unknown classifier C_2 and the rotation classifier R_2 are trained to align the source and target distributions and to recognize the known classes while rejecting the unknowns. C_2 is trained using the original images from $\mathcal{D}_s$ and $\mathcal{D}_t^{unk}$, while R_2 is trained using the stacked features of the original and rotated known target samples

- extensive experiments that define ROS as the new state of the art on two popular OSDA benchmark datasets.

The content of this paper is based on the submitted paper

S. Bucci[1], M. Loghmani[1], and T. Tommasi, **On the Effectiveness of Image Rotation for Open Set Domain Adaptation**, Under submission at *European Conference on Computer Vision (ECCV)*.

6.1 Method

6.1.1 Problem formulation

Let us denote with $\mathcal{D}_s = \{(\boldsymbol{x}_j^s, y_j^s)\}_{j=1}^{N_s} \sim p_s$ the labeled source dataset drawn from distribution p_s and $\mathcal{D}_t = \{\boldsymbol{x}_j^t\}_{j=1}^{N_t} \sim p_t$ the unlabeled target dataset drawn from distribution p_t. In OSDA, the source domain is associated with a set of *known* classes $y^s \in \{1, \dots, |\mathcal{C}_s|\}$ that are shared with the target domain $\mathcal{C}_s \subset \mathcal{C}_t$, but the target covers also a set $\mathcal{C}_{t\setminus s}$ of additional classes, which are considered *unknown*. As in CSDA, it holds that $p_s \neq p_t$ and we further have that $p_s \neq p_t^{\mathcal{C}_s}$, where $p_t^{\mathcal{C}_s}$ denotes the distribution of the target domain belonging to the shared label space $\mathcal{C}_s$. Therefore, in OSDA we

[1]equal contributions

face both a domain gap ($p_s \neq p_t^{\mathcal{C}_s}$) and a category gap ($\mathcal{C}_s \neq \mathcal{C}_t$). OSDA approaches aim at assigning the target samples to either one of the $|\mathcal{C}_s|$ shared classes or to reject them as *unknown* using only annotated source samples, with the unlabeled target samples available transductively. An important measure characterizing a given OSDA problem is the *openness* that relates the size of the source and target class set. For a dataset pair $(\mathcal{D}_s, \mathcal{D}_t)$, following the definition of [49], the openness $\mathbb{O}$ is measured as $\mathbb{O} = 1 - \frac{|\mathcal{C}_s|}{|\mathcal{C}_t|}$. In CSDA $\mathbb{O} = 0$, while in OSDA $\mathbb{O} > 0$.

6.1.2 Overview

When designing a method for OSDA, we face two main challenges: *negative transfer* and *known/unknown separation*. Negative transfer occurs when the whole source and target distribution are forcefully matched, thus also the unknown target samples are mistakenly aligned with source data. To avoid this issue, cross-domain adaptation should focus only on the shared $\mathcal{C}_s$ classes, closing the gap between $p_t^{\mathcal{C}_s}$ and p_s. This leads to the challenge of known/unknown separation: recognizing each target sample as either belonging to one of the shared classes $\mathcal{C}_s$ (known) or to one of the target private classes $\mathcal{C}_{t\setminus s}$ (unknown). Following these observations, we structure our approach in two stages: (i) we separate the target samples into known and unknown, and (ii) we align the target samples predicted as known with the source samples (see figure 6.1 for a schematic overview). The first stage can be formulated as an anomaly detection problem where the unknown samples are considered as anomalies, while the second stage can be formulated as a CSDA problem between source and the known target distribution. Inspired by recent advances in anomaly detection and CSDA [28], [29], we propose to solve both stages using the power of self-supervision. More specifically, we use two variations of the rotation classification task to first compute a normality score for the known/unknown separation of the target samples and then to reduce the domain gap.

6.1.3 Rotation classification for open set domain adaptation

Let us denote with $rot90(\boldsymbol{x},i)$ the function that rotates clockwise a 2D image $\boldsymbol{x}$ by $i \times 90°$. Rotation classification is a self-supervised task that consists in rotating a given image x by a random $i \in [1,4]$ and use a CNN to predict i from the rotated image $\tilde{\boldsymbol{x}} = rot90(\boldsymbol{x},i)$. We indicate with $|r| = 4$ the cardinality of the label space for this classification task. In order to effectively apply rotation classification to OSDA, we introduce the following variations.

Relative rotation Consider the images in figure 6.2. Inferring by how much each image has been rotated without looking at its original (non-rotated) version is an ill-posed problem since the pens, as all the other object classes, are not presented with a coherent orientation in the dataset. On the other hand, looking at both original and rotated image to infer the relative rotation between them is well-defined. Following this logic, we modify the standard rotation classification task [27] by introducing the original image as an anchor and training the rotation classifier to predict the rotation given the

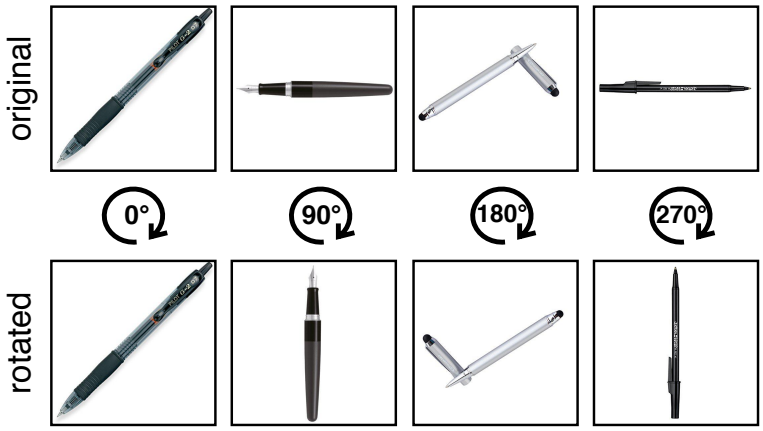


Figure 6.2: Are you able to infer the rotation degree of the rotated images without looking at the respective original one?

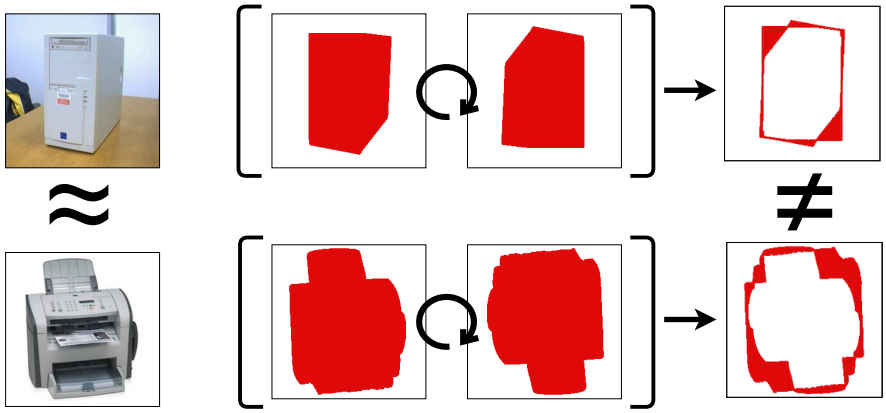


Figure 6.3: The objects on the left may be confused. The relative rotation guides the network to focus on discriminative shape information

concatenated feature of both original (anchor) and rotated image. This is essentially the same task as the relative rotation introduced in chapter 4, but for a single modality. As indicated by figure 6.3, the proposed relative rotation has the further effect of boosting the discriminative power of the learned features. It guides the network to focus more on specific shape details rather than on confusing texture information across different object classes.

Multi-rotation classification The standard setting of anomaly detection considers samples from one semantic category as the normal class and samples from other semantic categories as anomalies. Rotation classification has been successfully applied to this setting, but it suffers when including multiple semantic categories in the normal class [28]. This is the case when coping with the known/unknown separation of OSDA, where we have all the $|\mathcal{C}_s|$ semantic categories as known data. To overcome this problem, we propose a simple solution: we extend the rotation classification from a 4-class problem to a $(4 \times |\mathcal{C}_s|)$-class problem, where the set of classes represents the combination of semantic and rotation labels. For example, if we rotate an image of category $y^s = 2$

Algorithm 2 Compute normality score and generate $\mathcal{D}_t^{knw}$ & $\mathcal{D}_t^{unk}$

Require:
Trained networks E and R_1
Target dataset $\mathcal{D}_t = \{\boldsymbol{x}_j^t\}_{j=1}^{N_t}$
Ensure:
Known target dataset $\mathcal{D}_t^{knw} = \{\boldsymbol{x}_j^{t,knw}\}_{j=1}^{N_{t,knw}}$
Unknown target dataset $\mathcal{D}_t^{unk} = \{\boldsymbol{x}_j^{t,unk}\}_{j=1}^{N_{t,unk}}$

procedure GETROTATIONSCORE($\boldsymbol{z}$,i)
  $\mathbf{o} = zeros(|\mathcal{C}_s|)$ # vector of $|\mathcal{C}_s|$ zeros
  for each k **in** {1,...,$|\mathcal{C}_s|$} **do**
    $[\mathbf{o}]_k = [\boldsymbol{z}]_{k\times 4+i}$ # $[\mathbf{a}]_b$ indicated the b-th element of vector $\mathbf{a}$
  return o
procedure GETENTROPYSCORE($\boldsymbol{z}$)
  return $\boldsymbol{z} \cdot \log(\boldsymbol{z}) / \log(|\mathcal{C}_s|)$
procedure GETNORMALITYSCORE(E,R_1,$\mathcal{D}_t$)
  for each $\boldsymbol{x}_j^t$ **in** $\mathcal{D}_t$ **do**
    Initialize: $h = \{\}$, $\mathbf{o} = zeros(|\mathcal{C}_s|)$
    for each i **in** {1,...,4} **do**
      $\tilde{\boldsymbol{x}}_j = rot90(\boldsymbol{x}_j, i)$
      $\boldsymbol{z}_j = \text{softmax}\Big(R_1(E(\boldsymbol{x}_j)||E(\tilde{\boldsymbol{x}}_j))\Big)$
      $h \leftarrow getEntropyScore(\boldsymbol{z}_j)$
      o += $getRotationScore(\boldsymbol{z}_j, i)$ # element-wise sum of vectors
    $h = mean(h)$
    $o = max(\mathbf{o})$
    $\mathcal{N} \leftarrow \eta_j = max(o, 1-h)$
  return $\mathcal{N}$

procedure MAIN( )
  Initialize: $\mathcal{D}_t^{knw} = \{\}$, $\mathcal{D}_t^{unk} = \{\}$
  $\mathcal{A} = getNormalityScore(E, R_1, \mathcal{D}_t)$
  for each $(\boldsymbol{x}_j, \eta_j)$ **in** $(\mathcal{D}_t, \mathcal{N})$ **do**
    if $\eta_j \geq mean(\mathcal{N})$ **then**
      $\mathcal{D}_t^{knw} \leftarrow \boldsymbol{x}_j$
    else
      $\mathcal{D}_t^{unk} \leftarrow \boldsymbol{x}_j$

by $i = 3$, its label for the multi-rotation classification task is $z^s = (y^s \times 4) + i = 11$. In the following we use $\boldsymbol{y}$,$\boldsymbol{z}$ to express the one-hot encoding vectors for the class and multi-rotation labels.

6.1.4 Stage I: known/unknown separation

To distinguish between the known and unknown samples of $\mathcal{D}_t$, we train a CNN on the multi-rotation classification task using $\tilde{\mathcal{D}}_s = \{(\boldsymbol{x}_j^s, \tilde{\boldsymbol{x}}_j^s, z_j^s)\}_{j=1}^{4\times N_s}$. The network is composed of an encoder E and two heads: a multi-rotation classifier R_1 and a semantic label classifier C_1. The rotation prediction is computed on the stacked features of the original and rotated image produced by the encoder $\hat{\boldsymbol{z}}^s = \text{softmax}\Big(R_1([E(\boldsymbol{x}^s),E(\tilde{\boldsymbol{x}}^s)])\Big)$, while the semantic prediction is computed only from the original image features as $\hat{\boldsymbol{y}}^s = \text{softmax}\Big(C_1(E(\boldsymbol{x}^s)\Big)$. The network is trained to minimize the objective function $\mathcal{L}_1 = \mathcal{L}_{C_1} + \mathcal{L}_{R_1}$, where the semantic loss $\mathcal{L}_{C_1}$ is defined as a cross-entropy and the multi-rotation loss $\mathcal{L}_{R_1}$ combines cross-entropy and center loss [132]. More precisely,

$$\mathcal{L}_{C_1} = -\sum_{j\in\mathcal{D}_s} \hat{\boldsymbol{y}}_j^s \cdot \log(\boldsymbol{y}_j^s), \tag{6.1}$$

$$\mathcal{L}_{R_1} = \sum_{j\in\tilde{\mathcal{D}}_s} -\lambda_{1,1}\hat{\boldsymbol{z}}_j^s \cdot \log(\boldsymbol{z}_j^s) + \lambda_{1,2}||\hat{\boldsymbol{z}}_j^s - \gamma(\boldsymbol{z}_j^s)||_2^2, \tag{6.2}$$

where $||.||_2$ indicates the l_2-norm operator, and $\gamma(\boldsymbol{z}_j)$ indicates the centroid of the class associated with $\boldsymbol{z}_j$. By using the center loss we further encourage the network to minimize the intra-class variations while keeping the features of different classes separable to support the following use of the rotation classifier output as a metric to detect unknown category samples.

Once the training is complete, we use E and R_1 to compute the *normality score* $\mathcal{N} \in [0,1]$ for each target sample, with large $\mathcal{N}$ values indicating normal (known) samples and vice-versa. We start from the network prediction on all the relative rotation variants of a target sample $\hat{\boldsymbol{z}}_i{}^t = \text{softmax}\Big(R_1([E(\boldsymbol{x}^t),E(\tilde{\boldsymbol{x}}_i^t)])\Big)_i$ and their related entropy $H(\hat{\boldsymbol{z}}_i^t) = \Big(\hat{\boldsymbol{z}}_i^t \cdot \log(\hat{\boldsymbol{z}}_i^t)/\log|\mathcal{C}_s|\Big)_i$ with $i = 1,\ldots,|r|$. We indicate with $[\hat{\boldsymbol{z}}^t]_m$ the m-th component of the $\hat{\boldsymbol{z}}^t$ vector. The full expression of the normality score is:

$$\mathcal{N}(\boldsymbol{x}^t) = \max\left\{\max_{k=1,\ldots,|\mathcal{C}_s|}\left(\sum_{i=1}^{|r|}[\hat{\boldsymbol{z}}_i^t]_{k\times|r|+i}\right), \left(1 - \frac{1}{|r|}\sum_{i=1}^{|r|}H(\hat{\boldsymbol{z}}_i^t)\right)\right\}. \tag{6.3}$$

This formula is a function of the ability of the network to correctly predict the semantic class and orientation of a target sample (first term in the braces) as well as of its confidence evaluated on the basis of the prediction entropy (second term). We maximize over these two components with the aim of taking the most reliable metric in each case. Finally, the normality score is used to separate the target dataset into a known target dataset $\mathcal{D}_t^{knw}$ and an unknown target dataset $\mathcal{D}_t^{unk}$. The distinction is made directly through the data statistics using the average of the normality score over the whole target $\bar{\mathcal{N}} = \frac{1}{N_t}\sum_{j=1}^{N_t}\mathcal{N}_j$, without the need to introduce any further parameter:

$$\begin{cases} \boldsymbol{x}^t \in \mathcal{D}_t^{knw} & \text{if} \quad \mathcal{N}(\boldsymbol{x}^t) > \bar{\mathcal{N}} \\ \boldsymbol{x}^t \in \mathcal{D}_t^{unk} & \text{if} \quad \mathcal{N}(\boldsymbol{x}^t) < \bar{\mathcal{N}} \end{cases}. \tag{6.4}$$

It is worth mentioning that only R_1 is directly involved in computing the normality score, while C_1 is only trained for regularization purposes and as a warm up for the following stage. A pseudo-code on how to compute $\mathcal{N}$ and generate $\mathcal{D}_t^{knw}$ and $\mathcal{D}_t^{unk}$ is presented in algorithm 2.

6.1.5 Stage II: domain alignment

Once the target unknown samples have been identified, the scenario gets closer to that of standard CSDA. On one hand, we can use $\mathcal{D}_t^{knw}$ to close the domain gap without the risk of negative transfer and, on the other hand, we can exploit $\mathcal{D}_t^{unk}$ to extend the original semantic classifier, making it able to recognize the unknown category. Similarly to stage I, the network is composed of an encoder E and two heads: a rotation classifier R_2 and a semantic label classifier C_2. The encoder is inherited from the previous stage. The heads also leverage on the previous training phase but have two key differences with respect to stage I:

1. C_1 has a $|\mathcal{C}_s|$-dimensional output, while C_2 has a $(|\mathcal{C}_s|+1)$-dimensional output because of the addition of the unknown class;
2. R_1 is a multi-rotation classifier with a $(4 \times |\mathcal{C}_s|)$-dimensional output, R_2 is a rotation classifier with a 4-dimensional output.

The rotation prediction is computed as $\hat{\boldsymbol{q}} = \text{softmax}\Big(R_2([E(\boldsymbol{x}),E(\tilde{\boldsymbol{x}})])\Big)$ while the semantic prediction is $\hat{\boldsymbol{g}} = \text{softmax}\Big(C_2(E(\boldsymbol{x})\Big)$. The network is trained to minimize the objective function $\mathcal{L}_2 = \mathcal{L}_{C_2} + \mathcal{L}_{R_2}$, where $\mathcal{L}_{C_2}$ combines the supervised cross-entropy and the unsupervised entropy loss for the classification task, while $\mathcal{L}_{R_2}$ is defined as a cross-entropy for the rotation task. The unsupervised entropy loss is used to involve in the semantic classification process also the unlabeled target samples recognized as known. This loss enforces the decision boundary to pass through low-density areas. More precisely,

$$\mathcal{L}_{C_2} = -\sum_{j\in\{\mathcal{D}_s\cup\mathcal{D}_t^{unk}\}} \hat{\boldsymbol{g}}_j \cdot \log(\boldsymbol{g}_j) - \lambda_{2,1} \sum_{j\in\mathcal{D}_t^{knw}} \hat{\boldsymbol{g}}_j \cdot \log(\hat{\boldsymbol{g}}_j), \tag{6.5}$$

$$\mathcal{L}_{R_2} = -\lambda_{2,2} \sum_{j\in\mathcal{D}_t^{knw}} \hat{\boldsymbol{q}}_j \cdot \log(\boldsymbol{q}_j)\ . \tag{6.6}$$

Once the training is complete, R_2 is discarded and the target labels are simply predicted as $c_j^t = C_2(E(\boldsymbol{x}_j^t))$ for all $j = 1,\dots,N_t$.

6.2 Experiments

6.2.1 Reproducibility

In recent years, the machine learning community has become painfully aware of a reproducibility crisis [133]–[135]. Replicating the results of state-of-the-art deep learning models is seldom straightforward due to a combination of non-deterministic factors

in standard benchmark environments and poor reports from the authors. Although the problem is far from being solved, several efforts have been made to promote reproducibility through checklists [136], challenges [137] and by encouraging authors to submit their code. On our side, we contribute by re-running the state-of-the-art methods for OSDA and compare them with the results reported in the papers (see section 6.2). Our results are produced using the original public implementation together with the parameters reported in the paper and, in some cases, repeated communications with the authors. We believe that this practice, as opposed to simply copying the results reported in the papers, can be of great value to the community.

6.2.2 Setup: baselines, datasets

We validate ROS with a thorough experimental analysis on two widely used benchmark datasets, office-31 and office-home. *Office-31* [86] consists of three domains, Webcam (W), Amazon (A) and Dslr (D), each containing 31 object categories. We follow the setting proposed in [67], where the first 10 classes in alphabetic order are considered known classes and the last 11 classes are considered unknown. *Office-Home* [138] consists of four domains, *product* (Pr), *art* (Ar), *real world* (Rw) and *clipart* (Cl), each containing 65 object categories. Unless otherwise specified, we follow the setting proposed in [68], where the first 25 classes in alphabetic order are considered known classes and the remaining 40 classes are considered unknown. Both the number of categories and the large domain gaps make this dataset much more challenging than office-31.

We compare ROS against the state-of-the-art methods STA [68], OSBP [67], UAN [70], AoD [69]. For each of them, we run experiments using the official code provided by the authors, with the exact parameters declared in the relative paper. The only exception was made for AoD for which the authors have not released the code at the time of writing, thus we report the values presented in their original work. We also highlight that STA presents a practical issue related to the similarity score used to separate known and unknown categories. Its formulation is based on the *max* operator according to the equation in the paper[2], but appears instead based on *sum* in the implementation code. In our analysis we considered both the two variants (STA_{sum}, STA_{max}) for sake of completeness. All the results presented in this section, both for ROS and for the baseline methods, are the average over three independent experimental runs. We do not cherry pick the best out of several trials, but only run the three experiments we report.

6.2.3 Implementation Details

By following standard practice, we evaluate the performances of ROS on office-31 using two different backbones *ResNet-50* [98] and *VGGNet* [100], both pre-trained on ImageNet [4], and we focus on ResNet-50 for office-home.

Encoder E, ResNet-50 it is composed by all the layers of a standard ResNet-50 up to the average pooling layer. We start from the encoder model pre-trained on ImageNet

[2] see eq. (2) in [68]

[4] and we update only the last convolutional block, fine-tuning it with learning rate 0.0003.

Classifiers C_1, C_2, ResNet-50 they are both mainly composed by two Fully Connected (FC) layers. Specifically the first FC has output 256 and is followed by a Batch Normalization [139] layer and Leakly-ReLU (with negative slope angle as 0.2). The second FC changes depending on the classifier: for C_1 it has $|\mathcal{C}_s|$ outputs, while for C_2 it has $|\mathcal{C}_s| + 1$ outputs including the unknown category. All the layers are learned from scratch with learning rate 0.003.

Rotation classifiers R_1, R_2, ResNet-50 they both have the same structure of the classifiers described above. The only difference is in the number of outputs which is $4 \times |\mathcal{C}_s|$ for R_1 and 4 for R_2. All the layers are learned from scratch with learning rate 0.003.

Stage I and Stage II, ResNet-50 The network trained in Stage I is used as starting point for Stage II, and we know that for the semantic classifier the set of categories increases by one. To take it into consideration, in Stage II we set the learning rate of the new unknown class to twice that of the known classes (already learned in Stage I).

Encoder E, VGGNet it is composed by all the layers of a standard VGG-19 up to the second fully connected layer. We start from the encoder model pre-trained on ImageNet [4] and we update only the last two FC layers, finetuning it with learning rate 0.0003.

Classifiers C_1, C_2, R_1, R_2, VGGNet : they have exactly the same structure used for the ResNet-50 case described above.

Stage I and Stage II, VGGNet : The network trained in Stage I is not used as starting point for Stage II. Still we consider the learning rate of the extra unknown class in Stage II higher with respect to the other classes (1.5), but lower than the value used in case of ResNet-50 (2), where Stage II was inheriting the model of Stage I.

Office-31, ResNet-50 : batch size 32, learning rate defined as specified above and decreasing during training with inverse decay scheduling. We used SGD with momentum, setting the weight decay as 0.0005 and momentum as 0.9. The loss weights are set as $\lambda_{1,1} = \lambda_{2,2} = 3$ and $\lambda_{1,2} = \lambda_{2,1} = 0.1$. We ran ROS with 80 epochs for Stage I and 80 for Stage II. Each experiment is repeated three times taking the result on the target at the last epoch.

Office-31, VGGNet : batch size 32, learning rate defined as specified above and decreasing during training with inverse decay scheduling. We used SGD with momentum, setting the weight decay as 0.0005 and momentum as 0.9. The loss weights are set as $\lambda_{1,1} = \lambda_{2,2} = 3$ and $\lambda_{1,2} = \lambda_{2,1} = 0.1$. We ran ROS with 100 epochs for Stage I and 200

for Stage II. Each experiment is repeated three times taking the result on the target at the last epoch.

Office-Home, ResNet-50 : batch size 32, learning rate defined as specified above and decreasing during training with inverse decay scheduling. We used SGD with momentum, setting the weight decay as 0.0005 and momentum as 0.9. The loss weights are set as $\lambda_{1,1} = \lambda_{2,2} = 3$ and $\lambda_{2,1} = 0.1$. With respect to the previous cases, for this dataset adding the center loss to the rotation classifier R_1 seems less relevant: we kept it in the optimization process with a low weight $\lambda_{1,2} = 0.001$. We ran ROS with 150 epochs for Stage I and 45 for Stage II. Each experiment is repeated three times taking the result on the target at the last epoch.

It is worth noting that we essentially use the same set of parameters for all settings. This highlights that our method can generalize across datasets and network architectures without specific fine-tuning of the hyper-parameters.

6.2.4 Results

How does our method compare to the state-of-the-art? Table 6.1 and 6.2 show the average results over three runs on each of the domain shifts, respectively of office-31 and office-home. To discuss the results, we focus on the HOS metric since it is a synthesis of OS* and UNK, as discussed in section 6.2.1. Overall, ROS outperforms the state of the art on a total of 13 out of 18 domain shifts and presents the highest average performance on both office-31 and office-home, with an improvement of up to 2.2% HOS compared to the second best method OSBP. Indeed, we largely improve over STA, regardless of its specific max or sum implementation, while UAN is not a challenging competitor due to its low performance on the unknown class. We can compare against AoD only when using VGG for office-31: we report the original results in gray in Table 6.1, with the HOS value confirming our advantage.

A more in-depth analysis indicate that the advantage of ROS is largely related in its ability in separating known and unknown samples. Indeed, while our average OS* is similar to that of the competing methods, our average UNK is significantly higher. This characteristic is also visible qualitatively by looking at the t-SNE visualizations in Figure 6.4 where we focus on the comparison against the second best method OSBP. Here the features for the known (red) and unknown (blue) target data appear more confused than for ROS.

Is it possible to reproduce the reported results of the state-of-the-art? By analyzing the published OSDA papers, we noticed some incoherence in the reported results. For example, some of the results from OSBP are different between the pre-print [140] and the published [67] version despite having the same description for method and hyper-parameters. Also, AoD [69] compares against the pre-print results of OSBP, while omitting the results of STA. To dissipate these ambiguities and gain a better perspective on the current state-of-the-art methods, we compare in Table 6.3 the OS results on office-31 officially presented in previous works, with the one obtained by running their code. For this analysis we focus on OS since it is the only metric reported for some of the methods. The comparison shows that, despite using the original implementation and

Table 6.1: Accuracy (%) averaged over three runs of each method on office-31 dataset using ResNet-50 and VGGNet as backbones

Office-31

ResNet-50

		A → W			A → D			D → W			W → D		
		OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
STA_{sum}	[68]	92.1	58.0	71.0	95.4	45.5	61.6	97.1	49.7	65.5	96.6	48.5	64.4
STA_{max}	[68]	86.7	67.6	75.9	91.0	63.9	75.0	94.1	55.5	69.8	84.9	67.8	75.2
OSBP [67]		86.8	79.2	**82.7**	90.5	75.5	**82.4**	97.7	96.7	**97.2**	99.1	84.2	91.1
UAN [70]		95.5	31.0	46.8	95.6	24.4	38.9	99.8	52.5	68.8	81.5	41.4	53.0
ROS		88.4	76.7	82.1	87.5	77.8	**82.4**	99.3	93.0	96.0	100.0	99.4	**99.7**

		D → A			W → A			avg.		
		OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
STA_{sum}	[68]	94.1	55.0	69.4	92.1	46.2	60.9	94.6	50.5	65.5±0.3
STA_{max}	[68]	83.1	65.9	73.2	66.2	68.0	66.1	84.3	64.8	72.5±0.8
OSBP [67]		76.1	72.3	75.1	73	74.4	73.7	87.2	80.4	83.7±0.4
UAN [70]		93.5	53.4	68.0	94.1	38.8	54.9	93.4	40.3	55.1±1.4
ROS		74.8	81.2	**77.9**	69.7	86.6	**77.2**	86.6	85.8	**85.9±0.2**

VGGNet

	A → W			A → D			D → W			W → D		
	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
OSBP [67]	79.4	75.8	77.5	87.9	75.2	81.0	96.8	93.4	**95**	98.9	84.2	91.0
ROS	80.3	81.7	**81.0**	81.8	76.5	79.0	99.5	89.9	94.4	99.3	100.0	**99.7**
AoD [69]	87.7	73.4	79.9	92.0	71.1	**79.3**	99.8	78.9	88.1	99.3	87.2	92.9

	D → A			W → A			avg.		
	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
OSBP [67]	74.4	82.4	**78.2**	69.7	76.4	72.9	84.5	81.2	82.6±0.8
ROS	76.7	79.6	78.1	62.2	91.6	**74.1**	83.3	86.5	**84.4±0.2**
AoD [69]	88.4	13.6	23.6	82.6	57.3	67.7	91.6	63.6	71.9

the information provided by the authors, the OS obtained by re-running the experiments is between 1.3% and 4.9% lower than the originally reported results. The significance of this gap calls for greater attention in providing all the relevant information for reproducing the results of a published paper. A larger reproducibility study is provided in the supplementary material.

Why is it important to use the HOS metric? The most glaring example of why OS is not an appropriate metric for OSDA is provided by the results of UAN. In fact, when computing OS from the average (OS*,UNK) in Table 6.2 and 6.1, we can see that UAN has OS=72.5% for office-home and OS=91.4% for office-31. This is mostly reflective of the ability of UAN in recognizing the known classes (OS*), but it completely disregards its (in)ability to identify the unknown samples (UNK). For example, for most domain shifts in office-home, UAN does not assign (almost) any samples to the unknown class, resulting in UNK=0.0%. On the other hand, HOS better reflects the open set scenario and assumes a high value only when OS* and UNK are both high.

Is rotation classification effective for known/unknown separation in OSDA? To better understand the effectiveness of rotation classification for known/unknown separation,

Table 6.2: Accuracy (%) averaged over three runs of each method on office-home dataset using ResNet-50 as backbone

Office-Home												
	Pr → Rw			**Pr → Cl**			**Pr → Ar**			**Ar → Pr**		
	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
STA_{sum} [68]	78.1	63.3	69.7	44.7	71.5	55.0	55.4	73.7	63.1	68.7	59.7	63.7
STA_{max}	76.2	64.3	69.5	44.2	67.1	53.2	54.2	72.4	61.9	68.0	48.4	54.0
OSBP	76.2	71.7	73.9	44.5	66.3	53.2	59.1	68.1	**63.2**	71.8	59.8	65.2
UAN	84.0	0.1	0.2	59.1	0.0	0.0	73.7	0.0	0.0	81.1	0.0	0.0
ROS	70.8	78.4	**74.4**	46.5	71.2	**56.3**	57.3	64.3	60.6	68.4	70.3	**69.3**
	Ar → Rw			**Ar → Cl**			**Rw → Ar**			**Rw → Pr**		
	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
STA_{sum} [68]	81.1	50.5	62.1	50.8	63.4	56.3	67.9	62.3	65.0	77.9	58.0	66.4
STA_{max}	78.6	60.4	68.3	46.0	72.3	55.8	67.5	66.7	67.1	77.1	55.4	64.5
OSBP	79.3	67.5	72.9	50.2	61.1	55.1	66.1	67.3	66.7	76.3	68.6	72.3
UAN	88.2	0.1	0.2	62.4	0.0	0.0	77.5	0.1	0.2	85.0	0.1	0.1
ROS	75.8	77.2	**76.5**	50.6	74.1	**60.1**	67.0	70.8	**68.8**	72.0	80.0	**75.7**
	Rw → Cl			**Cl → Rw**			**Cl → Ar**			**Cl → Pr**		
	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**	OS*	UNK	**HOS**
STA_{sum}	51.4	57.9	54.2	69.8	63.2	66.3	53.0	63.9	57.9	61.4	63.5	62.5
STA_{max}	49.9	61.1	54.5	67.0	66.7	66.8	51.4	65.0	57.4	61.8	59.1	60.4
OSBP	48.0	63.0	54.5	72	69.2	**70.6**	59.4	70.3	**64.3**	67.0	62.7	64.7
UAN	66.2	0.0	0.0	80.6	0.1	0.2	70.5	0.0	0.0	74.0	0.1	0.2
ROS	51.5	73	**60.4**	65.3	72.2	68.6	53.6	65.5	58.9	59.8	71.6	**65.2**
	avg.											
	OS*	UNK	**HOS**									
STA_{sum}	63.4	62.6	61.9±2.1									
STA_{max}	61.8	63.3	61.1±0.3									
OSBP	64.1	66.3	64.7±0.2									
UAN	75.2	0.0	0.1±0.0									
ROS	61.6	72.4	**66.2± 0.3**									

we measure the performance of our Stage I and compare it to the Stage I of STA. Indeed, also STA has a similar two-stage structure, but uses a multi-binary classifier instead of a multi-rotation classifier to separate known and unknown target samples. To assess the performance, we compute the *area under receiver operating characteristic curve* (AUROC) over the normality scores $\mathcal{N}$ on office-31. Table 6.4 shows that the AUROC of ROS (0.9146) is significantly higher than that of the multi-binary used by STA (0.7999). Table 6.4 also shows the performance of Stage I when alternatively removing the center loss from equation (6.2) ($\lambda_{1,2} = 0$) and the anchor image when training R_1, thus passing from relative rotation to the more standard absolute rotation recognition task. In both these cases, the AUROC significantly drops compared to our full method, but still outperforms the multi-binary classifier of STA.

Why is the normality score defined the way it is? As defined in equation (6.3), our normality score is a function of the rotation score and entropy score. The rotation score is based on the ability of R_1 to predict the rotation of the target samples, while the

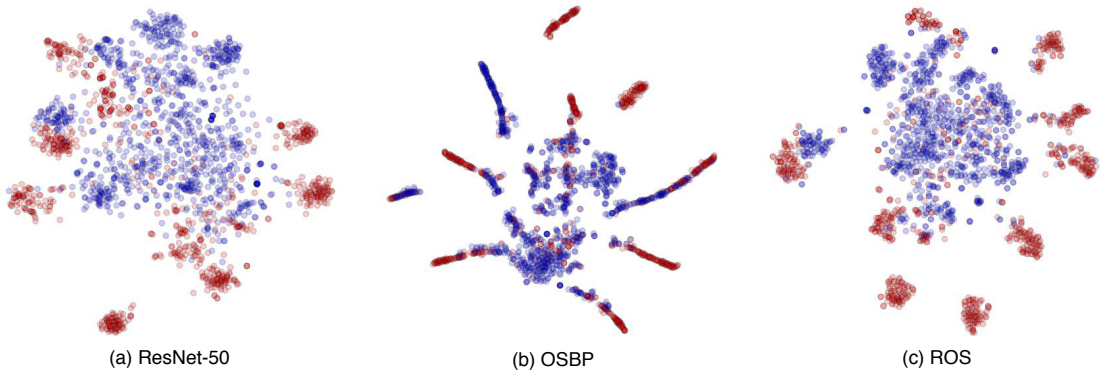


Figure 6.4: Visualization using t-SNE embeddings of the features extracted by ResNet-50(a), OSBP(b) and ROS(c) on W→A domain shift from office-31. Red points are target features of known classes, blue points are target features of unknown classes

Table 6.3: Reported vs reproduced OS accuracy (%) averaged over three runs on all the sub-domains of office-31 and office-home with the indicated backbones.

Reproducibility Study								
Office-31 (ResNet-50)						**Office-31 (VGGNet)**		
	STA_{sum}			UAN			OSBP	
$\text{OS}_{\text{reported}}$	OS_{ours}	gap	$\text{OS}_{\text{reported}}$	OS_{ours}	gap	$\text{OS}_{\text{reported}}$	OS_{ours}	gap
92.9	90.6±1.8	**2.3**	89.2	87.9±0.03	**1.3**	89.1	84.2 ±0.4	**4.9**
Office-Home (ResNet-50)								
	STA_{sum}			UAN				
$\text{OS}_{\text{reported}}$	OS_{ours}	gap	$\text{OS}_{\text{reported}}$	OS_{ours}	gap			
69.5	63.3±2.1	**6.2**	77.0	75.1 ±0.2	**1.9**			

Table 6.4: Ablation Analysis on Stage I and Stage II

Ablation Study							
Stage I	**A → W**	**A → D**	**D → W**	**W → D**	**D → A**	**W → A**	**avg.**
	AUROC	*AUROC*	*AUROC*	*AUROC*	*AUROC*	*AUROC*	*AUROC*
ROS	0.9006	0.8808	0.9940	0.9998	0.8747	0.8379	**0.9146**
Multi-Binary (from STA[68])	0.8315	0.8409	0.8679	0.7201	0.7566	0.7825	0.7999
ROS - No Center loss	0.8882	0.8319	0.988	0.9978	0.8473	0.8450	0.8997
ROS - No Anchor	0.8450	0.8487	0.9907	0.9987	0.8759	0.8620	0.9035
ROS - No Rotation Score	0.8630	0.8265	0.9948	0.9993	0.8626	0.8285	0.8958
ROS - No Entropy Score	0.8070	0.7870	0.9968	0.9989	0.8655	0.8442	0.8832
Stage II	**A → W**	**A → D**	**D → W**	**W → D**	**D → A**	**W → A**	**avg.**
	HOS	*HOS*	*HOS*	*HOS*	*HOS*	*HOS*	*HOS*
ROS	82.1	82.4	96.0	99.7	77.9	77.2	**85.9**
ROS Stage I - GRL[55] Stage II	83.5	80.9	97.1	99.4	77.3	72.6	85.1
ROS Stage I - No Anchor in Stage II	80.0	82.3	94.5	99.2	76.9	76.6	84.9

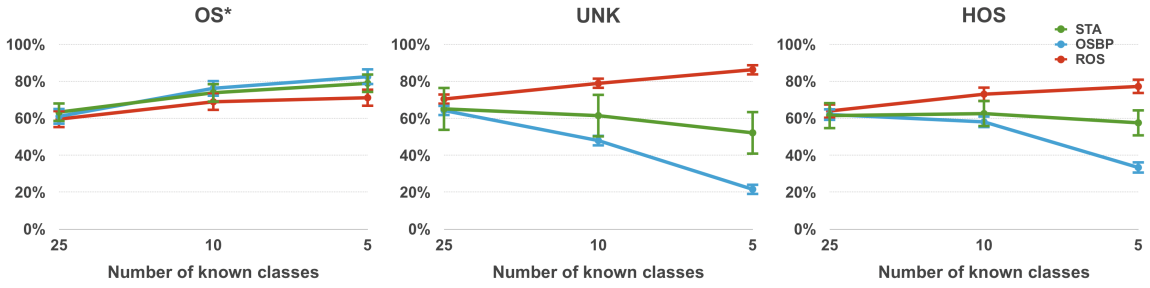


Figure 6.5: Accuracy (%) averaged over the three configurations designed for each degree of openness considered: with 25,10 and 5 known classes

entropy score is based on the confidence of such predictions. Table 6.4 shows the results of Stage I when alternatively discarding either the information of the rotation score (ROS - No Rotation Score) or the information of the entropy score (ROS - No Entropy Score). In both cases the AUROC significantly decreases compared to the full version, which supports our choice of combining these two score components.

Is rotation classification effective for domain alignment in OSDA? While rotation classification has already been used for CSDA [29], its application in OSDA, where the shared target distribution could be noisy (i. e. contain unknown samples) has not been studied. On the other hand, GRL [55] is used, under different forms, by all existing OSDA methods. We compare rotation classification and GRL in this context by evaluating the performance of our Stage II when replacing the R_2 with a domain discriminator. Table 6.4 shows that rotation classification performs on par with GRL, if not slightly better. Moreover we also evaluate the role of the relative rotation in the Stage II: the results in the last row of Table 6.4 confirms that it improves over the standard absolute rotation (No Anchor) even when the rotation classifier is used as cross-domain adaptation strategy.

Is our method effective on problems with a high degree of openness? The standard open set setting adopted in so far, presents a relatively balanced number of shared and private target classes with openness close to 0.5. Specifically it is $\mathbb{O} = 1 - \frac{10}{21} = 0.52$ for office-31 and $\mathbb{O} = 1 - \frac{25}{65} = 0.62$ for office-home. In real-world problems, we can expect the number of unknown target classes to largely exceed the number of known classes, with openness approaching 1. We investigate this setting using office-home and, starting from the classes sorted with ID from 0 to 64 in alphabetic order, we define the following settings with increasing openness: **25** known classes $\mathbb{O} = 0.62$, ID:{0-24, 25-49, 40-64}, **10** known classes $\mathbb{O} = 0.85$, ID:{0-9, 10-19, 20-29}, **5** known classes $\mathbb{O} = 0.92$, ID:{0-4, 5-9, 10-14}. Figure 6.5 shows that the performance of our best competitors, STA and OSBP, deteriorates with larger $\mathbb{O}$ due to their inability to recognize the unknown samples. On the other hand, ROS maintains a consistent performance.

6.3 Discussion

In this chapter, we present ROS: a novel method that tackles OSDA by using the self-supervised task of predicting image rotation. We show that, with simple variations of the rotation prediction task, we manage to first separate the target samples into known and unknown, and then align the target samples predicted as known with the source samples. Additionally, we propose HOS: a new OSDA metric defined as the harmonic mean between the accuracy of recognizing the known classes and rejecting the unknown samples. HOS overcomes the shortcoming of the current metric OS where the contribution of the unknown classes vanishes with increasing number of known classes.

We evaluate the perfomance of ROS and existing OSDA methods on the standard office-31 and office-home benchmarks. ROS outperforms the competing methods both on average and in 13 out of the 18 domain shifts tested. In addition, when tested on more realistic settings with increasing openness, ROS is the only method that maintains a steady performance. HOS reveals to be crucial in this evaluation to correctly assess the performance of the methods on both known and unknown samples. Finally, the failure in reproducing the reported results of existing methods exposes an important issue in OSDA that echoes the current reproducibility crisis in machine learning. We hope that our contributions can help laying a more solid foundation for the field of OSDA.

Highlights:

- ➔ There is a reproducibility crisis in OSDA, with existing methods not reaching the performance reported in the original paper when re-running the their code.
- ➔ The self-supervised task of relative rotation confirms its effectiveness and allowing both known/unknown discrimination and domain alignment.
- ➔ Using a self-supervised task to perform known/unknown separation produces very robust unknown recognition, allowing to deal with OSDA problems with high degree of openness.

Chapter 7

Conclusion

Object recognition in an unconstrained real-world environment is a capability where the human visual perception is still far superior to any robot vision system. Dealing with hundreds of different objects in different configurations, light conditions, and scale populate the scene, occluding each other in messy clutters, is extremely challenging. However, breaking down this complex problem into smaller sub-tasks gradually enables autonomous systems to comprehend and intelligently react to their surroundings.

Two important cornerstones on the way to more reliable and robust object recognition is (i) to integrate geometric information in the robot visual perception system through depth data and (ii) to adapt the system to the new domains the robot encounters without the need for manually annotated data. New insights on how to approach these two problems are the main contributions of this thesis.

7.1 Summary

In chapter 2, we have analyzed the capabilities of standard CNN-based object classification pipelines on robotic data to better understand their capabilities and how they can be improved. To this purpose, we have presented ARID: a large-scale, multi-view, RGB-D object dataset collected with a mobile robot in the wild. This dataset is designed to capture the challenges a robot faces when deployed in an indoor environment and fills the gap between research-oriented datasets and real-life data. With the support of other two datasets, WOD and ROD, we have assessed whether CNN can learn features that are discriminative in the robotic domain using training data from either the Web or the laboratory domain. In fact, WOD and ROD contain the same categories of ARID, but the former is composed of images downloaded from the Web while the latter contains data collected in a laboratory setting. With extensive experiments, we have shown that, despite being relatively easy to obtain, Web-based data allow the generation of more effective deep models than the lab-collected counterpart for the classification of robotic images. Nevertheless, this naive object classification pipeline presents results that are insufficient for the successful integration of robotic systems in our homes. Through a deeper analysis on ARID, we have identified small images and occlusions, as well as an overall domain gap between the Web and the robotic domain, to be the main causes of the unsatisfactory recognition accuracy. Preliminary experiments on potential solutions have shown that training the network with data augmentation and, especially,

DA strategies can significantly reduce the domain gap and provide better results on occluded objects. These approaches have instead a limited impact in improving the classification of small images that is better tackled using active vision strategies. The findings of this chapter inform the rest of the thesis and narrow the focus on (i) robustify the recognition of occluded object and (ii) use DA to effectively train CNN without the need of manually annotating a large amount of application-specific robotic data.

In chapter 3, we have proposed to robustify object recognition of occluded objects by incorporating depth information in the recognition process. More precisely, we have presented RCFusion: a multi-modal deep neural network for RGB-D object recognition. This method uses two streams of convolutional networks to extract RGB and depth features from multiple levels of abstraction. These features are concatenated and sequentially fed to an RNN to obtain a compact RGB-D feature that is used by a softmax classifier for the final classification. We have shown the validity of our approach by outperforming the existing methods for RGB-D recognition on two standard benchmarks, RGB-D Object Dataset and JHUIT-50. RCFusion presents compelling results also on OCID, a challenging dataset that specifically focuses on scenes with a high level of clutter and occlusion. But it is not all sunshine and rainbows: preliminary experiments on ARID reveal that the technology of current RGB-D cameras fails to provide reliable depth data in an unconstrained setup. Overall, the findings of this chapter reveal that when the depth information is reliable its addition to RGB data can be greatly beneficial for handling occlusions and object recognition in general.

In chapter 4, we have proposed the first method that brings DA in the context of RGB-D object recognition. Our approach consists of training a network to solve the self-supervised task of predicting the relative rotation between the RGB and depth image, in addition to the main object recognition task. To evaluate the performance of our method, we have defined two synthetic-to-real benchmarks for instance recognition and object categorization, using the existing HB and a newly collected dataset called synROD. We demonstrated empirically that our self-supervised task successfully reduces the domain shift and outperforms all considered baselines, indicating that exploiting the inter-modal relations is key to perform DA on RGB-D data. Being able to easily synthesize application-specific training data and make predictions on real data brings us a step closer to enable robot vision in the real world. However, when deployed in-the-wild, it is unrealistic to assume that the robot will only encounter objects from the training categories. This realization has motivated us to relax the closed-set assumption and explore solutions for OSDA.

In chapter 5, we have proposed a novel method to tackle the challenging problem of OSDA by casting it into the theoretical framework of PU learning. Our method, named OSDA-PURE, gets the best of both worlds: (a) it removes the SCAR assumption in PU learning by exploiting the self-supervised power of AEs and domain adversarial training, and (b) it isolates the unknown target samples reducing the effect of negative transfer through a novel reconstruction-based PU risk estimator. Additionally, we have proposed HOS: a new OSDA metric defined as the harmonic mean between the accuracy of recognizing the known classes and rejecting the unknown samples. HOS overcomes the shortcoming of the current metric OS where the contribution of the unknown classes vanishes with increasing number of known classes. Experiments in the

PU learning setting have shown that our AE-based risk estimator is clearly superior to the standard logarithmic instantiation when the P and U sets belong to different domains. Experiments in the OSDA setting have shown that OSDA-PURE performs competitively on standard digit recognition and object classification benchmarks. The results have also shown that there is space for improvement. In fact, training a network on the sample reconstruction task may be difficult in case of data scarcity due to the large number of trainable parameters introduced by an AE architecture.

In chapter 6, using the lessons learned in the previous two chapters, we have presented ROS: a novel method that tackles OSDA by using the self-supervised task of predicting image rotation. We have shown that, with simple variations of the rotation prediction task, we manage to first separate the target samples into known and unknown, and then align the target samples predicted as known with the source samples. We evaluate the performance of ROS and existing OSDA methods on the standard Office-31 and Office-Home benchmarks. Extensive experiments have shown that (i) replicating the results of existing methods is problematic due to a lack of adherence to reproducibility practices and (ii) ROS outperforms the competing methods in the tested benchmarks. In addition, when testing more realistic settings with increasing openness, ROS is the only method that maintains a steady performance. HOS confirms to be crucial in this evaluation to correctly assess the performance of the methods on both known and unknown samples.

7.2 Outlook

To conclude, we provide our outlook on how to build on the findings of this thesis to effectively use them in an unconstrained real-world robotic application.

7.2.1 From open set to universal domain adaptation

In chapter 5 and 6 we moved from closed set to the more realistic open set scenario for DA. In this scenario, the target domain includes all the classes of the source domain and additional unknown classes. However, this is not the only scenario that a robot could encounter. Another branch of DA, called partial DA, considers the inverse case where the target domain only contains a subset of the classes of the source domain. CSDA can be considered as a particular case of both OSDA and partial DA.

When the robot is deployed in the wild, the prior information on the label set required by each of these scenarios (open set, closed set, partial) might be unavailable. For example, let us assume that we have access to a large annotated dataset of objects collected in a traditional cubicle office environment. If we have to deploy a robot in an eccentric and modern open office space, the difference in the design of the furniture and objects, as well as the background creates a domain gap with the training data. In addition, the open office might include a table soccer or communal charging towers that are not present in the cubicle office, while the cubicle office includes compact disks and webcams might not be present in the open office. We cannot select the proper DA method without knowing the relation between the label set of source and target

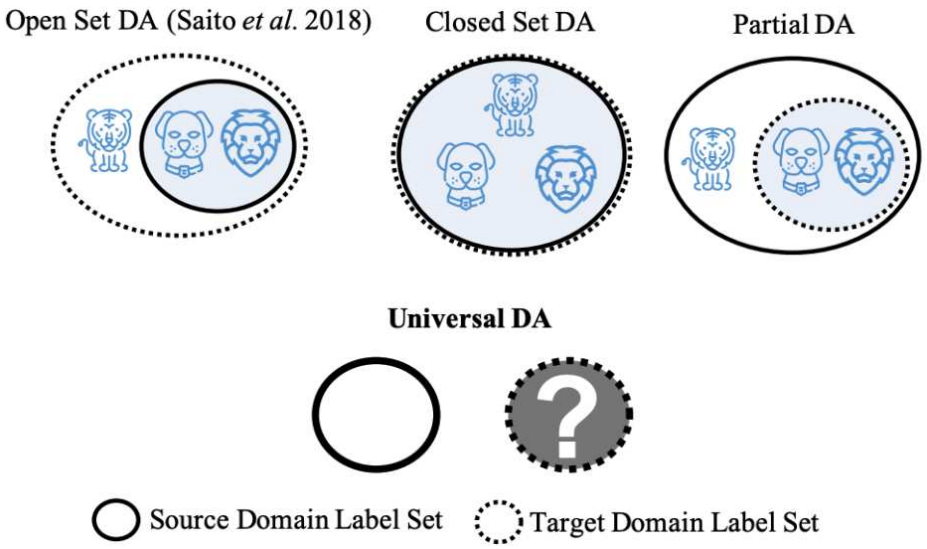


Figure 7.1: Existing domain adaptation settings with respect to label sets of source and target domains. Blue shades indicate the shared label set. Image from [70].

domain. This problem is solved by universal DA [70], which is a generalization of the DA scenarios discussed so far (see figure 7.1). A DA method designed for this setting needs to predict, for both source and target samples, whether they belong to the private or shared label space in order to avoid negative transfer.

At the time of writing, the only published method for universal DA is UAN [70]. This method is a first step in the right direction but, as we have seen in chapter 6, it performs poorly in the open set scenario. The findings of this thesis suggest that a promising possibility is to use a self-supervised task, such as image rotation, to produce a score that indicates whether a sample belongs to the private source, private target, or shared label space.

7.2.2 From recognition to detection and beyond

As it should be obvious by now, object recognition is a classification task that assigns a semantic label to the entire image. In order to locate and recognize several objects in the image, we need to transition from the task of object recognition to object detection. This is necessary for real-world applications since the frames acquired by the robot capture complex scenes with several objects that need to be recognized. It is therefore important to transfer our findings on RGB-D recognition and DA from a classification to a detection setting.

The literature of RGB-D object detection based on deep learning is very limited. However, advances in point cloud classification have inspired new approaches for feature extraction and fusion based on PointNet [141]. The most recent example is the work of He *et al.*[142], that extracts RGB and depth features using a standard CNN and PointNet and uses a dense fusion mechanism to generate pixel-wise multi-modal features. These algorithmic insights could be integrated with RCFusion (see chapter 3) to further advance the state of the art in RGB-D object detection.

Similarly, our findings in the field of DA for object recognition need to be adapted to object detection. Xu *et al.*[29] have shown that the self-supervised task of image rotation can be applied with minor adjustments to perform CSDA on full scenes. In the past years, other methods have been proposed to perform DA for higher-level tasks, such as semantic segmentation [143]. However, none of these methods explicitly deal with the open set (or universal) scenario and would greatly benefit from integrating the findings of chapter 5 and 6.

7.2.3 Multi-modal universal domain adaptation for object detection

Progressing from open set to universal DA and from recognition to detection while being able to process multi-modal data are natural follow-ups of this thesis in the quest of designing a robot visual system that can operate in the wild. After achieving these milestones, the challenge would be to integrate these components into a single module. The order in which the different components are integrated can vary based on algorithmic dependencies. For example, since the methods proposed in chapter 4 and 6 both use the self-supervised task of relative rotation, it might be convenient to first integrate them into an RGB-D OSDA method before transitioning to UDA and object detection. Ultimately, we want to build a module that can process the RGB and depth scenes provided by the RGB-D camera, adapt them to the domain of the training data without prior knowledge on the label sets, while achieving high recognition and detection rate. Such a module is an essential component in the lasting dream of creating intelligent robots.

Bibliography

[1] K. S. Kraebel, R. N. West, and P. Gerhardstein, „The influence of training views on infants' long-term memory for simple 3D shapes," *Developmental Psychobiology: The Journal of the International Society for Developmental Psychobiology*, vol. 49, no. 4, pp. 406–420, 2007.

[2] J. J. Peissig and M. J. Tarr, „Visual object recognition: Do we know more now than we did 20 years ago?" *Annu. Rev. Psychol.*, vol. 58, pp. 75–96, 2007.

[3] A. Krizhevsky, I. Sutskever, and G. Hinton, „Imagenet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25 (NIPS)*, 2012, pp. 1097–1105.

[4] J. Deng, W. D. R. Socher, L. Li, K. Li, and F. Li, „Imagenet: A large-scale hierarchical image database," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[5] L. Bo, X. Ren, and D. Fox, „Depth kernel descriptors for object recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 821–826.

[6] K. Lai, L. Bo, X. Ren, and D. Fox, „A large-scale hierarchical multi-view RGB-D object dataset," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1817–1824.

[7] L. Bo, X. Ren, and D. Fox, „Hierarchical Matching Pursuit for Image Classification: Architecture and Fast Algorithms," in *Advances in Neural Information Processing Systems (NIPS)*, 2011, pp. 2115–2123.

[8] M. Blum, J. T. Springenberg, J. Wülfing, and M. Riedmiller, „A learned feature descriptor for object recognition in RGB-D data," in *IEEE International Conference on Robotics and Automationm (ICRA)*, 2012, pp. 1298–1303.

[9] R. Socher, B. Huval, B.Bhat, C. Manning, and A. Ng, „Convolutional-Recursive Deep Learning for 3D Object Classification," in *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, 2012, pp. 656–664.

[10] A. Eitel, T. Springenberg, L. S. M. Riedmiller, and W. Burgard, „Multimodal Deep Learning for Robust RGB-D Object Recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 681–687.

[11] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, „Learning Rich Features from RGB-D Images for Object Detection and Segmentation," in *Computer Vision – ECCV 2014*, 2014, pp. 345–360.

[12] L. Bo, X. R. D., and Fox, „Unsupervised Feature Learning for RGB-D Based Object Recognition," in *Experimental Robotics: The 13th International Symposium on Experimental Robotics*. 2013, pp. 387–402.

[13] F. M. Carlucci, P. Russo, and B. Caputo, „$(DE)^2CO$: Deep Depth Colorization," *IEEE Robotics and Automation Letter (RA-L)*, vol. 3, no. 3, pp. 2386–2396, 2018.

[14] 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, pp. 506–511.

[15] X. L. T. H. S. Tang X. Wang, J. Keller, and S. L. Z. He M. Skubic, „Histogram of Oriented Normal Vectors for Object Recognition with a Depth Sensor," in *Asian Conference on Computer Vision (ACCV)*, 2013, pp. 525–538.

[16] F. Perronnin, J. Sánchez, and T. Mensink, „Improving the Fisher Kernel for Large-Scale Image Classification," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 143–156.

[17] F. M. Carlucci, P. Russo, and B. Caputo, „A deep representation for depth images from synthetic data," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1362–1369.

[18] A. Aakerberg, K. Nasrollahi, and T. Heder, „Improving a Deep Learning based RGB-D Object Recognition Model by Ensemble Learning," in *IEEE International Conference on Image Processing Theory, Tools and Applications*, 2017, pp. 1–6.

[19] A. Wang, J. Lu, J. Cai, T. J. Cham, and G. Wang, „Large-Margin Multi-Modal Deep Learning for RGB-D Object Recognition," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1887–1898, 2015.

[20] Z. Wang, R. Lin, J. Lu, J. Feng, and J. Zhou, „Correlated and Individual Multi-Modal Deep Learning for RGB-D Object Recognition," *CoRR*, vol. abs/1604.01655, 2016. arXiv: `1604.01655`.

[21] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, „Hypercolumns for object segmentation and fine-grained localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 447–456.

[22] S. Bell, C Lawrence Zitnick, K. Bala, and R. Girshick, „Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2874–2883.

[23] C. Doersch, A. Gupta, and A. A. Efros, „Unsupervised visual representation learning by context prediction," in *ICCV*, 2015, pp. 1422–1430.

[24] M. Noroozi and P. Favaro, „Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, Springer, 2016, pp. 69–84.

[25] R. Zhang, P. Isola, and A. A. Efros, „Colorful image colorization,“ in *ECCV*, Springer, 2016, pp. 649–666.

[26] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, „Context encoders: Feature learning by inpainting,“ in *CVPR*, 2016, pp. 2536–2544.

[27] S. Gidaris, P. Singh, and N. Komodakis, „Unsupervised representation learning by predicting image rotations,“ *arXiv preprint arXiv:1803.07728*, 2018.

[28] I. Golan and R. El-Yaniv, „Deep anomaly detection using geometric transformations,“ in *NeurIPS*, 2018, pp. 9758–9769.

[29] J. Xu, L. Xiao, and A. M. López, „Self-Supervised Domain Adaptation for Computer Vision Tasks,“ *IEEE Access*, vol. 7, pp. 156 694–156 706, 2019.

[30] A. Zimek, E. Schubert, and H.-P. Kriegel, „A survey on unsupervised outlier detection in high-dimensional numerical data,“ *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, pp. 363–387, 2012.

[31] J. Kim and C. D. Scott, „Robust Kernel Density Estimation,“ *J. Mach. Learn. Res.*, vol. 13, no. 1, 2529â€“2565, 2012.

[32] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, „Deep Structured Energy Based Models for Anomaly Detection,“ in *ICML*, 2016.

[33] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, „Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection,“ in *ICLR*, 2018.

[34] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, „A Geometric Framework for Unsupervised Anomaly Detection,“ in *Applications of Data Mining in Computer Security*, D. Barbará and S. Jajodia, Eds. Springer US, 2002, pp. 77–101.

[35] E. J. Candès, X. Li, Y. Ma, and J. Wright, „Robust Principal Component Analysis?“ *J. ACM*, vol. 58, no. 3, 2011.

[36] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun, „Learning Discriminative Reconstructions for Unsupervised Outlier Removal,“ in *ICCV*, 2015.

[37] C. Zhou and R. C. Paffenroth, „Anomaly Detection with Robust Deep Autoencoders,“ in *ACM SIGKDD*, 2017.

[38] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, „Unsupervised anomaly detection with generative adversarial networks to guide marker discovery,“ in *IPMI*, 2017.

[39] J. An and S. Cho, „Variational autoencoder based anomaly detection using reconstruction probability,“ *Special Lecture on IE*, vol. 2, no. 1, 2015.

[40] L. Deecke, R. Vandermeulen, L. Ruff, S. Mandt, and M. Kloft, „Image Anomaly Detection with Generative Adversarial Networks,“ in *MLKDD*, 2019.

[41] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, „Support Vector Method for Novelty Detection,“ in *NeurIPS*, 1999.

[42] L. M. Manevitz and M. Yousef, „One-class Svms for Document Classification," *J. Mach. Learn. Res.*, vol. 2, pp. 139–154, 2002.

[43] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, „Deep One-Class Classification," in *ICML*, 2018.

[44] D. Hendrycks and K. Gimpel, „A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks," in *ICLR*, 2017.

[45] S. Liang, Y. Li, and R. Srikant, „Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks," in *ICLR*, 2018.

[46] W. J. Scheirer, L. P. Jain, and T. E. Boult, „Probability Models for Open Set Recognition," *IEEE T-PAMI*, vol. 36, no. 11, pp. 2317–2324, 2014.

[47] P. R. Mendes Júnior, R. M. de Souza, R. d. O. Werneck, B. V. Stein, D. V. Pazinato, W. R. de Almeida, O. A. B. Penatti, R. d. S. Torres, and A. Rocha, „Nearest neighbors distance ratio open-set classifier," *Machine Learning*, vol. 106, no. 3, pp. 359–386, 2017.

[48] H. Zhang and V. M. Patel, „Sparse Representation-Based Open Set Recognition," *IEEE T-PAMI*, vol. 39, no. 08, pp. 1690–1696, 2017.

[49] A. Bendale and T. Boult, „Towards Open Set Deep Networks," in *CVPR*, 2016.

[50] Z. Ge, S. Demyanov, and R. Garnavi, „Generative OpenMax for Multi-Class Open Set Classification," in *BMVC*, 2017.

[51] R. Yoshihashi, W. Shao, R. Kawakami, S. You, M. Iida, and T. Naemura, „Classification-Reconstruction Learning for Open-Set Recognition," in *CVPR*, 2019.

[52] M. Long, Y. Cao, J. Wang, and M. I. Jordan, „Learning Transferable Features with Deep Adaptation Networks," in *ICML*, 2015.

[53] B. Sun, J. Feng, and K. Saenko, „Return of Frustratingly Easy Domain Adaptation," in *AAAI*, 2016.

[54] R. Xu, G. Li, J. Yang, and L. Lin, „Larger Norm More Transferable: An Adaptive Feature Norm Approach for Unsupervised Domain Adaptation," in *ICCV*, 2019.

[55] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, „Domain-adversarial training of neural networks," *JMLR*, vol. 17, no. 1, pp. 2096–2030, 2016.

[56] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, „Adversarial Discriminative Domain Adaptation," in *CVPR*, 2017.

[57] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo, „From source to target and back: symmetric bi-directional adaptive GAN," in *CVPR*, 2018.

[58] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, „Deep Reconstruction-Classification Networks for Unsupervised Domain Adaptation," in *ECCV*, 2016.

[59] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, „Domain Separation Networks," in *NeurIPS*, 2016.

[60] F. M. Carlucci, A. D'Innocente, S. Bucci, B. Caputo, and T. Tommasi, „Domain generalization by solving jigsaw puzzles," in *CVPR*, 2019, pp. 2229–2238.

[61] F. Qi, X. Yang, and C. Xu, „A unified framework for multimodal domain adaptation," in *ACM Multimedia*, 2018, pp. 429–437.

[62] L. Spinello and K. O. Arras, „Leveraging RGB-D data: Adaptive fusion and domain adaptation for object detection," in *ICRA*, IEEE, 2012, pp. 4469–4474.

[63] J. Hoffman, S. Gupta, J. Leong, S. Guadarrama, and T. Darrell, „Cross-modal adaptation for RGB-D detection," in *ICRA*, IEEE, 2016, pp. 5032–5039.

[64] X. Li, M. Fang, J.-J. Zhang, and J. Wu, „Domain adaptation from RGB-D to RGB images," *Signal Processing*, vol. 131, pp. 27–35, 2017.

[65] W. Jing and Z. Kuangen, „Unsupervised Domain Adaptation Learning Algorithm for RGB-D Staircase Recognition," *arXiv preprint arXiv:1903.01212*, 2019.

[66] P. Panareda Busto and J. Gall, „Open set domain adaptation," in *ICCV*, 2017.

[67] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, „Open Set Domain Adaptation by Backpropagation," in *ECCV*, 2018.

[68] H. Liu, Z. Cao, M. Long, J. Wang, and Q. Yang, „Separate to Adapt: Open Set Domain Adaptation via Progressive Separation," in *CVPR*, 2019.

[69] Q. Feng, G. Kang, H. Fan, and Y. Yang, „Attract or Distract: Exploit the Margin of Open Set," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7990–7999.

[70] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, „Universal domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2720–2729.

[71] M. Everingham, L. V. Gool, C. K. I. W. J., Winn, and A. Zisserman, „The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision (IJCV)*, vol. 88, no. 2, pp. 303–338, 2010.

[72] G. Griffin, A. Holub, and P. Perona, „griffin2007caltech-256 object category dataset," Technical Report 7694, California Institute of Technology, 2007.

[73] A. Torralba and A. A. Efros, „Unbiased look at dataset bias," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1521–1528.

[74] L. Bo, K. Lai, X. Ren, and D. Fox, „Object recognition with hierarchical kernel descriptors," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1729–1736.

[75] N. Massouh, F. Babiloni, T. Tommasi, J. Young, N. Hawes, and B. Caputo, „Learning deep visual object models from noisy web data: How to make it work," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5564–5571.

[76] *Sloth*, https://github.com/cvhciKIT/sloth, Accessed: 2017-09-05.

[77] K. Lai, L. Bo, and D. Fox, „Unsupervised feature learning for 3D scene labeling," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 3050–3057.

[78] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, „BigBIRD: A large-scale 3D database of object instances," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 509–516.

[79] P. Ammirato, P. Poirson, E. Park, and A. Berg, „A Dataset for Developing and Benchmarking Active Vision," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, in press, 2017.

[80] C. Li, A. Reiter, and G. D. Hager, „Beyond spatial pooling: Fine-grained representation learning in multiple domains," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4913–4922.

[81] C. Li, J. Bohren, E. Carlson, and G. D. Hager, „Hierarchical semantic parsing for object pose estimation in densely cluttered scenes," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 5068–5075.

[82] G. Pasquale, C. Ciliberto, L. Rosasco, and L. Natale, „Object identification from few examples by improving the invariance of a Deep Convolutional Neural Network," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4904–4911.

[83] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, and L. Montesano, „The iCub humanoid robot: An open-systems platform for research in cognitive development," *Neural Networks*, vol. 23, no. 8, pp. 1125 –1134, 2010.

[84] B. Chu, V. Madhavan, O. Beijbom, J. Hoffman, and T. Darrell, „Best Practices for Fine-Tuning Visual Classifiers to New Domains," in *Computer Vision – ECCV 2016 Workshops, Proceedings, Part III*, 2016, pp. 435–442.

[85] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, „How transferable are features in deep neural networks?" In *Advances in Neural Information Processing Systems 27 (NIPS)*, 2014, pp. 3320–3328.

[86] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, „Adapting Visual Category Models to New Domains," in *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision (ECCV), Part IV*, 2010, pp. 213–226.

[87] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, „Going deeper with convolutions," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[88] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, „Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," *CoRR*, vol. abs/1412.3555, 2014.

[89] M. Suchi, T. Patten, D. Fischinger, and M. Vincze, „EasyLabel: A Semi-Automatic Pixel-wise Object Annotation Tool for Creating Robotic RGB-D Datasets," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6678–6684.

[90] M. Loghmani, B. Caputo, and M. Vincze, „Recognizing Objects in-the-Wild: Where do we Stand?" In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2170–2177.

[91] *Tensorflow*, https://www.tensorflow.org/, Accessed: 24-04-2018.

[92] M. Zeiler and R. Fergus, „Visualizing and Understanding Convolutional Networks," in *Computer Vision – ECCV 2014: 13th European Conference, Proceedings, Part I*, 2014, pp. 818–833.

[93] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, „Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[94] S. Hochreiter and J. Schmidhuber, „Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[95] J. Schmidhuber and S. Heil, „Sequential neural text compression," *IEEE Transactions on Neural Networks*, vol. 7, no. 1, pp. 142–146, 1996.

[96] M. Mahoney, „Fast Text Compression with Neural Networks," in *Proceedings of the Thirteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2000, pp. 230–234.

[97] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, „Benchmarking in Manipulation Research: Using the Yale-CMU-Berkeley Object and Model Set," *IEEE Robotics Automation Magazine*, vol. 22, no. 3, pp. 36–52, 2015.

[98] K. He, X. Zhang, S. Ren, and J. Sun, „Deep Residual Learning for Image Recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[99] *Pretrained ResNet-18*, https://github.com/HolmesShuan/ResNet-18-Caffemodel-on-ImageNet, Accessed: 21-04-2018.

[100] K. Simonyan and A. Zisserman, „Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1556, 2014.

[101] *Blender*, http://www.blender.org, Accessed: 2020-01-30.

[102] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, „CyCADA: Cycle-Consistent Adversarial Domain Adaptation," in *International Conference on Machine Learning (ICML)*, 2018.

[103] R. Kaskman, S. Zakharov, I. Shugurov, and S. Ilic, „HomebrewedDB: RGB-D Dataset for 6D Pose Estimation of 3D Objects," in *ICCV Workshops*, 2019, pp. 0–0.

[104] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, „3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015, pp. 1912–1920.

[105] L. Yi, L. Shao, M. Savva, H. Huang, Y. Zhou, Q. Wang, B. Graham, M. Engelcke, R. Klokov, V. Lempitsky, *et al.*, „Large-scale 3d shape reconstruction and segmentation from shapenet core55," *arXiv preprint arXiv:1710.06104*, 2017.

[106] M. R. Loghmani, M. Planamente, B. Caputo, and M. Vincze, „Recurrent Convolutional Fusion for RGB-D Object Recognition," *RA-L*, vol. 4, no. 3, pp. 2878–2885, 2019.

[107] 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 *ECCV*, Springer, 2014, pp. 740–755.

[108] D. Li, Y. Yang, Y.-Z. Song, and T. Hospedales, „Deeper, Broader and Artier Domain Generalization," in *ICCV*, 2017.

[109] A. Eitel, T. Springenberg, L. S. M. Riedmiller, and W. Burgard, „Multimodal Deep Learning for Robust RGB-D Object Recognition," in *IROS*, 2015, pp. 681–687.

[110] P. Morerio, J. Cavazza, and V. Murino, „Minimal-Entropy Correlation Alignment for Unsupervised Deep Domain Adaptation," *ICLR*, 2018. [Online]. Available: `https://openreview.net/forum?id=rJWechg0Z`.

[111] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, „Striving for Simplicity: The All Convolutional Net," in *ICLR (workshop track)*, 2015. [Online]. Available: `http://lmb.informatik.uni-freiburg.de/Publications/2015/DB15a`.

[112] L. v. d. Maaten and G. Hinton, „Visualizing data using t-SNE," *JMLR*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[113] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, „To transfer or not to transfer," in *NeurIPS Workshop on Transfer Learning*, 2005.

[114] F. Denis, R. Gilleron, and F. Letouzey, „Learning from Positive and Unlabeled Examples," *Theoretical Computer Science*, vol. 348, no. 1, pp. 70–83, 2005.

[115] C. Elkan and K. Noto, „Learning classifiers from only positive and unlabeled data," in *ACM SIGKDD*, 2008.

[116] M. C. du Plessis, G. Niu, and M. Sugiyama, „Convex formulation for learning from positive and unlabeled data," in *ICML*, 2015.

[117] R. Kiryo, G. Niu, M. C. du Plessis, and M. Sugiyama, „Positive-unlabeled learning with non-negative risk estimator," in *NeurIPS*, 2017.

[118] M. Kato, L. Xu, G. Niu, and M. Sugiyama, „Alternate estimation of a classifier and the class-prior from positive and unlabeled data," *Preprint arXiv:1809.05710*, 2018.

[119] J. Zhao, M. Mathieu, and Y. LeCun, „Energy-based Generative Adversarial Networks," in *ICLR*, 2017.

[120] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. J. Huang, „A tutorial on energy-based learning," in *Predicting Structured Data*, MIT Press, 2006.

[121] E. Merdivan, M. R. Loghmani, and M. Geist, „Reconstruct & Crush Network," in *NeurIPS*, 2017.

[122] M. Kato, T. Teshima, and J. Honda, „Learning from Positive and Unlabeled Data with a Selection Bias," in *ICLR*, 2019.

[123] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, „Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[124] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning*. Springer series in statistics, 2001.

[125] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, „Reading Digits in Natural Images with Unsupervised Feature Learning," in *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[126] A. Krizhevsky and G. Hinton, „Learning multiple layers of features from tiny images," *Master's thesis, Dep. Computer Science, Univ. of Toronto*, 2009.

[127] G. French, M. Mackiewicz, and M. Fisher, „Self-ensembling for visual domain adaptation," in *ICLR*, 2018.

[128] L. P. Jain, W. J. Scheirer, and T. E. Boult, „Multi-class Open Set Recognition Using Probability of Inclusion," in *ECCV*, 2014.

[129] M. Baktashmotlagh, M. Faraki, T. Drummond, and M. Salzmann, „Learning Factorized Representations for Open-set Domain Adaptation," in *ICLR*, 2019.

[130] S. Cicek and S. Soatto, „Unsupervised Domain Adaptation via Regularized Conditional Alignment," in *ICCV*, 2019.

[131] L. Bergman and Y. Hoshen, „Classification-Based Anomaly Detection for General Data," in *International Conference on Learning Representations*, 2020.

[132] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, „A Discriminative Feature Learning Approach for Deep Face Recognition.," in *ECCV*, 2016.

[133] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, „Deep reinforcement learning that matters," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[134] J. Dodge, S. Gururangan, D. Card, R. Schwartz, and N. A. Smith, „Show Your Work: Improved Reporting of Experimental Results," in *Proceedings of EMNLP*, 2019.

[135] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, „Are gans created equal? a large-scale study," in *Advances in neural information processing systems*, 2018, pp. 700–709.

[136] *The Machine Learning Reproducibility Checklist*, https://www.cs.mcgill.ca/ jpineau/ReproducibilityC Accessed: 4-3-2020.

[137] *Reproducibility Challenge*, https://reproducibility-challenge.github.io/neurips2019/, Accessed: 4-3-2020.

[138] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, „Deep hashing network for unsupervised domain adaptation," in *CVPR*, 2017.

[139] S. Ioffe and C. Szegedy, „Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Preprint arXiv:1502.03167*, 2015.

[140] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, „Open Set Domain Adaptation by Backpropagation," *arXiv preprint arXiv:1804.10427*, 2018.

[141] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, „Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[142] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, „PVN3D: A Deep Point-wise 3D Keypoints Voting Network for 6DoF Pose Estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2020, in press.

[143] Y. Li, L. Yuan, and N. Vasconcelos, „Bidirectional learning for domain adaptation of semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6936–6945.